Exercise 1)

Given the following data samples

$$\mathbf{X} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 0 & 0 \\ 0 & 2 \\ 1 & 2 \\ 2 & 1 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ -1 \end{bmatrix},$$

$-0.1+0.1=0 \rightarrow 0$ ✓
$-0.1+0.1=0 \rightarrow 0$ ✓
$0.1 \rightarrow 1$ ✓
$0.7+0.1,0.3 \rightarrow 1$ ✗
$0.4 \rightarrow 1$ ✗
$0.4 \rightarrow 1$ ✗

find the linear discriminant function using the **Batch perceptron algorithm** and the criterion function

$$L_p(\boldsymbol{w}, b) = -\sum_{i:y_i f(\boldsymbol{x}_i)<0} y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b)$$

Initialize $\boldsymbol{w} = [0.1, 0.1]^T,\ b = 0.1,\ \eta = 1,\ \theta = 0$

Use the L_1 norm to compute the termination condition

begin initialize $\boldsymbol{w}, b, \eta, \theta$
  do
    select samples for which $y_i f(\boldsymbol{x}_i) < 0$
    update $\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \nabla_{\boldsymbol{w}} L_p(\boldsymbol{w}, b)$ and $b \leftarrow b - \eta \nabla_b L_p(\boldsymbol{w}, b)$
      (using the selected samples)
  until $\eta(|\nabla_{\boldsymbol{w}} L_p(\boldsymbol{w}, b)| + |\nabla_b L_p(\boldsymbol{w}, b)|) \leqslant \theta$
end

Be careful it is L1-norm, it is the termination criterion

With respect to

- Here we have to compute the derivatives w.r.t $\boldsymbol{w}$ and b:

$$\nabla_{\boldsymbol{w}} L_p(\boldsymbol{w}, b) = -\sum_{i:y_i f(\boldsymbol{x}_i)<0} y_i \boldsymbol{x}_i = -\left(\begin{bmatrix} 0 \\ -2 \end{bmatrix} + \begin{bmatrix} -1 \\ -2 \end{bmatrix} + \begin{bmatrix} -2 \\ -1 \end{bmatrix} \right) = \begin{bmatrix} 3 \\ 5 \end{bmatrix}$$

$$\nabla_b L_p(\boldsymbol{w}, b) = -\sum_{i:y_i f(\boldsymbol{x}_i)<0} y_i = -(-1-1-1) = 3$$

Recall that we are considering a **batch** algorithm

- all samples are considered for updating $\boldsymbol{w}$ and b

Exercise 2)

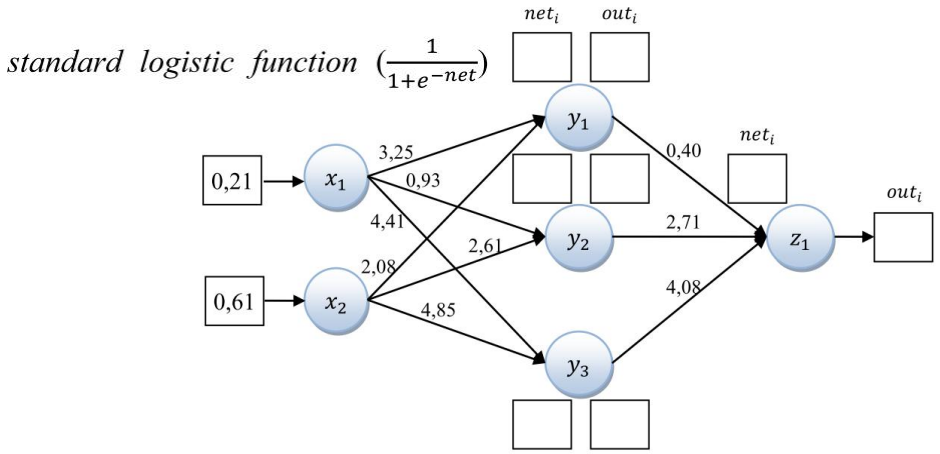
Given a 3-level MLP neural network (with bias) made up of:
6 neurons for the input layer
8 neurons for the hidden layer
5 output neurons
How many sums and multiplications are necessary for the forward step of a generic pattern (neglecting the operations performed by the activation function)?
Justify your answer.

mult add bias mult add bias

$6 \cdot 8 + 8 + 8 + 8 \cdot 5 + 5 + 5$

Exercise 3)

Given the following neural network, calculate the values net and out of each neuron, using the standard logistic function



Exercise 4)

A multiclassifier, it is built by 5 classifiers, combined at the decision level using the Majority vote rule, is used to recognize patterns belonging to 4 classes. The following table shows the outputs returned by the single classifiers (C_i) input data for 3 different patterns (p_j).
Which are the output classes returned by the multiclassifier?

	C_1	C_2	C_3	C_4	C_5
$\boldsymbol{p}_1$	1	2	1	1	3
$\boldsymbol{p}_2$	1	2	2	3	4
$\boldsymbol{p}_3$	3	2	1	3	3

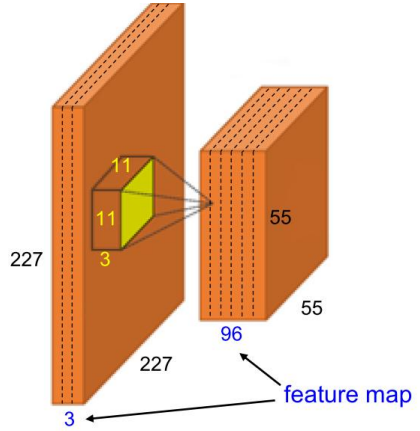
Exercise 5)

A multiclassifier, it is composed by 3 classifiers combined at the confidence level, is used to recognize patterns belonging to 3 classes (A, B, C). The following table shows the confidences returned by the classifiers (C_i) considering as input data 3 different patterns (p_j). Complete the table by reporting, for each fusion method (Sum, Product, Maximum and Minimum), the confidences obtained and the output class

	C_1			C_2			C_3		
	A	B	C	A	B	C	A	B	C
$\boldsymbol{p}_1$	0,15	0,81	0,04	0,02	0,56	0,42	0,54	0,12	0,34
$\boldsymbol{p}_2$	0,31	0,24	0,45	0,54	0,41	0,05	0,02	0,03	0,95
$\boldsymbol{p}_3$	0,42	0,46	0,12	0,77	0,21	0,02	0,41	0,30	0,29

Exercise 6)

Given an input volume and an output volume, related to a convolution layer in a CNN, having the following dimensions:
Input: 3 X 227 X 227
Output: 96 x 55 x 55
Considering that "Filter Size": 3 X 11 X 11
Calculate the total number of connections and weights (<u>without</u> considering the bias). Justify your answer.



Exercise 7)

Suppose you trained a given classifier (spam vs not spam), the result on the test set is summarized in the following table.

		Predicted class	
		Spam	not Spam
Actual class	Spam	8	2
	not Spam	16	974

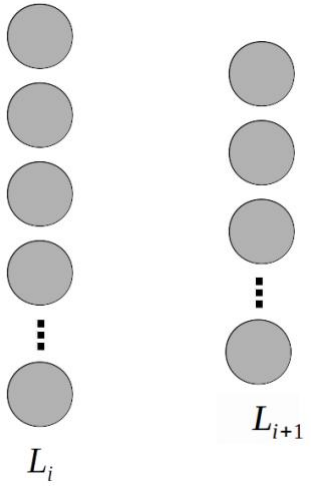
Calculate precision and recall

$Pr = \frac{tp}{tp+fp} = \frac{8}{8+16} = \frac{1}{3}$

$Re = \frac{tp}{tp+fn} = \frac{8}{8+2} = \frac{4}{5}$

Exercise 8)

Given two levels L_i and L_{i+1}, of a neural network, consisting of 10 and 8 neurons respectively, indicate:
The number of connections;
The number of distinct weights.
Solve both if the two levels constitute a portion of an MLP network and if they belong to a CNN where each neuron of level i + 1 is connected to 3 neurons of level i (receptive field = 3). Justify the answer.



Exercise 9)

Assuming to use K-fold Cross-Validation with K = 4 and to divide 6000 patterns into training and validation sets, how many different trainings (runs) are performed? At each run, how many patterns are used for training and how many for validation?

Exercise 10)

There are 100 samples in total, 50 are Yes and 50 are No (Yes and No are the two classes);

The results of the classification are as follows:

55 are classified as Yes (45 of them are in line with reality, and 10 are not in line with reality);

45 are classified as No (40 of them are in line with the actual situation, and 5 are not in line with the reality);

Calculate Precision and Recall of this algorithm.

$tp = 45 \quad fp = 10$

$tn = 40 \quad fn = 5$

$Pr = \frac{tp}{tp+fp} = \frac{45}{55} = \frac{9}{11}$

$Re = \frac{tp}{tp+fn} = \frac{45}{50} = \frac{9}{10}$

Exercise 11)

Given a 3-level MLP neural network (with bias) made up of:
6 Input neurons
8 Intermediate (hidden) neurons
5 Output neurons
Calculate, justifying the answer, the total number of weights.

Exercise 12)

A given system has a precision of 0.5 and a recall of 0.35. Assuming that in such a system: (true positive + false negative) = 20. Based on these information, how many patterns does the system select as relevant (i.e. assign to the positive class)? justify the result.

$P_r = 0.5$, $Re = 0.35$

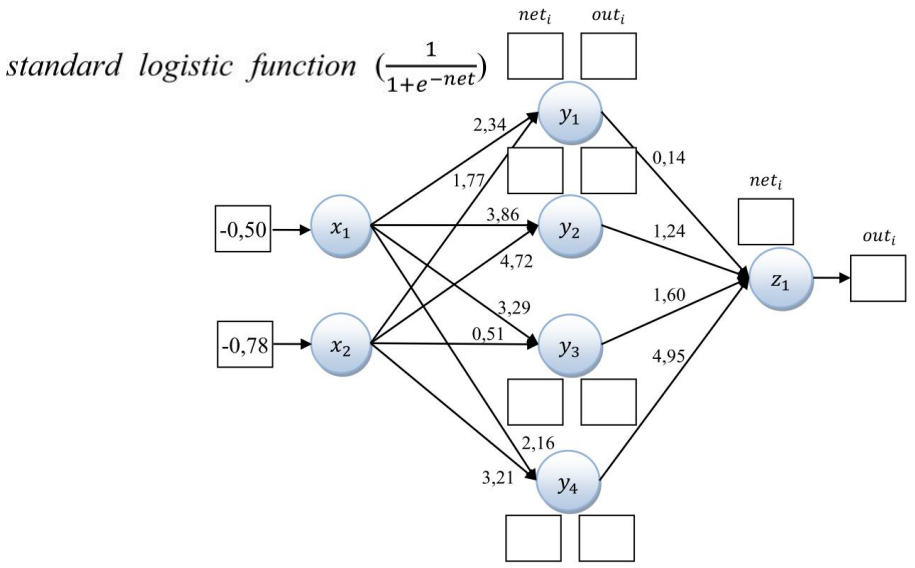
$$\begin{cases} tp + fn = 20 \\ \frac{tp}{tp+fp} = 0.5 \\ \frac{tp}{tp+fn} = 0.35 \end{cases}$$

$tp = 20 \cdot 0.35 = 7$

$tp + fp = 7 / 0.5 = 14$

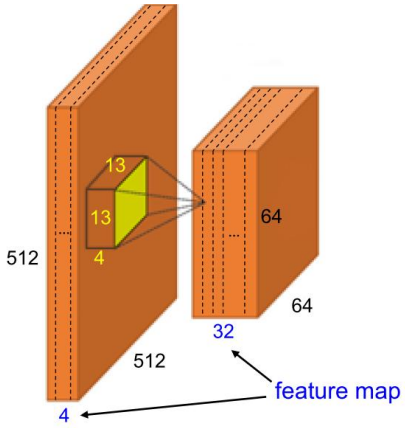
Exercise 13)

Given the following neural network, calculate the net and out of each neuron, using the standard logistic function



Exercise 14)

Given an input volume and an output volume, related to a convolution layer in a CNN, having the following dimensions:
Input: 4 x 512 x 512
Output: 32 x 64 x 64
Considering that “Filter Size”: 4 x 13 x 13
Calculate the total number of connections and weights (considering the bias). Justify your answer.



Exercise 15)

A multiclassifier, it is built by 5 classifiers, combined at the decision level using the Majority vote rule, is used to recognize patterns belonging to 4 classes. The following table shows the outputs returned by the single classifiers (C_i) input data for 3 different patterns (p_j).
Which are the output classes returned by the multiclassifier?

	C_1	C_2	C_3	C_4	C_5
$\boldsymbol{p_1}$	3	1	1	3	3
$\boldsymbol{p_2}$	2	3	1	4	2
$\boldsymbol{p_3}$	4	1	4	2	3

Exercise 16)

Given a convolution layer of a CNN with an input of 28 x 32 x 4 (Width x Height x Depth format) and filters having dimensions of 6 x 6 x 4. Calculate the size(Width x Height) of each feature map in the Output volume, considering a Padding = 3 and Stride = 2.

Exercise 17)

Consider a supervised classification problem in which the training set is made up of the following seven examples that belong to two classes with labels A and B, and are described by two real-valued attributes: $(0.5, 0.5, \mathtt{A})$, $(1, 0.75, \mathtt{A})$, $(2.0, 0.5, \mathtt{A})$, $(1, 0.25, \mathtt{B})$, $(2, 0.25, \mathtt{B})$, $(2.5, 0.75, \mathtt{B})$, $(3, 0.5, \mathtt{B})$. Determine whether it is possible to obtain a classifier consistent with the above training set using:

multi-layer perceptron network.

For each affirmative answer determine *how many* distinct, consistent classifiers of the corresponding type can be built, and define one such classifier.

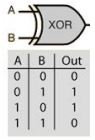
Notation:

- attribute vector: $\mathbf{x} = (x_0, x_1, ..., x_d)$
- weight vector: $\mathbf{w} = (w_0, w_1, ..., w_d)$
- perceptron input: $a(\mathbf{x}, \mathbf{w}) = w_0 x_0 + w_1 x_1 + \ldots + w_d x_d$
- perceptron output (activation):

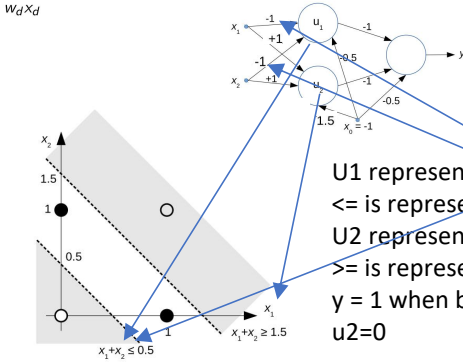
$$y = g(a) = \begin{cases} 1, & \text{if } a \geq 0 \\ 0, & \text{if } a < 0 \end{cases}$$

Take a cue from this Exercise:

For instance, for Boolean inputs x_1, x_2 it is easy to see that the simple perceptron network below implements the XOR function



A	B	Out
0	0	0
0	1	1
1	0	1
1	1	0



U1 represents x1+x2<=0.5
<= is represented by -1,
U2 represents x1+x2>=1.5
>= is represented by +1
y = 1 when both u1=0 and u2=0

Exercise 18)

Below is a table showing how two human judges rated the relevance of a set of 12 documents to a particular information need (0 = nonrelevant, 1 = relevant). Let us assume that you've written an AI system that returns (i.e. it classifies them as relevant) the following set of documents {4, 5, 6, 7, 8}. Calculate precision and recall of your system if a document is considered relevant only if the two judges agree (i.e. the true label is the rate of the judges).

DocID	1	2	3	4	5	6	7	8	9	10	11	12
Judge 1	0	0	1	1	1	1	1	1	0	0	0	0
Judge 2	0	0	1	1	0	0	0	0	1	1	1	1

predicted relevant

fn

tp

really relevant

fp

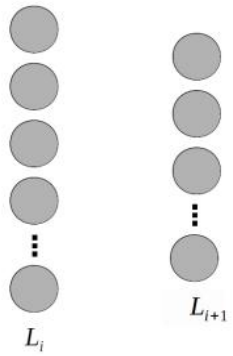
tn

$Pr = \frac{tp}{tp+fp} = \frac{1}{5}$

$Re = \frac{tp}{tp+fn} = \frac{1}{2}$

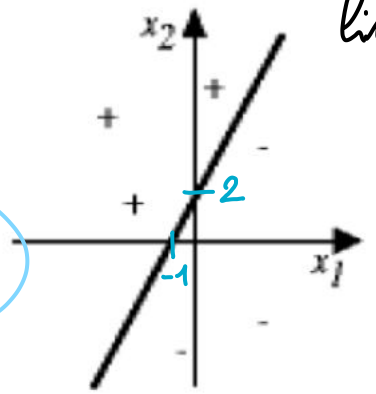
Exercise 19)

Given two layers L_i and L_{i+1} , of a neural network, consisting of 12 and 9 neurons respectively, indicate:
The number of connections;
The number of distinct weights.
Solve both if the two levels constitute a portion of an MLP network and if they belong to a CNN where each neuron of level i + 1 is connected to 5 neurons of level i (receptive field = 5). Justify the answer.



Exercise 20)

What are the values of weights w_0, w_1, and w_2 for the perceptron whose decision surface is illustrated in the figure? Assume the surface crosses the x_1 axis at -1 and the x_2 axis at 2.



line : (-1,0) , (0,2)

$\frac{y-0}{2-0} = \frac{x+1}{1} \Rightarrow y = 2x+2$

The equation of a line passing through two points:

$\frac{y-y_1}{y_2-y_1} = \frac{x-x_1}{x_2-x_1}, \quad x_2 \neq x_1 , \; y_2 \neq y_1$

$w_0 = -2$
$w_1 = -2$
$w_2 = 1$

$w_0 < 0$
perché (0,0) ha label −1

The output of the perceptron is

$o = sgn(w_0 + w_1x_1 + w_2x_2)$

The equation of the decision surface (the line) is

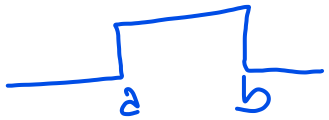
$$w_o + w_1x_1 + w_2x_2 = 0$$

Exercise 21)

We define Randomized Leaky ReLU in the following way:

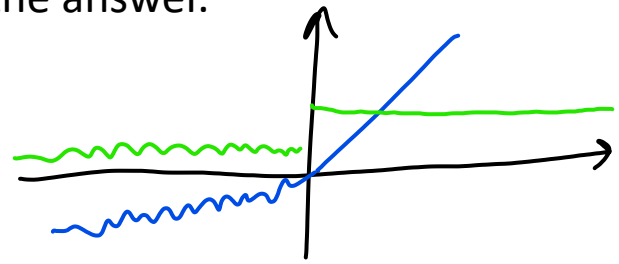
α drawn from a uniform distribution

$$\alpha \sim \mathcal{U}(a, b)$$



$$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$

where a=0.125, b=0.375
plot f(x) and f'(x), where $x \in [-3,3]$
Justify the answer.



Exercise 22)

Given a 3-level MLP neural network (with bias) made up of:
24 Input neurons
48 Intermediate neurons
3 Output neurons
Calculate, justifying the answer, the total number of weights.

Exercise 23)

Given a CNN network, given an Input image of size 7 x 7 x 3 (Width x Height x Depth) and a convolution layer composed by a single filter of size 3 x 3 x 3 with padding = O and stride = 2, calculate the elements of the output volume of the highlighted cells, justifying the answer.

Input

Depth 0

197	103	42	252	27	78	205
114	57	2	195	7	1	130
97	71	179	60	187	22	21
86	84	187	229	208	167	237
25	177	236	250	25	9	87
217	175	190	175	23	10	69
67	127	246	142	4	125	87

Depth 1

124	164	158	18	229	152	110
19	111	22	75	167	224	88
136	21	201	237	248	43	136
151	245	140	163	12	207	19
212	197	87	203	42	149	157
12	78	232	52	113	232	198
64	167	99	112	42	236	186

Depth 2

105	45	160	12	81	207	228
174	18	111	216	200	91	62
170	191	128	124	74	187	123
199	224	184	134	66	193	87
77	41	50	226	226	88	106
151	182	191	216	198	184	93
60	120	91	168	141	136	150

Filter

Depth 0

0.89	0.87	0
0	0.3	0.52
0	0	0.38

Depth 1

0.24	0.9	0
0.07	0	0.64
0.01	0	0.41

Depth 2

0	0.71	0
0.4	0	0
0	0.1	0.3

Exercise 24)

Given an input **x,** a set of weights $W^{(1)}$ and a bias $b^{(1)}$
Assuming ReLu as the activation function, what is the value of the next layer neurons?

Exercise 25)

The Leaky ReLU activation function is defined as:
where a is a small number, here a=0.01
In the backward step, how Leaky ReLu modifies the value of an input (of the activation function) x if:
a) x=-0.7
b) x=5;
?

Justify the answer.

$$y_i = f(x_i) = \begin{cases} ax_i, & x_i < 0 \\ x_i, & x_i \geq 0 \end{cases}$$

Exercise 26)

You have a 32x32x5 image and a 5x5x5 filter, what is the size of the convolution output, without using padding and stride=1, between the image and the filter? Justify your answer.

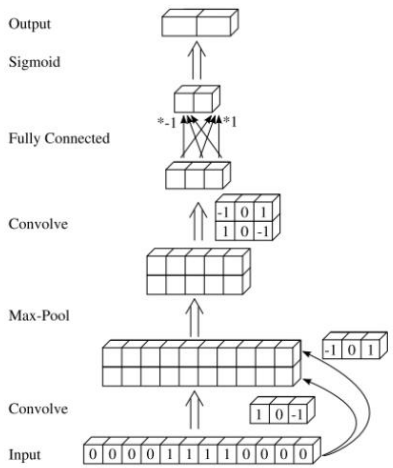
Exercise 27)

In this exercise we address a convolutional neural network (CNN) with one-dimensional input. While two-dimensional CNNs can be used for example for grayscale images, one-dimensional CNNs could be used for time-series such as temperature or humidity readings. Concepts for the 1D-case are equivalent to 2D networks. We interpret data in our network as three-dimensional arrays where a row denotes a feature map, a column denotes a single dimension of the observation, and the depth of the array represents different observations. As we will only work with a single input vector, the depth will always be one.

Let the following CNN be given:

- Input I: Matrix of size $1 \times 12 \times 1$. We therefore have an input with twelve dimensions consisting of a single feature map.
- First convolutional layer with filters $F_0^1 = (-1, 0, 1)$ and $F_1^1 = (1, 0, -1)$ that generates two output feature maps from a single input feature map.
- Max-pooling layer with stride 2 and filter size 2. Note that max-pooling pools each feature map separately.
- Convolutional layer with convolutional kernel $F_0^2 = ((-1, 0, 1), (1, 0, -1))$ of size $2 \times 3 \times 1$.
- Fully connected layer that maps all inputs to two outputs. The first output is calculated as the negative sum of all its inputs, and the second layer is calculated as the positive sum of all its inputs.
- Sigmoidal activation function $\phi(z) = \frac{1}{1+e^{-z}}$

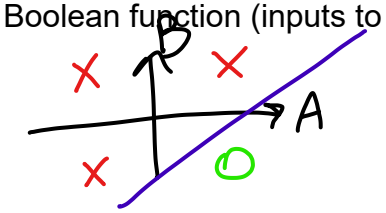
Calculate the response of the CNN for the input $(0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0)$



Exercise 28)

Report weights of a perceptron (showing and explaining the procedure adopted to obtain them) which implements the following Boolean function (inputs to the network are A and B):

A	B	O = A∧¬B
-1	-1	-1
-1	1	-1
1	-1	1
1	1	-1



Draw the decision surface of your perceptron in a simple 2D plot

$w_0 = -\frac{1}{2}$ $w_1 = 1$ $w_2 = -1$

(w₀ preso con il segno e X)

$-x_2 + x_1 - \frac{1}{2} = 0$

Exercise 29)

Given:

- a pattern x_1 with value -1 (single feature) of class l;
- a perceptron with a single weight $w_1 = 0.5$, classification threshold set at zero, learning rate set at 0.6 and without bias.

Apply an iteration of the learning algorithm (using x_1 as the only training pattern) and report the new value of w_1.

$\alpha = 0.6$ learning rate, $y_1 = 1(0.5 \cdot (-1)) = 0$ (w_1 = 0.5, x_1 = -1)

$\Delta w_1 = \alpha \cdot (t_1 - y_1) \cdot x_i = 0.6 \cdot 1 \cdot -1 = -0.6$ (t_1 = 1, y_1 = 0, x_i = -1)

$w_1' = w_1 - \Delta w_1 = -0.1 \longrightarrow y_1' = 1(-0.1 \cdot -1) = 1$ new count

Exercise 30)

Draw a network made up of 3 neurons (no bias, a two-neuron input layer and an output neuron), having three binary inputs X_1 , X_2 and 1 (i.e. a fixed input with a value of 1), output Y of the network implements the function X_1 XOR X_2. To report all the weights of the net.

The activation function is the following (a=2):

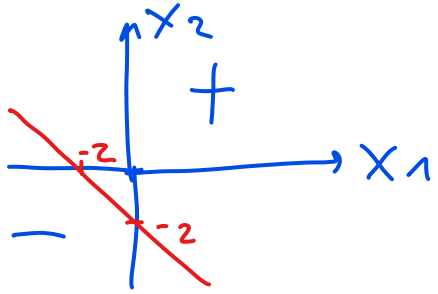
$$h(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w}^T\mathbf{x} >= a \\ 0 & \text{otherwise} \end{cases}$$

$w_2 x_2 + w_1 x_1 + w_0$

Exercise 31)

Given a perceptron with weights $w_0 = 2$; $w_1 = 1$; $w_2 = 1$

show the partition of R^2 (i.e. the areas assigned to the two classes) created by the given perceptron, specify which is the area where the output of the perceptron is 1.



$x_2 + x_1 + 2 = 0$

$x_2 = -x_1 - 2$

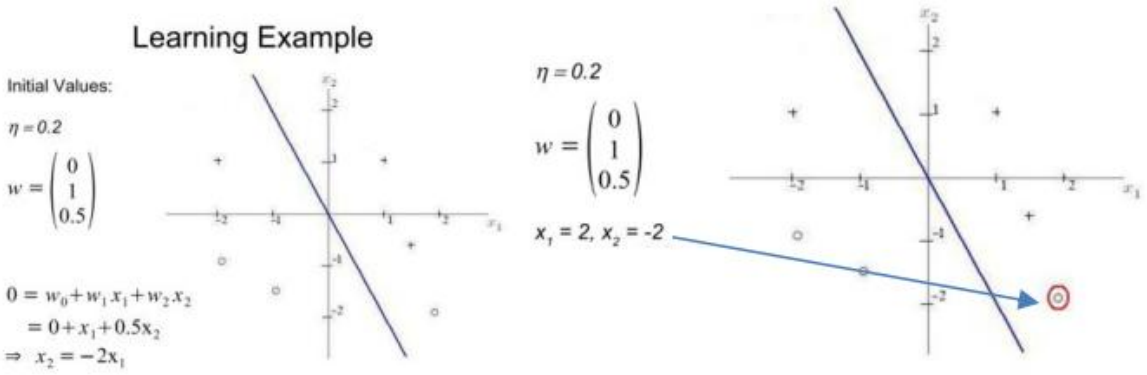
Exercise 32)

Given a perceptron with the following weights ($w_0 = 2$; $w_1=1$; $w_2=1$) which of the following perceptrons create the same hyperplane and which classify the patterns in the same way (i.e. assign the same class to a given pattern).

$(w_0, w_1, w_2)^T$

(I) $(1, 0.5, 0.5)^T$
(II) $(200, 100, 100)^T$
(III) $(\sqrt{2}, \sqrt{1}, \sqrt{1})^T$
(IV) $(-2, -1, -1)^T$

Exercise 33)



Given the figure on the left, which contains the initial situation of the perceptron, update the weights considering only the pattern circled in red (in the figure on the right), uncorrectly classified as “positive”. Learning rate = 0.2.

Exercise 34)

The SReLU activation function is defined as follows

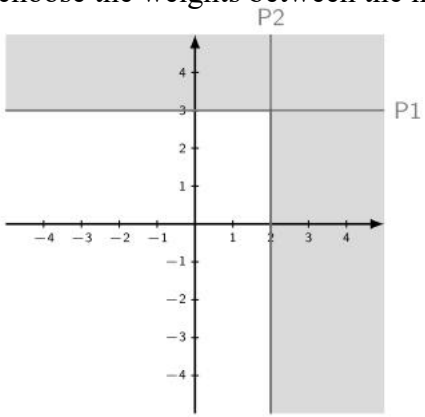
$$h(x_i) = \begin{cases} t_i^r + a_i^r(x_i - t_i^r), & x_i \geq t_i^r \\ x_i, & t_i^r > x_i > t_i^l \\ t_i^l + a_i^l(x_i - t_i^l), & x_i \leq t_i^l \end{cases}$$

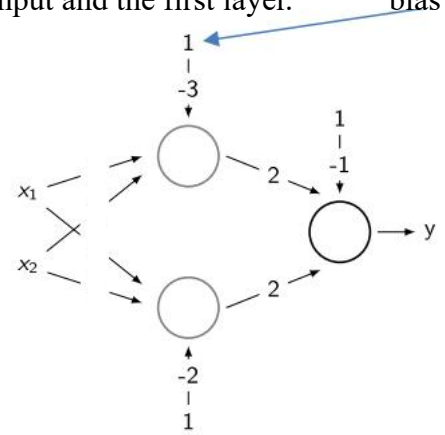
The function has 4 sets of parameters to optimize during the training: t^l, t^r, a^l and a^r. Report the gradient for each parameter of SReLU.

Exercise 35)

To develop a neural network with two input (x_1 and x_2), three neurons (two in the first layer and single output neuron) and a bias, ReLu as activation function. This network has to create the decision surface reported in the below figure, start to the draft reported in the figure (right). I.e. you have to choose the weights between the input and the first layer. bias

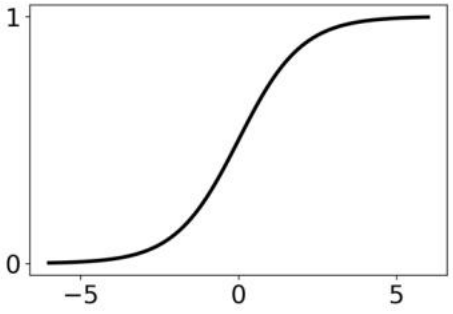
gray area and white area are the two partitions created by the net



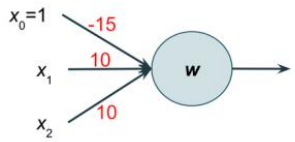


Exercise 36)

Recall the sigmoid function



Consider the neural network below that has three input units: x_0 is the bias term and is always 1, x_1 can take on a value of 0 or 1, and x_2 can take on a value of 0 or 1. The single unit in the output layer uses a sigmoid activation function.



Which Boolean logic gate functions does this neural network model? Why?

Exercise 37)

Given a MLP network and a training set of 1000 patterns, we optimize the network using SGD for 8 epochs with a batch size of 25 patterns. How many times the weight vectors are updated in the training step?

Exercise 38)

If the derivative of the activation function is f (x) (1-f (x)) what activation function are we talking about? Mathematically justify the answer.

Exercise 39)

We define the following similarity between two patterns x and y:
sim = f(A*(|g(x)-g(y)|)+b)

g: neural network, g (x) its output given input x
A: weights
b: bias
f: activation function

In case A has positive values, is sim a metric? Justify your solution mathematically.

Exercise 40)

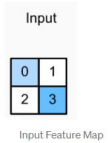
In reinforcement learning, what happens in the discounted future reward if γ = 0 or γ = 1? Explain how the rewards are considered in both cases.

Exercise 41)

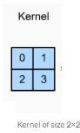
Given the following explanation (layer used in segmentation networks):

Transposed Convolutions

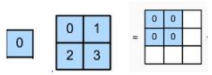
Transposed Convolutions are used to upsample the input feature map to a desired output feature map using some learnable parameters.
The basic operation that goes in a transposed convolution is explained below:
1. Consider a 2x2 encoded feature map which needs to be upsampled to a 3x3 feature map.



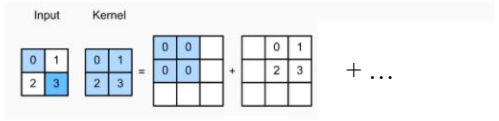
2. We take a kernel of size 2x2 with unit stride and zero padding.



3. Now we take the upper left element of the input feature map and multiply it with every element of the kernel as shown in figure

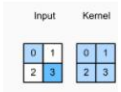


4. Similarly, we do it for all the remaining elements of the input feature map as depicted in figure



5. As you can see, some of the elements of the resulting upsampled feature maps are over-lapping. To solve this issue, we simply add the elements of the over-lapping positions.

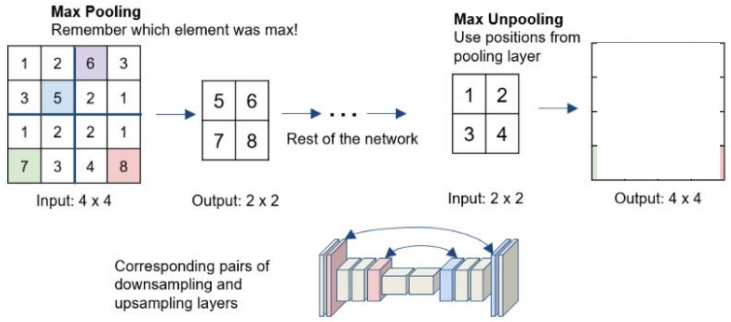
What is the 3 × 3 output of the transposed convolution between the given input and the given kernel? Justify the answer



Exercise 42)

Given the following definition of max unpooling, to report the final output of the below example

Max-Unpooling: The Max-Pooling layer in CNN takes the maximum among all the values in the kernel. To perform max-unpooling, first, the index of the maximum value is saved for every max-pooling layer during the encoding step. The saved index is then used during the Decoding step where the input pixel is mapped to the saved index, filling zeros everywhere else.



Exercise 43)

Given the predicted probability $\mathbf{p} \in \mathbb{R}^n$ and ground-truth label $\mathbf{y} \in \mathbb{R}^n$, calculate the gradient of cross entropy loss $H(\mathbf{y}, \mathbf{p}) = -\sum_{i=1}^{n} y_i \log p_i$ wrt. every element of $\mathbf{p}$, i.e. $\frac{\partial H}{\partial p_i}(\mathbf{y}, \mathbf{p})$.

Exercise 44)

Consider an 1-layer neural network $\mathbf{y} = g(\mathbf{A}\mathbf{x})$ with input $\mathbf{x}$, output $\mathbf{y}$, network weight $\mathbf{A}$ and output function g. Let's first assume the input and the network weights are

$$\mathbf{A} = \begin{pmatrix} 3.0 & 2.0 \end{pmatrix} \in \mathbb{R}^{1\times 2} \qquad \mathbf{x} = \begin{pmatrix} 2.0 & 1.0 & -1.0 \\ 4.0 & -2.0 & 0.0 \end{pmatrix} \in \mathbb{R}^{2\times 3}$$

where $\mathbf{x}$ are 2D points with a **minibatch** size of 3.

i) Assume the function g is linear, i.e. $g(\mathbf{A}\mathbf{x}) = \mathbf{A}\mathbf{x}$. Given the target output $\mathbf{t}$ and the loss function $\mathcal{L}$, perform weight update. Assume a learning rate of 1. Please provide the loss *before* and *after* the weight update.

$$\mathbf{t} = \begin{pmatrix} 15 & 3 & 1 \end{pmatrix} \in \mathbb{R}^{1\times 3} \qquad \mathcal{L} = \frac{1}{2}\sum_j (y_j - t_j)^2$$

First calculate the forward pass and obtain the loss. Next, you can derive the gradients of loss wrt. to every element of the network weight $\frac{\partial \mathcal{L}}{\partial a_i}$. Make sure to include all steps of your derivation. Once the network weight is updated, calculate the forward pass again to acquire the loss.

update rule (A=(a_1 a_2)):

$$a_1^{t+1} = a_1^t - \eta \cdot \frac{1}{3}\frac{\partial \mathcal{L}^t}{\partial a_1} =$$

$$a_2^{t+1} = a_2^t - \eta \cdot \frac{1}{3}\frac{\partial \mathcal{L}^t}{\partial a_2} =$$

chain rule for:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{A}} = \frac{\partial \mathcal{L}}{\partial \mathbf{y}}\frac{\partial \mathbf{y}}{\partial \mathbf{A}}$$

Exercise 45)

Now we set the function g to be ReLU. For $\mathbf{A}\mathbf{x} \in \mathbb{R}^{1\times 3}$, ReLU is applied to every element. Now please calculate the gradients wrt. the network weight $\frac{\partial \mathcal{L}}{\partial a_i}$. You can use the chain rule $\frac{\partial \mathcal{L}}{\partial a_i} = \frac{\partial \mathcal{L}}{\partial g}\frac{\partial g}{\partial a_i}$. What do you notice in the calculated gradient?

Exercise 46)

Prove the equation: $\text{Softmax}(\mathbf{x}) = \text{Softmax}(\mathbf{x} + c)$, where c is a constant.

Exercise 47)

Derivation of the normalization term in Adam.

$$\mathbf{m}^{t+1} = \beta_1 \mathbf{m}^t + (1-\beta_1)\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t)$$
$$\mathbf{v}^{t+1} = \beta_2 \mathbf{v}^t + (1-\beta_2)\left(\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t) \odot \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t)\right)$$
$$\hat{\mathbf{m}}^{t+1} = \frac{\mathbf{m}^{t+1}}{1-\beta_1^{t+1}}, \hat{\mathbf{v}}^{t+1} = \frac{\mathbf{v}^{t+1}}{1-\beta_2^{t+1}}$$
$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta\frac{\hat{\mathbf{m}}^{t+1}}{\sqrt{\hat{\mathbf{v}}_{t+1}}+\epsilon}$$

Derive the explicit form of $\mathbf{v}^t$ w.r.t. $\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t) \odot \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t)$. $\mathbf{v}^t$ in the form of $\sum_t f^t(\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t) \odot \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^t), \beta_2)$.

In mathematics, an explicit function is defined as a function in which the dependent variable can be explicitly written in terms of the independent variable. In standard form, we can write an explicit function as y = f(x). V is not in explicit form, as v(t+1) depends on v(t)

$\mathbf{v}^0 = 0$

$\mathbf{v}^1 = \beta_2 \mathbf{v}^0 +$

$\mathbf{v}^t = \sum_{i=1}^{t}$

v(0) = 0 and V(1) = Beta_2 multiplied by v(0) plus something else, try expressing v(t) as a summation from 1 to t where there is no variable v.

Hadamard or Element-wise product

For example, the Hadamard product for a 3 × 3 matrix A with a 3 × 3 matrix B is

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \circ \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} & a_{13}b_{13} \\ a_{21}b_{21} & a_{22}b_{22} & a_{23}b_{23} \\ a_{31}b_{31} & a_{32}b_{32} & a_{33}b_{33} \end{bmatrix}$$

Exercise 48)

In the following, we investigate a tiny CNN. It consists of a convolutional layer, followed by a ReLU, followed by max-pooling. The convolutional layer has a kernel w of size 3 (and no bias term), a stride of 1, and a zero-padding of 1 (i.e., the boundary is padded with 0's). The max-pooling has a kernel size of 2 (the size of the window to take a max over). The input x is a 2d matrix (or an image with a single channel, i.e., gray image). We want to transform the image into the target t. To measure the difference between x and t, we use the mean absolute error (or L1 loss). For simplicity, we do not use a regularization term. $\mathcal{L}_{L1}$.

maxpooling uses a stride = 2

$$x = \begin{pmatrix} 2 & 7 & 6 & 4 \\ 6 & 5 & 0 & 4 \\ 0 & 3 & 8 & 4 \\ 0 & 4 & 1 & 2 \end{pmatrix} \qquad w = \begin{pmatrix} 0.5 & 0.3 & 0.5 \\ 0.8 & 1.1 & -1.7 \\ -1.0 & 1.0 & 1.3 \end{pmatrix} \qquad t = \begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}$$

Perform a forward pass through the tiny CNN to calculate $\mathcal{L}_{L1}$.

$$L1LossFunction = \sum_{i=1}^{n} |y_{true} - y_{predicted}|$$

Exercise 49)

The gradient for a max function with respect to a single x_i will be

$$\frac{\partial \max(\mathbf{x})}{\partial x_i}$$

single x_i can be the maximum of several patches, e.g. for a max pool operation with a kernel size 3 and stride 1.

Max-pooling applies the max operations to several patches in the input. We split the input $\mathbf{x}$ into N patches $\mathbf{z}_k$. Then, the gradient for a single patch $\mathbf{z}_k$ wrt. to an input x_i is

$$\frac{\partial \max(\mathbf{z}_k)}{\partial x_i}$$

Compute the analytical gradient for the max-pool layer.

$$\frac{\partial \text{ReLU}(\mathbf{x})}{\partial \mathbf{x}} = \begin{cases} 1 & \mathbf{x} > 0 \\ 0 & \mathbf{x} \leq 0 \end{cases}$$

The idea here is similar - for some entries the gradient will be backpropagated, for some it won't.

Exercise 50)

Derive the gradient for a 1D convolutional layer where the input is $\mathbf{x} = [x_1, x_2, x_3, x_4]^\top$, the weights are $\mathbf{w} = [w_1, w_2]^\top$, and the output is $\mathbf{y} = \mathbf{x} * \mathbf{w}$. Derive the gradient for the filter weights $\frac{\partial \mathbf{y}}{\partial \mathbf{w}}$.

It should be noted that the formalization used in exercises 51-54 is detailed below

We aim to calculate several properties for different configurations of convolutional and max-pooling layers. For the layers, we use the same notation as PyTorch - input channels C_{in}, output channels C_{out}, kernel size K, stride S, and padding P. We use square kernels and equal stride/padding; hence, we only need to specify scalars.

for instance:

- $\text{Conv}(C_{in} = 3, C_{\text{out}} = 16, K = 5, S = 2, P = 1)$: a convolutional layer with 3 input channels and 16 output channels, kernel size 5, stride 2, and padding 1.
- $\text{MaxPool}(K = 2, S = 1)$: max-pooling with a (square) kernel size of 2 with stride 1
- $\text{FC}(C_{in} = 25088, C_{\text{out}} = 4096)$: a fully-connected layer with 25088 input channels, 4096 output channel.
- ReLU, Tanh, LeakyReLU: an activation function. Here, no arguments are needed. An activation function is an elementwise operation. Using the terminology from above, it has a kernel size 1, stride 1, and padding 0.

To avoid clutter, we assume in the following that the arguments as shown above are positional, i.e. $\text{Conv}(C_{in} = 3, C_{\text{out}} = 16, K = 5, S = 2, P = 1) \equiv \text{Conv}(3, 16, 5, 2, 1)$

Ex. 51)

The receptive field R_k is defined as the size of the region in the input that produces a feature in the feature map f_k. Express the receptive field $\mathcal{R}_k$ of a convolutional or pooling layer as function of the stride S, the kernel size K, and the receptive field of the previous layer $\mathcal{R}_{k-1}$. Let $\mathcal{R}_0 = 1$. Then, $\mathcal{R}_1$ corresponds to the receptive field of the *first* layer.
Hint: Sketch a 1D grid and apply several consecutive 1D convolutions with $K = \{2, 3\}$ and $S = \{1, 2\}$. For each feature, answer how many other features influence its value. Based on these examples, you can see a recursive formula emerge.

Exercise 52)

Calculate the receptive field R_k at each layer for the following architectures.

i) $\text{Conv}(32, 128, 3, 1, 1)$ - Relu - $\text{Conv}(128, 128, 4, 4, 1)$

ii) $(\text{Conv}(3, 3, 3, 2, 1)$ - Relu$)*6$ (The same block 6 times)

iii) $\text{Conv}(3, 64, 3, 2, 1)$ - Relu - $\text{MaxPool}(4, 3)$ - $\text{Conv}(64, 128, 3, 1, 1)$ - Relu - $\text{MaxPool}(2, 2)$

$\text{Conv}(C_{in} = 3, C_{\text{out}} = 16, K = 5, S = 2, P = 1) \equiv \text{Conv}(3, 16, 5, 2, 1)$

$\text{MaxPool}(K = 2, S = 1)$: max-pooling with a (square) kernel size of 2 with stride

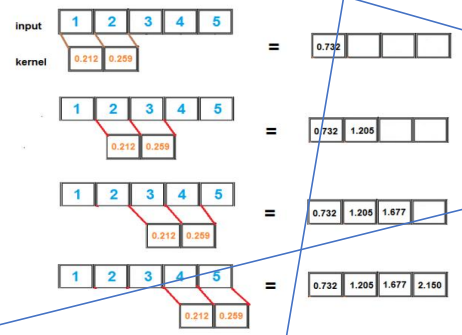
$$\mathcal{R}_k = \mathcal{R}_{k-1} + (K - 1) \prod_{i=1}^{k-1} S_i$$

Exercise 53)

We pass an image of size $C = 3, W = 320, H = 320$ through the network below. Fill in values for the ?'s so that the architecture is valid. Then report the tensor size (C_{out}, H, W) after each layer when the image is passed through the network.

$\text{Conv}(C_{in} = 3, C_{\text{out}} = 16, K = 5, S = 2, P = 1) \equiv \text{Conv}(3, 16, 5, 2, 1)$

$\text{MaxPool}(K = 2, S = 1)$: max-pooling with a (square) kernel size of 2 with stride



Layer
$\text{Conv}(?, ?, 3, 1, 1)$
ReLU
$\text{MaxPool}(2, 2)$
$\text{Conv}(64, ?, 3, 1, 1)$
ReLU
$\text{MaxPool}(2, 2)$
$\text{Conv}(128, 256, 3, 1, 1)$
ReLU
$\text{MaxPool}(2, 2)$
$\text{Conv}(?, 512, 3, 1, 1)$
ReLU
$\text{MaxPool}(2, 2)$

We are asking about Effective Receptive Field.
Effective Receptive Field (ERF): is the area of the original image that can possibly influence the activation of a neuron. The base RF is simply equal to filter size over the previous layer but ERF traces the hierarchy back to the input image and indicates the extent of the input image which can modulate the activity of a neuron. Here, we focus on ERF calculation. It is worth noting that ERF and RF are sometimes used interchangeably (and hence confused) in the computer vision community.

Exercise 54)

Considering the topology reported in the figure, calculate the number of trainable parameters for the first and second convolutional layers. What is the number of parameters of a fully connected layer with 25088 input and an output of 4096 elements?

Layer	Output Shape
Conv(3, 64, 3, 1, 1)	(64, 320, 320)
ReLU	(64, 320, 320)
MaxPool(2, 2)	(64, 160, 160)
Conv(64, 128, 3, 1, 1)	(128, 160, 160)
ReLU	(128, 160, 160)
MaxPool(2, 2)	(128, 80, 80)
Conv(128, 256, 3, 1, 1)	(256, 80, 80)
ReLU	(256, 80, 80)
MaxPool(2, 2)	(256, 40, 40)
Conv(256, 512, 3, 1, 1)	(512, 40, 40)
ReLU	(512, 40, 40)
MaxPool(2, 2)	(512, 20, 20)

- Conv($C_{in} = 3, C_{\text{out}} = 16, K = 5, S = 2, P = 1$): a convolutional layer with 3 input channels and 16 output channels, kernel size 5, stride 2, and padding 1.

MaxPool($K = 2, S = 1$): max-pooling with a (square) kernel size of 2 with stride

Exercise 55)

A hidden layer in an RNN receives the following gradient during backprop:

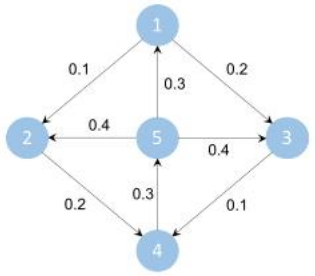
$$\mathbf{g} = \begin{bmatrix} 0.1 & 0.2 & 0.0 & 1.2 & 10.0 & 0.2 \end{bmatrix}^\top$$

Apply gradient clipping with $\tau = 2.0$ τ is the hyperparameter of the clipping

Exercise 57)

For the graph on the left, write down the adjacency matrix and for the adjacency matrix on the right, draw the graph. In both cases, also compute the degree matrix and graph Laplacian.

In this exercise $A_{ji}=w_{ji}$ instead in the lecture notes $A_{ij}=w_{ij}$



(i)

$$\mathbf{A} = \begin{pmatrix} 0 & 0.1 & 0 & 0.3 \\ 0 & 0 & 0.5 & 0 \\ 0.2 & 0 & 0 & 0 \\ 0 & 0 & 0.4 & 0 \end{pmatrix}$$

(ii)

Exercise 58)

For the graph shown above on the left, execute the diffusion process 3 times. Consider the Laplacian as the graph shift operator and the input graph signal $\mathbf{x} = [5, 4, 3, 2, 1]^\top$

$$\mathbf{L} = \begin{pmatrix} 0.3 & 0 & 0 & 0 & -0.3 \\ -0.1 & 0.5 & 0 & 0 & -0.4 \\ -0.2 & 0 & 0.6 & 0 & -0.4 \\ 0 & -0.2 & -0.1 & 0.3 & 0 \\ 0 & 0 & 0 & -0.3 & 0.3 \end{pmatrix}$$

Exercise 56) for a recall of GRU see page 281 of lecture notes

the b-th training pattern described with C_{in} elements

matrices A store the weights

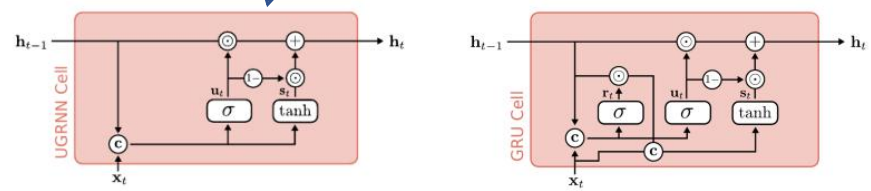
UGRNN. It can be expressed in Einstein Notation as follows:

$$U_t[b, c_{\text{out}}] = \sigma(A_{uh}[c_{\text{out}}, C_{\text{in}}] H_{t-1}[b, C_{\text{in}}] + A_{ux}[c_{\text{out}}, C_{\text{in}}] X_t[b, C_{\text{in}}] + b_u[c_{\text{out}}])$$
$$S_t[b, c_{\text{out}}] = \tanh(A_{sh}[c_{\text{out}}, C_{\text{in}}] H_{t-1}[b, C_{\text{in}}] + A_{sx}[c_{\text{out}}, C_{\text{in}}] X_t[b, C_{\text{in}}] + b_s[c_{\text{out}}])$$
$$H_t[b, c_{\text{out}}] = U_t[b, c_{\text{out}}] H_{t-1}[b, c_{\text{out}}] + (1 - U_t[b, c_{\text{out}}]) S_t[b, c_{\text{out}}]$$

where U_t is the updated gate, S_t is the next target state, and H_t is the hidden state (the output layer is omitted). Below you see the illustration of a UGRNN cell and its (more complicated) predecessor, the GRU.

- Write the equations for the GRU in Einstein Notation.
- What are the changes relative to the UGRNN?
- What can a GRU do that the UGRNN cannot?

Update Gate Recurrent Neural Network (UGRNN)



The convolution equation can be written more compactly in Einstein notation.

Capital letter indices will used to indicate subtensors (slices) so that, for example, $M[I, J]$ denotes a matrix while $M[i, j]$ denotes one element of the matrix, $M[i, J]$ denotes the ith row, and $M[I, j]$ denotes the jth collumn.

By abuse of notation, capital letters will also be used to indicate the range of the indices. A vector $x[I]$ has I dimensions with values $x[0], \ldots, x[I-1]$. A Matrix $M[I, J]$ has $I \times J$ entries.

Einstein notation is the convention that repeated capital indeces in a product of tensors are implicitly summed.

$y = Wx$ abbreviates $y[i] = \sum_j W[i, j]x[j] = W[i, J]x[J]$.

$y = x^\top W$ abbreviates $y[j] = \sum_i W[i, j]x[i] = W[I, j]x[I]$.

$L[b, x, y, i]$ is the value of feature i for batch element b at image position $\langle x, y \rangle$.

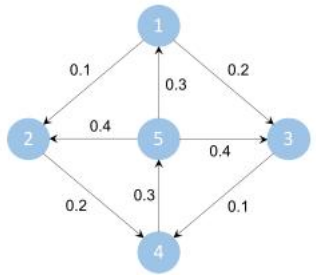
Convolution in Einstein Notation

$$L_{\ell+1}[b, x, y, j]$$
$$= \sigma\left(\left(\sum_{\Delta x, \Delta y, i} W[\Delta x, \Delta y, i, j]\, L_\ell[b, x + \Delta x, y + \Delta y, i]\right) - B[j]\right)$$
$$= \sigma\left(W[\Delta X, \Delta Y, I, j]\, L_\ell[b, x + \Delta X, y + \Delta Y, I] - B[j]\right)$$

Einstein went back to explicit index notation (Einstein notation) when working with the higher order tensors in his theory of gravitation.

Exercise 59)

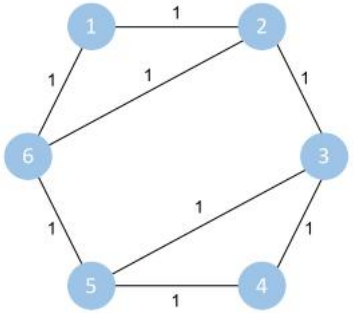
For the same graph, compute the output of the graph convolution with the filter $\mathbf{h} = \{0.2, 0.3, 0.5\}$ for the graph signal $\mathbf{x} = [1, 2, 3, 4, 5]^\top$. Consider the Laplacian as the shift operator.



$$\mathbf{L} = \begin{pmatrix} 0.3 & 0 & 0 & 0 & -0.3 \\ -0.1 & 0.5 & 0 & 0 & -0.4 \\ -0.2 & 0 & 0.6 & 0 & -0.4 \\ 0 & -0.2 & -0.1 & 0.3 & 0 \\ 0 & 0 & 0 & -0.3 & 0.3 \end{pmatrix}$$

Exercise 60)

Consider the graph shown below. Compute the output of the graph convolution for the filter $\mathbf{h} = \{1, 1\}$ for graph signals $\mathbf{x}_1 = [1, 2, 0, 0, 0, 3]^\top$ and $\mathbf{x}_2 = [0, 0, 3, 1, 2, 0]^\top$. Which property of graph convolutions is illustrated based on the nature of the two graph signals? Assume the adjacency matrix as the shift operator.



Exercise 61)

You come across a nonlinear function that passes 1 if its input is nonnegative, else evaluates to 0, i.e.

$$f(x) = \begin{cases} 1 & | \, x >= 0 \\ 0 & | \, x < 0 \end{cases}$$

A friend recommends you use this non-linearity in your convolutional neural network with the Adam optimizer. Would you follow their advice? Why or why not?

Exercise 62)

You are training a single-layer, feedforward neural network with a softmax activation function in the final layer to classify among 99 classes, with a *cross-entropy loss* training objective. Recall, the cross-entropy loss function:

$$L(\mathbf{y}, \hat{\mathbf{y}}) = -\mathbf{y} \cdot \log(\hat{\mathbf{y}})$$

where $\mathbf{y}$ is the *one-hot encoded* label, and $\hat{\mathbf{y}}$ is the *predicted* probability distribution over labels.

You decide to independently sample your initial weights from a Gaussian of mean 0, standard deviation 0.0001. You can assume perfect class balance in dataset.

You *accidentally* set a **0 learning rate**. What would you expect your average loss after the first training epoch to be? Provide a brief explanation (1-2 sentences) as to why this is so. You don't have to calculate the exact numerical value - feel free to leave your answer as a fraction.

Exercise 63)

Consider the convolutional neural network defined by the layers in the left column below. Fill in the shape of the output volume and the number of parameters at each layer. You can write the activation shapes in the format (H, W, C), where H, W, C are the *height, width* and *channel* dimensions, respectively. Unless specified, *assume padding 1, stride 1 where appropriate.*

Notation:

- CONVx-N denotes a convolutional layer with N filters with height and width equal to x. each filter has a number of channels equal to those of the tensor to which the filter is applied
- POOL-n denotes a n×n max-pooling layer with stride of n and 0 padding.
- FLATTEN flattens its inputs, identical to `torch.nn.flatten` / `tf.layers.flatten`
- FC-N denotes a fully-connected layer with N neurons

Layer	Activation Volume Dimensions	Number of parameters
Input	$32 \times 32 \times 3$	0
CONV3-8		
Leaky ReLU		
POOL-2		
BATCHNORM		
CONV3-16		
Leaky ReLU		
POOL-2		
FLATTEN		
FC-10		

Exercise 64)

You decide to use cross entropy loss to train your network. Recall that the cross-entropy loss for a single example is defined as follows: $L_{CE}(\hat{y}, y) = -\sum_{i=1}^{n_y} y_i log(\hat{y}_i)$. where $\hat{y} = (\hat{y_1}, \hat{y_2}...\hat{y_{n_y}})^T$ represents the predicted probability distribution over the classes and $y = (y_1, y_2...y_{n_y})^T$ represents the ground truth vector, which is zero everywhere except for the correct class (eg $y = (1, 0, 0, 0)^T$ for comedy, and $y = (0, 0, 1, 0)^T$ for action).

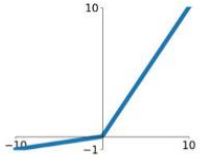
Suppose you're given an example poster of a horror movie. If the model correctly predicts the resulting probability distribution as $\hat{y} = (0.1, 0.4, 0.3, 0.2)$, what is the value of the cross-entropy loss? You can give an answer in terms of logarithms.

to build a classifier that takes in an image of a movie poster and classifies it into one of four genres: comedy, horror, action, and romance. You have been provided with a large dataset of movie posters where each movie poster corresponds to a move with exactly one of these genres.

After some training, the model now incorrectly predicts romance with distribution ⟨0, 0.4, 0, 0.6⟩ for the same poster. What is the new value of the cross-entropy loss for this example?

Leaky ReLU

$\max(0.1x, x)$



Exercise 65)

Prove the following lower bound on the cross-entropy loss for an example with K correct classes:

$$L_{CE}(\hat{y}, y) \geq KlogK$$

Assume you are still using softmax activation with cross-entropy loss with ground truth vector $y \in \{0,1\}^{n_y}$ with K nonzero components.

use Jensen's inequality

$$\log\left(\frac{\sum_{i=1}^{n} x_i}{n}\right) \geq \frac{\sum_{i=1}^{n} \log(x_i)}{n} \quad \text{or} \quad \frac{x_1 + x_2 + \cdots + x_n}{n} \geq \sqrt[n]{x_1 \cdot x_2 \cdots x_n}$$

Exercise 66)

Given the following data

Item	x1	x2	Class
A	1	2	yes =1
B	2	1	yes =1
C	1	1	no =0
D	1	0	no =0

a) Are the data linearly separable? State reasons for your answer.

b) We will train a perceptron on the data. We add a bias $x_0 = -1$ to each of the data points. Suppose the current weigths to be $\boldsymbol{w} = (0, -1, 1)$. Assume a learning rate of 0.1. How should the weights be updated if point A is considered? How would the weights have been updated if the algorithm instead had considered point B?

Exercise 67)

You are solving the binary classification task of classifying images as cat vs. non-cat. 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:

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

You classify all inputs with a final value $\hat{y} \geq 0.5$ as cat images. What problem are you going to encounter?

Exercise 68)

Given the gradient calculated at a point, the Adam optimizer has three distinct steps. First, update the moving averages. Second, apply the bias correction. Third, update the parameters.
Consider the moving average of the square of the gradients. It is given by the recursive formula:

$$s_t = \beta_2 s_{t-1} + (1 - \beta_2) g_t^2$$

Write down the expression for s_t only in terms of the gradients $g_0, g_1, \ldots, g_t$

$$i.e.\ s_t = (something) \sum something \times g_i^2$$

Exercise 69)

we have a neural network with a single hidden layer and the following activation function $\cos(5x^2)$, apply automatic differentiation to find the derivative of this function at the point x = 2.
see page 245-248 of the lecture notes.

Exercise 70) Reverse Mode

Let's try to evaluate the function

$$f(x, y) = xy + \exp(xy)$$

and its gradient at the point $(1, 2)$. We'll use reverse mode this time.
In reverse mode, instead of storing full derivative information at each node, *only the partial derivatives of nodes relative to its children are stored*. For example, if node x_3 has inputs nodes x_1 and x_2, only the partial derivatives $\frac{\partial x_3}{\partial x_1}$ and $\frac{\partial x_3}{\partial x_2}$ are stored. Contrast this with forward mode, where for a function with inputs x and y, this node would store $\frac{\partial x_3}{\partial x}$ and $\frac{\partial x_3}{\partial y}$ via the chain rule.

The reverse mode consists of two passes through the graph. The forward pass first builds the computational graph while storing just the partial derivative information. The reverse pass then starts at the output node and traverses the graph in the reverse direction to find the full partial derivatives.

The "bar notation" is commonly used to denote our backward pass tangents, $\bar{x_i} = \frac{\partial f}{\partial x_i}$. These are sometimes also called the adjoint variable. At the final node in the graph, $f = x_N$, we have $\bar{x_N} = \frac{\partial f}{\partial x_N} = 1$. We then traverse backward through the graph to construct the partial derivative from the chain rule. $\bar{x_{N-1}} = \bar{x_N} \frac{\partial x_N}{\partial x_{N-1}}$. Note that the partial derivative is exactly the value that has already been stored by the forward pass of the graph.

This process is relatively straightforward for nodes with only one child. When we encounter nodes with multiple children, we must perform a summation over the children, which follows directly from the multivariate chain rule.

For x_i with children x_j and x_k, we have

$$\bar{x_i} = \bar{x_j} \frac{\partial x_j}{\partial x_i} + \bar{x_k} \frac{\partial x_k}{\partial x_i}.$$

II. Summary Sketch of Reverse Mode

1. Create the evaluation graph
2. The forward pass does function evaluations
3. The forward pass also saves the partial derivatives of the elementary function at each step
    - **It does not do the chain rule!**
    - It only stores the partial derivatives
    - Examples:
        - If $v_3 = v_1 v_2$ is a node, then we store $\frac{\partial v_3}{\partial v_1}$ and $\frac{\partial v_3}{\partial v_2}$. That's it.
        - If $v_3 = \sin(v_2)$ is a node, then we store $\cos(v_2)$. Notice that there is no $\dot{v}_2$.
4. The reverse pass starts with $\bar{v}_N = \frac{\partial f}{\partial v_N} = 1$ (since f is v_N).
5. Next, the reverse pass gets $\bar{v}_{N-1} = \frac{\partial f}{\partial v_N} \frac{\partial v_N}{\partial v_{N-1}}$.
    - **Note:** $\frac{\partial v_N}{\partial v_{N-1}}$ is already stored from the forward pass.
6. The only trick occurs when we get to a branch in the graph. That is, when the node we're on has more than one child. In

    that case, we sum the two paths. For example, if v_3 has v_4 and v_5 as children, then we do:

$$\bar{v}_3 = \frac{\partial f}{\partial v_3} = \frac{\partial f}{\partial v_4} \frac{\partial v_4}{\partial v_3} + \frac{\partial f}{\partial v_5} \frac{\partial v_5}{\partial v_3}$$

- **Note:** This summation is a manifestation of the chain rule.

The partial derivative of f with respect to u_i can be written as

$$\frac{\partial f}{\partial u_i} = \sum_{j \text{ a child of } i} \frac{\partial f}{\partial u_j} \frac{\partial u_j}{\partial u_i}.$$

At each node i we compute

$$\bar{u}_i += \frac{\partial f}{\partial u_j} \frac{\partial u_j}{\partial u_i}.$$

The $\bar{u}_i$ variable stores the current value of the partial derivative at node i. As mentioned previously, it is sometimes called the adjoint variable.

Exercise 71)

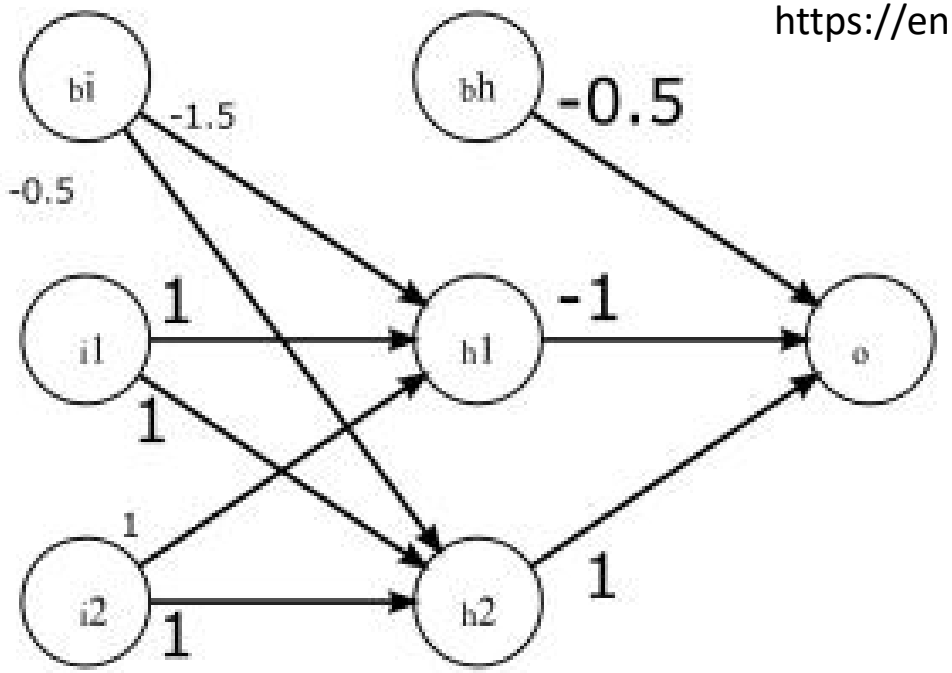
apply the reverse mode for calculating the derivative of
x/(1+|x|)
in x=4

Exercise 72)

The figure below shows a multilayer perceptron that constructs the XOR function. How would you rewrite it to construct the binary equivalence function (i.e. the output is above threshold when both inputs are either 0 or 1)? Can you construct it so that it will detect equivalence for any combination of integer inputs?

The activation function of each neuron has an output of 1 if its input is >=0.5 (otherwise output=0)
bi=bh=1;

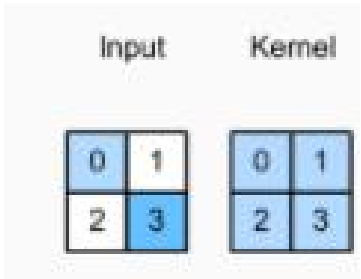
https://en.wikipedia.org/wiki/XNOR_gate



Input		Output
A	B	A XNOR B
0	0	1
0	1	0
1	0	0
1	1	1

Exercise 73)

What is the 4 × 4 output of the transposed convolution (see exercise 41 for details on transposed convolution) between the given input and the following kernel (with stride = 2)



Exercise 74)

Consider an initial parameter vector $\mathbf{w}^{(0)} = [1\ 0\ 2]^T$ and a loss function of the form:

$$J(\mathbf{w}, x_i, t_i) = \frac{1}{4}(\mathbf{x_i}^T\mathbf{w} - t_i)^4$$

for a linear regression model $\hat{t}_i = \mathbf{x_i}^T\mathbf{w}$.

1. Derive the generic update formula for the parameter vector $\mathbf{w}^{(i)}$, given the datum $(\mathbf{x_i}, t_i)$, provided by the gradient descent.
2. Apply the derived formula to the datum $x_1 = [2\ 1\ 3]^T$, $t_1 = 7$ and a learning rate $\alpha = 0.5$.

$$\mathbf{w}^{(i)} = \mathbf{w}^{(i-1)} - \alpha \frac{\partial J(\mathbf{w}, x_i, t_i)}{\partial \mathbf{w}}$$

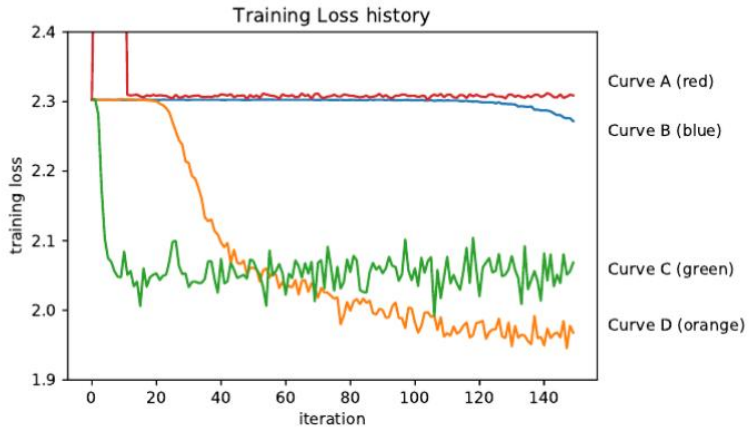
Exercise 75)

We have a system that obtains the following performance indicators (applying it on a dataset consisting of 50 patterns of class '1' and 50 of class '0'):
Accuracy=0.85;
Precision=0.818;
False positives=10
What is recall of this approach?

Exercise 76)

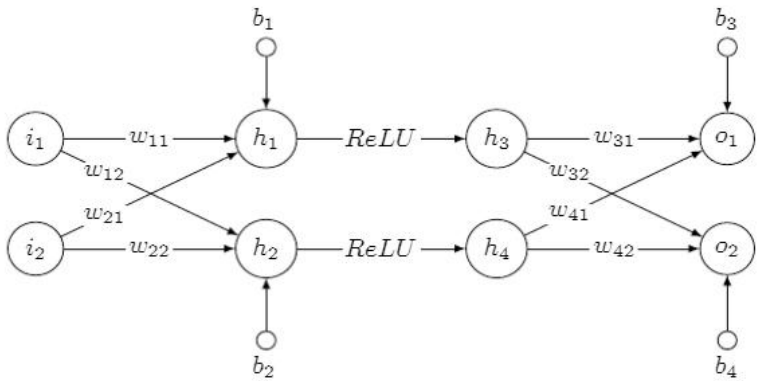
Your colleague trained a neural network using standard stochastic gradient descent and L2 weight regularization with four different learning rates (shown below) and plotted the corresponding loss curves (also shown shown below). Unfortunately he forgot which curve belongs to which learning rate. Please assign each of the learning rate values below to the curve (A/B/C/D) it probably belongs to and explain your thoughts.

learning_rates = [3e−4, 4e−1, 2e−5, 8e−3]



Exercise 77)

Given the following neural network with fully connection layer and ReLU activations, including two input units (i_1, i_2), four hidden units (h_1, h_2) and (h_3, h_4). The output units are indicated as (o_1, o_2) and their targets are indicated as (t_1, t_2). The weights and bias of fully connected layer are called w and b with specific sub-descriptors.



The values of variables are given in the following table:

Variable	i_1	i_2	w_{11}	w_{12}	w_{21}	w_{22}	w_{31}	w_{32}	w_{41}	w_{42}	b_1	b_2	b_3	b_4	t_1	t_2
Value	2.0	-1.0	1.0	-0.5	0.5	-1.0	0.5	-1.0	-0.5	1.0	0.5	-0.5	-1.0	0.5	1.0	0.5

Compute the output (o_1, o_2) with the input (i_1, i_2) and network paramters as specified above. Write down all calculations, including intermediate layer results.

Compute the mean squared error of the output (o_1, o_2) calculated above and the target (t_1, t_2).

Update the weight w_{21} using gradient descent with learning rate 0.1 as well as the loss computed previously. (Please write down all your computations.)

the mean squared error is the loss function

If a vector of n predictions is generated from a sample of n data points on all variables, and Y is the vector of observed values of the variable being predicted, with $\hat{Y}$ being the predicted values (e.g. as from a least-squares fit), then the within-sample MSE of the predictor is computed as

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2.$$

Backward pass (Applying chain rule):

$$\frac{\partial MSE}{\partial w_{21}}$$

Exercise 78)

Suppose a deep network layer performs

$$\mu = \frac{1}{N}\sum_{i=1}^{N} X_i$$

$$Y_i = X_i - \mu \quad \text{for} \quad i = 1, \ldots, N$$

Where N is the dimension of input X and output Y. Note that the sum is taken over the coordinates of the input sample, not the elements of a minibatch as in batch normalization.

Compute the jacobian

The Jacobian $J_Y(X)$ is the matrix of partial derivatives $\frac{\partial Y_i}{\partial X_j}$.

Exercise 79)

Suppose the first qubit is in the state $3/5\,|0\rangle + 4/5\,|1\rangle$ and the second qubit is in the state $1/\sqrt{2}\,|0\rangle - 1/\sqrt{2}\,|1\rangle$, then the joint state of the two qubits is $(3/5\,|0\rangle + 4/5\,|1\rangle)(1/\sqrt{2}\,|0\rangle - 1/\sqrt{2}\,|1\rangle) = 3/5\sqrt{2}\,|00\rangle - 3/5\sqrt{2}\,|01\rangle + 4/5\sqrt{2}\,|10\rangle - 4/5\sqrt{2}\,|11\rangle$

Exercise 80)

Suppose you have a general two-qbit state:

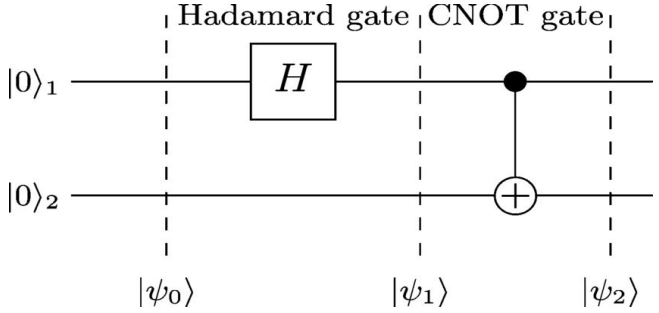
$$|\psi\rangle = \begin{bmatrix} \alpha_{00} \\ \alpha_{01} \\ \alpha_{10} \\ \alpha_{11} \end{bmatrix} = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$$

Now suppose you measure the most-significant (leftmost) qbit in the computational basis (as in, collapse it to either $|0\rangle$ or $|1\rangle$). There are two questions we might ask:

1. What is the probability that the measured qbit collapses to $|0\rangle$? What about $|1\rangle$?
2. What is the state of the 2-qbit system after measurement?

Exercise 81)

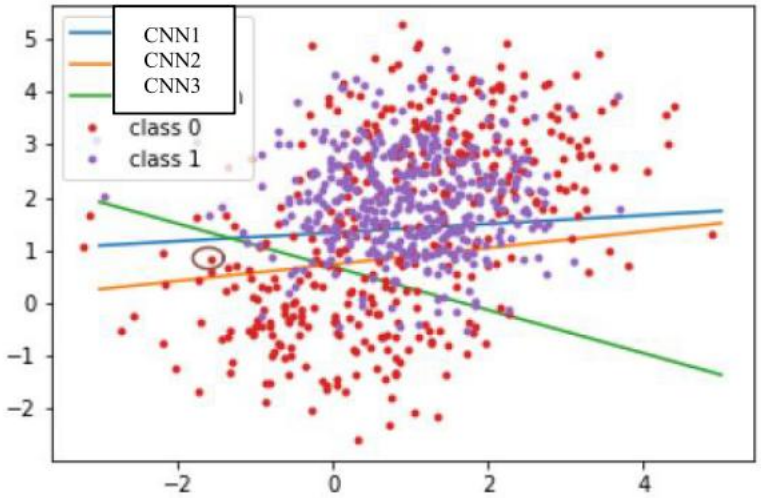
What is the output of this quantum gate?



This is quantum circuit, starting with the initial state $|\psi_0\rangle = |00\rangle = |0\rangle \otimes |0\rangle$. Then it apply the Hadamard gate to the first qubit.

Exercise 82)

We have trained 3 CNNs. We have plotted the decision boundaries for all three classifiers on the training data in the figure. They all classify the points above their boundaries as class 1 (purple) and the points below the boundary as class 0 (red).
By referring to the figure, what is the ensemble decision boundary obtained by combining by voting rule the three CNNs? Explain your answer.



Exercise 83)

If a data block in a convolutional network has dimension $H \times W \times D = 200 \times 200 \times 128$, and we apply a convolutional filter to it of dimensions $H_F \times W_F \times D = 7 \times 7 \times 128$, what is the dimension of the output data block?

We assume: padding=0; stride=1.

Explain your answer

Exercise 84)

You come up with a CNN classifier. For each layer, calculate the number of weights (with bias) and the size of the associated activation maps (output shape).

- CONV-K-N denotes a convolutional layer with N filters, each them of size $K \times K$, Padding and stride parameters are always 0 and 1 respectively.
- POOL-K indicates a $K \times K$ pooling layer with stride K and padding 0.
- FC-N stands for a fully-connected layer with N neurons.

Layer	Activation map dimensions	Number of weights
INPUT	$128 \times 128 \times 3$	0
CONV-9-32		
POOL-2		
CONV-5-64		
POOL-2		
CONV-5-64		
POOL-2		
FC-3		

Explain your answer

Exercise 85)

Find the Jacobian, at the point (1,2), of:

$$f(x, y) = (x^4 + 3y^2 x \ , \ 5y^2 - 2xy + 1)$$

Explain your answer

Exercise 86)

Let's assume we have a two layer siamese neural network, as defined below:

$$\begin{aligned}
z_1 &= W_1 x^{(i)} + b_1 \\
a_1 &= ReLU(z_1) \\
z_2 &= W_1 x'^{(i)} + b_1 \\
a_2 &= ReLU(z_2) \\
a &= a_1 - a_2 \\
z_3 &= W_2 a + b_2 \\
\hat{y}^{(i)} &= \sigma(z_3) \\
L^{(i)} &= y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \\
J &= -\frac{1}{m} \sum_{i=1}^{m} L^{(i)}
\end{aligned}$$

Note that $x^{(i)}, x'^{(i)}$ represents a pair of single input examples, and are each of shape $D_x \times 1$. Further $y^{(i)}$ is a single output label and is a scalar. There are m examples in our dataset. We use D_{a_1} nodes in our first hidden layers; that is, z_1's and z_2's shape is $D_{a_1} \times 1$. Note that the first two layers share the same weights.

What is $\partial a / \partial z_2$?

Explain your answer

Exercise 87)

Consider the input dataset $X \in \mathbb{R}^{n \times d}$ with n samples of size d, a target vector $y \in \mathbb{R}^n$, a weight vector $w \in \mathbb{R}^d$ and a prediction $\hat{y} = Xw$. The mean squared error (MSE) is defined as the sum over the squared differences between the prediction $\hat{y}_i$ and the true values y_i for each instance:

$$L(\boldsymbol{y},\hat{\boldsymbol{y}}) = \frac{1}{n}(\hat{\boldsymbol{y}} - \boldsymbol{y})^T(\hat{\boldsymbol{y}} - \boldsymbol{y})$$

Find the vector **w** that the minimizes the loss L, i.e, the gradient of the loss, with respect to ..., is zero.

Hint:

$(A \pm B)^T = A^T \pm B^T \qquad (AB)^T = B^T A^T$

Exercise 88)

Imagine that we pass an EPR pair $\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$ through the quantum gate $H \otimes H$; that is, we apply a separate Hadamard gate to each qubit. Recall that $H|0\rangle = |+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ and $H|1\rangle = |-\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$.

Show that

$$(H \otimes H)\left(\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle\right) = \frac{1}{\sqrt{2}}(|+\rangle \otimes |+\rangle) + \frac{1}{\sqrt{2}}(|-\rangle \otimes |-\rangle)$$

Is an Einstein-Podolsky-Rosen (EPR) pair.

Exercise 89)

Given the function $f(x) : \mathbb{R}^5 \mapsto \mathbb{R}$ with

$$f(x) = x_1 x_2 x_3 x_4 x_5,$$

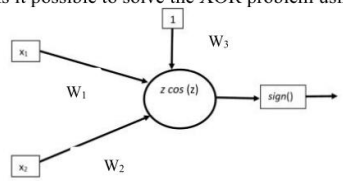
compute the gradient ∇f evaluated at the point $x = [2, 1, 1, 1, 1]^\intercal$.

1. Draw the computational graph.
2. Compute the gradient using forward mode.

Notice that: gradient, non only a partial derivative

Exercise 90)

Is it possible to solve the XOR problem using a single neuron and the activation function $z \cdot \cos(z)$, with the constraint $w_1=w_2=w_3$? Yes or No? Explain your answer.



Exercise 91)

Consider a single neuron with weight vector $\mathbf{w}$ and bias b using an activation function that is monotonically strictly increasing with $g(0) = 0$. The class label assigned to an input $\mathbf{x}$ by this neuron is defined to be $C(\mathbf{x}) = sign(\mathbf{w}^T\mathbf{x} + b)$. If a point $\mathbf{x}^1$ assigned to a particular class is at a distance d_1 from the hyperplane $H = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{w}^T\mathbf{x} + b = 0\}$, then any other point $\mathbf{x}^2$ at a distance $d_2 > d_1$ in the same halfspace as $\mathbf{x}^1$ will be assigned to the same class by this neuron.

Consider the case where $\mathbf{x}^1 \in H_+$ AND $\mathbf{x}^2 \in H_+$

By assumption, $z_1 = \mathbf{w}^T\mathbf{x}^1 + b > 0$ AND $z_2 = \mathbf{w}^T\mathbf{x}^2 + b > 0$.

For definition, the distance $d(\mathbf{x},H)=(\mathbf{w}^t\mathbf{x}+b)/\|w\|$

Remember that

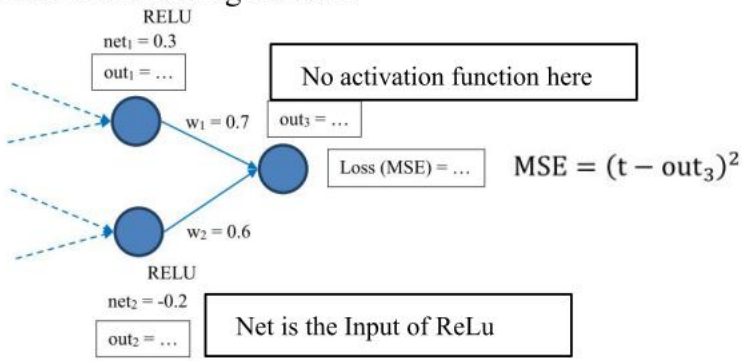
Since g is strictly increasing and $g(0) = 0$ ...

Exercise 92)

Given the following activation function: $x \cdot \cos(x)$.
Plot the values of that activation function in the backward step of the backpropagation algorithm and compare it with the values of ReLu in the backward step, comment the plot. Suppose that $x \epsilon [-4,4]$.

Exercise 93)

Given the following network



Considering a training pattern whose truth value is t=2, run the backpropagation with a learning rate of 0.1, what are the new values of w_1 and w_2?

Exercise 94)

We have three classifiers combined by weighted vote rule, we have a test set of eight patterns that belong to two classes: black and white. The weights, of the weighted vote rule, are w_1, w_2 and w_3.
What is the classification output of the ensemble?

$\psi_1(x)$, $w_1 = .1$

$\psi_2(x)$, $w_2 = .3$

$\psi_3(x)$, $w_3 = .6$

Exercise 95)

Given the following loss, defined:

$$Loss(y,z) = \begin{cases} z - zy + \log(1+e^{-z}) & \text{if } z \geqslant 0 \\ -zy + \log(e^{z}+1) & \text{if } z < 0 \end{cases}$$

make it neater, combining the two equations into a single equation.
Hint: the equation representing the two equations have to contain the term log(1+e^-|z|)

Exercise 96)

Given:

$$J = -\frac{1}{N}\sum_{i=1}^{N} y_i \ln(p_i) + (1-y_i)\ln(1-p_i)$$

and

$$p_i = \frac{1}{1+e^{-z_i}}$$

and

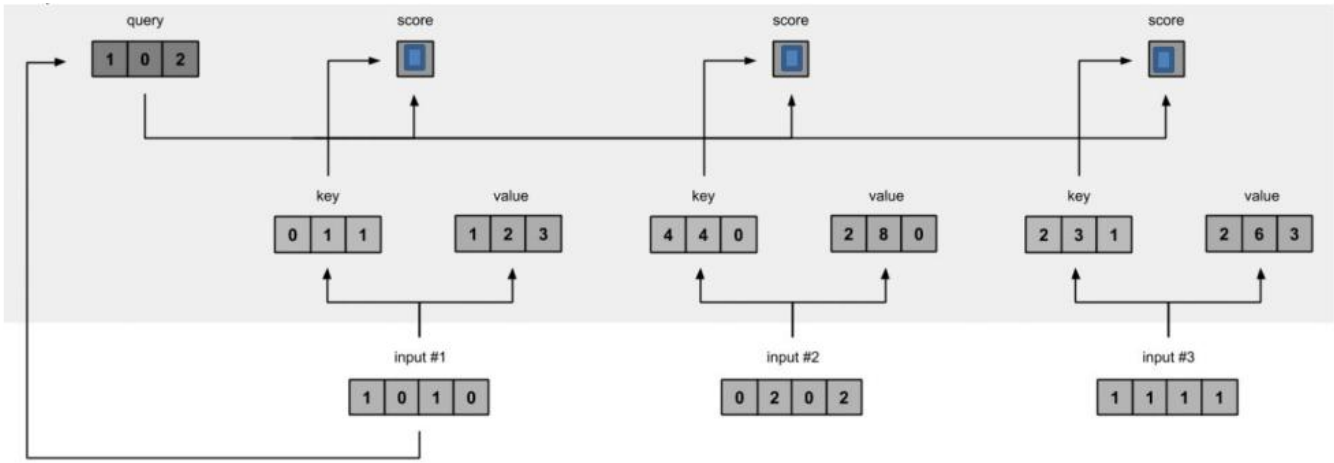
$$z_i = X_i w^T + b$$

with $X = \begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,n} \\ x_{2,1} & x_{2,2} & \dots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N,1} & x_{N,2} & \dots & x_{N,n} \end{bmatrix}$, where n is representing the number of independent variables and N the number of samples,
the weight vector $w = \begin{bmatrix} w_0 & \dots & w_{n-1} \end{bmatrix}$ and a scalar b representing the bias term.

solve $\frac{\partial J}{\partial w_j}$

I want a neat result that follows the following formula:
$a\sum_{b=c}^{d}(e-f)g$
Replace a, b, c, d, e, f and g with the right values.
Hint:
$\frac{d}{dx}(\log(1-x)) = -\frac{1}{1-x}$

Exercise 97)



Calculate the scores, i.e. the three missing values. With score I mean the value obtained by considering only Q and K term in the attention formula without considering the softmax, so: no softmax; no normalization, instead to divide by $\sqrt{d}$ you can divide by 1, no multiplication by V.

Exercise 98)

denote a layer as $l_{\boldsymbol{A}}(\boldsymbol{x}) := \boldsymbol{A}\boldsymbol{x}$, the ReLU-activation (Rectified Linear Unit) – which applies $\max(\boldsymbol{x}_i, 0)$ elementwise – $\mathrm{ReLU}(\boldsymbol{x}) = \max(\boldsymbol{x}, \boldsymbol{0})$. We write

$$N(\boldsymbol{x}) := (l_{\boldsymbol{C}} \circ \mathrm{ReLU} \circ l_{\boldsymbol{B}} \circ \mathrm{ReLU} \circ l_{\boldsymbol{A}})(\boldsymbol{x})$$

where $\circ$ denotes function composition and $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{10\times 10}$ as well as $\boldsymbol{C} \in \mathbb{R}^{3\times 10}$ are matrices. Further, we let $\mathrm{softmax}(\boldsymbol{x})$ denote the softmax-function for $\boldsymbol{x} \in \mathbb{R}^n$,

$$\mathrm{softmax}(\boldsymbol{x})_i = \frac{e^{\boldsymbol{x}_i}}{\sum_{j=1}^{n} e^{\boldsymbol{x}_j}}$$

for $i \in \{1, \dots, n\}$.

Answer the following questions:

1. For the given matrices $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ what is the space of possible inputs to N, i.e. the domain of N?
2. For the given matrices $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ what is the space of possible outputs of N, i.e. the codomain of N? Can the output be seen as a vector of probabilities corresponding to the class probabilities of a categorical distribution?
3. For the given matrices $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ what is the space of possible outputs of $(\mathrm{softmax} \circ N)$, i.e. the codomain of $\mathrm{softmax} \circ N$? Can the output be seen as a vector of probabilities corresponding to the class probabilities of a categorical distribution?

function composition is an operation $\circ$ that takes two functions f and g, and produces a function $h = g \circ f$ such that $h(x) = g(f(x))$.

Exercise 99)

Given:

$g(\boldsymbol{\eta}) = \sum_{i=0}^{n} \max(\boldsymbol{\eta}_i - \tau, 0)$ for some constant τ.

Calculate

$$\frac{\partial}{\partial \boldsymbol{\eta}_i} g(\boldsymbol{\eta})$$

Exercise 100)

Given:

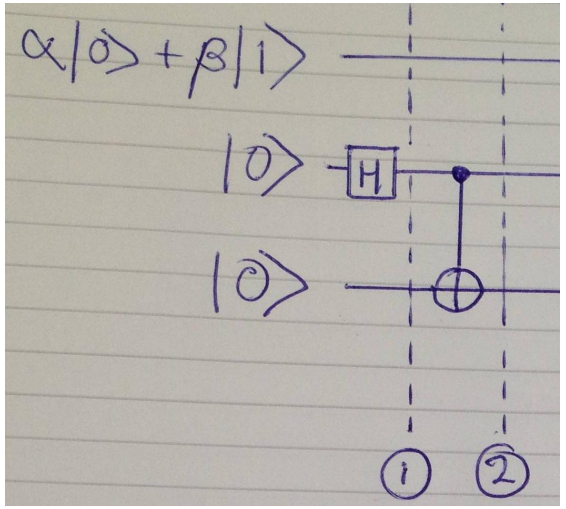
$$|v\rangle = \alpha|0\rangle + \beta|1\rangle \text{ and } |v^{\perp}\rangle = -\beta|0\rangle + \alpha|1\rangle$$

Show that:

$$\frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right) = \frac{1}{\sqrt{2}}\left(|vv\rangle + |v^{\perp}v^{\perp}\rangle\right)$$

Exercise 101)

What is the output of the following quantum circuit?



Exercise 102)

We define the following activation function, which input is a vector:

$h(\boldsymbol{\eta}) = \|\boldsymbol{\eta}\|_{\infty}$

Calculate

$$\frac{\partial}{\partial \boldsymbol{\eta}_i} h(\boldsymbol{\eta})$$

Hint:

$\|x\|_{\infty} := \max\left(|x_1|, \ldots, |x_n|\right).$

Exercise 103)

You are tasked with training a neural network to approximate the value of a definite integral of a given function. The function to be integrated is:

$$f(x) = \sin(x^2) + e^{-x}\cos(2x)$$

Your goal is to create a neural network that can estimate the definite integral of $f(x)$ over a specified interval $[a, b]$

How would you create/train such a network?

Exercise 104)

Given the following operation

$$Y = m * a(WX),$$

where, $*$ symbolizes the element-wise multiplication and m denotes a n-dimension binary vector.

where, a symbolizes the activation function, and W denotes an $n \times D$ weight matrix.

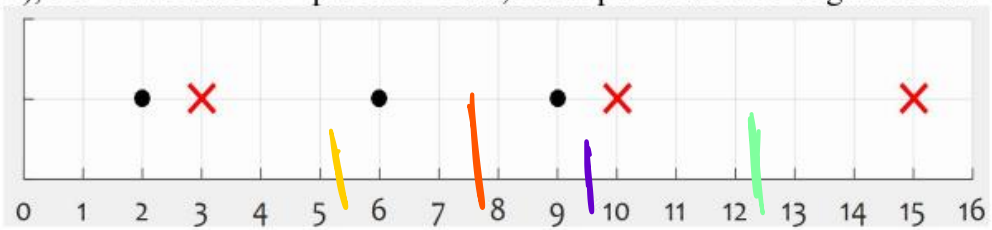
$$m_i \sim Bernoulli(1-p), i \in \{1, 2, \ldots n\}$$

What kind of layer (among those explained in the handouts) does Y implement? Justify the answer.

the Bernoulli distribution is the discrete probability distribution of a random variable which takes the value 1 with probability p and the value 0 with probability (1-p)

Exercise 105)

Build a ROC curve (y-axis: sensitivity; x-axis: (1-specificity)) for the following CNN output scores (x-axis of the following figure, e.g. score of the last red cross is 15), red crosses are the 'positive' class, black points are the 'negative class':



ROC sens = recall, 1 - spec

$t < 2$ $Sens = \frac{tp}{tp+fn} = 1$

$Spec = \frac{tn}{tn+fp} = 0$

$2 < t < 3$ sens = 1, spec = 2/3

$3 < t < 6$ sens = 2/3, spec = 1/3

$6 < t < 9$ sens = 2/3, spec = 2/3

$9 < t < 10$ sens = 2/3, spec = 1

$10 < t < 15$

Exercise 106)

The Bloch Sphere, given:

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle$$

find the angles theta and phi corresponding to the "1 state" ("1 state" has a 100% probability of measuring "1", and 0% probability of measuring "0"). Then plot the "1 state" as a vector in the Bloch sphere.
Motivate your answer.