VIR 2019 **Name:** ____________________

Exam test

Variant: A Points ____________________

1. Consider the following network.

$$y = \sin(\mathbf{w}^\top \mathbf{x}) - b \tag{1}$$

- Draw the computational graph of the forward pass of this network. Note that **every** operator is a node with a given arity and output. For example, the + operator is a node which has two input arguments and a single output argument, etc...

- Consider an input $\mathbf{x} = [2, 1]\mathbf{w} = [\frac{\pi}{2}, \pi], b = 0$ and label $l = 2$.
  - Compute the forward pass of the network.
  - Use an L_2 loss (Mean square error) to compute the loss value between the forward prediction y and label l. Add this loss to the computation graph.
  - Use the chain rule to compute the gradient $\frac{\partial L(y,l)}{\partial \mathbf{w}}$ and estimate an update of parameters $\mathbf{w}$ with learning rate $\alpha = 0.5$.

2. You are given an input volume X of dimension $[batch \times width \times height \times depth \times channel] = [3 \times 6 \times 6 \times 7 \times 1]$
   Consider a 3D convolutional filter F of size $[width \times height \times depth] = [3 \times 3 \times 3]$

   - What is the size of padding, which ensures the same spatial resolution of the output feature map? Note: A padding size of 1 for a $[30 \times 30]$ image gives it a resulting size of $[32 \times 32]$, in other words, zeros are added on both sides.

   - Calculate the total memory in bytes of the learnable parameters of the filter, assuming that each weight is a half-precision float (FP16) which takes up 2 bytes each

   - Calculate the amount of operations performed by a single applciation of the filter (just one stamp). Each addition or multiplication counts as a single operation.

     For example: $\alpha x + \beta y + c$ amounts to 2 multiplication and 3 addition operations, totaling 5 operations.

   - Considering the entire input dimensions of X, given a stride of 1 in all dimensions, no padding and only valid convolutions, calculate the amount of filter applications (”stamps”) that you have to perform to process the entire input.

3. Activation function maps single input $\mathbf{x}$ on a single output value $\mathbf{y}$.

   - Define a Leaky Rectified Linear Unit $\mathbf{lrelu}(\mathbf{x})$ activation function in pseudocode, with $\alpha = 0.1$. The function has a single argument $\mathbf{x}$ and output $\mathbf{y} = \mathbf{lrelu}(\mathbf{x})$.

   - Define the gradient of the $\mathbf{lrelu}(\mathbf{x})$ activation function in pseudocode. The function has a single argument $\mathbf{x}$ and outputs $\frac{\partial lrelu(x)}{\partial \mathbf{x}}$. Hint: Break up the function into two separate cases (if-else).

4. You are given batch of two one-dimensional training examples $\mathbf{B} = \{x_1 = 2, x_2 = 4\}$. The network consists of:

   - Batch-norm layer $BN_{\gamma,\beta}(\mathbf{B})$ with two learnable parameters $\gamma = 6$, $\beta = -1$.
   - L2-norm layer $\|\mathbf{y}\|_2^2 = \sum_i y_i^2$

   Compute gradient of $\|BN_{\gamma,\beta}(\mathbf{B})\|_2^2$ with respect to the parameter γ.
   Hint: Output of the batch-norm layer for this batch is two-dimensional.

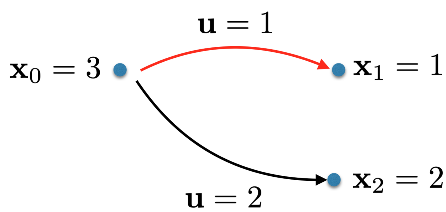
5. You are given batch of two one-dimensional training examples $x_1 = 2, x_2 = 4$. The batch-norm layer has two learnable parameters $\gamma = 6$, $\beta = -1$. Compute jacobian of the batch-norm layer with respect to the parameter γ.
   Hint: Output of the batch-norm layer for this batch is two-dimensional.

6. You are given batch of three one-dimensional training examples $x_1 = 5, x_2 = 2, x_3 = 1$. The batch-norm layer has two learnable parameters $\gamma = 6$, $\beta = -1$. Compute jacobian of the batch-norm layer with respect to the parameter β.
   Hint: Output of the batch-norm layer for this batch is three-dimensional (you do not have to compute full feed-forward pass).

7. Consider MDP consisting of three states $\mathbf{x}_0 = 3, \mathbf{x}_1 = 1, \mathbf{x}_2 = 2$ and two types of actions $\mathbf{u} = 1$ and $\mathbf{u} = 2$, see image below. Agent selects action $\mathbf{u}$ in the state $\mathbf{x}$ according the following stochastic policy

$$\pi_\theta(\mathbf{u}|\mathbf{x}) = \begin{cases} \sigma(\theta\mathbf{x}) & \text{if } \mathbf{u} = 1 \\ 1 - \sigma(\theta\mathbf{x}) & \text{if } \mathbf{u} = 2 \end{cases}$$

with scalar parameter $\theta = 2$. This policy maps one-dimensional state $\mathbf{x}$ on the probability distribution of two possible actions $\mathbf{u} = 1$ or $\mathbf{u} = 2$.



Consider trajectory-reward function defined as follows:

$$r(\tau) = \sum_{\mathbf{x}_i \in \tau} (\mathbf{x}_i - 1)^2$$

Given training trajectory $\tau = [(\mathbf{x}_0 = 3), (\mathbf{u} = 1), (\mathbf{x}_1 = 1)]$, which consists of the single transition (outlined by red color), estimate REINFORCE policy gradient.

- Policy gradient

$$\left.\frac{\partial \log \pi_\theta(\mathbf{u}|\mathbf{x})}{\partial \theta}\right|_{\substack{\mathbf{x} = \mathbf{x}_0 \\ \mathbf{u} = \mathbf{u}_0}} \cdot r(\tau) =$$

- Updated weights with learning rate $\alpha = 1$

$$\theta^{\text{updated}} =$$

8. You are given network (without loss layer) which consists of the convolutional layer and the max-pooling layer. The structure is defined as follows:

$$f(\mathbf{x}, \mathbf{w}) = \text{maxp}\Big(\text{conv}(\mathbf{x},\ \mathbf{w}),\ 1 \times 2\Big)$$

Given the input feature map (image) $\mathbf{x}$ and convolutional kernel $\mathbf{w}$:

$$\mathbf{x} = \boxed{2 \mid 1 \mid 2} \qquad \mathbf{w} = \boxed{1 \mid 0}$$

estimate gradient $\frac{\partial f(\mathbf{x},\mathbf{w})}{\partial \mathbf{w}}$ of its output wrt kernel $\mathbf{w}$.
Hint: Draw computational graph, perform feed-forward pass, compute local gradients for the max-pooling layer and the convolutional layer and substitute edge-values computed in the feed-forward pass.

9. You are given the following network

$$f(\mathbf{x}, \mathbf{w}) = \frac{1}{2}\|\mathbf{w}^\top \mathbf{x}\|_2^2 = \frac{1}{2}((w_1 x_1)^2 + (w_2 x_2)^2)$$

and a single training example $\mathbf{x} = [\sqrt{3}, 1]^\top$. Consider Stochastic Gradient Descend algorithm, which updates the weights as follows:

$$\mathbf{w}^k = \mathbf{w}^{k-1} - \alpha \left.\frac{\partial f^\top(\mathbf{w})}{\partial \mathbf{w}}\right|_{\mathbf{w}=\mathbf{w}^{k-1}},$$

where α denotes its learning rate.

- For which α the SGD converges (at least slowly) in both dimensions? **Hint:** Derive formula for weight values in k-th iteration

$$w_1^k = \rho_1(\alpha)^k w_1^0$$

$$w_2^k = \rho_2(\alpha)^k w_2^0,$$

where $\rho_i(\alpha)$ denotes convergence rate in dimension $i = 1, 2$. Find α for which both formulas converges to zero (i.e. the global optimum).

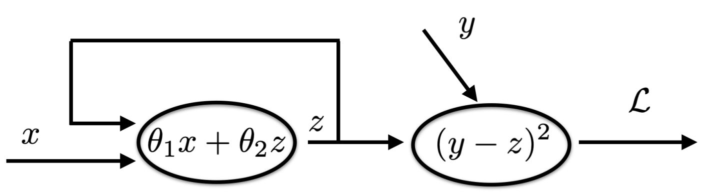
- What is the best learning rate α^*, which guarantees the fastest convergence rate for arbitrary weight initialization $\mathbf{w}^0$ and this particular training example.
  Hint: The smaller the $|\rho_i(\alpha)|$, the faster the convergence. Choose alpha, which minimize maximal convergence rate:

$$\alpha^* = \arg\min_{\alpha} \max\{|\rho_1(\alpha)|, |\rho_2(\alpha)|\}$$

10. Consider linear recurrent neural network with L2 loss depicted on the image below. The network is initialized with parameters $\theta_1 = 1, \theta_2 = 0, z_0 = 0$. You are given the following training sequence:

time=1	time=2
$x_1 = 0$	$x_2 = 1$
$y_1 = 1$	$y_2 = 3$

Estimate gradient of the overall loss (computed over all available outputs y_i for both available times $i = 1, 2$) with respect to θ_1.



Hint: Unroll the network in time, to obtain a usual feedforward network with two loss nodes. Do the backpropagation as usual.