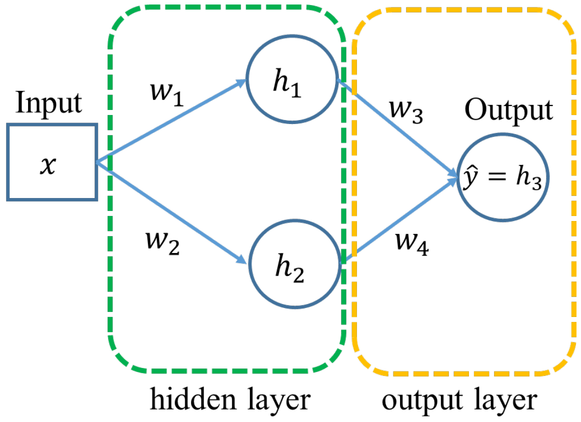
Homework Assignment 4

Part-1: Basic Concepts

1. Backpropagation in A Neural Network (7 points)



$$h_1 = f_1(w_1 x + b_1)$$

$$h_2 = f_2(w_2 x + b_2)$$

$$\hat{y} = f_3(w_3 h_1 + w_4 h_2 + b_3)$$

$$f_n' = \frac{\partial f_n(v)}{\partial v}, \text{ n} = 1,2,3$$

$x, w_1, w_2, w_3, w_4, b_1, b_2, b_3, h_1, h_2, h_3$ are scalars

Compute the derivatives of the loss L with respect to parameters and input, assuming $\frac{\partial L}{\partial h_3}$ is known.

Example:

This is correct: $\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial h_3}\frac{\partial h_3}{\partial h_1}\frac{\partial h_1}{\partial w_1} = \frac{\partial L}{\partial h_3} f_3' w_3 f_1' x$ you get 1 point

This partial solution is not acceptable: $\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial h_3}\frac{\partial h_3}{\partial h_1}\frac{\partial h_1}{\partial w_1}$, you get 0 point

$\frac{\partial L}{\partial w_2}$

$\frac{\partial L}{\partial w_3}$

$\frac{\partial L}{\partial w_4}$

$\frac{\partial L}{\partial b_1}$

$\frac{\partial L}{\partial b_2}$

$\frac{\partial L}{\partial b_3}$

$\frac{\partial L}{\partial x}$

2. Computational Graph (20 points)

$h = 2x + 1$

$z = x^2 + h$

$y = \dfrac{1}{1 + e^{-h}}$

(1) Draw the computational graph based on the above three equations (1 point)

(2) What is $\frac{\partial y}{\partial z}$ from the graph? (19 points)

3. Target (output) Normalization for a Neural Network

Usually, we need to apply normalization/standardization to the inputs for classification and regression tasks, so that the input will be in the range of 0 to 1, or -1 to +1. For example, if the input is an image, then every pixel value is divided by 255, so that the pixel values of the normalized image are in the range of 0 to 1. Input normalization facilitates the convergence of training algorithms.

We may also need to apply normalization to the output. Assume the input is an image of a person, the output vector has two components, $\hat{y}_{(1)}$ and $\hat{y}_{(2)}$: $\hat{y}_{(1)}$ is the monthly income (in the range of 0 to 10,000), and $\hat{y}_{(2)}$ is the age (in the range of 0 to 100). The MSE loss for a single data sample is

$$L = (\hat{y}_{(1)} - y_{(1)})^2 + (\hat{y}_{(2)} - y_{(2)})^2$$

where $y_{(1)}$ and $y_{(2)}$ are ground truth values of an input data sample.

Question: is output (i.e., the output target $y_{(1)}, y_{(2)}$) normalization necessary for this task? Why?

If it is necessary, what normalization can be applied?

4. Activation Functions for Regression

Neural networks can be used for regression. To model nonlinear input-output relationship, a neural network needs nonlinear activation functions in the hidden layers. Usually, the output layer does not need nonlinear activation functions. However, sometimes, there are requirements for outputs. For example, if the output is the sale price of a house, then the output should be nonnegative.
Assume $\boldsymbol{z}$ is the scalar output of a network, and the network does not have nonlinear activation function in the output layer. Now, there is some requirement for output, and you decide to add a nonlinear activation function.

You design nonlinear activation functions for three different requirements:

(1) the final output y should be nonnegative ($y \geq 0$), then what is the activation function $y = f(z)$?

(2) the final output y should be nonpositive ($y \leq 0$), then what is the activation function $y = f(z)$?

(3) the final output y should be $a \leq y \leq b$, then what is the activation function $y = f(z)$?

You may use a combination of the basic activation functions that you can find in the lecture notes or the documents of Keras and Pytorch. Do NOT use if statements, for example:

```
def activation(z):
   if z < a:     return a
   elif z > b:  return b
   else: return z
```

This is not an acceptable answer.

5. Normalization Inside a Neural Network

To facilitate the convergence of training a deep neural network, it is necessary to normalize the output or input of each layer of a neural network.

Read the paper https://arxiv.org/abs/1803.08494 and answer the questions:

(1) Batch Normalization will be highly unstable if batch_size is very small. Why?

(2) Why is Layer Normalization independent of batch_size?

6. Skip Connections in a Deep Neural Network

Read: https://arxiv.org/abs/1512.03385 and answer the question

Why are skip/residual connections useful to build a deep network?

7. Randomness of a Neural Network

We can train the same network three times on the same dataset to get model-1, model-2, and model-3. However, the performance of these models on the test set could be different.

What is the cause of this randomness? If you write a technical paper, which model should you choose to report on the paper? The worst model? or the best model? or all of the three models?

8. ReLU and Piecewise Linear

Prove that an MLP with ReLU activations is a piecewise linear function of the input

Note: this is not the answer "it is because ReLU is piecewise linear"

Part-2 Programming

Programming tasks: H4P2T1 and H4P2T2 using the template ECG_Keras_template.ipynb or ECG_Pytorch_template.ipynb. You may choose to use Keras or Pytorch.

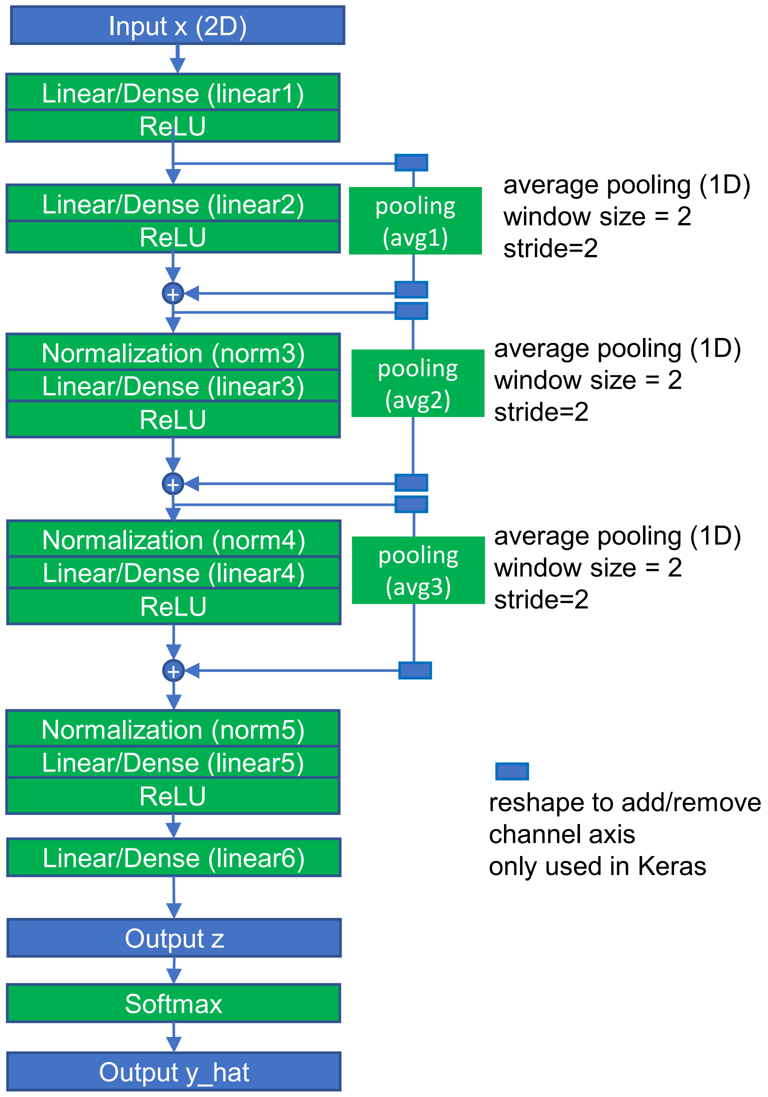
Grading

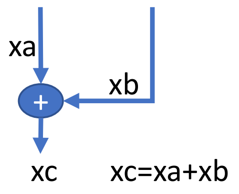
The number of points for each question/task

	Undergrad Student	Graduate Student
1. Backpropagation	7	7
2. Computational Graph	20	20
3. Output Target Normalization	2	2
4. Activations for Regression	extra 5 points	6
5. Normalization inside Network	2	2
6. Skip connection	2	2
7. Randomness	2	1
8. ReLU and piecewise linear	N.A.	extra 5 points
H4P2T1 (MLP)	30	30
H4P2T2 (CNN)	35	30

Read the instructions about H4P2T1 and H4P2T2 on the following pages.

In H4P2T1, you will implement an MLP with residual connections for ECG signal classification according to the diagram below. You will lose points if your network deviates from the diagram: one deviation costs 10 points. The following actions are deviations: miss a connection, add an extra connection, miss a layer, add an extra layer, or use different parameters (not those defined in the diagram) for the pooling layers. Note: Softmax is optional in Pytorch, so missing Softmax in Pytorch is not a deviation.





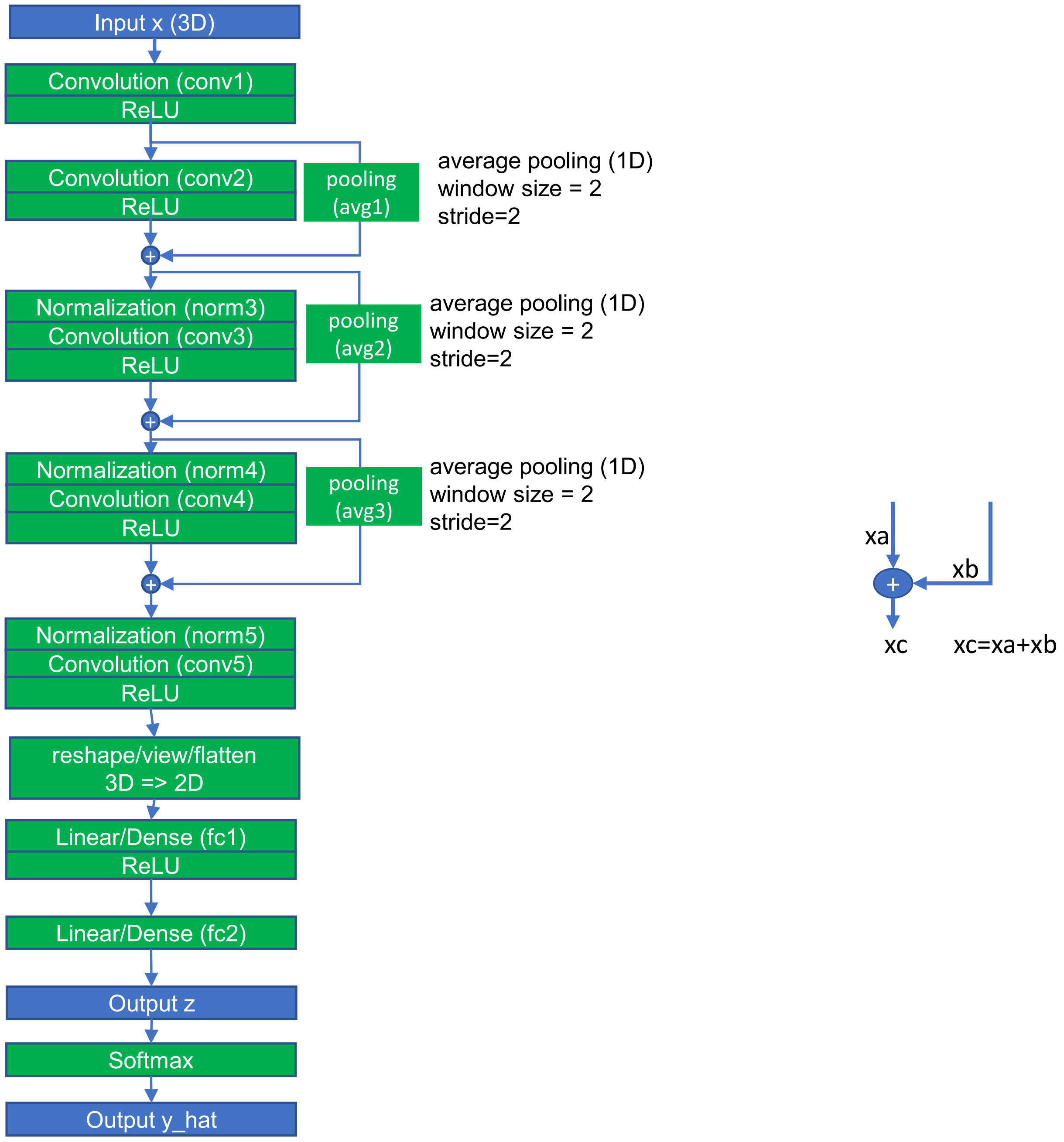
For these linear/dense layers: you are free to choose the internal parameters (e.g., the number of units in a layer).

For these normalization layers: you choose a feasible normalization method from GroupNorm, InstanceNorm, BatchNorm, and LayerNorm (see Q5).

For these average pooling layers: the pooling window size (named pool_size in Keras, kernel_size in Pytorch) is fixed to 2, and the stride is fixed to 2 (it is named strides in Keras). You may add padding if necessary.

The accuracy on the test set is about 89%. You will get zero score if the test accuracy < 85%

In H4P2T2, you will implement a CNN with residual connections for ECG signal classification according to the diagram below. You will lose points if your network deviates from the diagram: one deviation costs 10 points. The following actions are deviations: miss a connection, add an extra connection, miss a layer, add an extra layer, or use different parameters (not those defined in the diagram) for the pooling layers. Note: Softmax is optional in Pytorch, so missing Softmax in Pytorch is not a deviation.



For these convolution and linear/dense layers: you are free to choose the internal parameters (e.g., the number of kernels/filters, the number of input channels, the number of output channels, stride, padding, etc).

For these normalization layers: you choose a feasible normalization method from GroupNorm, InstanceNorm, BatchNorm, and LayerNorm (see Q5).

For these average pooling layers: the pooling window size (named pool_size in Keras, kernel_size in Pytorch) is fixed to 2, and the stride is fixed to 2 (it is named strides in Keras). You may add padding if necessary.

The accuracy on the test set is about 90%. You will get zero score if the test accuracy < 85%