MASTER OF SCIENCE
IN ENGINEERING

Teachers: J. Hennebert, M. Melchior
Assistants: C. Gisler, M. Willi, Y.-I. Beffa

MSE

TSM Deep Learning

Practical Work 06 – 01/04/2021
Deep Learning Frameworks

Objectives

The main objective of this PW is to understand the overall functioning of TensorFlow 2, a computational graph framework used for machine learning and deep learning. More specifically, we want to experiment with the definition of a graph composing a multi-layer perceptron (MLP).

Submission

— **Deadline** : Wednesday 14th of April, 12 :00 (noon)

— **Format** :

— Zip with report and iPython notebook.

Exercise 1   Computational graph exercise - pen & paper

The logistic regression (LR) is a well known 2-class classification algorithm. LR is actually similar to the generalised perceptron - cf week 2. A LR outputs values between 0 and 1.0 with the following equation :

$$\hat{y} = h_\theta(\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^T\mathbf{x} + b)} \tag{1}$$

where the parameters θ of the model are defined with a weight vector $\mathbf{w}$ and a scalar value b. Assuming an input space with 2 dimensions, the LR function becomes :

$$h_\theta(x_1, x_2) = \frac{1}{1 + \exp(-(w_1x_1 + w_2x_2 + b))} \tag{2}$$

a) Revise slides 6–19 from the class and propose a computational graph with an input space of 2 dimensions (Eq. 2). Use operators as granular as you can. You should have about 7 nodes in your graph.
b) On the graph, illustrate the forward pass using numerical values.
c) Illustrate the back-propagation of a gradient value of 1.0 at the output of the graph using the numerical values of b).

Exercise 2   TensorFlow 2 installation

Install TensorFlow 2 on your machine. Follow the instructions given in slide 24 and under https://www.tensorflow.org/install/pip. We recommend that you use a virtual environment to isolate your Python packages in a directory.

As alternative you can also :

— Use notebooks in Google Colaboratory.
— Use Docker with an image including Python, numpy, Jupyter, TensorFlow 2, etc.

Exercise 3   MLP with TensorFlow 2

The Figure 1 below illustrates a Multi-Layer Perceptron. The network takes inputs of dimension D, has H hidden neurons and K output neurons. In this case we assume that all activation functions of the neurons are ReLU for the hidden layer and sigmoids for the output layer. The objective of the exercise is to implement computational graphs in TensorFlow 2 to train and use such an architecture. The constraints we put ourselves is to use "low-level" functions of TensorFlow, i.e. we will not use high-level Keras functions to compose layers and to train the parameters.

a) Revise slides 31–41 from the class on TensorFlow and make sure you understand the principles and the different steps that were described.
b) Use the iPython notebook provided on Moodle and complete the missing parts of the code in order to perform a training and testing phase. We assume that the loss function used will be the Mean Squared Error (MSE) loss function.
c) Make sure that the network is converging and compute the error rate on the test set. You should observe an error rate at about 8-12%.

Optional. Implement a simple learning rate schedule strategy, i.e. reducing the α along the training (piecewise).

Exercise 4   MLP with TensorFlow 2 and Keras

Analyse the code provided in slides 51-53 and implement the MLP of Exercise 3 using Keras. Note : let Keras handle the batch size in the `model.fit()` method.
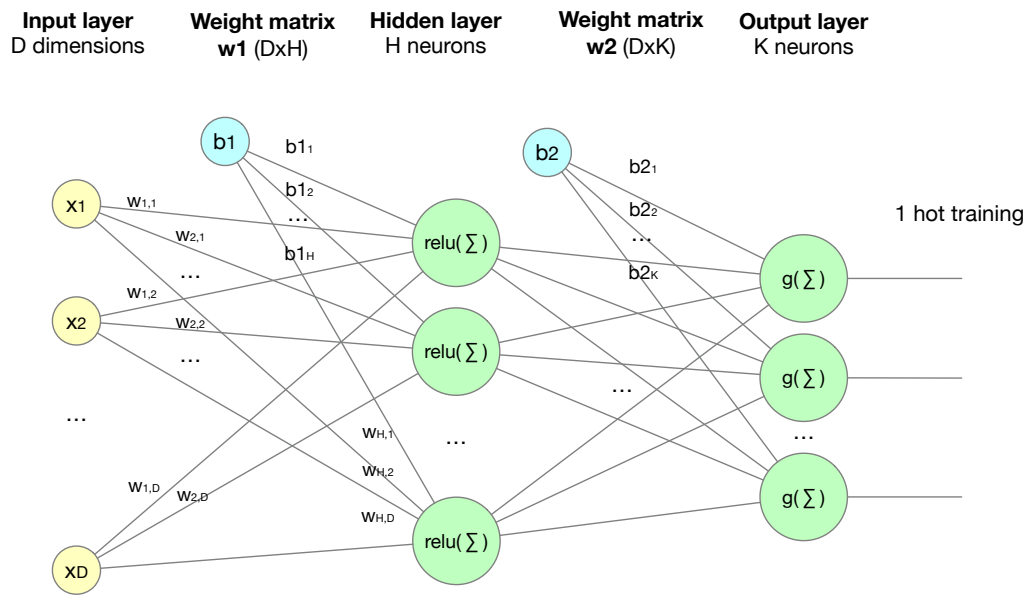
FIGURE 1 – Multi-layer perceptron with one hidden layer.

a) Provide the code in new cells at the end of your notebook of Exercise 3.

b) Compare the performances with your implementation of Exercise 3 (cpu, accuracy) and comment.

Optional. Redo this exercise using a softmax activation at the output layer and with a cross-entropy loss function. Compare the performances with the other implementations.

Exercise 5   Optional : Review Questions

a) In computational graphs, we say that the "+" gate is a *gradient distributor*. Why? How can we qualify the multiplication gate and the max gate?

b) Give 4 advantages of using computational graphs for learning strategies.

c) What are the expected advantages / disadvantages of a static graph strategy (TensorFlow) versus a dynamic graph stategy (Pytorch)?

d) What is the use of the `@tf.function` decorator for functions in TensorFlow 2.0?

e) What does the function `tf.gradients(ys, xs)`? Describe precisely the output.

f) Explain the difference between the Keras sequential and functional API.