Master ST Computer Science | **University of Bordeaux**
Faculty S&T
Deep Learning in Computer Vision | Academic year 2022 – 2023

Lab 3
(2h00 hours, evaluated)

Neural Networks
Implementing a Neural Network with Keras & PyTorch

Introduction

In this lab, we will see how to build a neural network with the Keras & PyTorch frameworks in Python. As usual, we give an incomplete python program. Some instructions will be used as they are and you do not need to change them. We will comment some of them but we do not need to understand them all. We will use the dataset MNIST, one of the most used datasets in machine learning research. This dataset consists of 70000 images of handwritten digits (of size 28x28).

So first, you need to download the files lab3_skeleton*.py and lab3_3_skeleton*.py available at:
/net/ens/DeepLearning/DLCV2021/lab3_skeleton_keras.py
/net/ens/DeepLearning/DLCV2021/lab3_3_skeleton_keras.py
/net/ens/DeepLearning/DLCV2021/lab3_skeleton_pytorch.py
/net/ens/DeepLearning/DLCV2021/lab3_3_skeleton_pytorch.py
or http://dept-info.labri.fr/~mansenca/DLCV2021/lab3_skeleton_keras.py
http://dept-info.labri.fr/~mansenca/DLCV2021/lab3_3_skeleton_keras.py
http://dept-info.labri.fr/~mansenca/DLCV2021/lab3_skeleton_pytorch.py
http://dept-info.labri.fr/~mansenca/DLCV2021/lab3_3_skeleton_pytorch.py

You will need to activate the work environment with:
source /net/ens/DeepLearning/python3/tensorflow2/bin/activate

Basics:
Recall that a neural network (NN for short) consists of a set of layers disposed linearly, and each layer is a set of (artificial) neurons. The model is simplified so that signals can only circulate from the bottom layer to the top layer. Each neuron in the k-th layer of the neural network is connected to all the neurons in the (k − 1)-th layer, and the neurons in a given layer are all independent from each other.

A neural network consists of the following components

- An **input layer**, **x**
- An arbitrary amount of **hidden layers**
- An **output layer**, **y**
- A set of **weights** and **biases** between each layer, **w and b**
- A choice of **activation function** for each hidden layer, **σ**.

1. Single Neuron:
As a first exercise, we will consider a single neuron with 28x28=784 inputs and a single sigmoid unit generating the output. Our neuron will simply learn to distinguish between the digit 0 and the other digits (in the MNIST dataset).

1. Complete the file with the model definition
2. Train your network and measure its accuracy.
3. Try various batch sizes.

2. A Neural Network with One Hidden Layer:
In this section, we will add a hidden layer with 64 units.

1. Modify your file accordingly to implement this NN.

2. Train your network and measure its accuracy.
3. Try various number of neurons on hidden layer
4. Try various activation functions

3. Multiclass Neural Network:

Now, we will adapt our previous network to recognize all the 10 digits.

1. Modify your file accordingly to implement this NN.
2. Train your network and measure its accuracy.
3. Try various optimizers
4. Search the best network architecture that maximize accuracy

4. Previous Networks with pytorch:

Exercises 1, 2 and 3 but coded with pytorch

5. [Optional] Try on FashionMNIST dataset.

You must send an archive with your code. You must submit two files for each question: one coded with keras and one coded with pytorch. You can work by group of two if you want.
This archive must be named LASTNAME_firstname_Lab3_DLCV.tar.bz2 or LASTNAME1_firstname1_LASTNAME2_firstname2_Lab3_DLCV.tar.bz2 and sent to: boris.mansencal@labri.fr with DLCV, your group, name and lab number in the e-mail subject: for example "DLCV-mansencal: DOE_John_Lab3".
In you report, for each method, please try different hyper-parameters, note the loss value for 40 epochs, plot the loss over epochs, compute the accuracies, make a summary table with the obtained accuracies and discuss your results. Document the differences between the two versions of your code if any.