Master ST Computer Science | **University of Bordeaux**
Faculty S&T
Deep Learning in Computer Vision | Academic year 2019 – 2020

Lab 2
(2h40 hours, evaluated)

Neural Networks
Implementing a Neural Network from scratch

Introduction

In this lab, we will see how to build a neural network from scratch in Python with numpy. 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 MNIST dataset, one of the most used datasets in machine learning research. This dataset consists of 70000 grayscale images of handwritten digits (of size 28x28).

So first, you need to download the file lab2_skeleton.py available at:
http://dept-info.labri.fr/~mansenca/DLCV2019/lab2_skeleton.py

You will need to activate the work environment with:
source /net/ens/DeepLearning/tensorflow/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 definition of the sigmoid function
2. Give the mathematical expression of y in means of x, W, b, and σ.
3. Give the mathematical expression of the loss function L and its implementation. We will use the cross-entropy.

For backpropagation, we will need to know how the loss function changes with respect to each component wi of w.

4. Give the expression of the derivative of L with respect to a component w.
5. Complete the file with the implementation of the necessary elements to train your model and test it

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. Try your network and measure its accuracy.

3. Multiclass Neural Network:

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

1. Explain, with a figure, the architecture of the new network
2. In your file, define the new multiclass loss function
3. Train and test your network.

You must send an archive with your code.
This archive must be named LASTNAME_firstname_Lab2_DLCV.tar.bz2 and send to : boris.mansencal@labri.fr with DLCV in the e-mail subject.
In the e-mail, please note the loss value you get for 500 epochs for each method.