

Lab 4
(2h00, evaluated)

Neural Networks
Implementing a Convolutional Neural Network with Keras & Pytorch

Introduction

In this lab, we will see how to build a convolutional 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 datasets MNIST and CIFAR10.

So first, you need to download the file lab4_skeleton.py available at:

/net/ens/DeepLearning/DLCV2021/lab4_skeleton_keras.py
/net/ens/DeepLearning/DLCV2021/lab4_2_skeleton_keras.py
/net/ens/DeepLearning/DLCV2021/lab4_skeleton_pytorch.py
/net/ens/DeepLearning/DLCV2021/lab4_2_skeleton_pytorch.py
or http://dept-info.labri.fr/~mansenca/DLCV2021/lab4_skeleton_keras.py
http://dept-info.labri.fr/~mansenca/DLCV2021/lab4_2_skeleton_keras.py
http://dept-info.labri.fr/~mansenca/DLCV2021/lab4_skeleton_pytorch.py
http://dept-info.labri.fr/~mansenca/DLCV2021/lab4_2_skeleton_pytorch.py

You will need to activate the work environment with:

source /net/ens/DeepLearning/python3/tensorflow2/bin/activate

1. Convolutional Neural Network on MNIST dataset

Program your first convolutional neural network to recognize the 10 digits.

1. Complete the file to define a model that use a convolution layer.
2. Train your network and measure its accuracy.
3. Compare with the results you got with your best model on lab3.3
4. Try various network architectures and hyper-parameters to improve accuracy

2. Convolutional Neural Network on CIFAR10 dataset

CIFAR10 dataset consists of color images in 10 classes (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck). Program a convolutional neural network to recognize the 10 classes.

5. Explore the data set and display some images
6. Complete the file to define a model that use a convolution layer.
7. Train your network and measure its accuracy.
8. Compare with the results you get adapting your best model of lab3.3
9. Try various network architectures and hyper-parameters to improve accuracy

3. Data augmentation

Try to apply data augmentation to improve your results on CIFAR10.

4. Transfer learning / fine-tuning on CIFAR10 dataset

Get a previously trained network (ResNet trained on ImageNet for example) and try to apply transfer learning/fine-tuning to improve your results on CIFAR10.

You must send an archive with your code and report. 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](#)”.

The report should contain a description of the network architectures and hyper-parameters that you tried, and for each try: the plot of the loss function of the corresponding training, the final accuracy on the test set, the confusion matrix and the 10 worst classified images (with, for each image, the category and probability that you get and the actual category). Document the differences between the two versions of your code if any. Document any new concept that was not explained during the lab and that your code may use.