APAI Lab02: DNN Definition and Training

Davide Nadalini, Ilenia Carboni, Alberto Dequino, Luca Bompani, Lorenzo Lamberti, Francesco Conti.
(University of Bologna)

d.nadalini@unibo.it

In this Hands-on session:

A first-time user of Pytorch framework will learn how to :

- define a Neural Network in PyTorch;
- train a NN;
- test a NN.

LAB2 DEADLINE:

17/10/2025

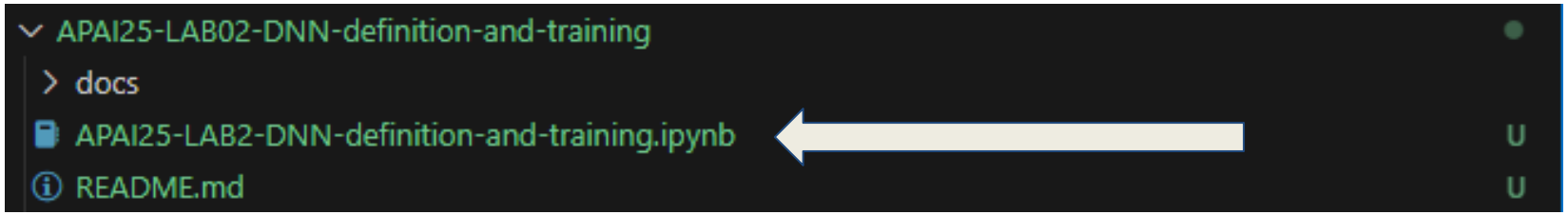
Tasks:

1. PyTorch definition of a NN model;
2. Count network's parameters and MAC operations;
3. Data loader for Fashion-MNIST
4. Code for testing a neural network on Fashion MNIST dataset;
5. Code for training a neural network on Fashion MNIST;
6. Save and load model's trained weights;

All the details about the tasks are explained in the pdf document attached.

How to deliver the assignment

You will **ONLY deliver the Jupyter Notebook** of your experiments, **rename it as** LAB<number_of_the_lesson>_APAI_<your_name>.ipynb



Such .ipynb file **SHOULD BE ALREADY PRE-RUN**

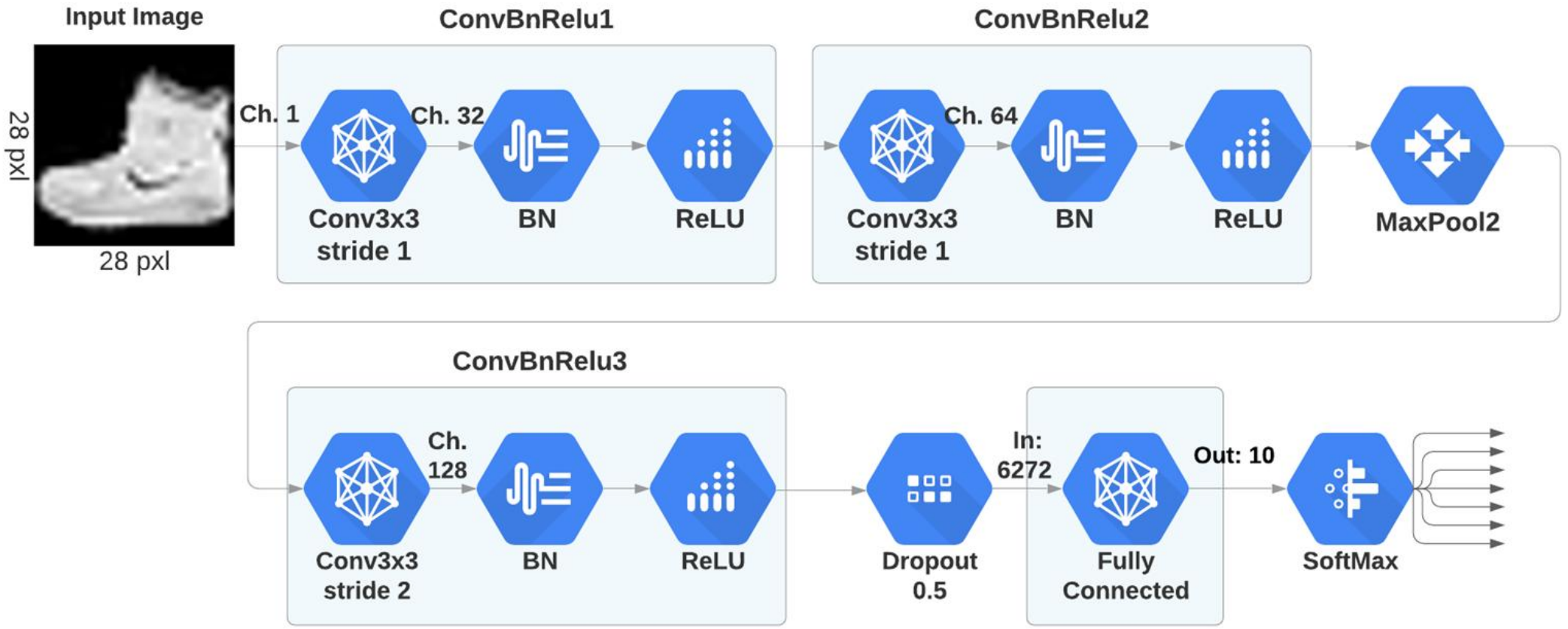
```
#basic
import os
from os.path import join
import numpy as np
from tqdm import tqdm
import time

#plotting
import matplotlib.pyplot as plt

#torch
import torch; print('\nPyTorch version in use:',
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
from torch.utils.data import DataLoader

#torchvision
import torchvision
from torchvision import transforms, datasets

# others
from copy import deepcopy
from tqdm.autonotebook import tqdm
from sklearn.metrics import precision_score, rec
import inspect


PyTorch version in use: 2.4.1+cu121
cuda avail:  False
```

Pre-run boxes look like this!

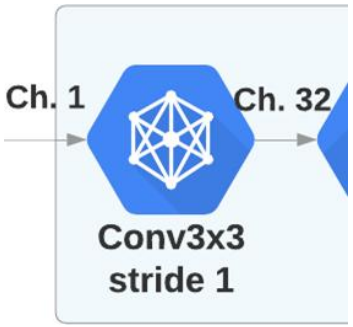
ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

A BIT OF THEORY

LAB1 - CNN


Input Image
28 pxl
28 pxl
Ch. 1
ConvBnRelu1
Conv3x3
stride 1
Ch. 32
BN
ReLU
ConvBnRelu2
Conv3x3
stride 1
Ch. 64
BN
ReLU
MaxPool2
ConvBnRelu3
Conv3x3
stride 2
Ch.
128
BN
ReLU
Dropout
0.5
In:
6272
Fully
Connected
Out: 10
SoftMax

LAB1 - CNN



2D Convolution (Visualizer: https://ezyang.github.io/convolution-visualizer/)
Kernel: size of the convolution kernel (height, width)
Channels (input / output): channels of the input/output feature map
Stride: how many “pixels” of the input image you skip when the kernel slides
Padding: how many zeroes to add on the border of the input feature map

Formula for the size (height/width) of the output feature map (dilation here is 1):

- Input: $(N, C_{in}, H_{in}, W_{in})$ or (C_{in}, H_{in}, W_{in})
- Output: $(N, C_{out}, H_{out}, W_{out})$ or $(C_{out}, H_{out}, W_{out})$, where

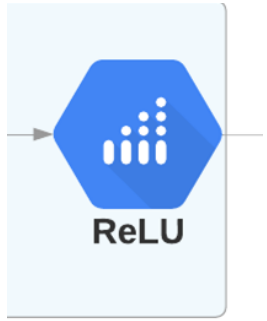
$$H_{out} = \left\lfloor \frac{H_{in} + 2 \times \text{padding}[0] - \text{dilation}[0] \times (\text{kernel\_size}[0] - 1) - 1}{\text{stride}[0]} + 1 \right\rfloor$$

$$W_{out} = \left\lfloor \frac{W_{in} + 2 \times \text{padding}[1] - \text{dilation}[1] \times (\text{kernel\_size}[1] - 1) - 1}{\text{stride}[1]} + 1 \right\rfloor$$

Batch Normalization
Layer that adjusts the statistics of the input feature map at training time. Input and output sizes are the same (does not change the NCHW shape)

num_features: channels of the input feature map.

LAB1 - CNN



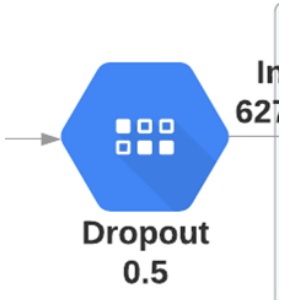
Rectified Linear Unit Activation
Activation function that passes all values above 0, zeroes values below zero.

Max Pooling
Reduces the size of the input feature map according to stride and kernel size.

- Input: (N, C, H_{in}, W_{in}) or (C, H_{in}, W_{in})
- Output: (N, C, H_{out}, W_{out}) or (C, H_{out}, W_{out}), where

$$H_{out} = \left\lfloor \frac{H_{in} + 2 * \text{padding}[0] - \text{dilation}[0] \times (\text{kernel\_size}[0] - 1) - 1}{\text{stride}[0]} + 1 \right\rfloor$$

$$W_{out} = \left\lfloor \frac{W_{in} + 2 * \text{padding}[1] - \text{dilation}[1] \times (\text{kernel\_size}[1] - 1) - 1}{\text{stride}[1]} + 1 \right\rfloor$$



Dropout
Layer used to prevent overfitting. In each training iteration, zeroes the channels of the input feature map with probability p.

Softmax Activation
Transforms the input feature array into a probability array.

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

The LAB starts now !