Exercise Session
CIVIL-459
Spring 2023
Lab 2
Deadline: 24 March

EPFL
School of Civil Engineering
Prof. Alexandre Alahi

Problem Set 2 - Deep Learning in PyTorch

The main goal of this exercise is to help you get familiar with PyTorch.

Getting Started

Familiarize yourself with pytorch using the *Tutorial 1* and *Tutorial 2* notebooks. Additionally, you can check the tutorials found on the Pytorch official website.

Part 1: Fully Connected Neural Network

The problem set contains two notebooks. The first notebook explains how to use Pytorch to build a simple fully connected neural network.The second one is to evaluate the performance of the trained models. Follow the problem statement of each exercise. Here are some notes:

1. The neural network should have at least 2 layers but can be as large you want.
2. Set up the forward pass by connecting the different layers of the neural network. Do not forget to use activation layers.
3. Create the network by specifying the input and output size. Then choose the optimizer you want to use (e.g., SGD or Adam). You should tune the parameters of the optimizer.
4. Implement the training process and save your best model.
5. Finally, for us to test your code and get an accuracy, you should fill the evaluator notebook where you should load the saved model, input to it the data and get the prediction labels. We will be using our own test data for grading.

Deliverables

You need to submit two jupyter notebooks and the trained models into the moodle.

Part 2: Convolutional Neural Network

This time we'll check out how to build a convolutional network to classify CIFAR10 images. By using weight sharing - multiple units with the same weights - convolutional layers are able to learn repeated patterns in your data. For example, a unit could learn the pattern for an eye, or a face, or lower level features like edges.

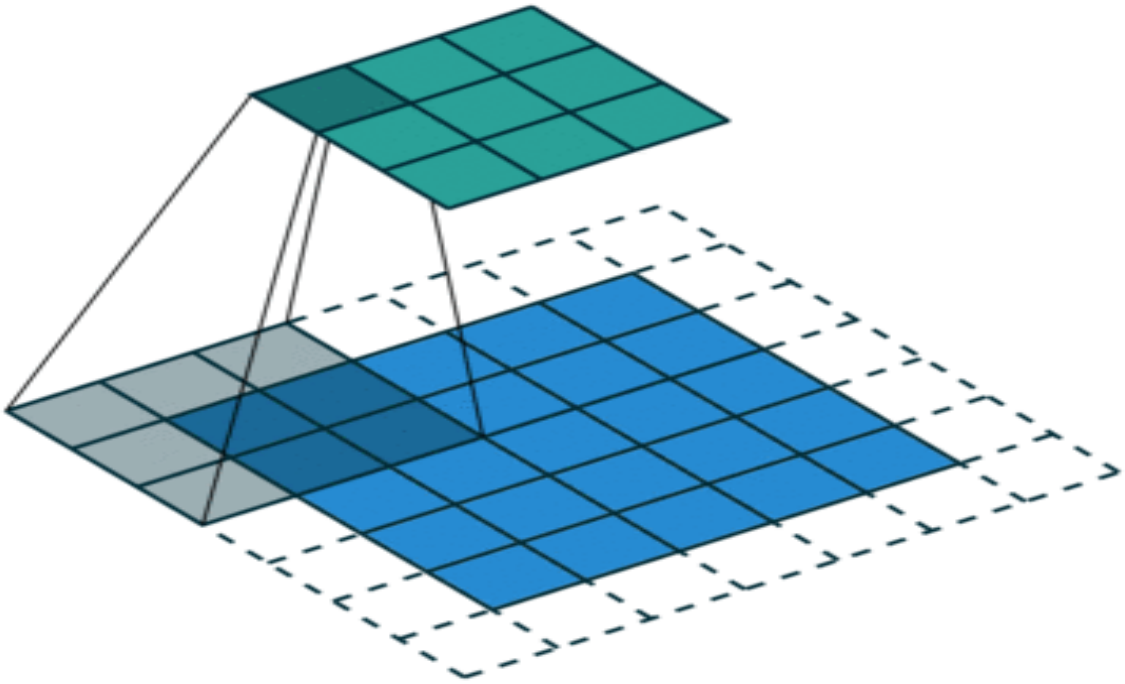

Figure 1: Convolution

Traditionally, convolutional layers are followed by max-pooling, where values in the convolutional layer are aggregated into a smaller layer shown in Figure 2. These types of networks become

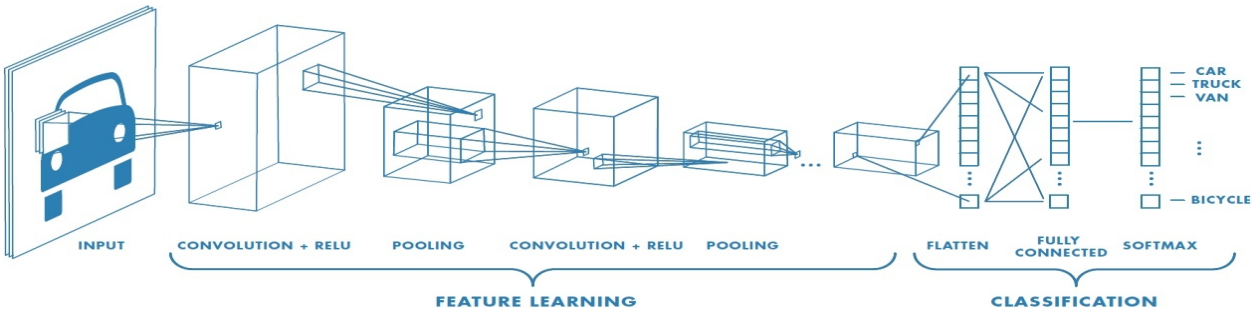

Figure 2: Convolutional Neural Network

really useful when you stack a bunch of layers, you make the network deeper. Typically you'll use a convolutional layer to change the depth, ReLU activation, then a max-pooling layer to reduce the height and width.

To finish off the network, you need to flatten the final convolutional layer into a normal fully-connected (dense) layer, then add more fully-connected layers as a classifier. The convolutional layers act as feature detectors that the classifier uses as inputs. Convolutional networks have been massively effective in image classification, object recognition, and even in natural language processing applications like speech generation.

Building this new network is pretty much the same as before, but we use nn.Conv2d for the

convolutional layers and nn.MaxPool2d for the max-pooling layer.

What to do

Get the codes from "https://github.com/vita-epfl/DLAV-2023" and implement the convolutional neural network using pytorch in **CNN Exercise.ipynb**

1. Define the different layers of the neural network. The neural should have at least 2 layers but can be as large as you want.
2. Set up the forward pass by connecting the different layers of the neural network. **Note:** Do not forget to use activation layers.
3. Choose a loss function and an optimizer.You can tune the parameters of the optimizer.
4. Implement the training process and save your best model.
5. Finally, for us to test your code and get an accuracy, you should fill the evaluator notebook where you should load the saved model, input to it the data and get the prediction labels. We will be using our own test data for grading.

Deliverables

You need to submit the jupyter notebooks and the trained models into the moodle.

Helpful References

- Pytorch Documentation: https://pytorch.org/docs/stable/
- Pytorch Tutorial: Official link, Collection
- Pytorch Convolution Layers: http://pytorch.org/docs/master/nn.html#convolution-layers