**Exercise Session**
CIVIL-459
Spring 2025
Lab 3

**EPFL**
School of Civil Engineering
**Prof. Alexandre Alahi**

# Problem Set 3 - Convolutional Neural Network

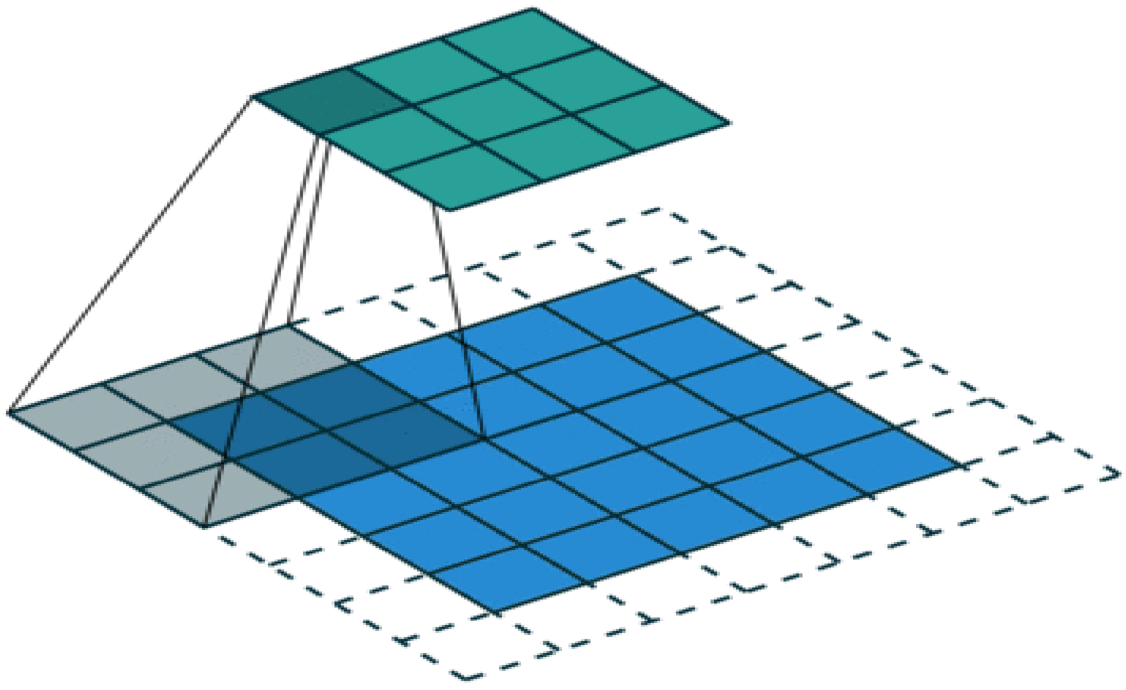

Figure 1: Convolution

This time we'll check out how to build a convolutional network to classify CIFAR10 images (The CIFAR10 dataset consists of 60,000 color images (32x32 pixels) divided into 10 classes of Airplane, Bird, Cat, Dog etc). 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.

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 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

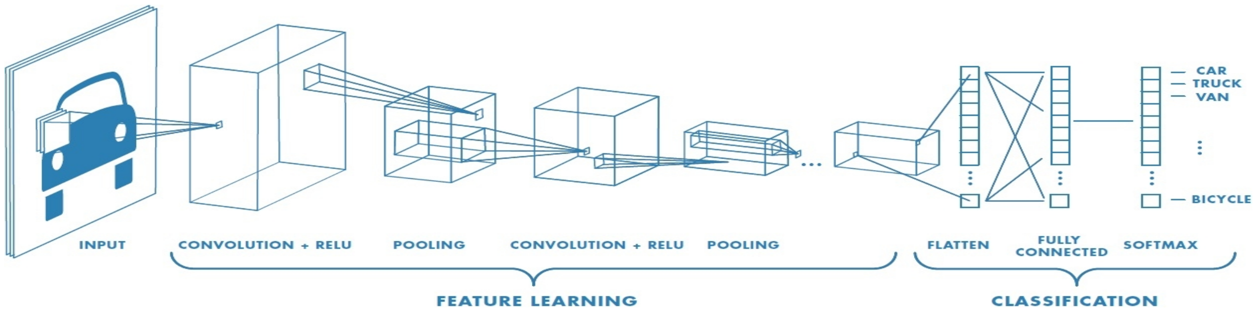

Figure 2: Convolutional Neural Network

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

Understand and implement the convolutional neural network using pytorch in **CNN Exercise.ipynb** (You can get the notebook from “https://github.com/vita-epfl/DLAV-2025” or on Moodle). There are several theory questions designed to evaluate your understanding of CNN architecture, accounting for 40% of the notebook assessment. The remaining 60% will be based on the coding portion.

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.

## Deliverables

You need to submit the jupyter notebooks containing the code and answers to theory questions 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`