Technische Universität München

ConvNets with Memory-Efficient Implicit Padding

Practical Course:

Hands-on Deep Learning for Computer Vision and Biomedicine

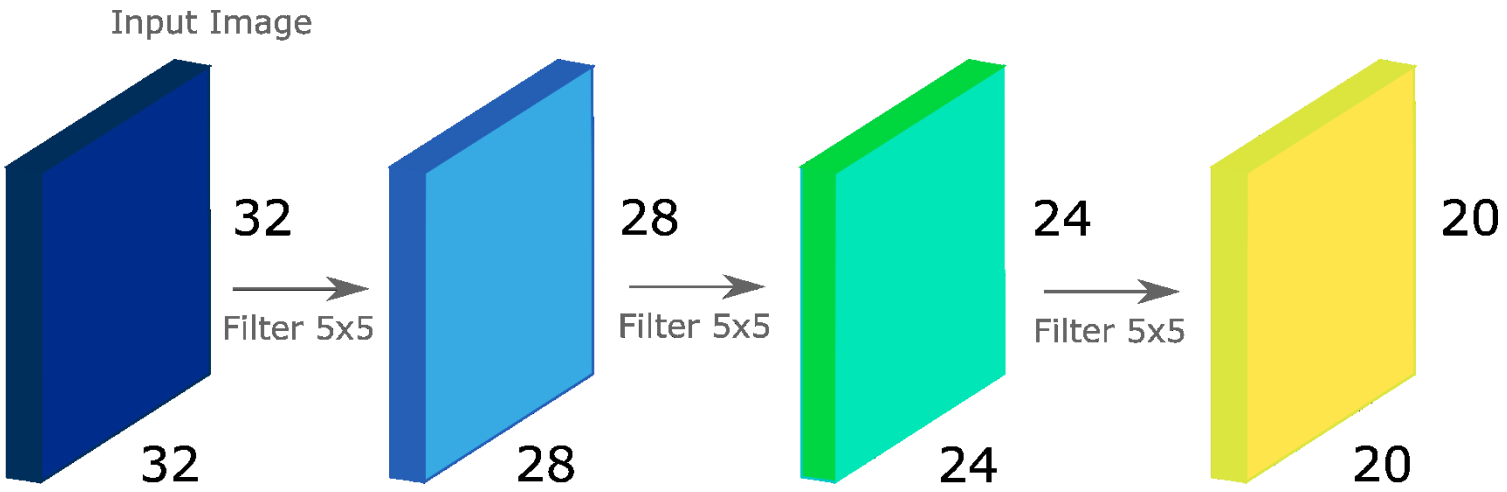
Maria Dreher

Summer Term 2018

18/09/2018

Motivation

Dimensions in Convolutional Layers



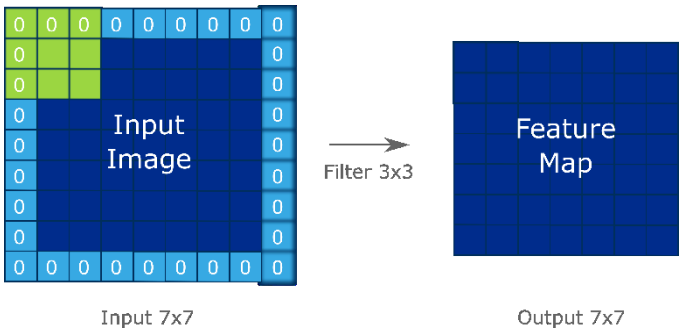
Quickly decreasing image size: 32 → 28 → 24 → 20

Motivation

Goal: Keep image size

1. Option: (Zero) padding in every layer

- Repeat this in every layer
- Output of every ConvLayer is as large as Input
- Overall network output has same size as well

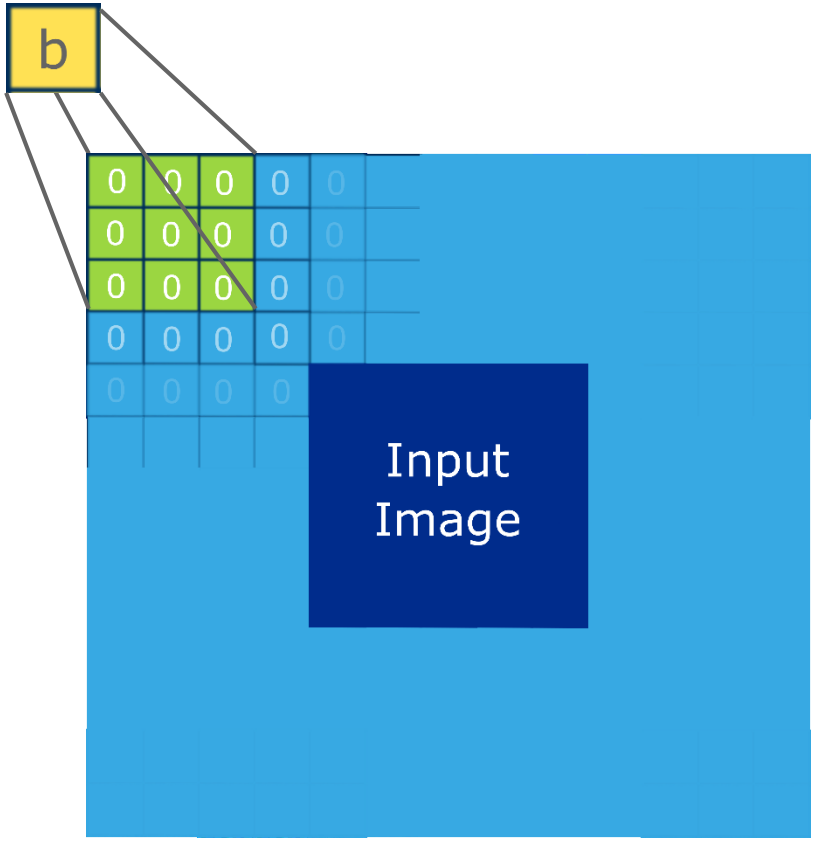


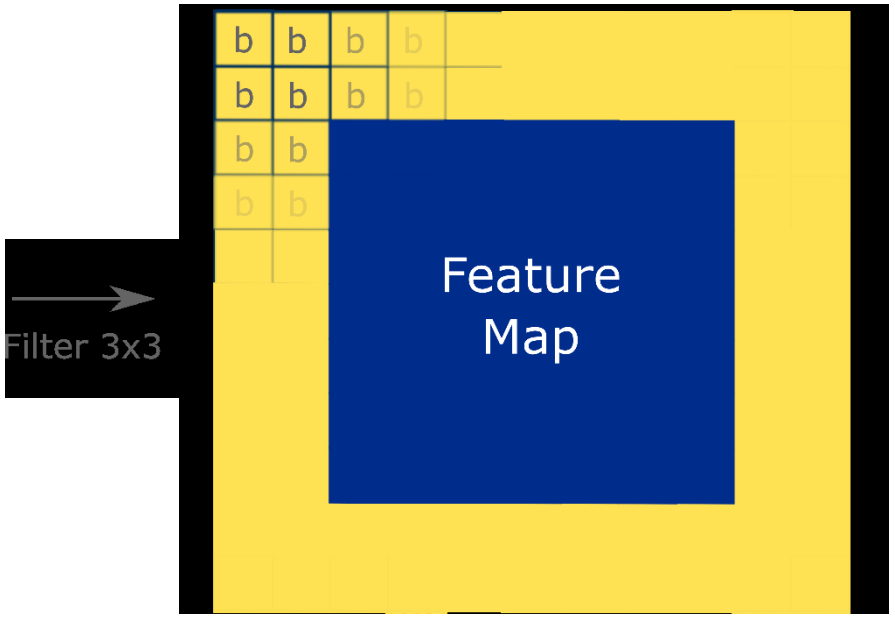
BUT: Repeatedly adding fake zeros to the input!

Explicit and Implicit Padding

Back to Basics of ConvLayers

$$z_i = w^T x_i + b$$





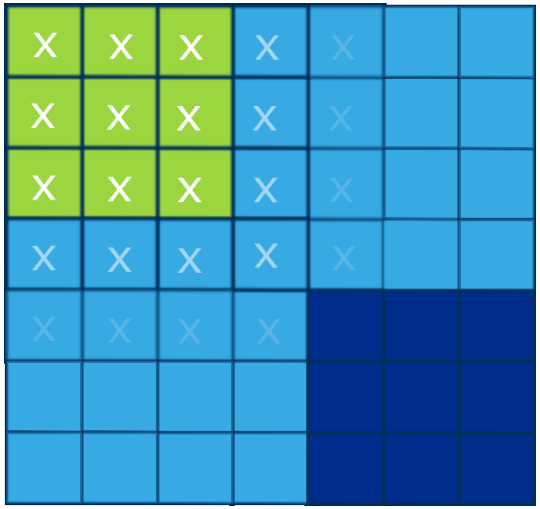
Explicit and Implicit Padding

Back to Basics of ConvLayers

$$z_i = w^T x_i + b$$

For other values than zero:

$$z_i = (w_1 + w_2 + \cdots + w_n)\, x + b$$

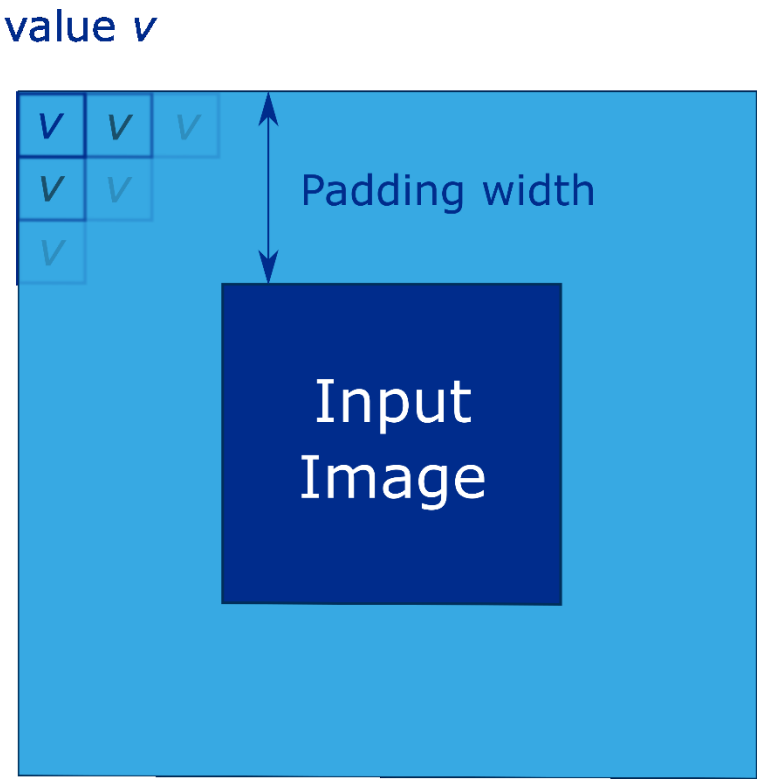


Explicit and Implicit Padding

Implicit Padding

Only store two values:

- **Padding width**
- **Fill value**



PyTorch

Convolutional Networks

PyTorch

Convolutional Layer

class **torch.nn.Conv2d**(*in_channels, out_channels, kernel_size, stride=1, padding=0, dilation=1, groups=1, bias=True*) [source]

*„**padding** controls the amount of*

***implicit zero-paddings** on both sides*

*for **padding** number of points for each dimension“* [1]

[1] https://pytorch.org/docs/stable/nn.html

PyTorch

Extend Conv2d class in PyTorch?

- **Idea**: Extend Conv2d to take a fill value
- **But**:
  - Only API is in Python
  - All operations are done in C/C++ (including convolution)
  - Value has to be passed through whole pipeline