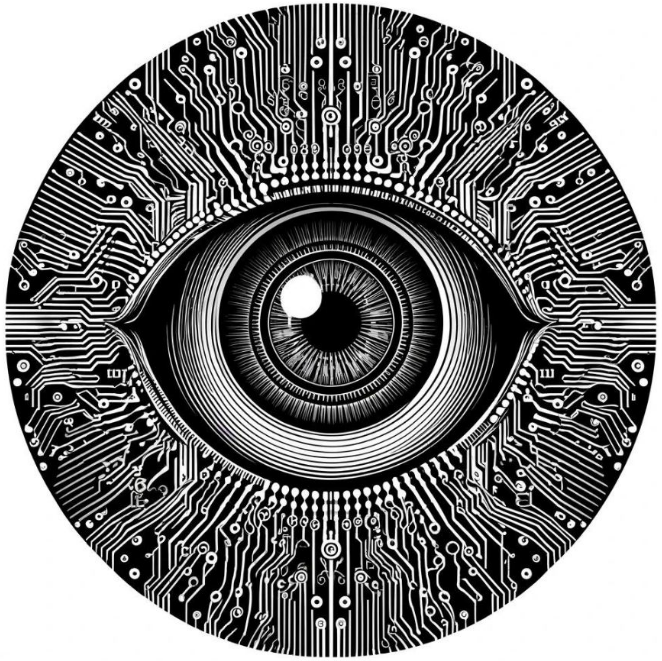
Fundamentals of Convolutions



Antonio Rueda-Toicen

SPONSORED BY THE
Federal Ministry
of Education
and Research

Learning goals

- Understand the role of convolutions in producing image features
- Evaluate effect of stride, padding, and filter size on output images

Convolution filters create new images

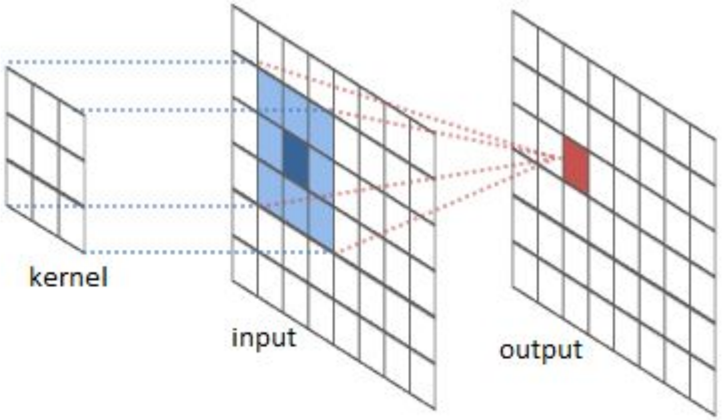
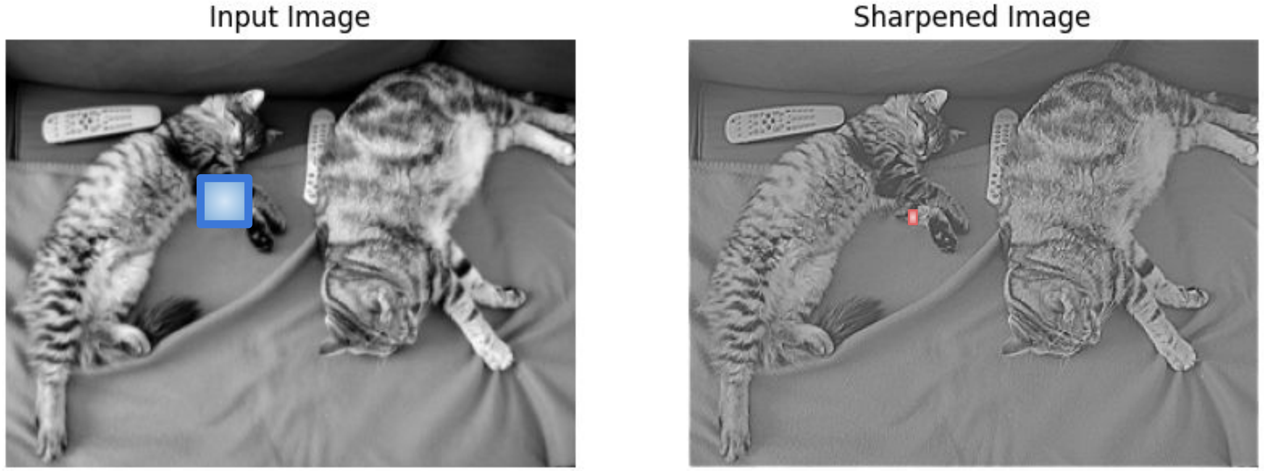


Image from source



$$\text{Sharpening Kernel} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \text{Image Patch (Input)} = \begin{bmatrix} 10 & 20 & 30 \\ 40 & 50 & 60 \\ 70 & 80 & 90 \end{bmatrix}$$

$$\text{Output} = (0 \cdot 10) + (-1 \cdot 20) + (0 \cdot 30) + (-1 \cdot 40) + (5 \cdot 50) + (-1 \cdot 60) + (0 \cdot 70) + (-1 \cdot 80) + (0 \cdot 90)$$

$$\text{Output} = 50$$

Convolutions were ‘traditionally’ handcrafted

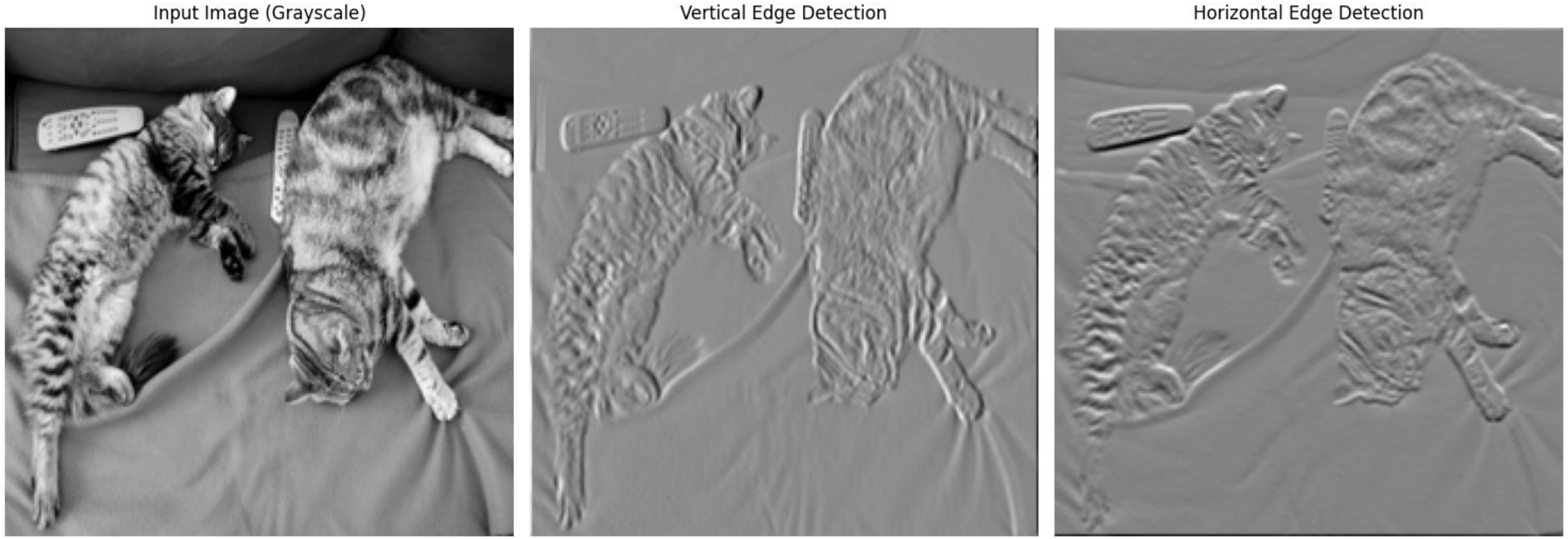
Input Image

Blurred Image with Convolution Filter

$$\begin{bmatrix} 0.111 & 0.222 & 0.111 \\ 0.222 & 0.444 & 0.222 \\ 0.111 & 0.222 & 0.111 \end{bmatrix}$$

Gaussian blur kernel

Handcrafted convolution kernels



Vertical edge detection:

$$K_v = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

Horizontal edge detection:

$$K_h = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

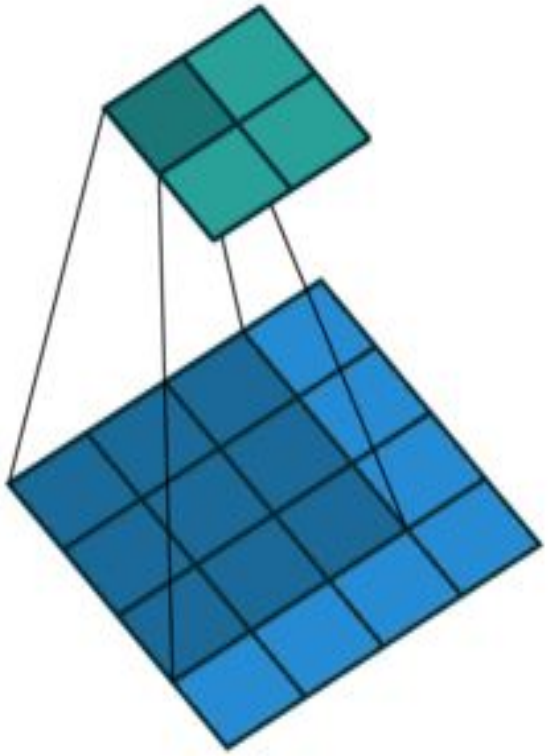
The receptive field size controls how many neighboring pixels are considered

3x3 vs 5x5 uniform blur kernels

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

$$\begin{bmatrix} \frac{1}{25} & \frac{1}{25} & \frac{1}{25} & \frac{1}{25} & \frac{1}{25} \\ \frac{1}{25} & \frac{1}{25} & \frac{1}{25} & \frac{1}{25} & \frac{1}{25} \\ \frac{1}{25} & \frac{1}{25} & \frac{1}{25} & \frac{1}{25} & \frac{1}{25} \\ \frac{1}{25} & \frac{1}{25} & \frac{1}{25} & \frac{1}{25} & \frac{1}{25} \\ \frac{1}{25} & \frac{1}{25} & \frac{1}{25} & \frac{1}{25} & \frac{1}{25} \end{bmatrix}$$

Padding and stride effects



Convolution of 3x3 and stride = 1 without padding

Effect: the output loses one pixel on each dimension

Border padding

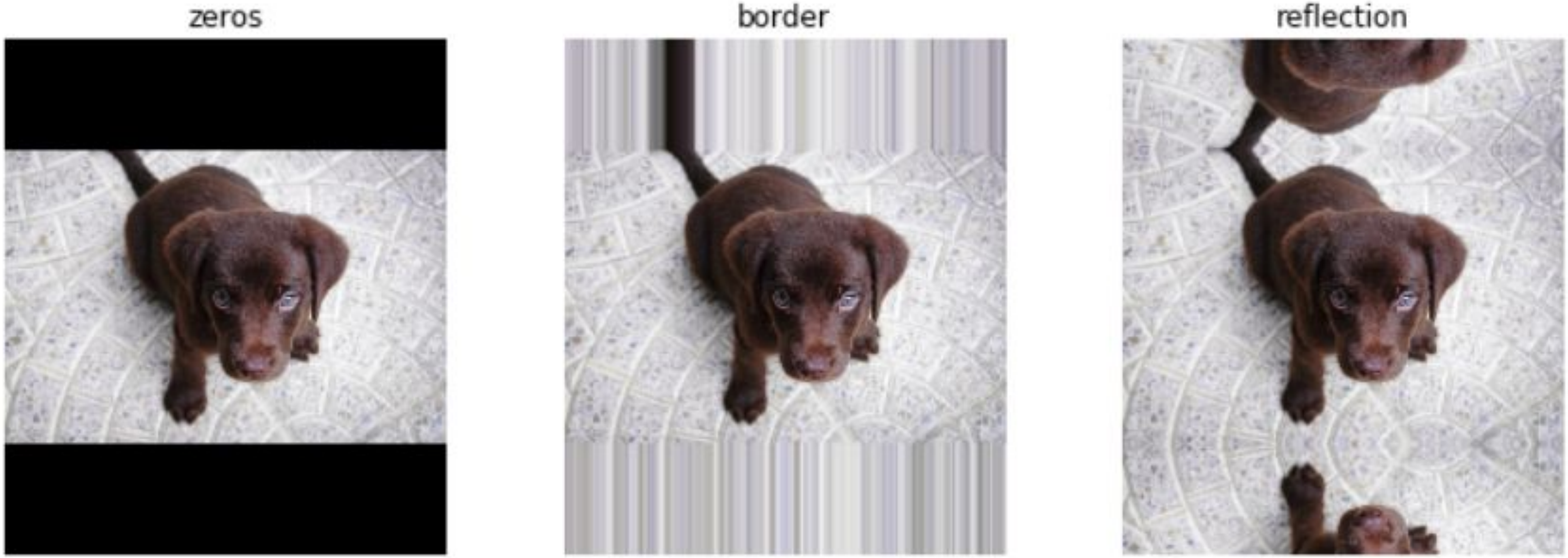
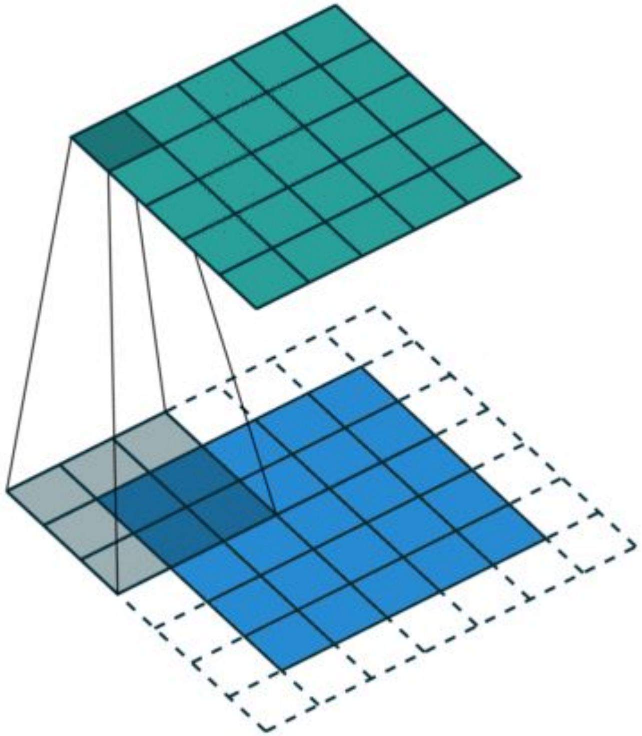


Image from [the fastai documentation]

“Automatic” feature learning

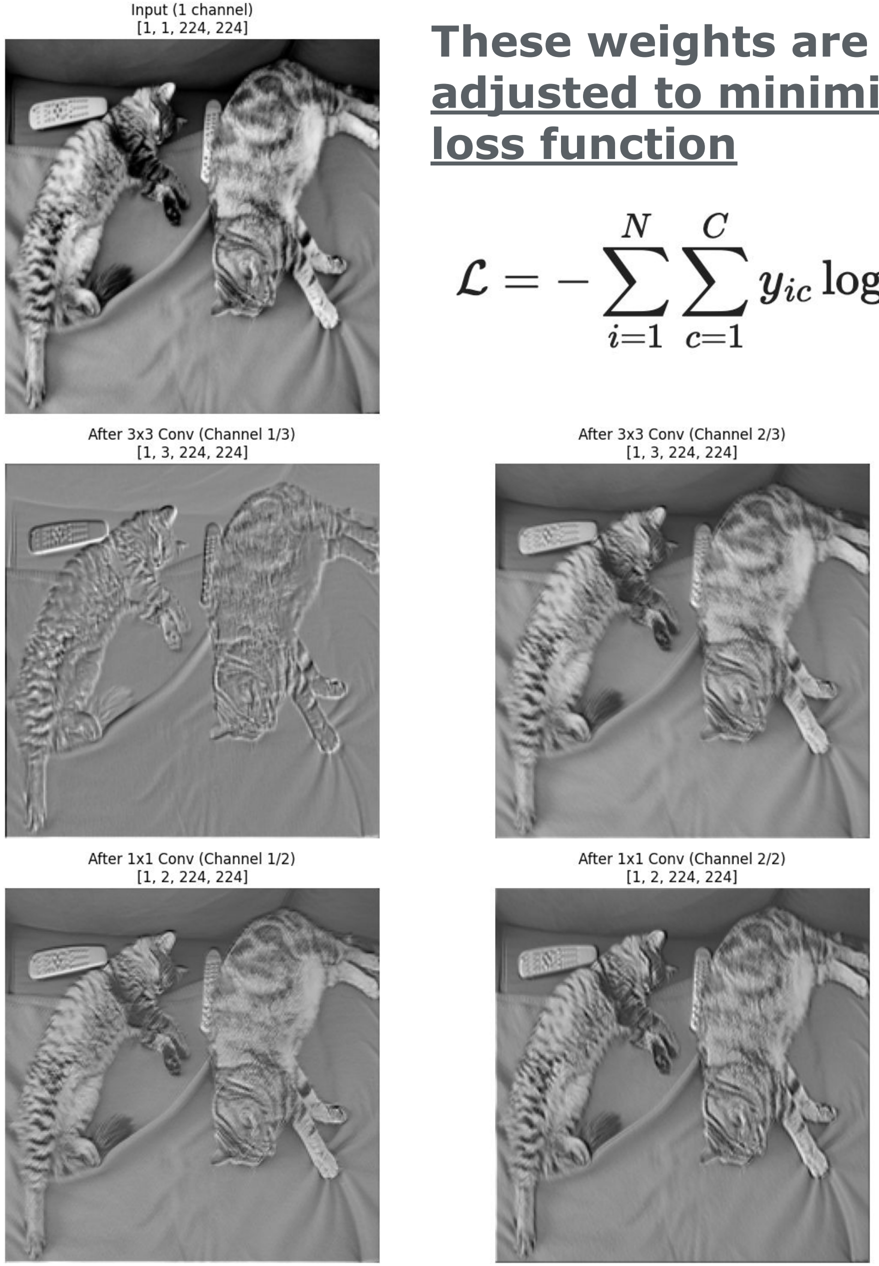


$$\begin{pmatrix} -0.64 & -0.74 & 0.91 \\ -0.96 & 0.48 & -0.46 \\ 0.56 & 0.67 & -0.08 \end{pmatrix}$$

$$\begin{pmatrix} -0.89 & 0.40 & 0.12 \\ -0.45 & 0.38 & 0.67 \\ 0.91 & -0.71 & -0.35 \end{pmatrix}$$

$$\begin{pmatrix} 0.65 & 0.39 & -0.75 \\ 0.36 & -0.57 & 0.34 \\ 0.61 & 0.69 & 0.43 \end{pmatrix}$$

$$\begin{pmatrix} -0.64 & -0.74 & 0.91 \\ -0.96 & 0.48 & -0.46 \\ 0.56 & 0.67 & -0.08 \end{pmatrix}$$



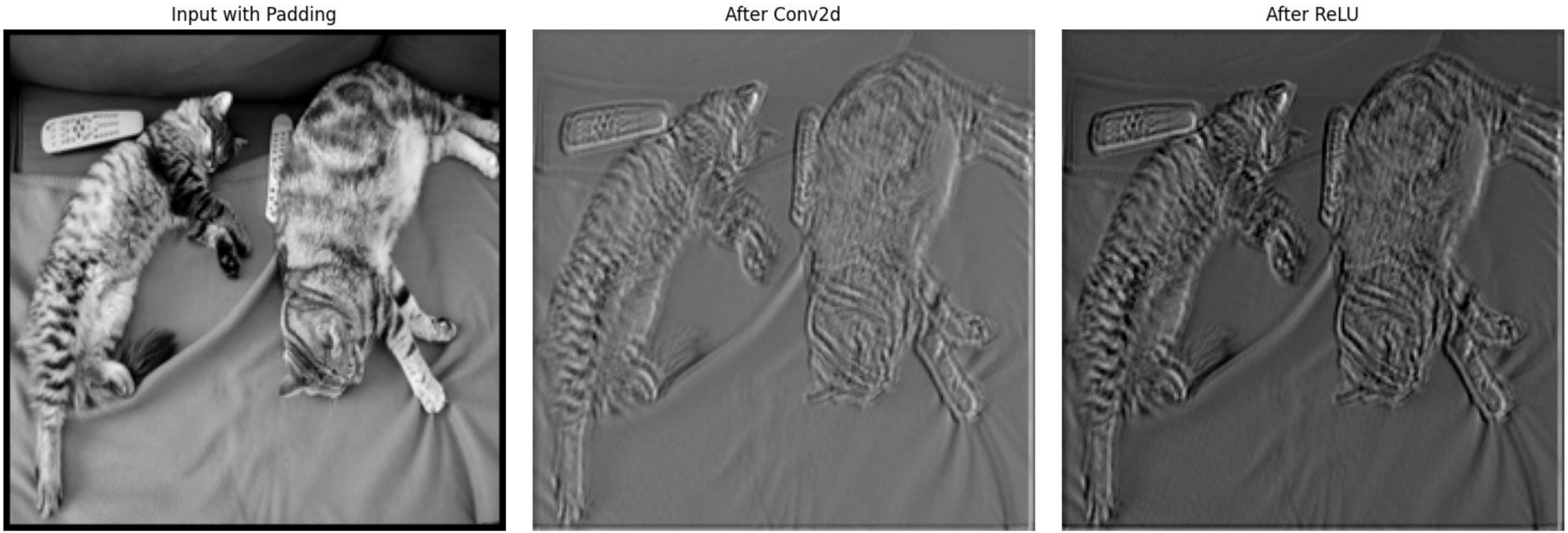
These weights are <u>adjusted to minimize the loss function</u>

$$\mathcal{L} = -\sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic} \log(\hat{y}_{ic})$$

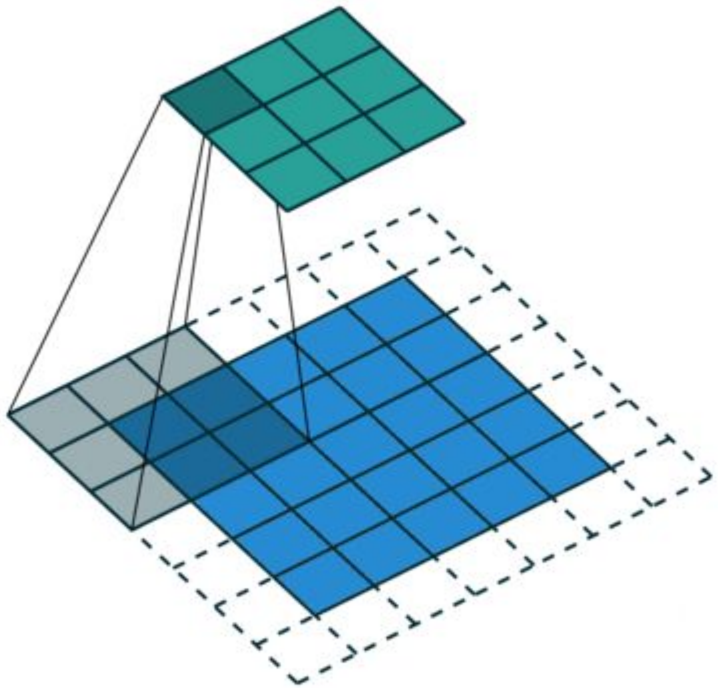
Convolutions of 3x3 and stride = 1 padded with zeros

Effect: the output preserves the original image size while producing a “different-looking” image (based on the values of the weights)

ReLU adds non-linearity to activations

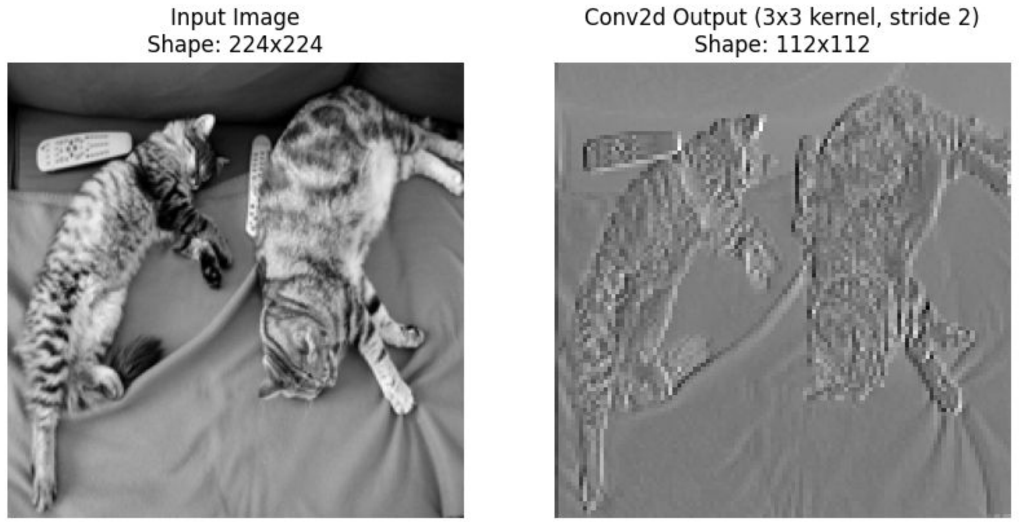


Downsampling images with stride > 1 convolutions

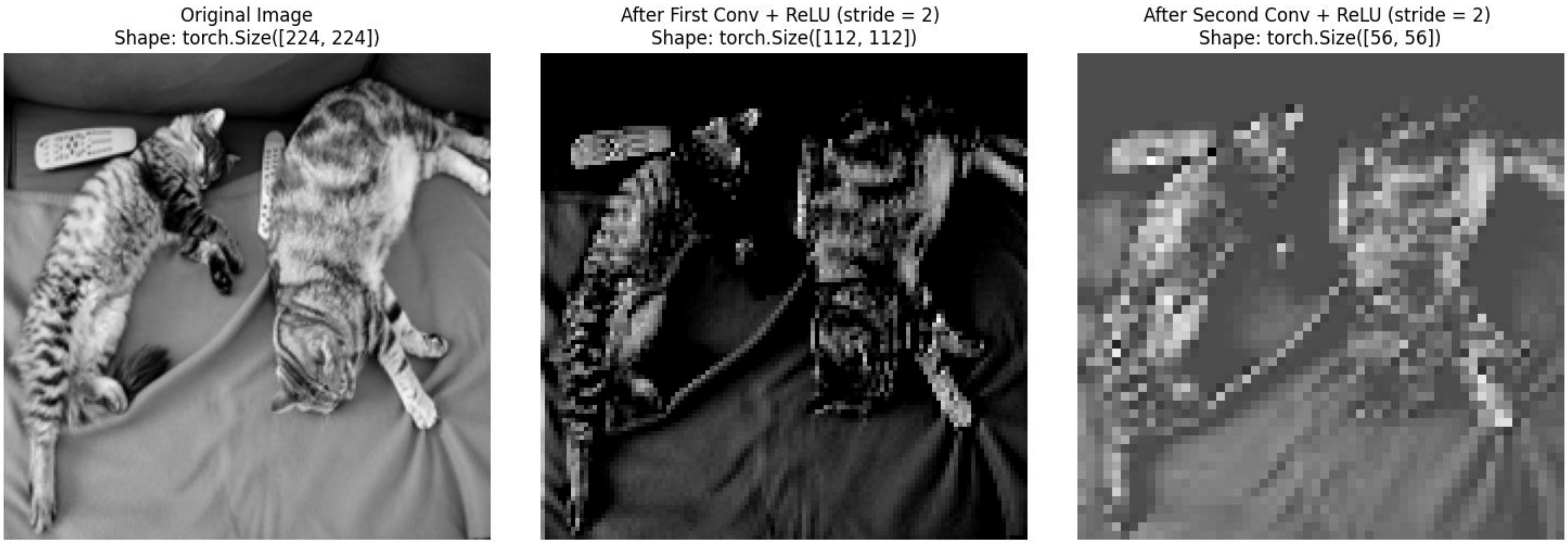


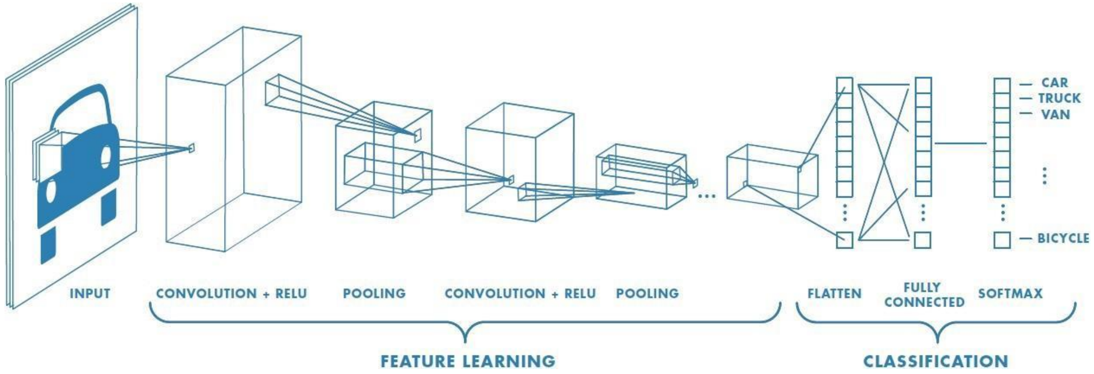
Convolution of 3x3 and stride = 2 padded with zeros

Effect: the output is downsampled to about half its size



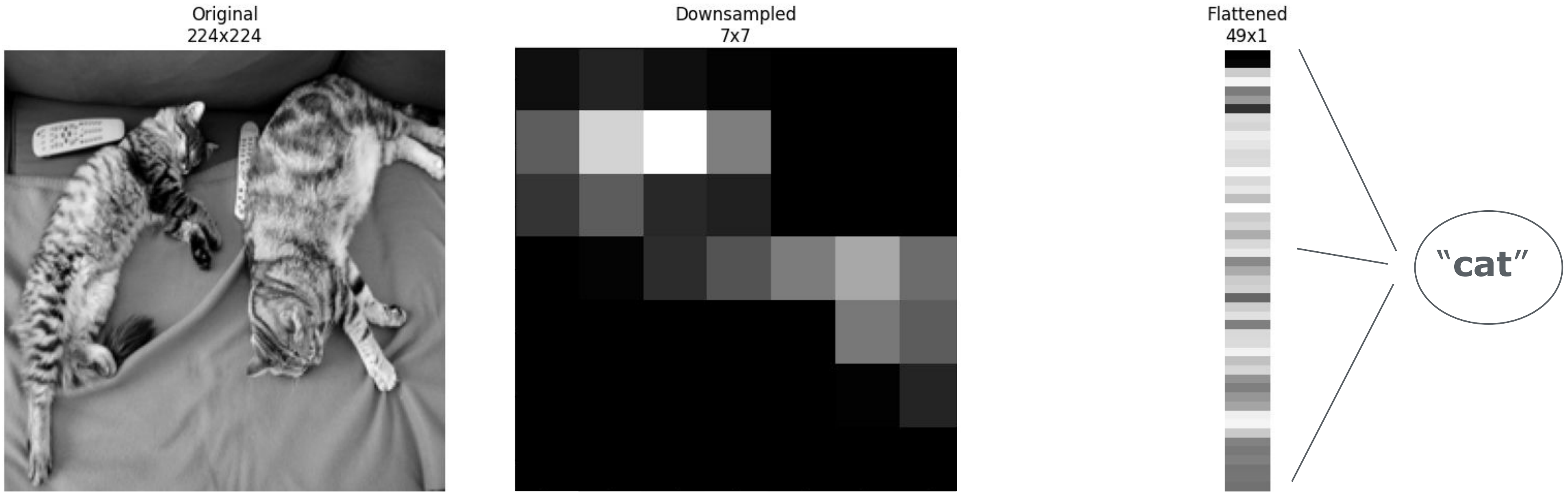
Convolutions are concatenated

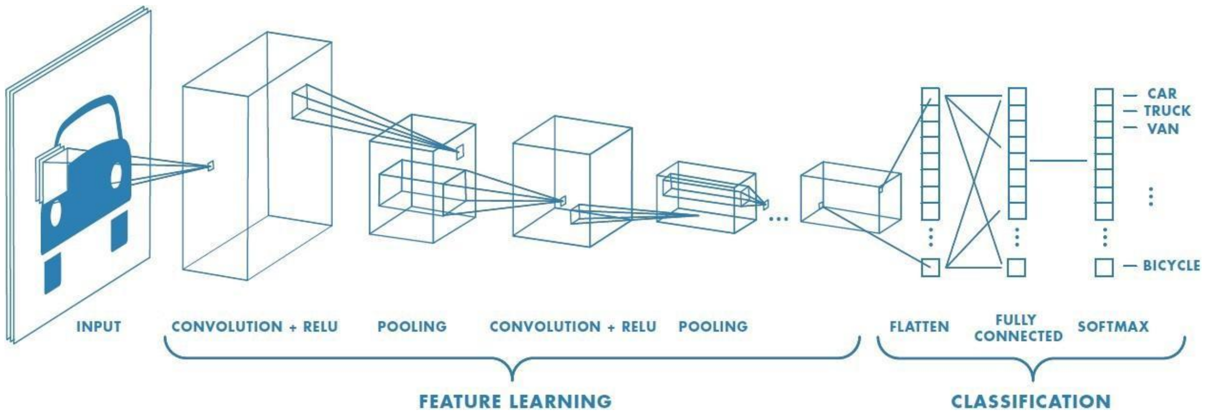




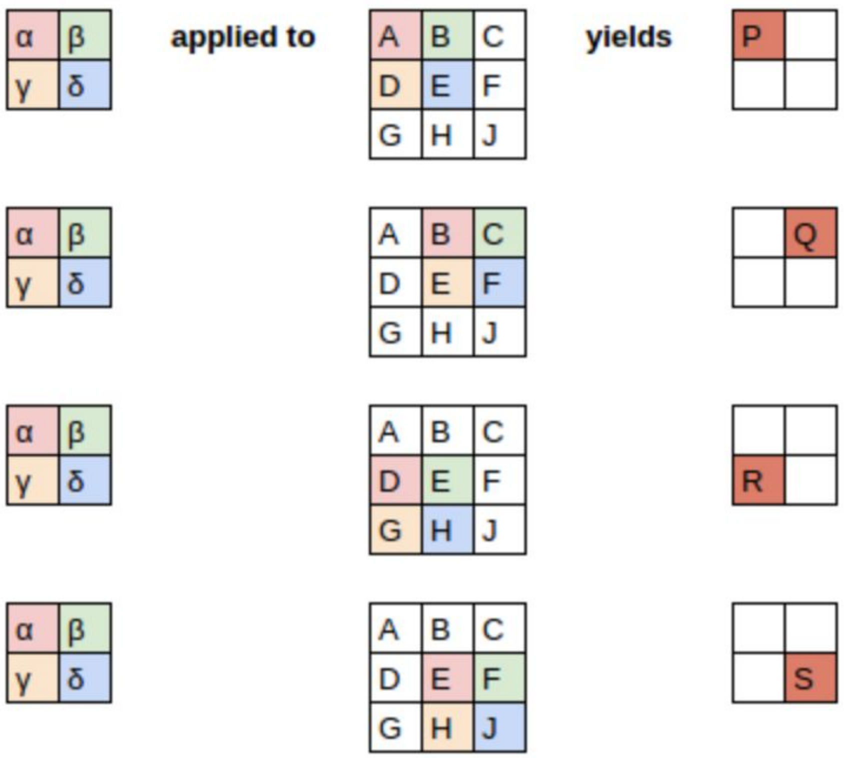
$$\mathcal{L} = -\sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic} \log(\hat{y}_{ic})$$

We use convolutions as “feature extractors” for classifiers





‘Unrolling’ convolutions



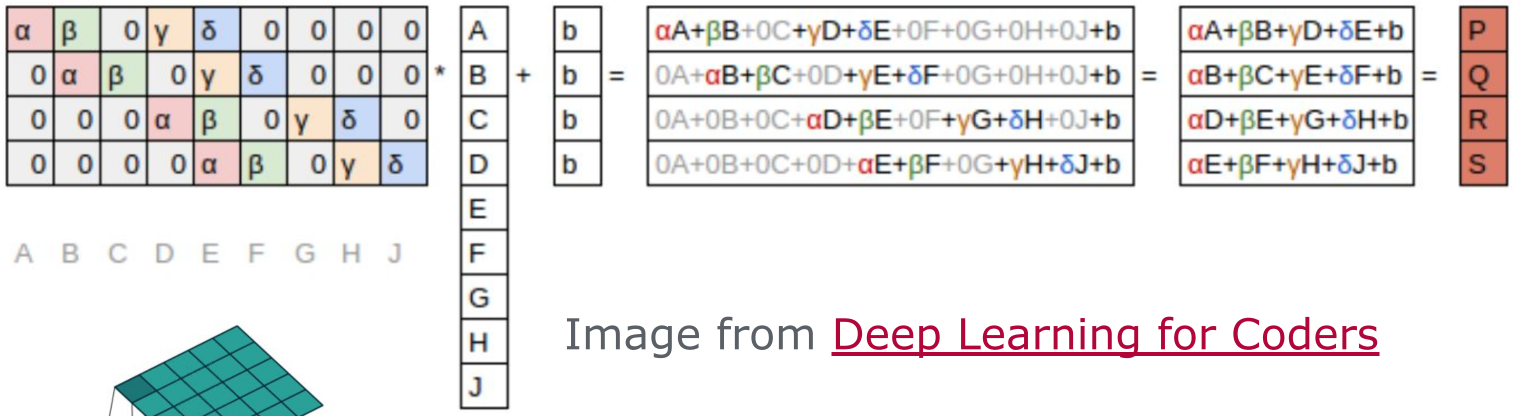
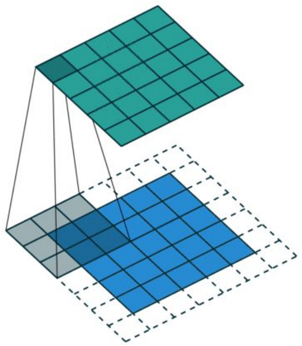


Image from [Deep Learning for Coders]



Even though we see convolutions happen as sequential animations, their values are ‘unrolled’ by PyTorch and applied as parallel matrix multiplications **simultaneously**

Summary

Convolutions and feature learning

- Convolutional layers create new images (“new features”) through learnable weights
- Convolutions were traditionally hand-crafted, now we use the neural network to set their weights
- Classifiers flatten these learned features before feeding them to a feedforward network
- Convolutions are “unrolled” by PyTorch in order to do parallel processing

Effects of receptive field size, padding, and stride

- The size of the receptive field influences how many pixels we consider
- Convolutions with padding preserve spatial dimensions
- Strided convolutions allow us to do learnable downsampling

SPONSORED BY THE
Federal Ministry
of Education
and Research

Further reading and references

A guide to convolution arithmetic for deep learning

- https://arxiv.org/abs/1603.07285

Network in network (1x1 convolutions)

- https://arxiv.org/abs/1312.4400

Hypercolumns for object segmentation and fine-grained localization

- https://openaccess.thecvf.com/content_cvpr_2015/papers/Hariharan_Hypercolumns_for_Object_2015_CVPR_paper.pdf

SPONSORED BY THE
Federal Ministry of Education and Research