Convolutional Neural Networks (CNN)

Readings: Murphy 14.

Images ≡ 2D array of pixels. Each pixel characterized by 3 integer values (encoding intensity levels of R, G, B)

Important structural knowledge:

-1- Spatial Locality: set of pixels will take into consideration to find, say a Cat, will be near one another in the image.

-2- Translation invariance: pattern of pixels that characterizes a Cat is the same no matter where in the image the cat occurs!

Filters - 1D

image filter: is a function that takes in a local spatial neighborhood of pixel values and detects the presence of some pattern in data.

Example: image = 1D binary array ; X

filter = $\boxed{F_1}$ of size two ; $Y_i = F_1(X_{i-1}, X_i)$ for pixel i

filter = $\boxed{F_2}$ of size three ; $Y_i = F_2(X_{i-1}, X_i, X_{i+1})$

input image:

0	0	1	1	1	0	1	0	0	0

F1:

-1	+1

F1: is a detector of "Left edges"

After convolution with F1

0	0	1	0	0	-1	1	-1	0	0

F2:

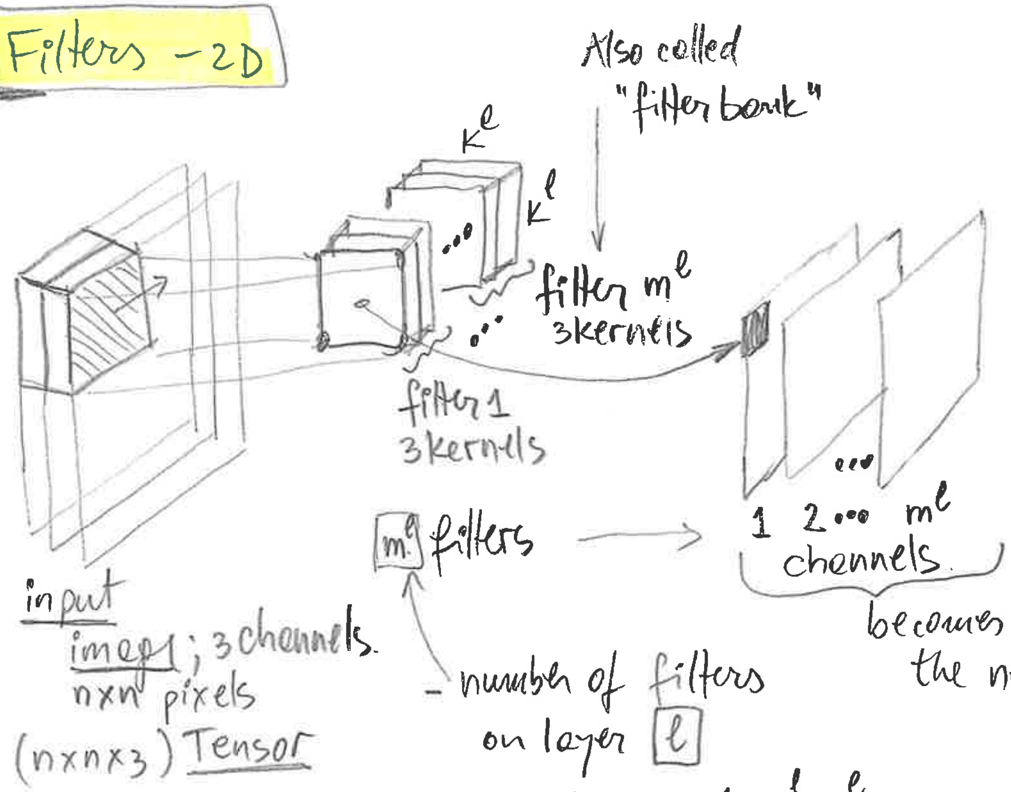
-1	+1	-1

F2: is a detector for isolated positive pixels in the binary image.

After convolution with F2

0	-1	0	-1	0	-2	1	-1	0	0

Filters – 2D



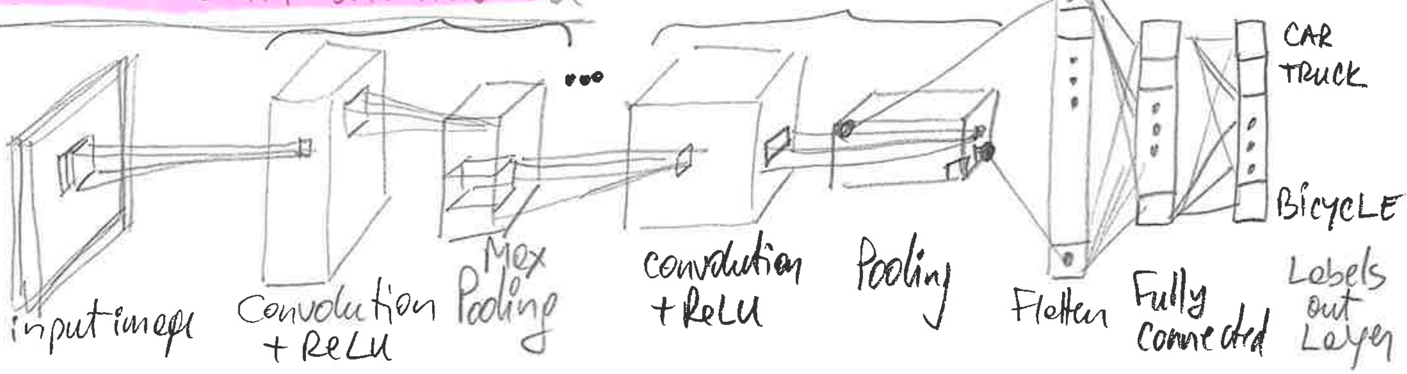
becomes the "input image" to the next convolution Layer.

- number of filters on layer $\boxed{\ell}$
- size of kernel $k^\ell \times k^\ell$
- stride is spacing we apply the filter to the image
- padding = how many extra pixels we add around the edge of input image.

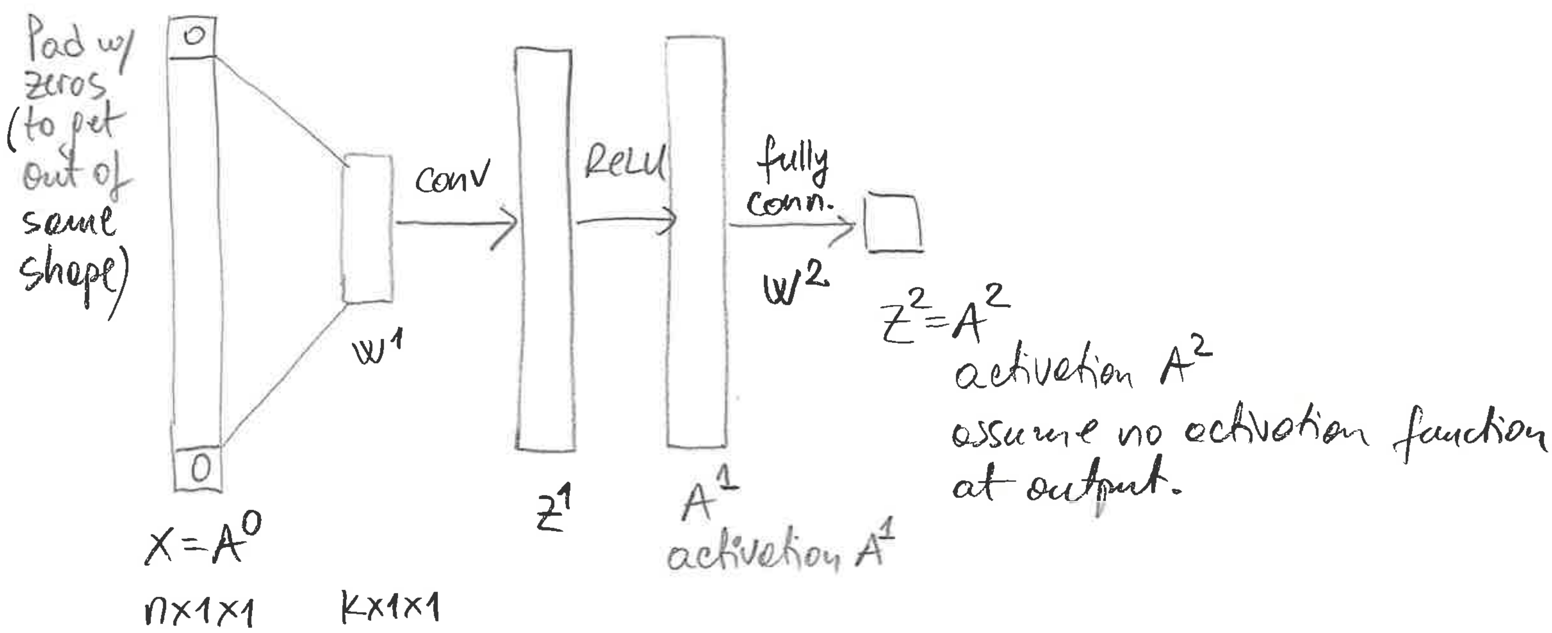
Max pooling **layer** operates like a filter, but has no weights.

- we can think of it as purely functional, like a ReLU in a fully connected network!
- is applied w/ a stride > 1, so that resulting image is smaller than the input image.
- $k \times k$ window, $k \geq$ stride, so that the whole image is covered.
- does not add additional bias or offset values!

Common CNN architecture



Backpropagation in a simple CNN



– Describe the forward pass as follows:

$$z_i^1 = w^1 \cdot A^0_{[i-\lfloor k/2 \rfloor : i+\lfloor k/2 \rfloor]}$$ Assume k is odd number.

$$A^1 = \mathrm{ReLU}(z^1)$$

$$A^2 = Z^2 = w^{2T} A^1$$

$$\mathcal{L}_{square}(A^2, y) = (A^2 - y)^2$$: squared Loss.

– Weight update: $\frac{\partial Loss}{\partial w^1} = \frac{\partial z^1}{\partial w^1} \circ \frac{\partial A^1}{\partial z^1} \circ \frac{\partial Loss}{\partial A^1}$

→ k×n matrix such that $\frac{\partial z_i^1}{\partial w_j^1} = X_{i-\lfloor k/2 \rfloor + j - 1}$

→ n×n matrix, diagonal, such that $\frac{\partial A_i^1}{\partial z_i^1} = \begin{cases} 1, & \text{if } z_i^1 > 0 \\ 0, & \text{otherwise.} \end{cases}$

→ $\frac{\partial Loss}{\partial A^1} = \frac{\partial Loss}{\partial A^2} \cdot \frac{\partial A^2}{\partial A^1} = 2(A^2 - y) \cdot w^2$; n×1 vector.