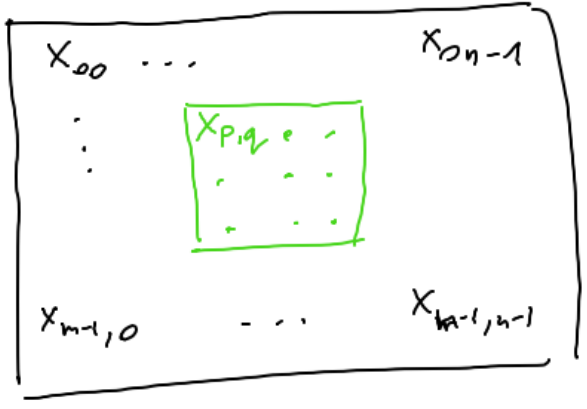
Convolutional Neural Networks (CNN)

Le Cun 180 (1989)

input – a matrix $m \times n$ $\quad (x_{ij})_{i=0,j=0}^{m-1,n-1}$



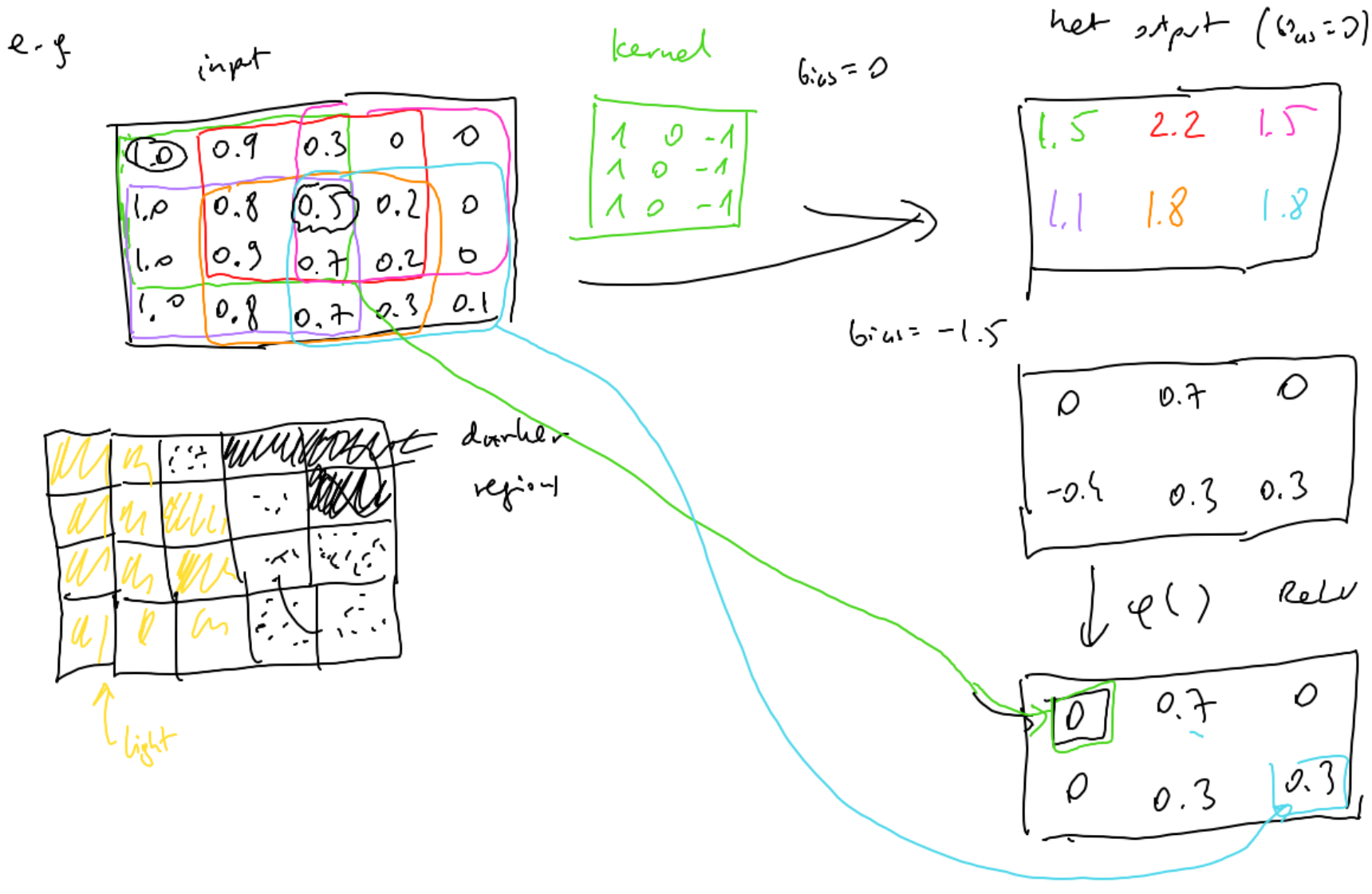
kernel of size $k \times k$ ⇝ weights $(a_{ij})_{i,j=0}^{k-1}$, b – bias

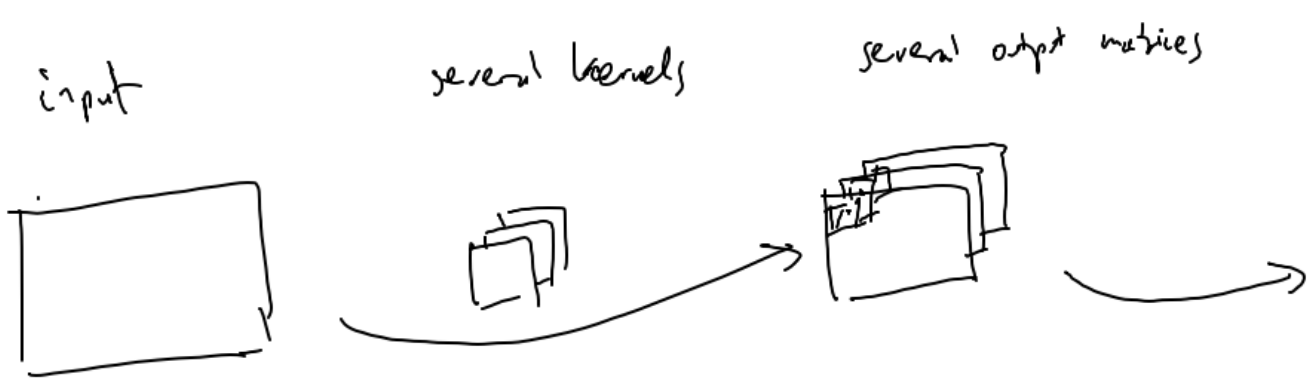
for $p = 0, 1, \ldots, m-k$, $q = 0, 1, \ldots, n-k$

$$net_{p,q} = \sum_{i=0}^{k-1} \sum_{j=0}^{k-1} a_{ij} x_{i+p, j+q} + b$$

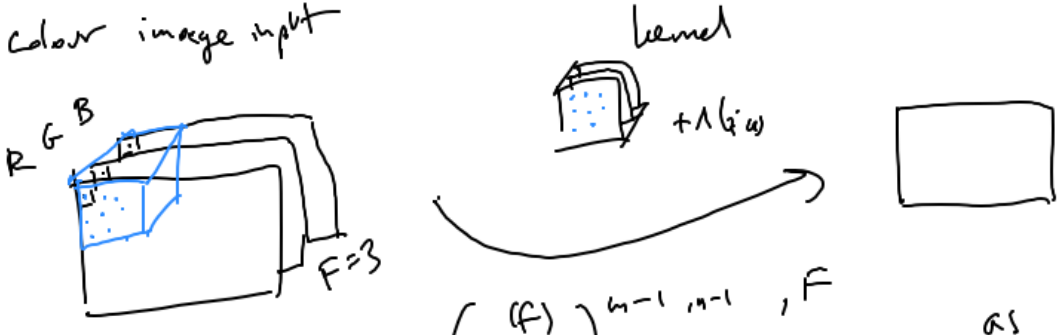
$$output_{p,q} = \varphi(net_{p,q})$$

φ – activation function, usually ReLU


e.g
input
kernel
bias = 0
het output (bias = 0)
1.5 2.2 1.5
1.1 1.8 1.8
bias = -1.5
0 0.7 0
-0.4 0.3 0.3
φ( ) ReLU
0 0.7 0
0 0.3 0.3
darker region
light



„3×3 kernel" means in fact „3×3×F" kernel

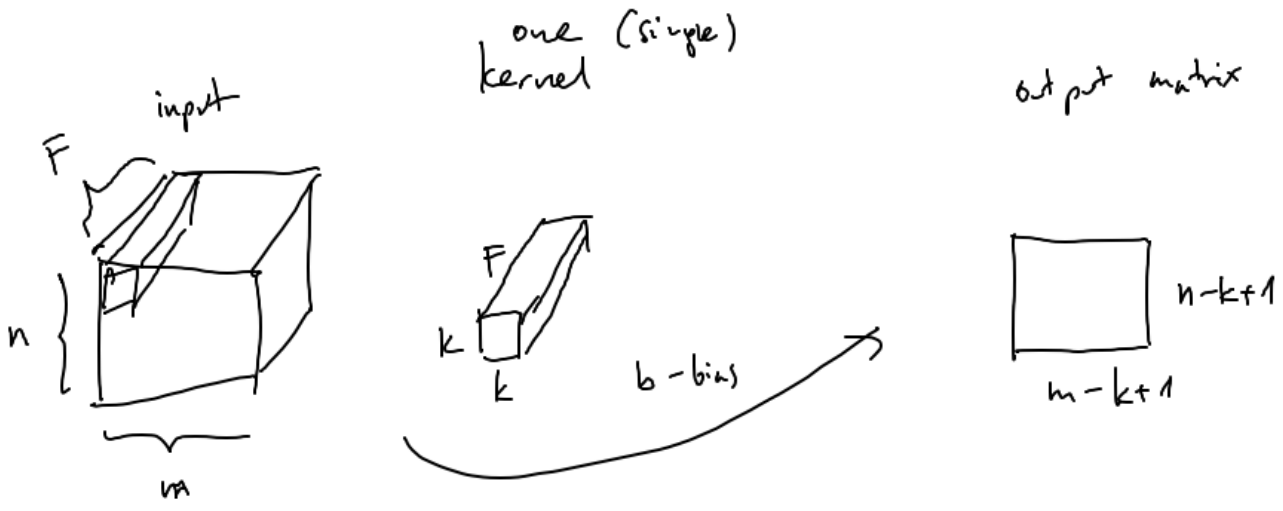


If we have $\left(x_{i,j}^{(f)}\right)_{i=0,j=0,f=1}^{m-1,n-1,F}$ as input (i.e., F channels of size $m \times n$ matrices),

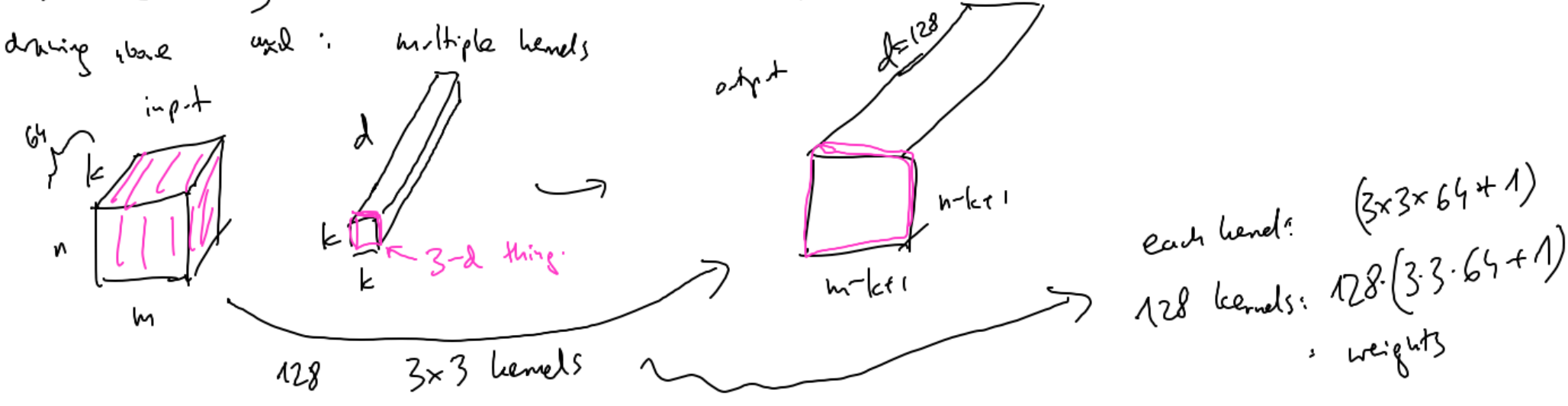
then the kernel should have $k \times k \times F + 1$ weights: $\left(a_{i,j}^{(f)}\right)_{i,j=0,f=1}^{k-1,F}$

for $p = 0, \dots, m-k$, $q = 0, 1, \dots, n-k$:

$$net_{p,q} = \left(\sum_{i=0}^{k-1} \sum_{j=0}^{k-1} \sum_{f=1}^{F} a_{i,j}^{(f)} x_{i+p,j+q}^{(f)} \right) + b$$



But one usually takes more than 1 kernel, so we skip the dimension F in the drawing above and use multiple kernels



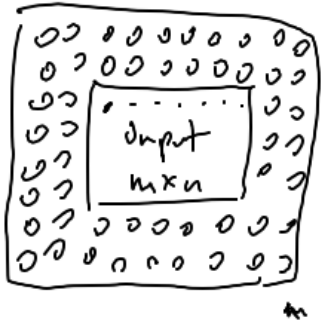
Padding – it allows one to keep the size of the original (image) matrix after convolution:

k is usually odd

"same padding"

pad the input $\frac{k-1}{2}$ - zeros in each direction

e.g. $k=5$



convolution with $k \times k$ kernels

after padding:

$m \times n \longrightarrow (m+k-1) \times (n+k-1) \longrightarrow m \times n$

the same size

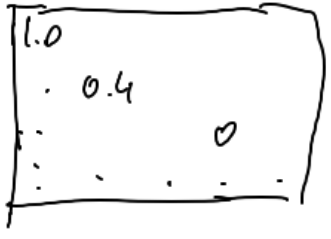
Another advantage of same padding: each input pixel influences ~~the same~~ a more similar number of outputs

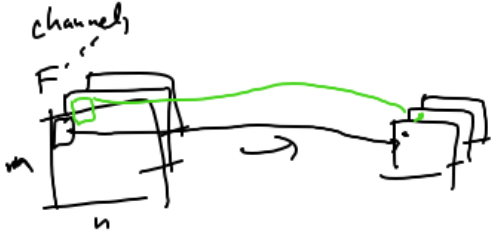
Pooling transforms an input matrix to a smaller matrix

no weights

Example: 10×10 matrix

5×5 output

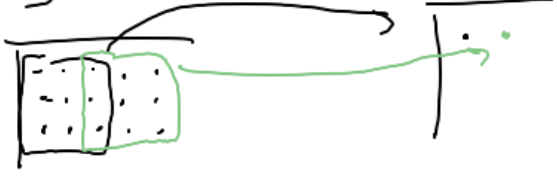




2×2 max pooling (without overlapping): split the input matrix into 2×2 squares, take maximum at each square as the output

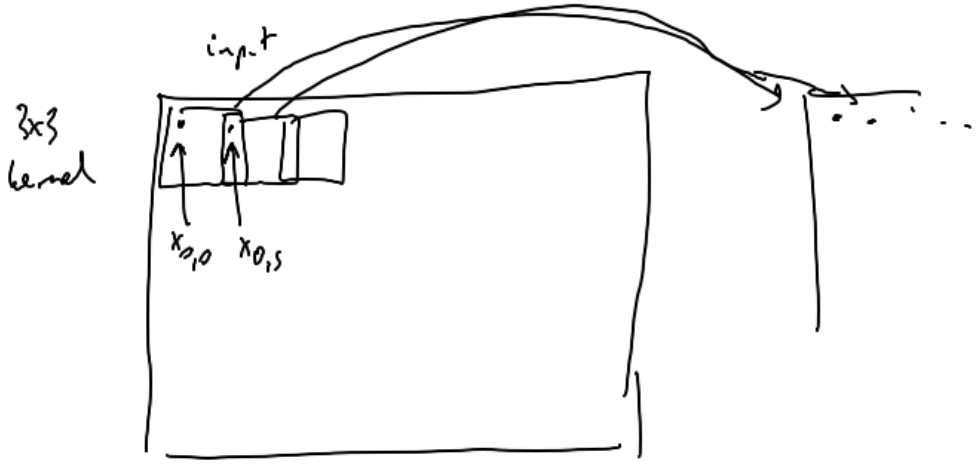
{ average pooling was used also

3×3 with overlapping



Strides

This means that we discard some outputs of the convolutional layer



stride = s means that we take for $p = 0, 1, \dots, \left\lfloor \frac{m-k}{s} \right\rfloor$, $q = 0, 1, \dots, \left\lfloor \frac{m-k}{s} \right\rfloor$:

$$net_{p,q} = \sum_{i=0}^{k-1} \sum_{j=0}^{k-1} \sum_{f=1}^{F} a_{i,j}^{(f)} \cdot x_{ps+i,\, qs+j}^{(f)}$$

previously: $s = 1$

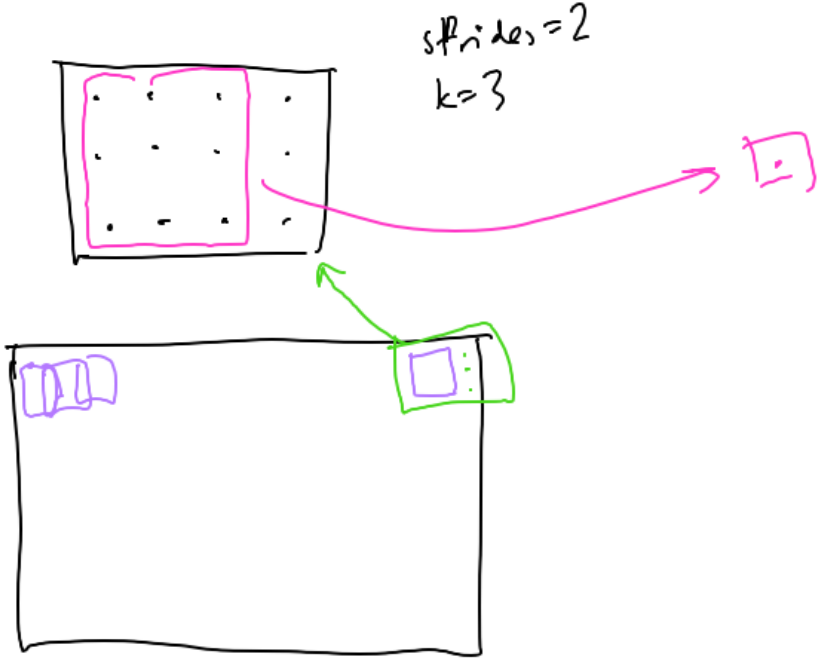
$ps + i \le m-1$

$ps + k-1 \le m-1$

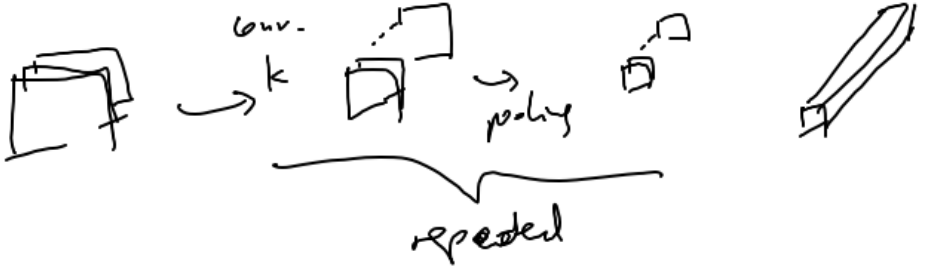
$ps \le m-k$

$p \le \frac{m-k}{s}$

Strides and padding should be chosen carefully to avoid loss of information:
(not necessarily a problem)



CNN:



at the end: usually fully connected layers