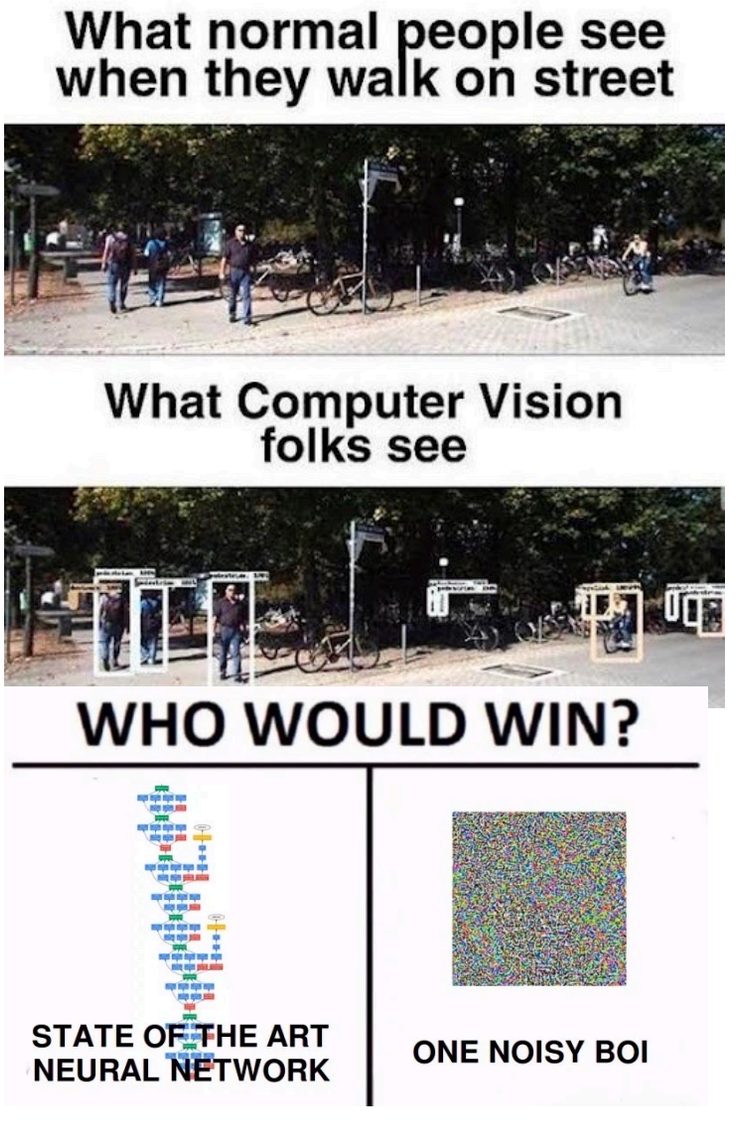
March 22, 2023

DSML: Computer Vision.

CNNs: Operations under the hood.

Class starts @ 9:05 pm.



Recap:

I] Operation

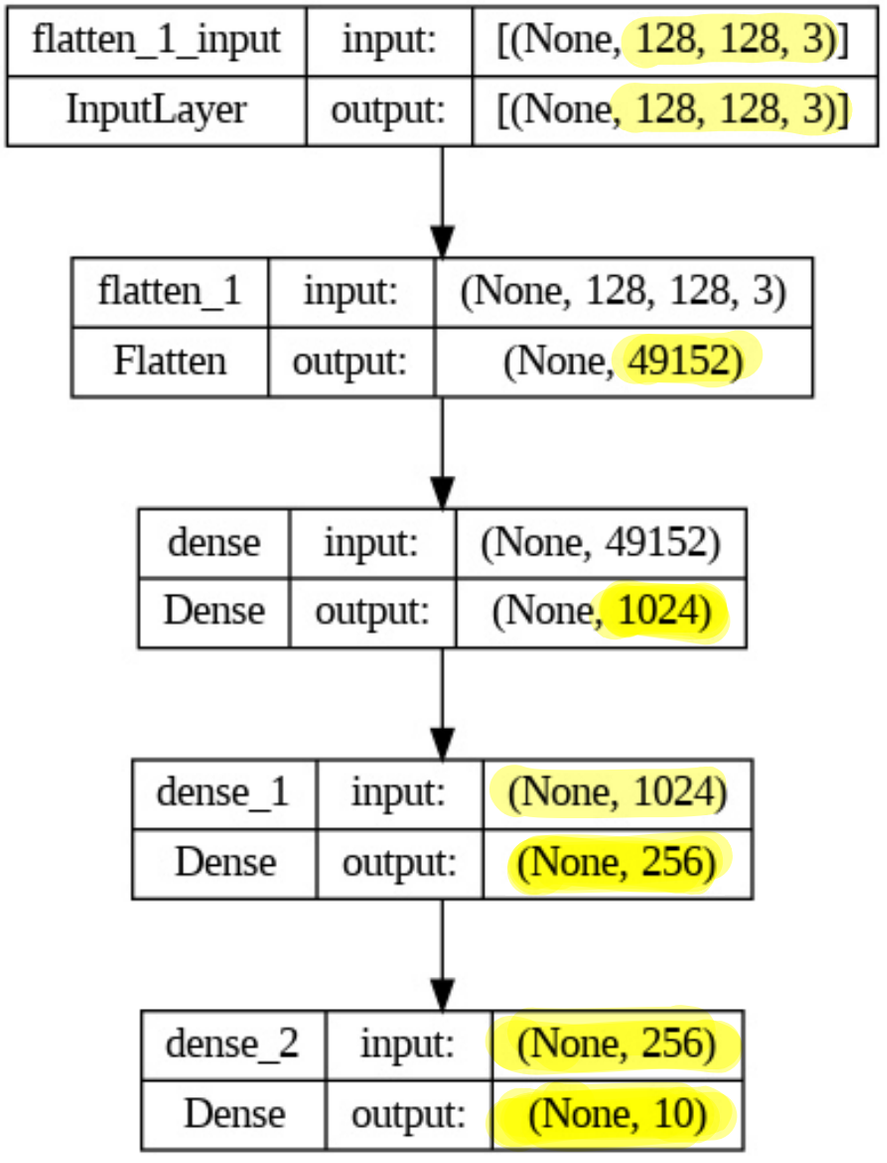
Operation	Input sizes: Kernel	Input sizes: Image	Output sizes
① Conv2D (valid)	$a \times b$	$m \times n$	$m-a+1 \times n-b+1$
② Conv2D (same)	$a \times b$	$m \times n$	$m \times n$
③ Conv2D (Full)	$a \times b$	$m \times n$	$m+a-1 \times n+b-1$
④ Conv2D (Strided)	$a \times b$	$m \times n$, $k \times k$	$\frac{m}{k} \times \frac{n}{k}$
⑤ Conv2D (Dilated)	$a \times b$	$m \times n$, $k \times k$	$m \times n$
⑥ MaxPool2D	$a \times b$	$m \times n$	$\frac{m}{a} \times \frac{n}{b}$
⑦ AvgPool2D	$a \times b$	$m \times n$	$\frac{m}{a} \times \frac{n}{b}$

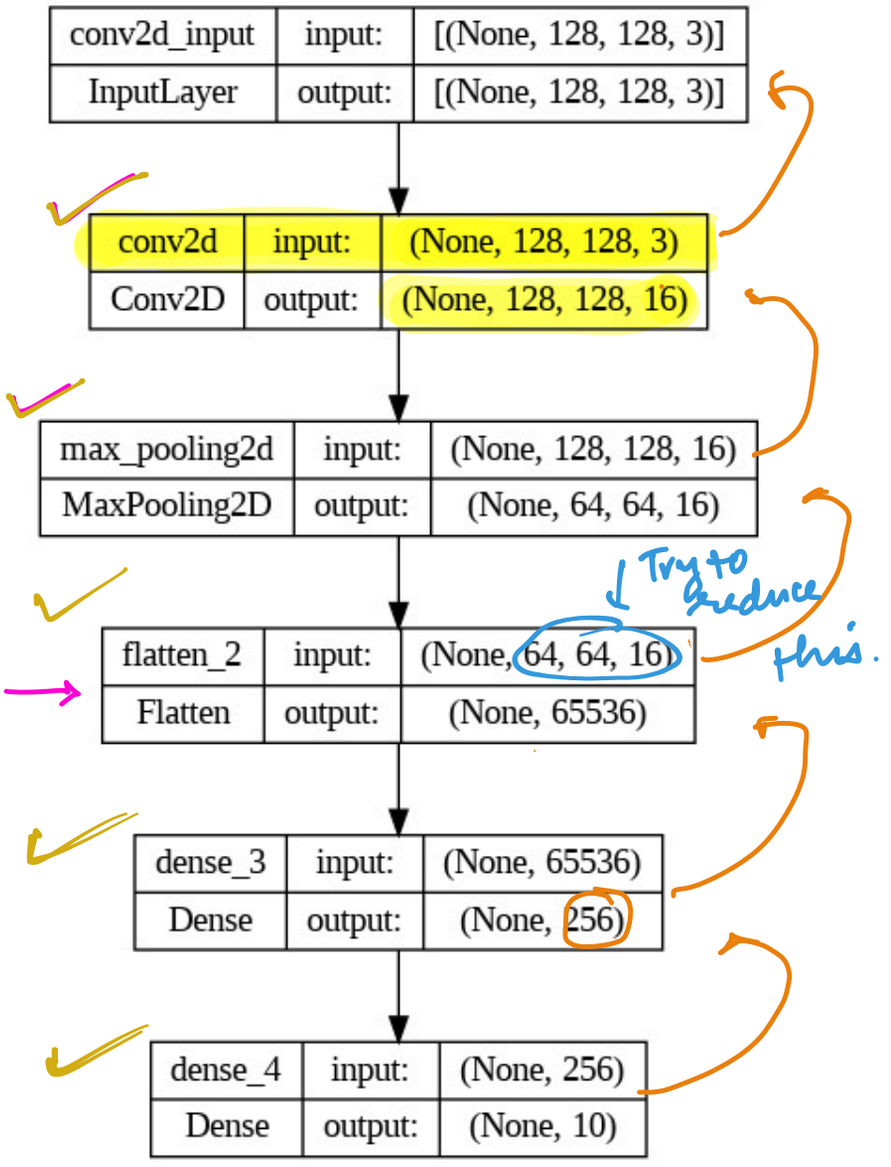
II] Apparel classification Case Study:

ANN : ≈ 40%

CNN : ≈ 90%

Architectures :





Overview: Major Components

Data: ✓

→ 128 × 128 × 3 Images.

→ 10 categories.

Model: ✓

Conv2D

↓

ReLU

↓

MaxPool

↓

Dense

Optimization: ✓

→ Adam.

→ categorical_crossentropy

→ Validation performance.

↓

CCE.

① Regularization (Dropout, Batchnorm)

② More data.

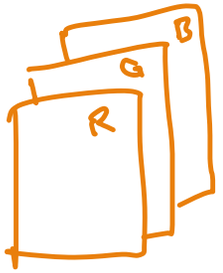
✓③ Reduce parameters. (decrease layers etc.

④ Changing opts, batch, learningrate etc, hyperparams; Early stopping.

The Conv2D block:

→ Please try implementing different conv2D operations from scratch.

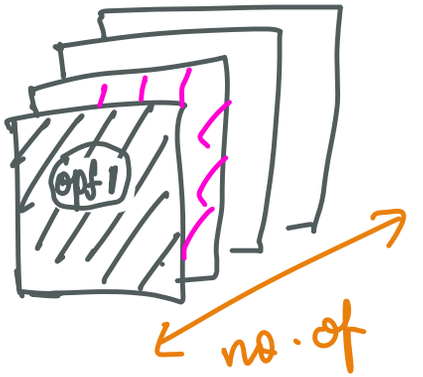
→ from scipy.signal import convolve2D



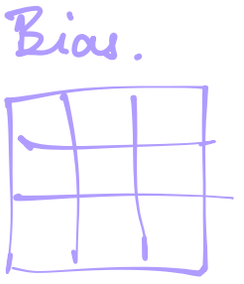
number of input channels. (m)

no. of output filters. (n)

3 × 3 × 1



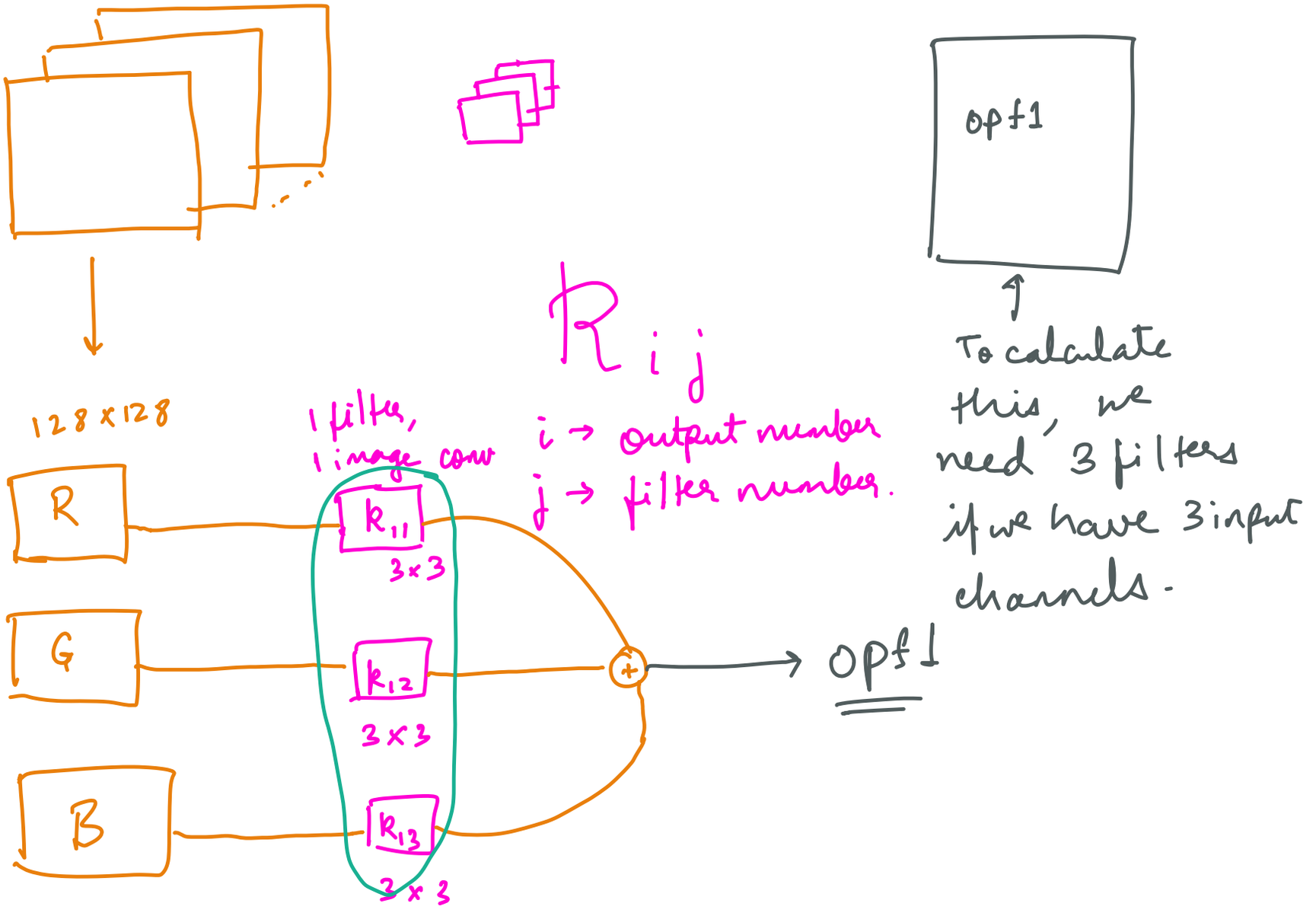
no. of o/p. channels.



Dilated convolution

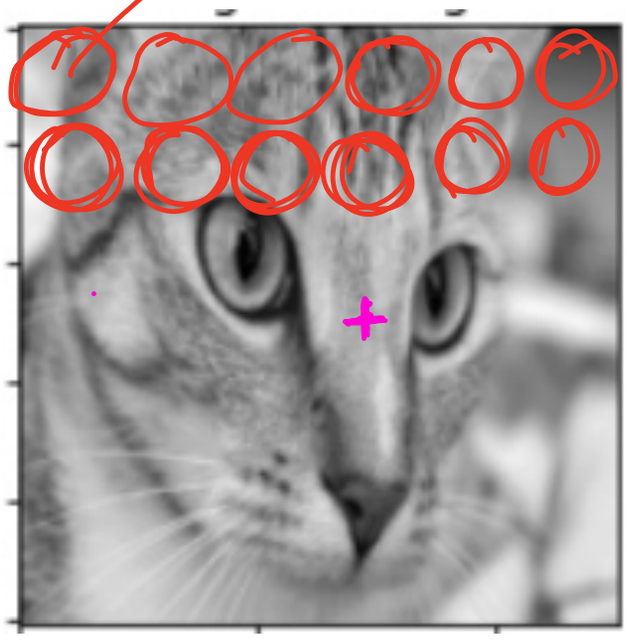
↓

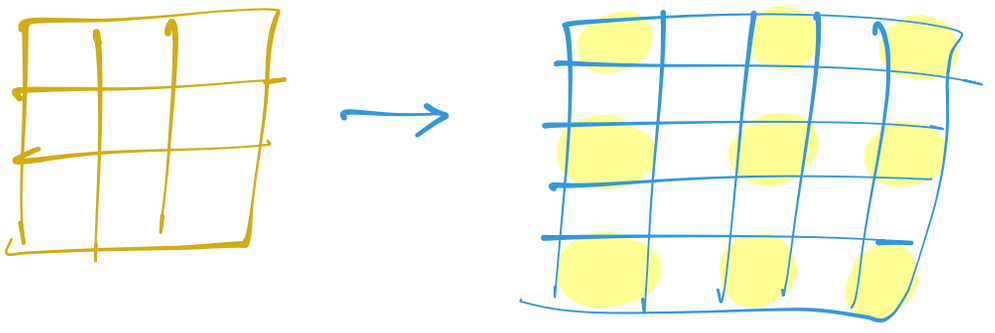
Increase the Receptive field size without increasing computational load.


opf1
To calculate this, we need 3 filters if we have 3 input channels.
128 x 128
1 filter, 1 image conv
R_{ij}
i → output number
j → filter number.
R
G
B
K_{11}
3×3
K_{12}
3×3
K_{13}
3×3
+
opf1

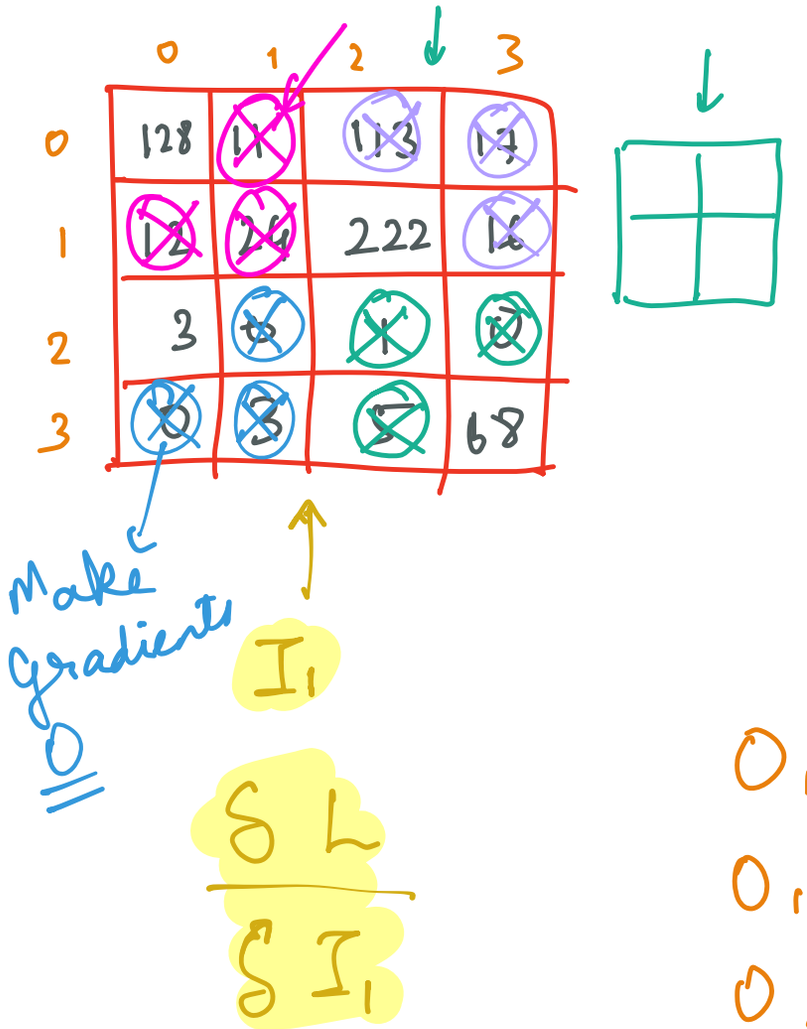
The small part of the image that the cell "sees" is called the receptive field in Neuroscience.

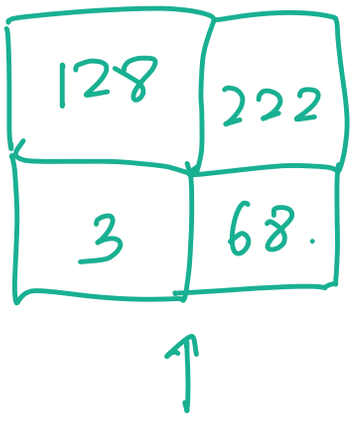
V1





Backprop for maxpool





O_1

$O_1[0,0] = I_1[0,0]$

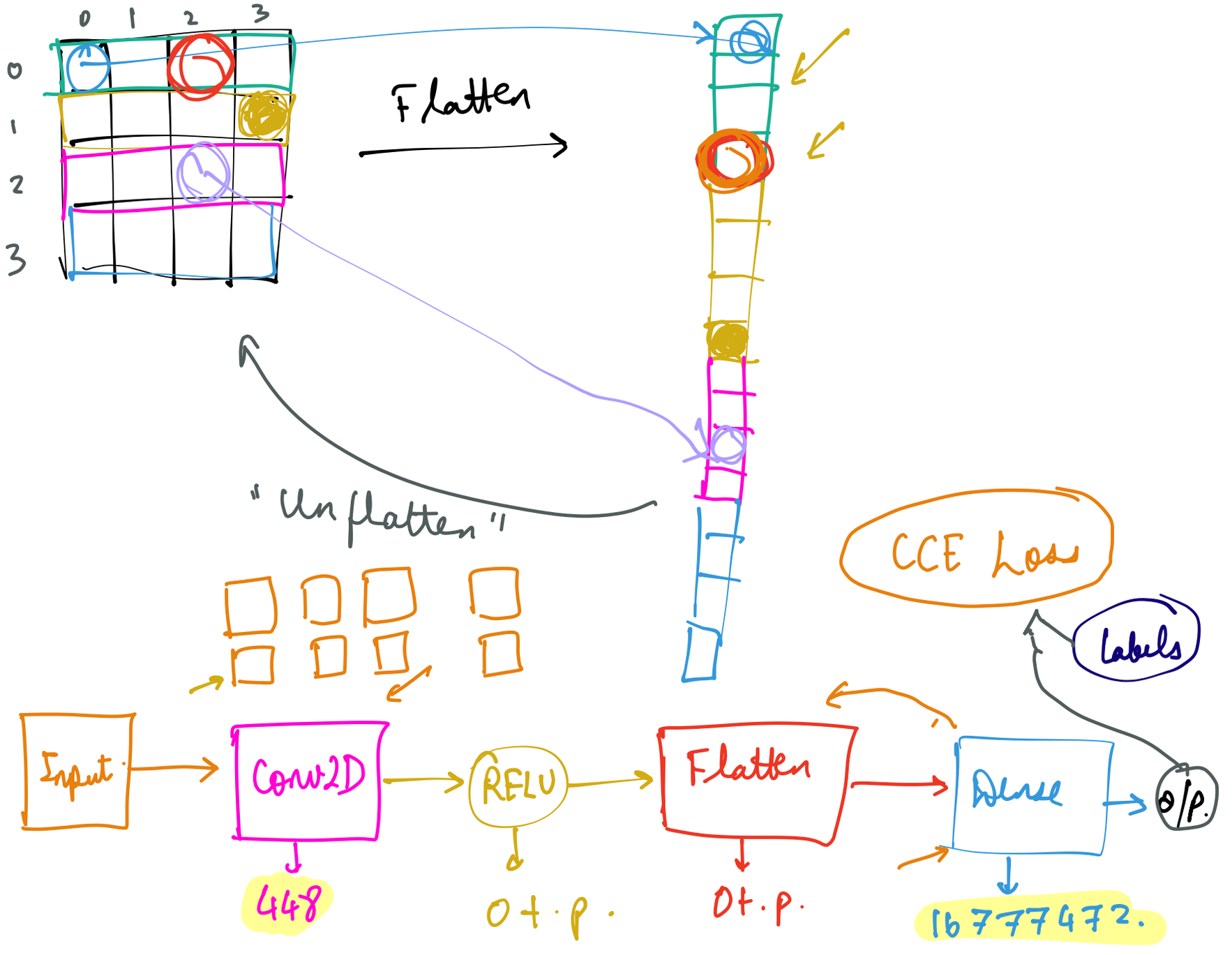
$O_1[0,1] = I_1[1,2]$

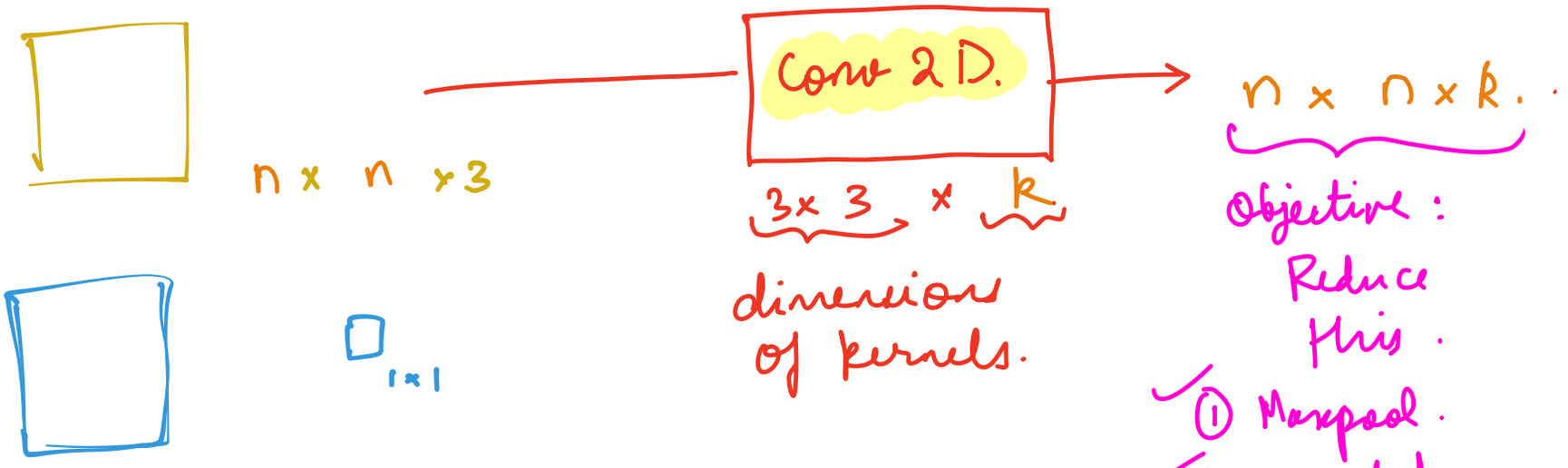
$O_1[1,0] = I_1[2,0]$

$O_1[1,1] = I_1[3,3]$.

$\frac{\delta L}{\delta I_1}$

For all indices of I_1 which are <u>not</u> picked, We assume the gradient contribution to be 0.


0 1 2 3
0
1
2
3
Flatten
"Unflatten"
CCE Loss
labels
Input
Conv2D
RELU
Flatten
Dense
o/p.
448
o/p.
o/p.
16777472.

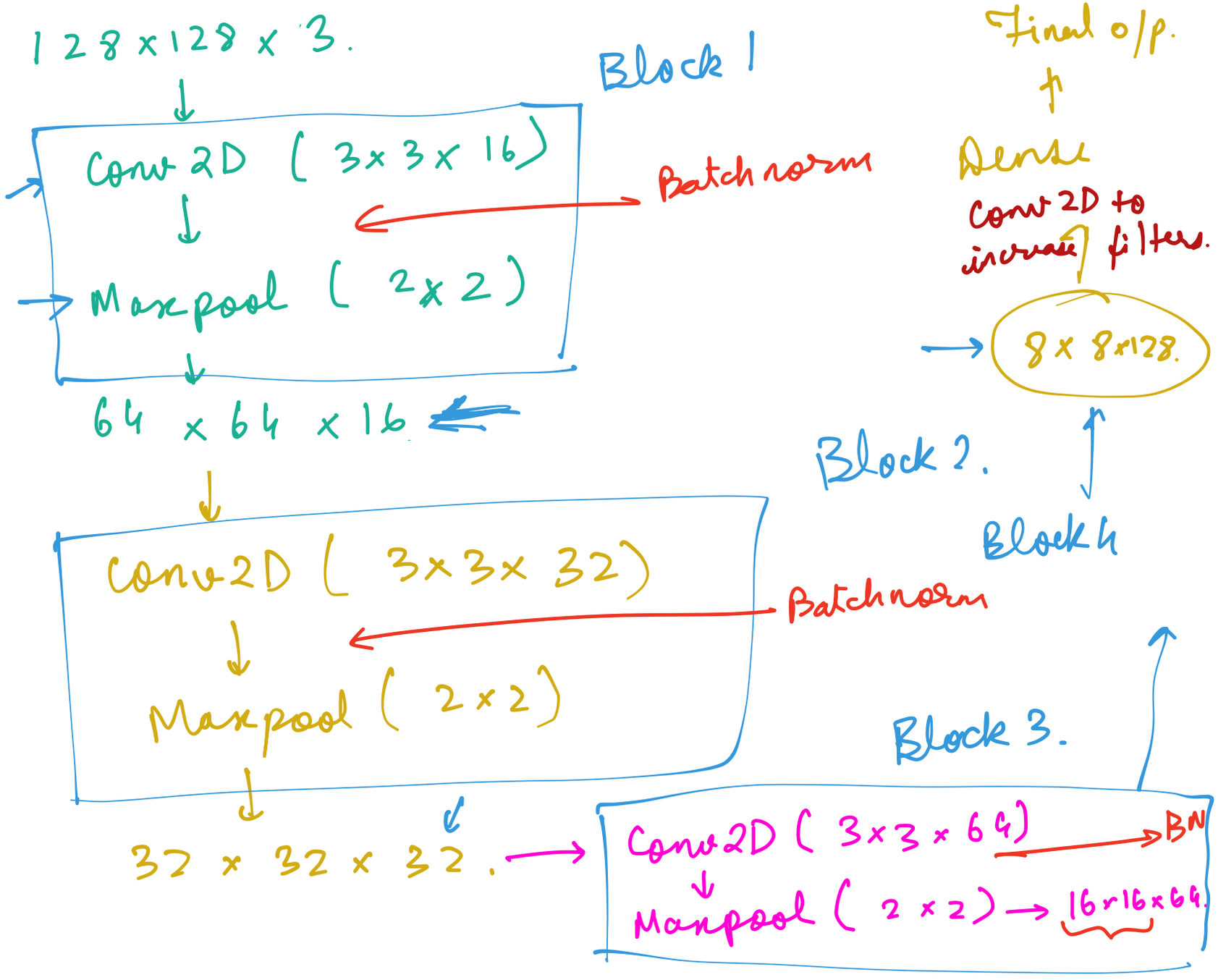


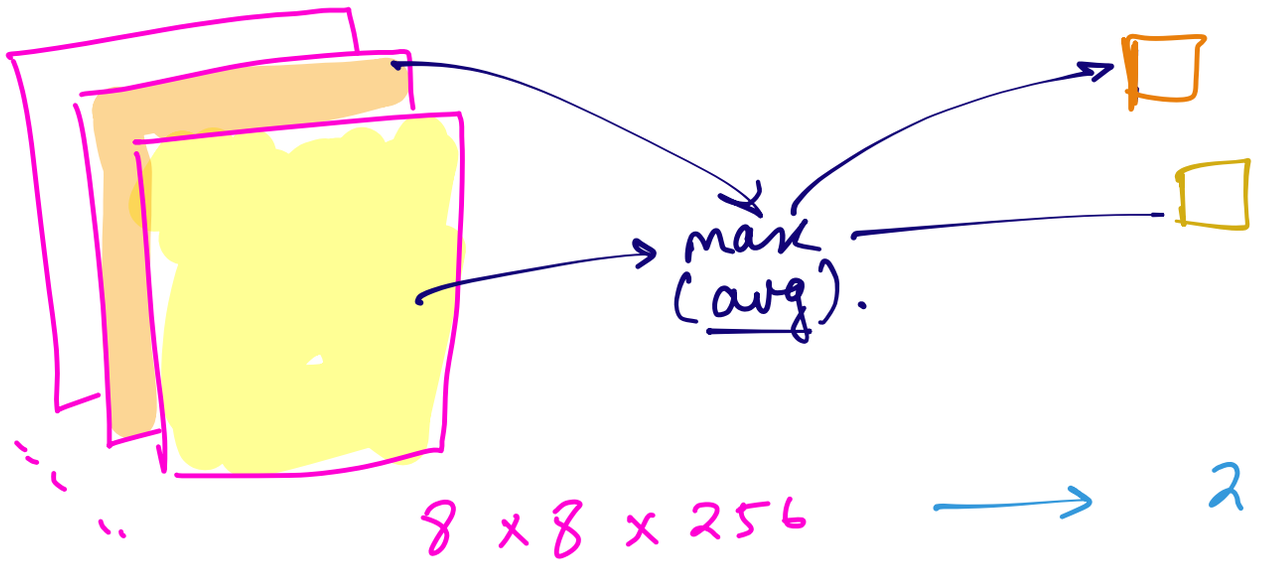
Solution 1 : Maxpool

$n \times n \times k \xrightarrow{\text{Maxpool2D } 2\times 2} \frac{n}{2} \times \frac{n}{2} \times k.$

Solution 2 : Strided conv.

$n \times n \times k \xrightarrow{\text{Conv2D } 1\times 1,\ \text{o.p. channels} \to k',\ \text{stride} : 2.} \frac{n}{2} \times \frac{n}{2} \times k'$


128 x 128 x 3.
Block 1
Conv 2D (3x3x16)
Batch norm
Maxpool (2x2)
64 x 64 x 16
Conv2D (3x3x32)
Batch norm
Maxpool (2x2)
32 x 32 x 32.
Block 3.
Conv2D (3x3x64)
BN
Maxpool (2x2) → 16x16x64.
Block 4
Block 2.
8 x 8x128.
Conv 2D to increase filters.
Dense
Final o/p.



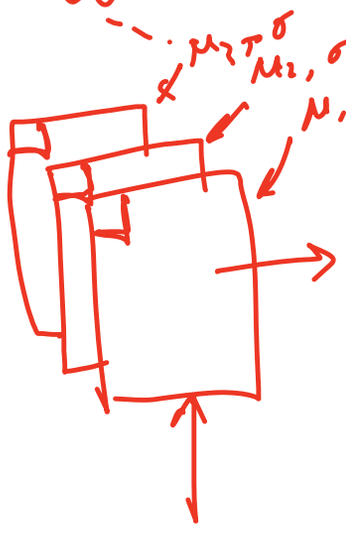
$8 \times 8 \times 256 \longrightarrow$ 256 dim vector.

Global Max. pooling

(Avg).

Batch norm

Conv2D → Relu → 128 × 128 × 16



μ_1, σ μ_2, σ μ_{16}, σ

μ

ith image

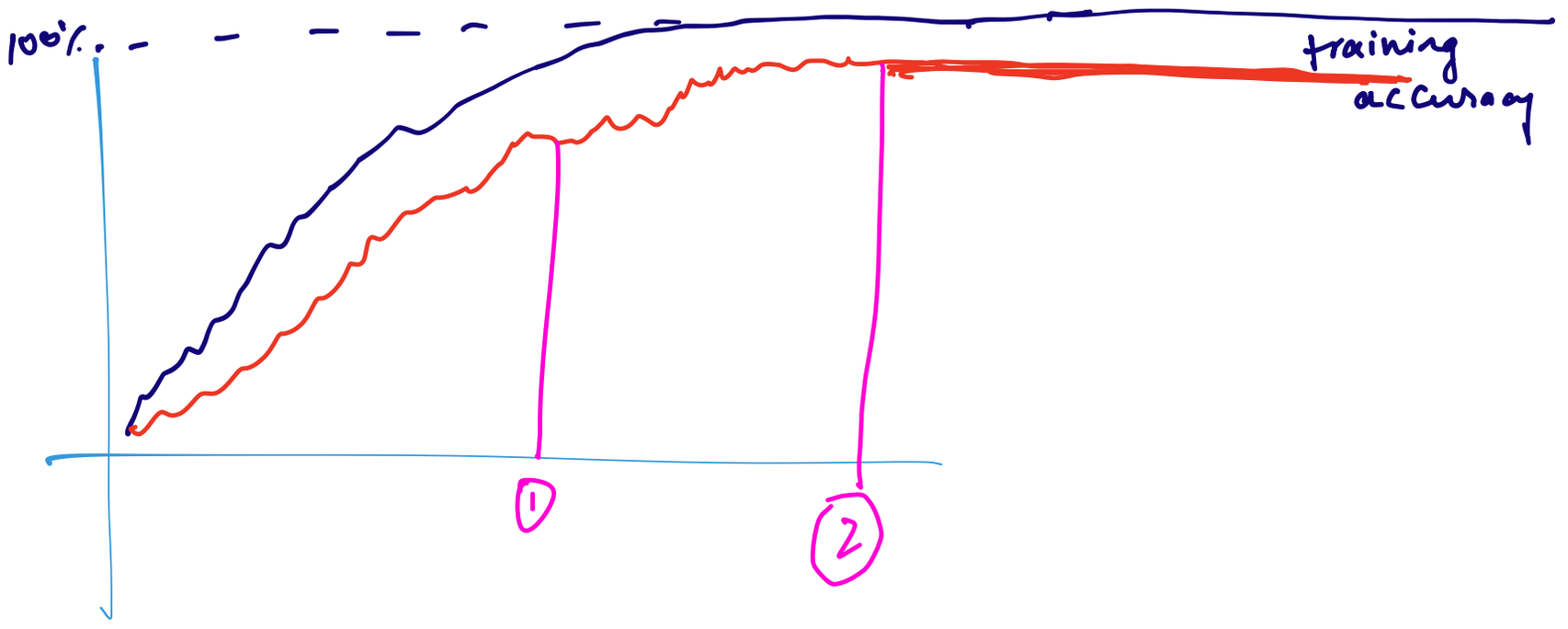
$i = 1, \ldots, 16.$

$$\frac{x_i - \mu}{\sigma}$$

$\mu = 0,$

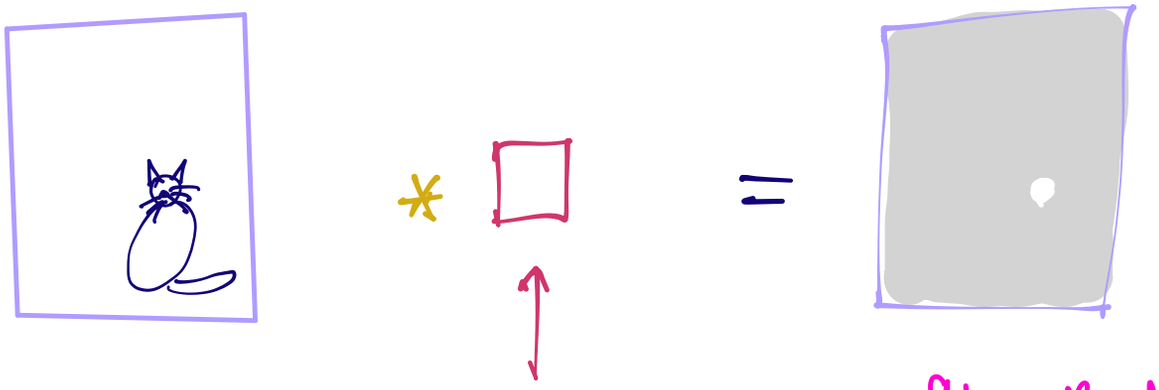
$\sigma = 1$

$[-3, 3]$

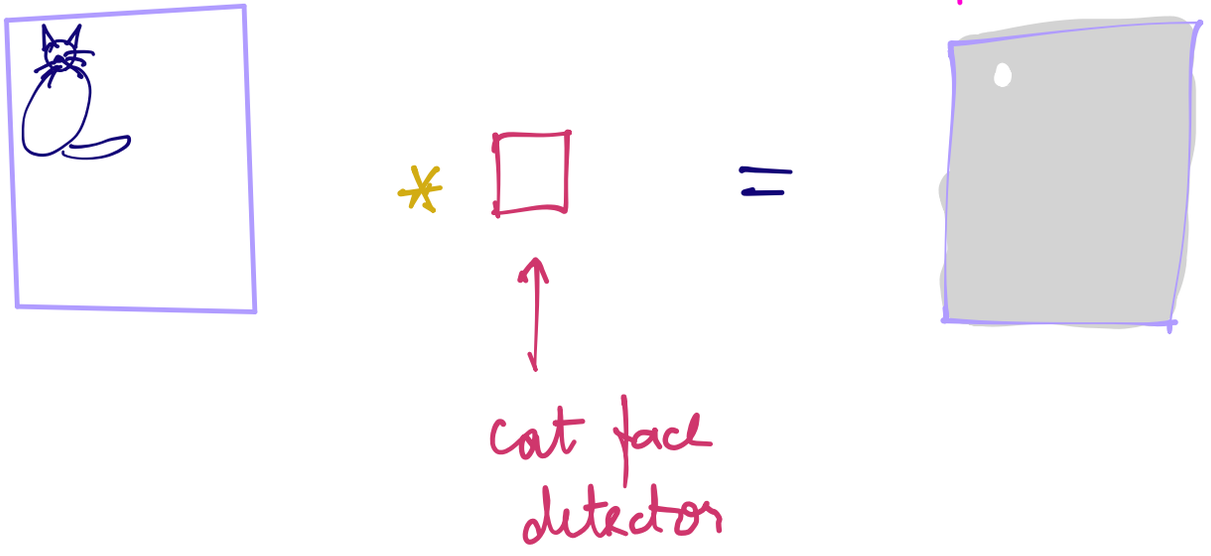


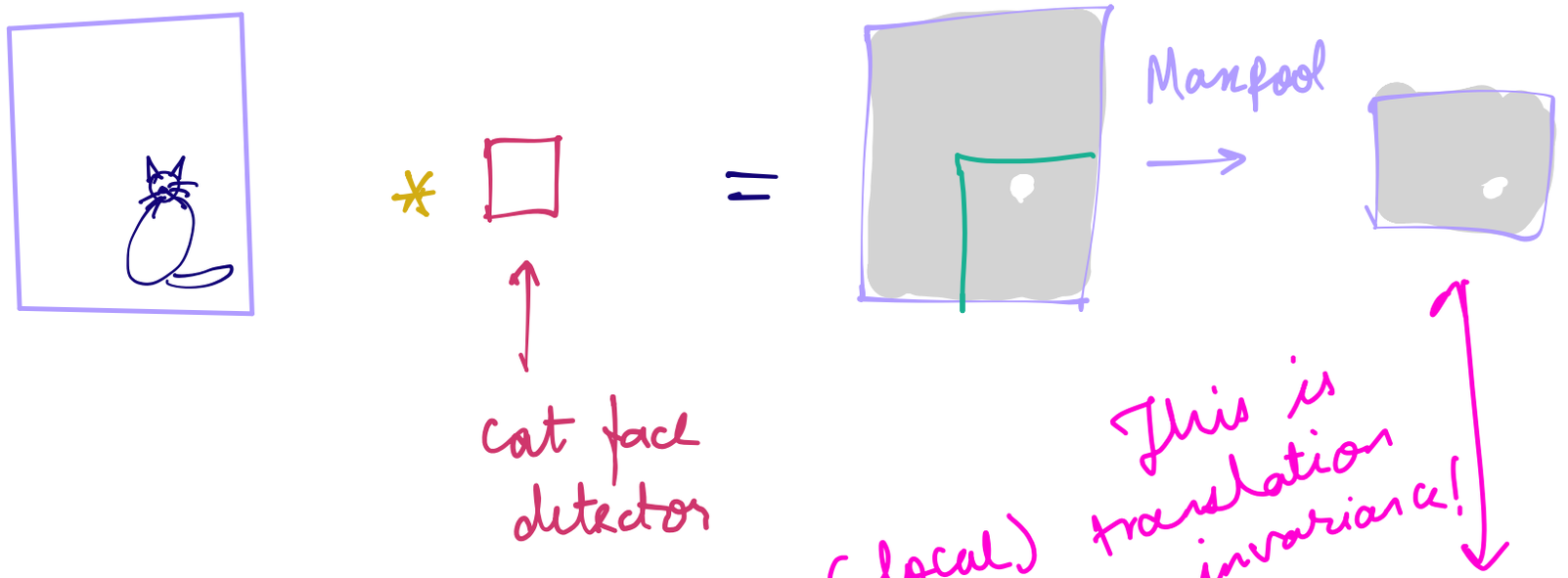
We will next try to stop training when the validation accuracy plateaus.

What is translation invariance.



If we shift the input, and see that the output also shifts, we say that the operation is covariant




Maxpool
cat face detector
This is translation invariance!
(local)
Maxpool
cat face detector

(Optional : Deriving gradients for a convolution).

Image $\begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix}_{3\times3}$ $*$ $\begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix}_{2\times2}$ kernel $= Y.$ $\begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{bmatrix}_{2\times2}$

$$y_{11} = k_{11}x_{11} + k_{12}x_{12} + k_{21}x_{21} + k_{22}x_{22}$$
$$y_{12} = k_{11}x_{12} + k_{12}x_{13} + k_{21}x_{22} + k_{22}x_{23}$$
$$y_{21} = k_{11}x_{21} + k_{12}x_{22} + k_{21}x_{31} + k_{22}x_{32}$$
$$y_{22} = k_{11}x_{22} + k_{12}x_{23} + k_{21}x_{32} + k_{22}x_{33}.$$

$\frac{\partial L}{\partial k_{11}}$ → what we want.

$= \left(\frac{\partial L}{\partial Y}\right)$ ← this is a matrix. $\cdot \frac{\partial Y}{\partial k_{11}}$ → $\begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{bmatrix}$.