Components of CNNs
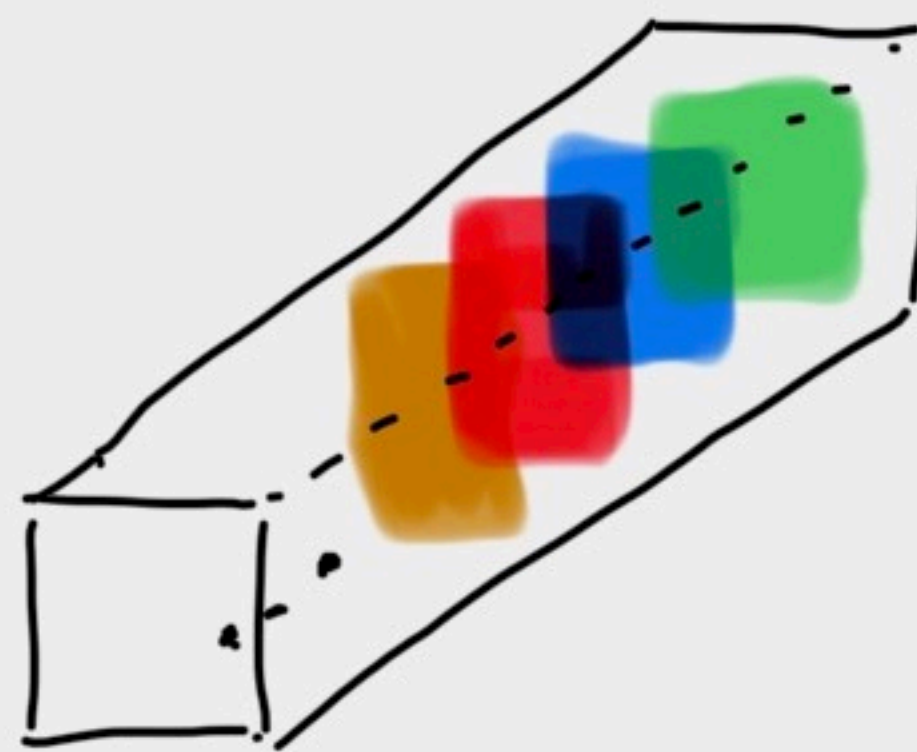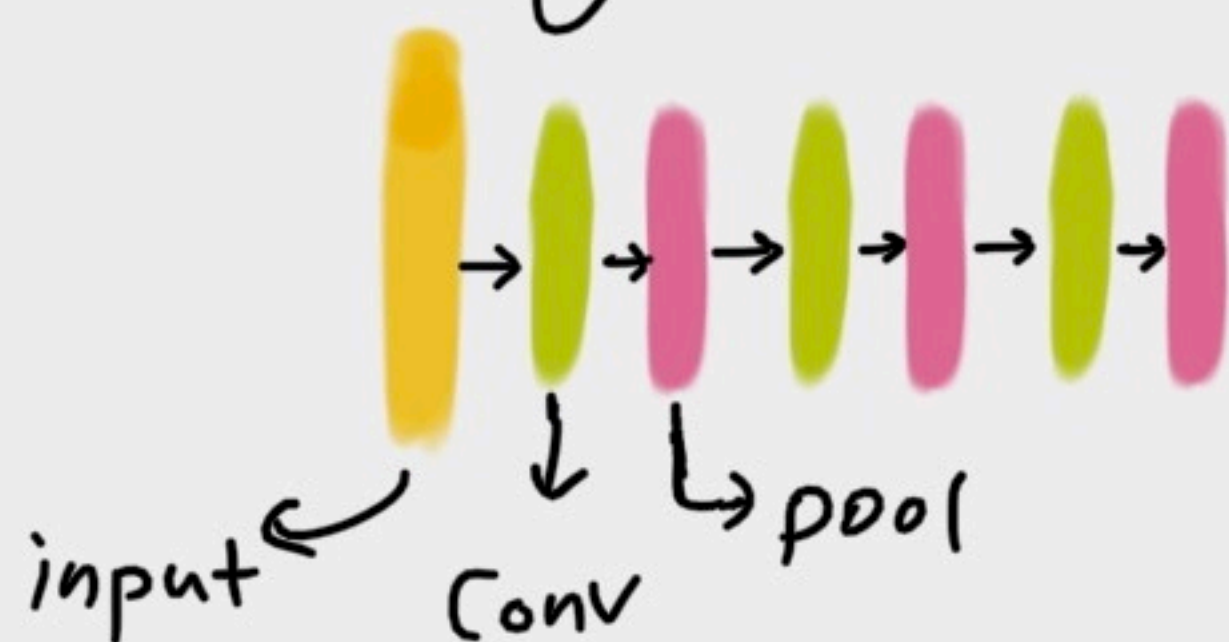
1. number of filters
2. size of each filter
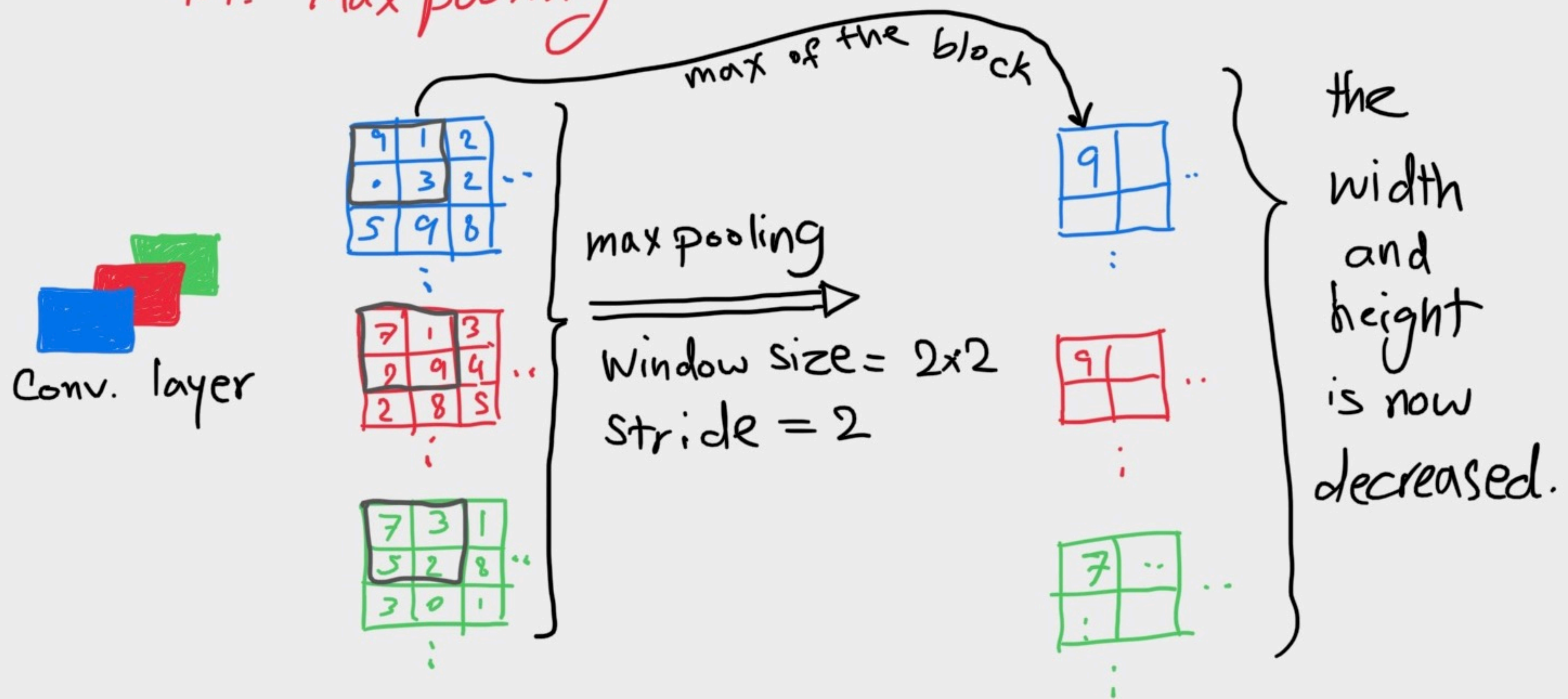3. stride → when sliding, we choose how much we slide at each step.
   when stride $= 1$, the filtered image will have the same size as the input image. The size of the filtered image decreases when we increase the stride.
4. pooling layers

input ⟵ Conv ↓ ↳ pool

Convolutional layers can get too large, i.e., high-param. We use pooling to deal with that.

max of the block

9	1	2
.	3	2
5	9	8

7	1	3
2	9	4
2	8	5

7	3	1
5	2	8
3	0	1

Conv. layer

max pooling $\Longrightarrow$

Window size = 2×2
Stride = 2

| 9 | |
| | |

| 9 | |
| | |

| 7 | |
| | |

the width and height is now decreased.
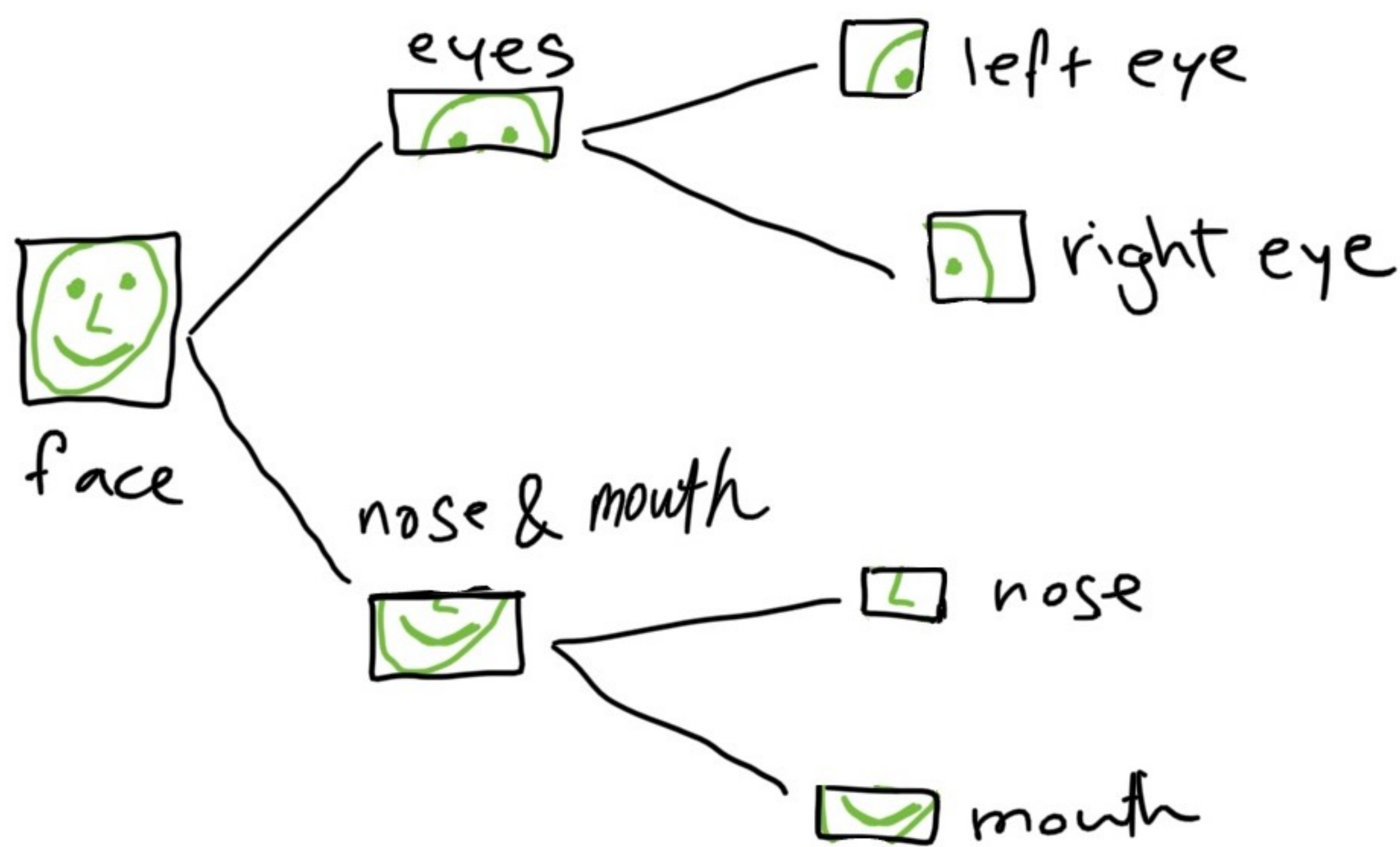
4.2. Average pooling

not usually for edge detection
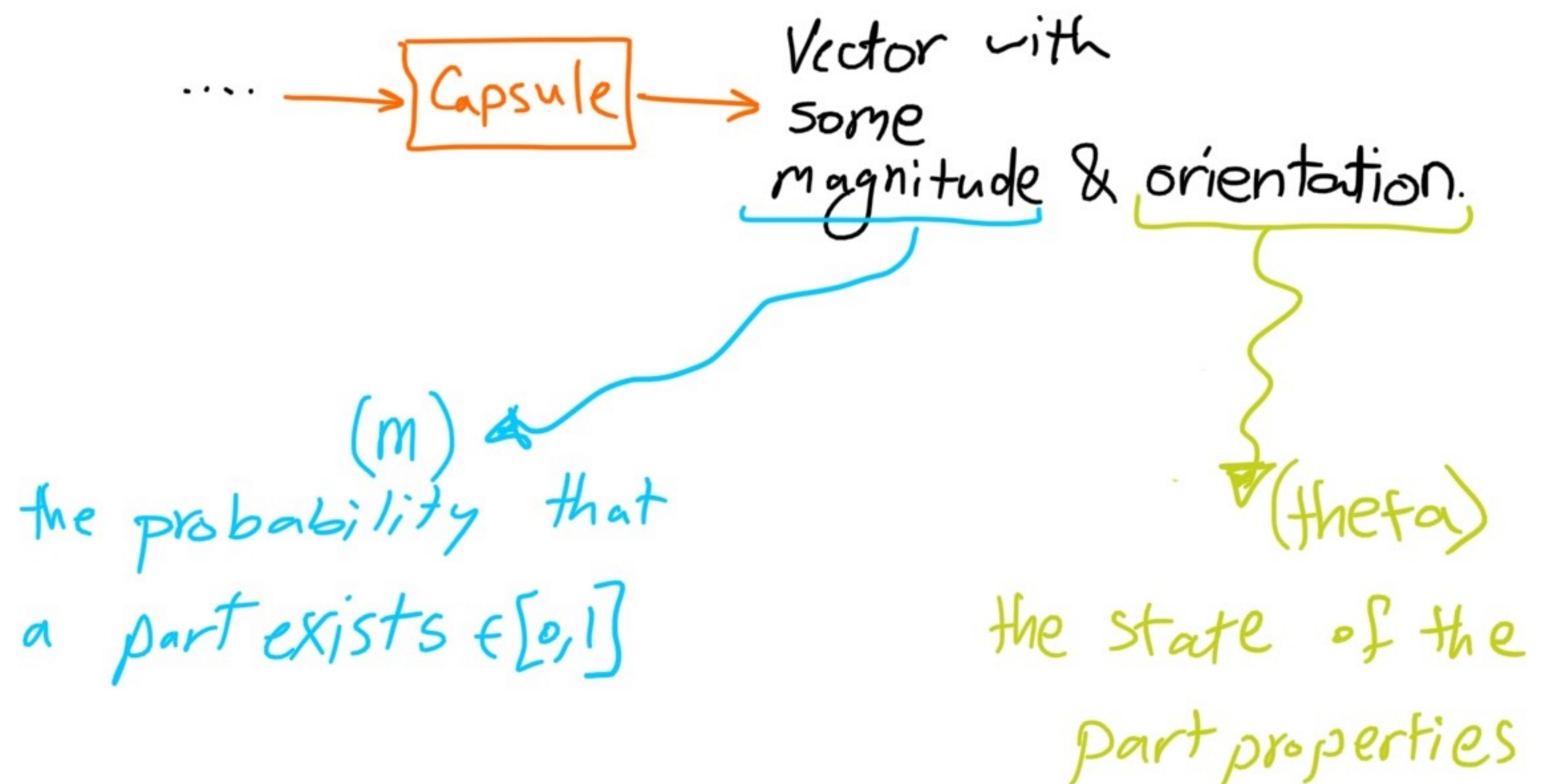
4.3. Alternatives to pooling:

Pooling throws away some info, i.e., discards some pixels.
In classification, pooling is fine.
But in something like face recognition it is bad.
Instead, some people use strategies to keep some spatial info. An example is Capsule Network, which learns spatial relationships between parts.

Capsule: Collection of nodes that Contain info about a sub-part of the image (width, orientation, color, ...)

.... → Capsule → Vector with some magnitude & orientation.

(m) ← the probability that a part exists $\in [0,1]$

(theta) ← the state of the part properties

The output of these Capsules Can be used to make a "tree".

Capusle

$m = 0.8$

θ_1

Capusle

$m = 0.8$

θ_2