Convolutional Neural Networks

Banyak diambil dari MIT Introduction to Deep Learning

Lecture 3 Deep Computer Vision

http://introtodeeplearning.com/

dan

Brandon Rohrer. How convolutional neural networks work

https://github.com/brohrer/public-hosting/blob/master/how_CNNs_work.pptx?raw=true

Kerangka Bahasan

1. Pengantar
2. Pembelajaran Fitur Visual
3. Ekstraksi Fitur dan Convolution
4. CNN
5. Sebuah Arsitektur untuk Banyak Aplikasi
6. Ringkasan

Arsitektur Dasar Feed Forward Network

- Convolutional Neural Network biasanya disingkat CovNet atau CNN
- Ada yang menyebut sebagai “scanning for patterns/objects”
- Kita akan lihat bahwa proses utama pada CNN yaitu “pelilitan” (convolutioning) ini adalah juga proses “pemindaian” (scanning).
- CNN merupakan sebuah feed forward nework (FFN), sebagaimana MLP.
- Dalam sebuah FFN, ada tiga bagian utama yaotu
  - Sebuah input layer
  - Satu atau lebih hidden layer
  - Sebuah output layer

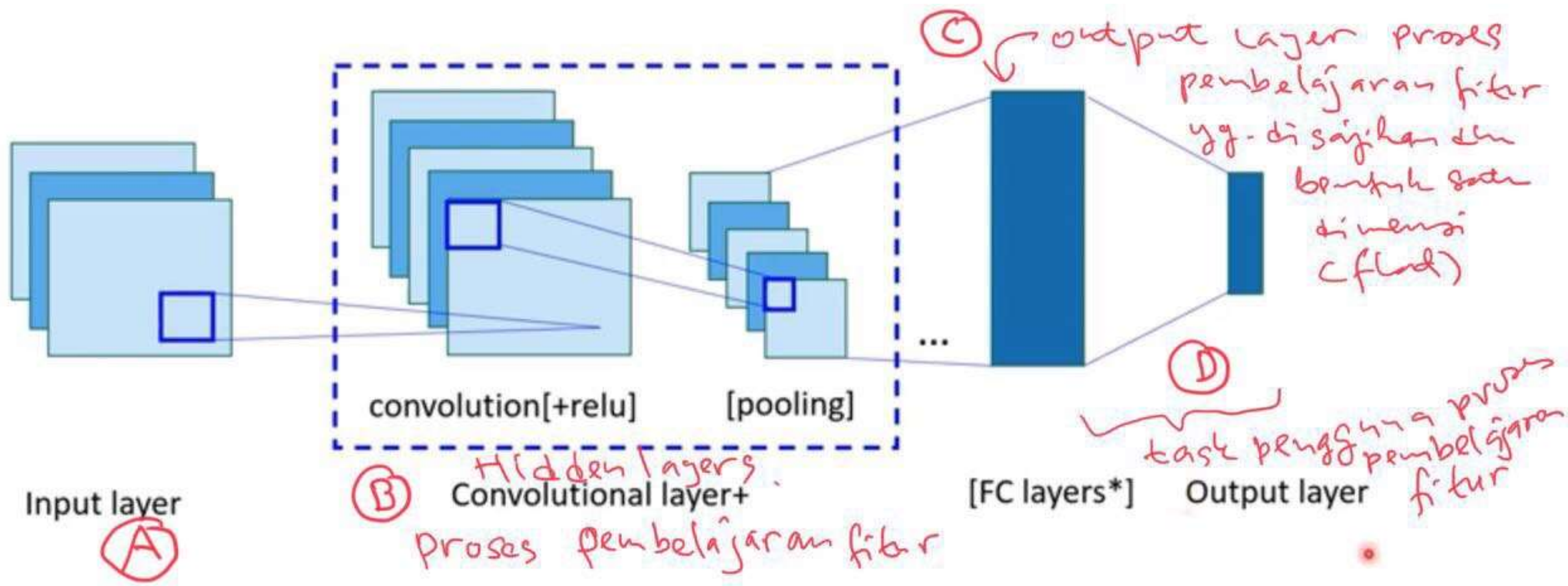
Arsitektur Dasar Feed Forward Network (cont)

- Dimana komputasi terjadi pada layer yang “dihubungkan” (connected) dengan layer sebelumnya. Hubungan (connection) itu artinya adalah adanya parameter (berupa matriks bobot).
- Komputasi terjadi pada hidden layer.
- Proses utama FDD adalah ada hidden layer, yang merupakan pembelajaran fitur (feature learning).
- Artinya model yang dihasilkan akan mentransformasikan setiap vektor fitur asli dari obyek menjadi

Arsitektur Dasar Feed Forward Network (cont)

- Shg pada prinsipnya dapat dibagi menjadi 5 bagian
  - A: input
  - B: pembelajaran fitur (dalam hidden layers)
  - C: output pembelajaran fitur
  - D: task pengguna hasil proses pembelajaran fitur. Untuk image misalnya bisa klasifikasi, segmentasi, atau deteksi obyek.

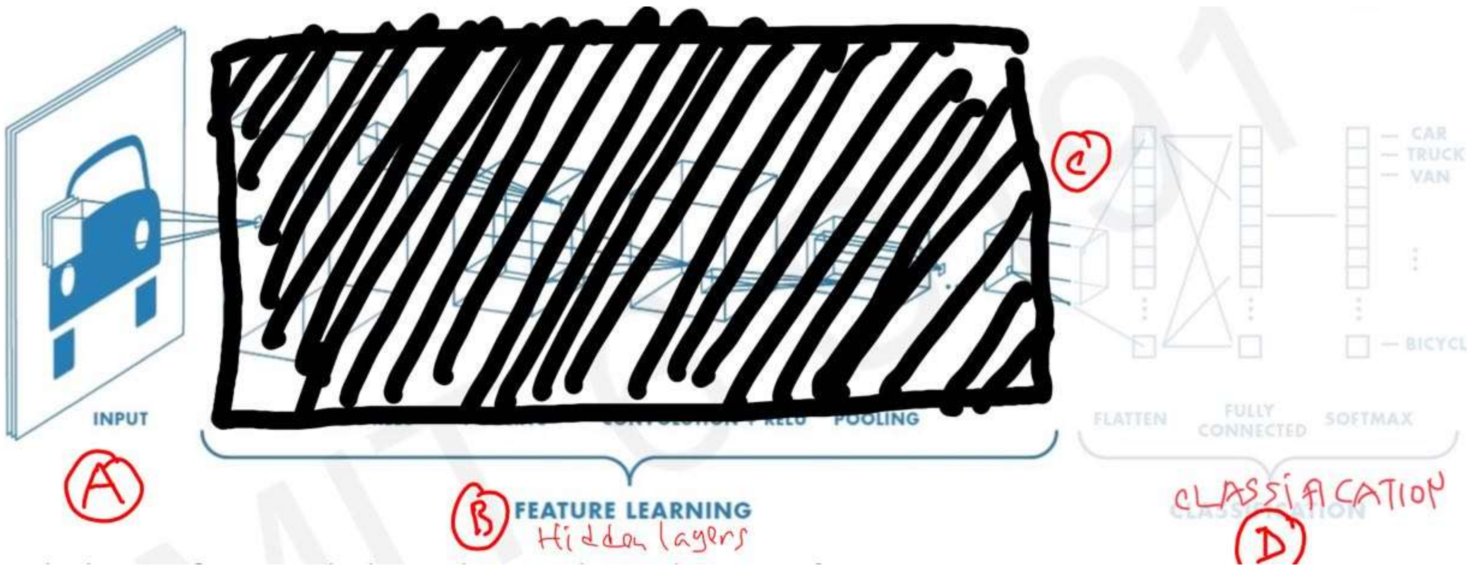
Arsitektur Dasar Feed Forward Network (cont)



Masayu Laylia Khodra. CNN: Architecture
https://youtu.be/Nma7D8hnZdg
Corat coret oleh MAB

Arsitektur Dasar Feed Forward Network (cont)

- Dengan gambar lain. Bagian pembelajaran fitur dibuat “black box”, krn pada bagian Pengantar ini kita blm melihat detilnya,.



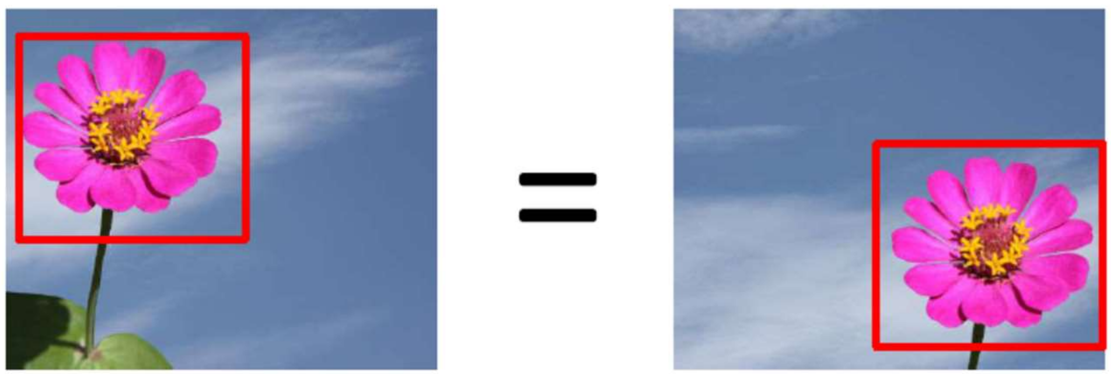
Gambar dari MIT. Introduction to Deep Learning: Deep Computer Vision)
Corat coret oleh MAB

Pembelajaran Fitur

- Pembelajaran fitur merupakan proses inti. Kita kosentrasi ke proses tsb.
- Sebuah proses perlu memperhatikan dua hal:
  - efektivitas
  - efisiensi
- Pada proses pembelajaran fitur
  - Efektif: menghasilkan vektor fitur yang mirip untuk obyek yang mirip, meski ada variasi, al. dengan mengandung informasi spatial.
  - Efisien: meminimalisir jumlah parameter, al. dengan parameter sharing
- Pemakaian CNN terutama untuk image yang merupakan obyek dua dimensi.

Spatial Invariant

- Perlu “shift/position invariant”, yaitu meski lokasi gambar tidak sama keduanya tetap dikenali sebagai gambar bunga.

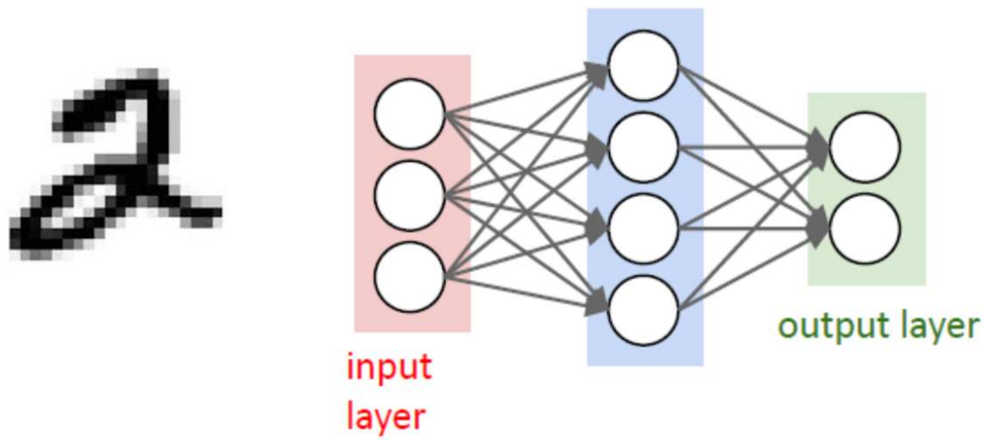


- MLP sensitif terhadap lokasi pola/obyek

Gambar dari CMU. Introduction to Deep larning:

Spatial Invariant (cont)

- MLP bisa mngenalo input vektor seperti angka tulisan tangan tsb, atau *vector input* secara umum. Namun tidak shift invariant.

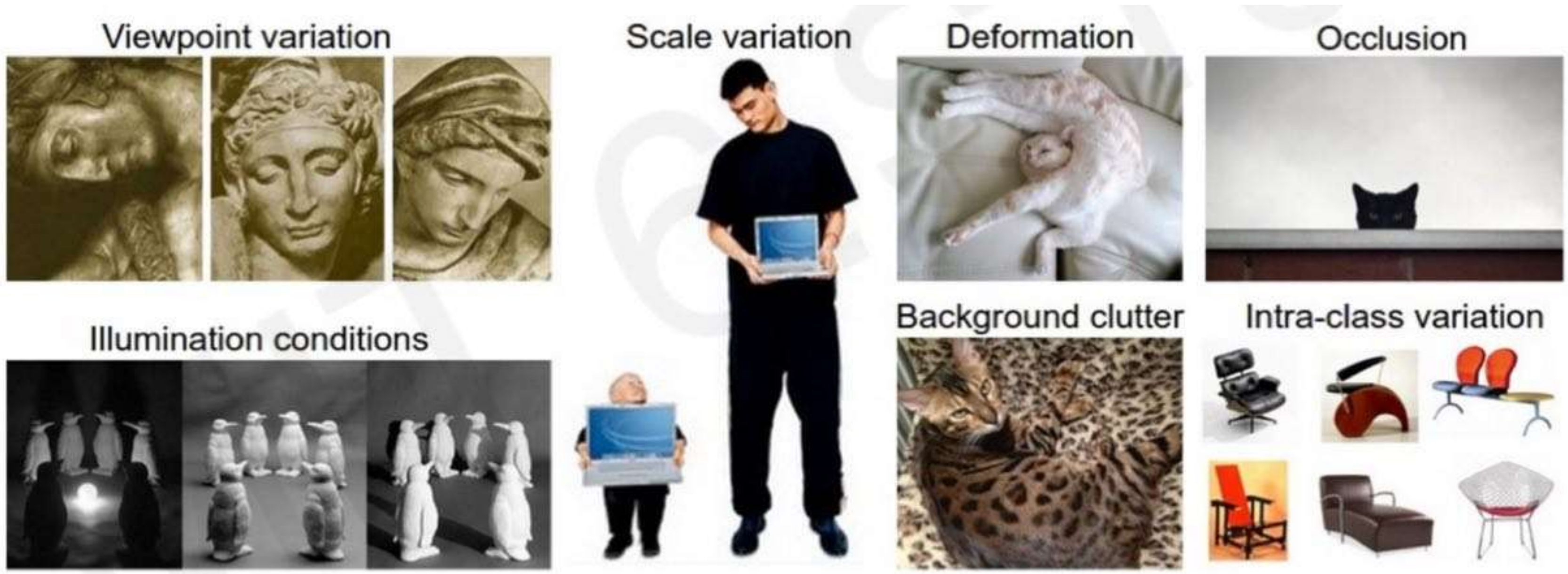


Spatial Invariant (cont)

- MNIST dataset: dataset tulisan tangan angka / digit 0 – 9. Tidak ada shift invariant.

Spatial Invariant (cont)

- Beberapa variasi lain



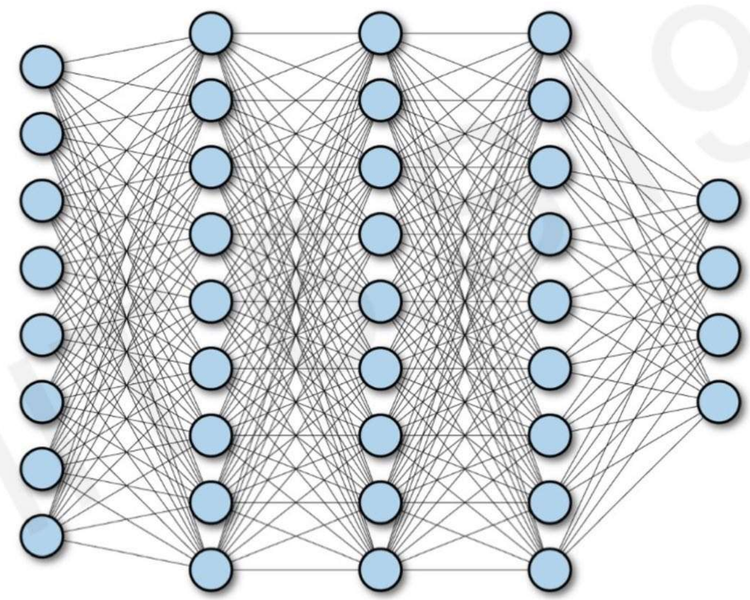
Li/Johnson/yeung CS231n

Kerangka Bahasan

1. Pengantar
2. Pembelajaran Fitur Visual
3. Ekstraksi Fitur dan Convolution: Contoh Sederhana
4. CNN
5. Sebuah Arsitektur untuk Banyak Aplikasi
6. Ringkasan

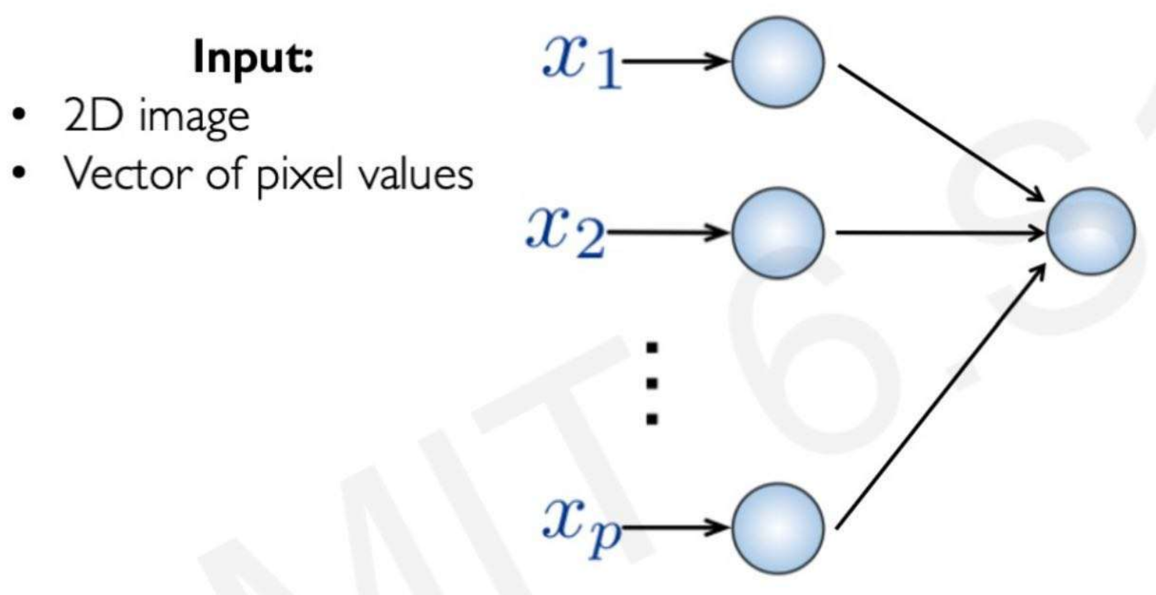
MLP tdk Efektif unt Obyek dg Info Spatial

- Pada jaringan yang fully conectted spt MLP
- Untuk input berupa image 2D
  - Informasi spatial hilang, karena flattening menjadi vektor 1D pada input layer (hasil kurang efektif).
  - Parameter sangat sangat banyak (proses kurang efisein, a.l. lambat)
- Bagaimana cara memasukkan informasi spatial?



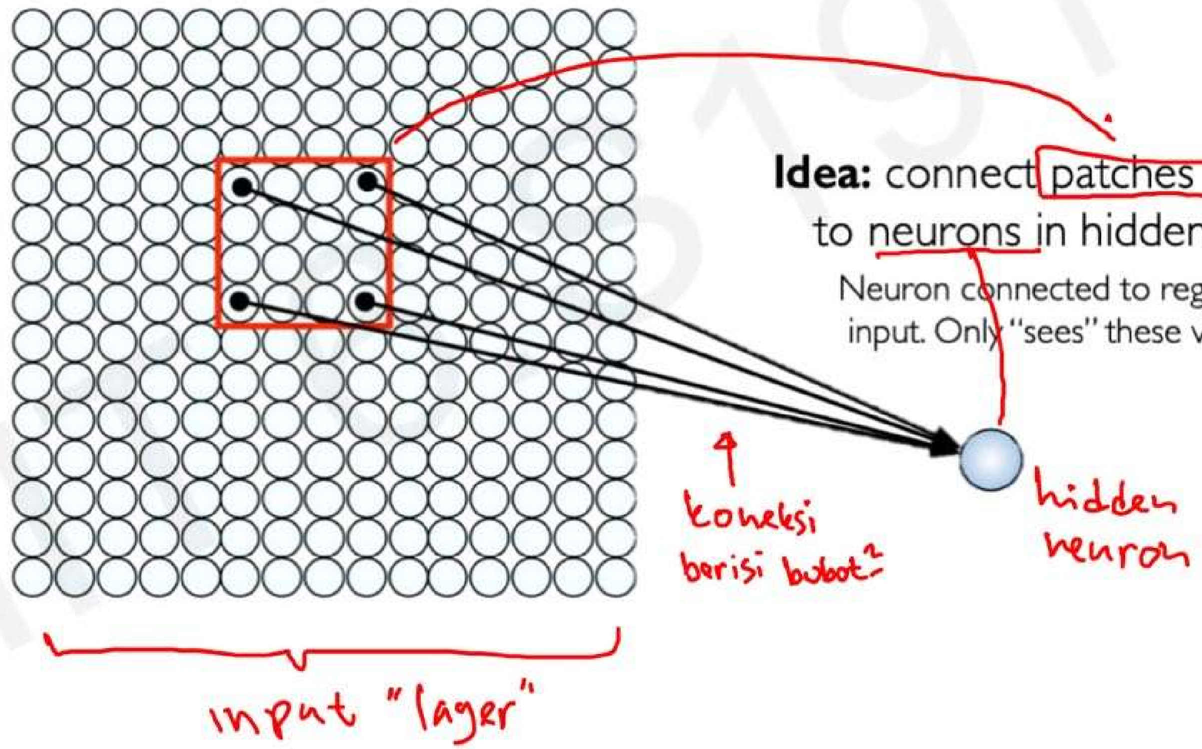
MLP tdk Efektif unt Obyek dg Info Spatial (cont)

- MLP



Menggunakan Struktur Spatial

Input: 2D image.
Array of pixel values

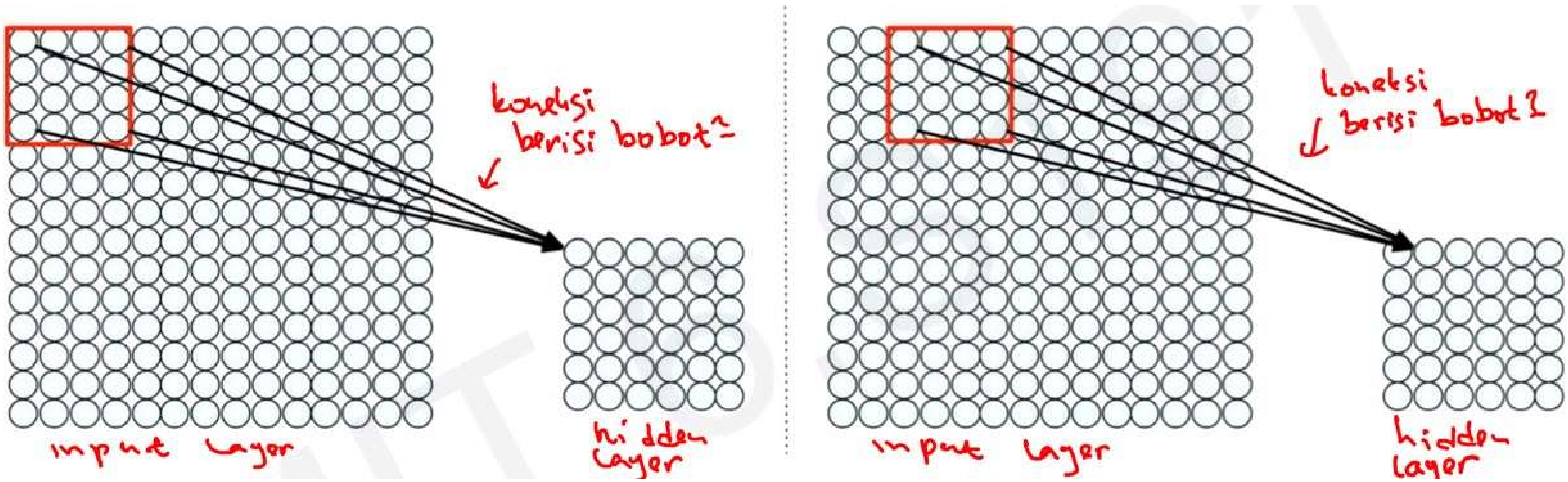


Idea: connect patches of input
to neurons in hidden layer.

Neuron connected to region of input. Only "sees" these values.

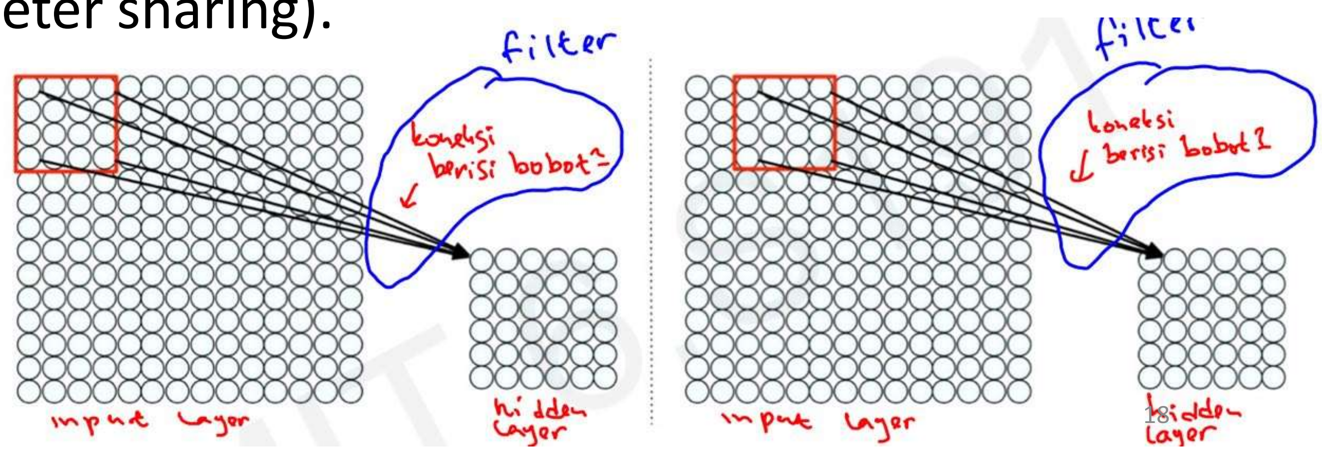
Menggunakan Struktur Spatial (cont)

- Proses scanning. Potongan/patch di input layer digeser ke kanan.

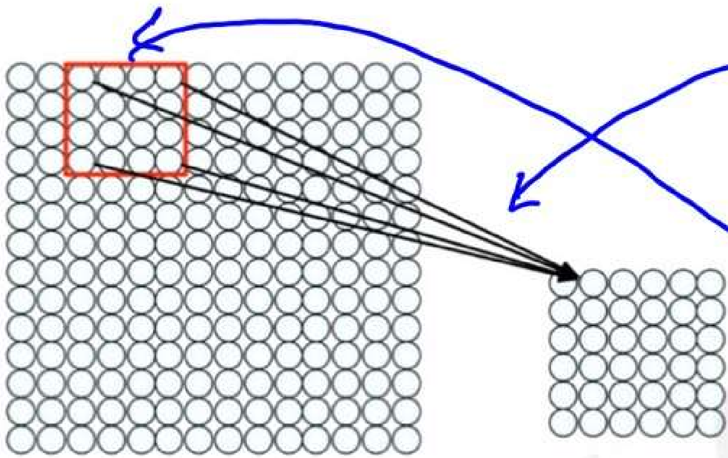


Menggunakan Filter untuk Mengekstrak Fitur

1. Menggunakan filter (yaitu sehimpunan bobot) untuk mengekstrak fitur lokal.
2. Gunakan berbagai filter (himpunan bobot) yg berbeda, unt menghasilkan berbagai kombinasi hiden layer.
3. Dalam satu siklus scanning, potongan digeser (scanning/convolution) menggunakan himpunan bobot (filter) yg nilainya sama (parameter sharing).



Ekstraksi Fitur dengan Convolution/Scanning



- Filter of size 4x4 : 16 different weights
- Apply this same filter to 4x4 patches in input
- Shift by 2 pixels for next patch

This "patchy" operation is **convolution**

1) Apply a set of weights – a filter – to extract **local features**

2) Use **multiple filters** to extract different features

3) **Spatially share** parameters of each filter

Kerangka Bahasan

1. Pengantar
2. Pembelajaran Fitur Visual
3. Ekstraksi Fitur dan Convolution: Contoh Kasus Sederhana
4. CNN
5. Sebuah Arsitektur untuk Banyak Aplikasi
6. Ringkasan

- Contoh kasus sederhana ini diambil dari
  Brandon Rohrer. How convolutional neural networks work

- Slides

https://github.com/brohrer/public-hosting/blob/master/how_CNNs_work.pptx?raw=true

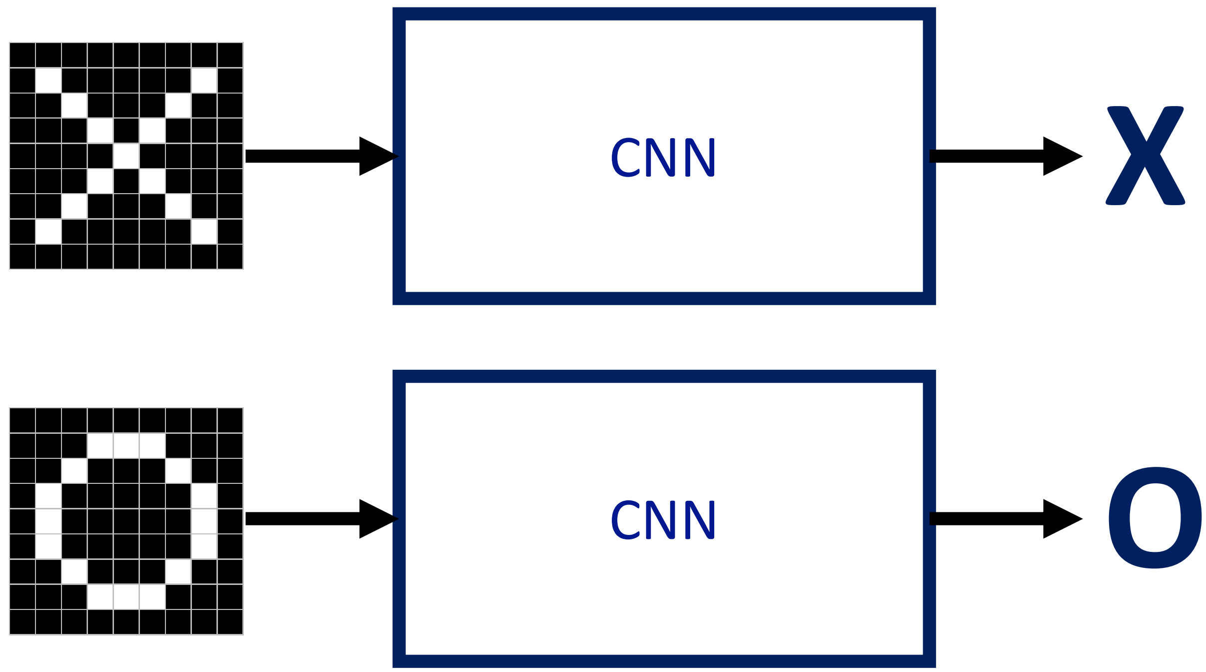
- Video

https://youtu.be/JB8T_zN7ZC0

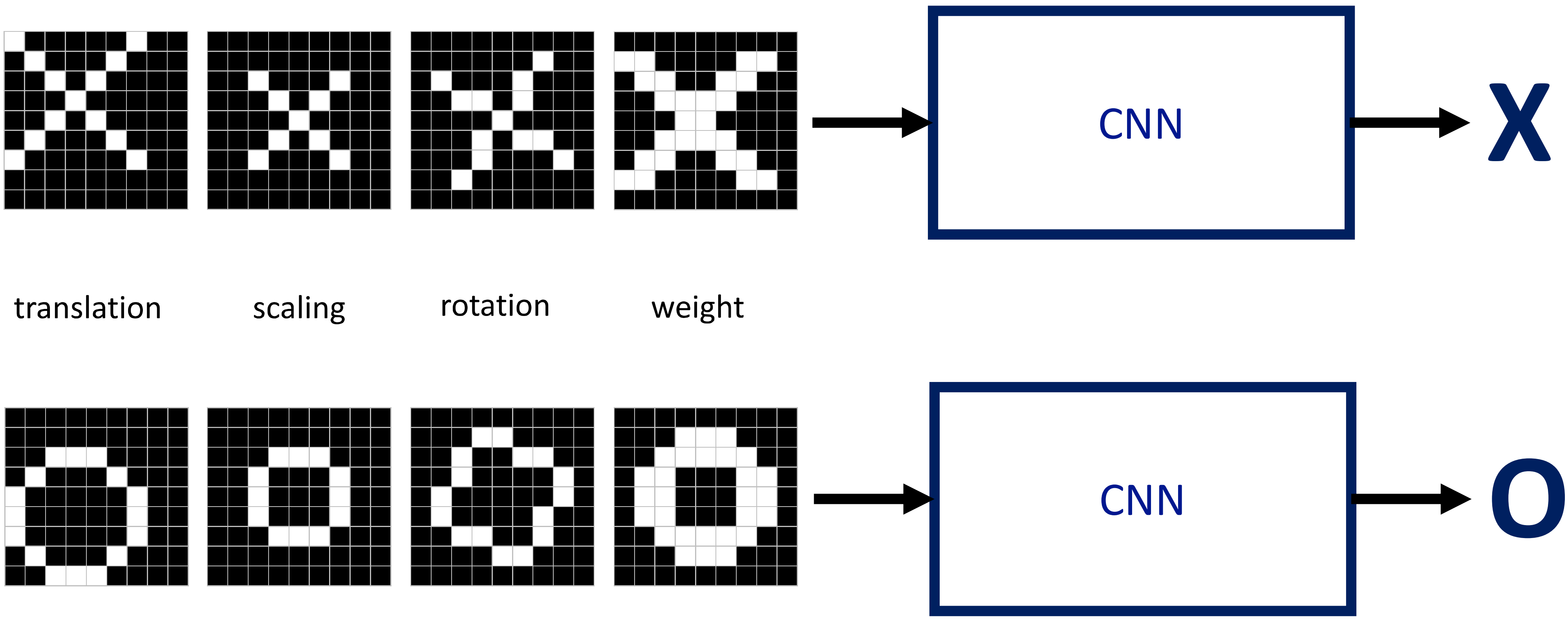
A toy ConvNet: X's and O's

Says whether a picture is of an X or an O

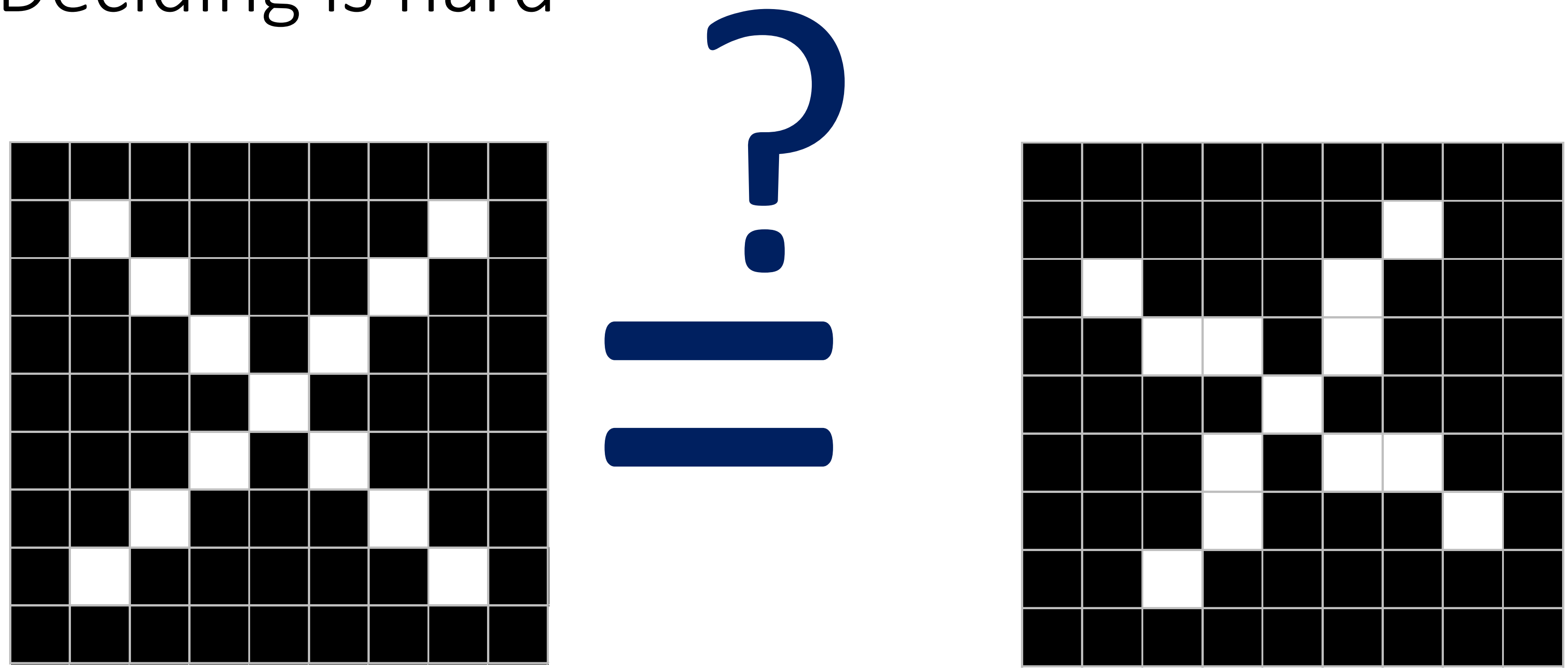
For example



Trickier cases



Deciding is hard



What computers see

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

?

=

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	1	-1	-1	-1
-1	-1	1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	1	-1	-1
-1	-1	-1	1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

What computers see

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	X	-1	-1	-1	-1	X	X	-1
-1	X	X	-1	-1	X	X	-1	-1
-1	-1	X	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	X	-1	-1
-1	-1	X	X	-1	-1	X	X	-1
-1	X	X	-1	-1	-1	-1	X	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

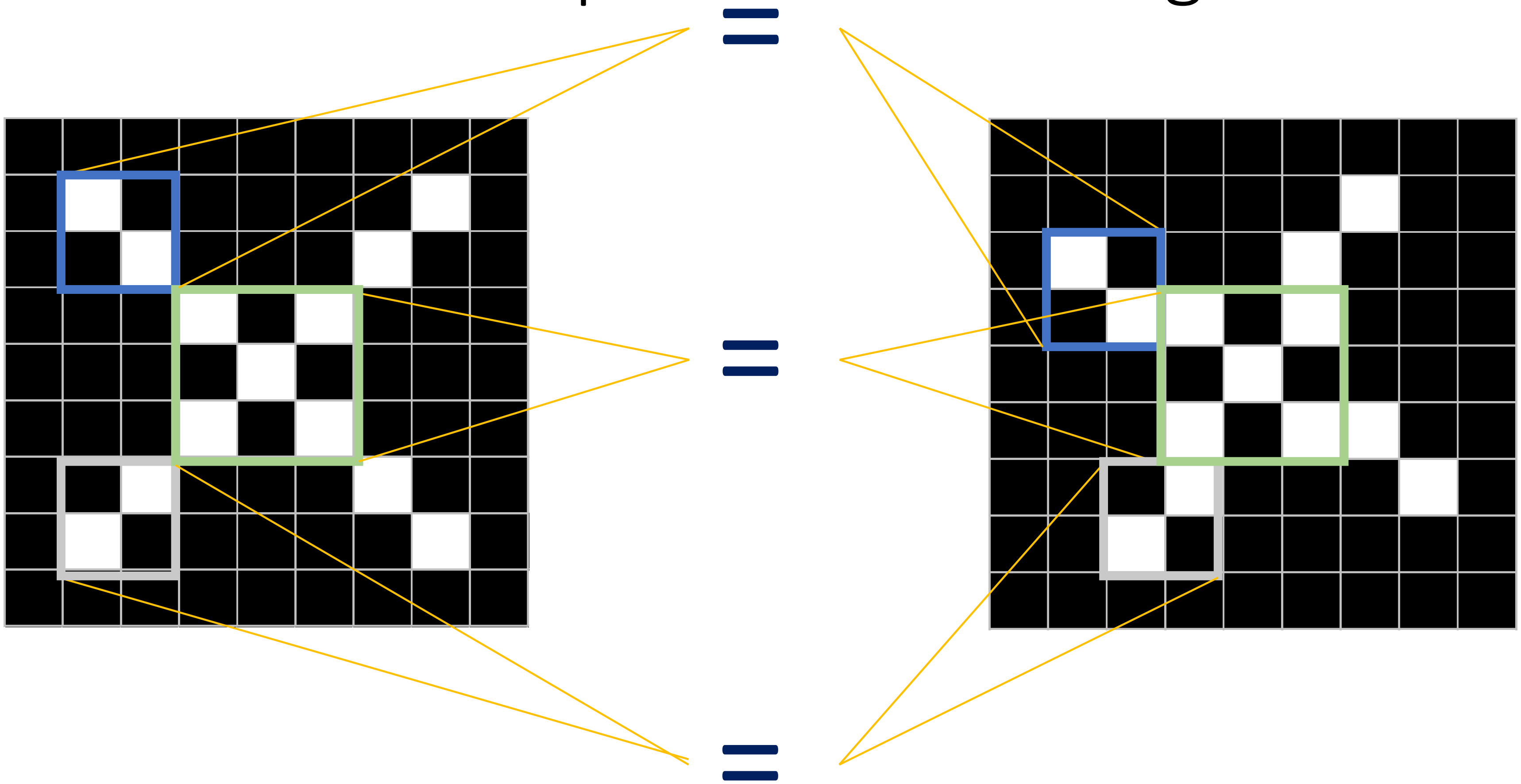
Computers are literal

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

$\neq$

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	1	-1	-1	-1
-1	-1	1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	1	-1	-1
-1	-1	-1	1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

ConvNets match pieces of the image

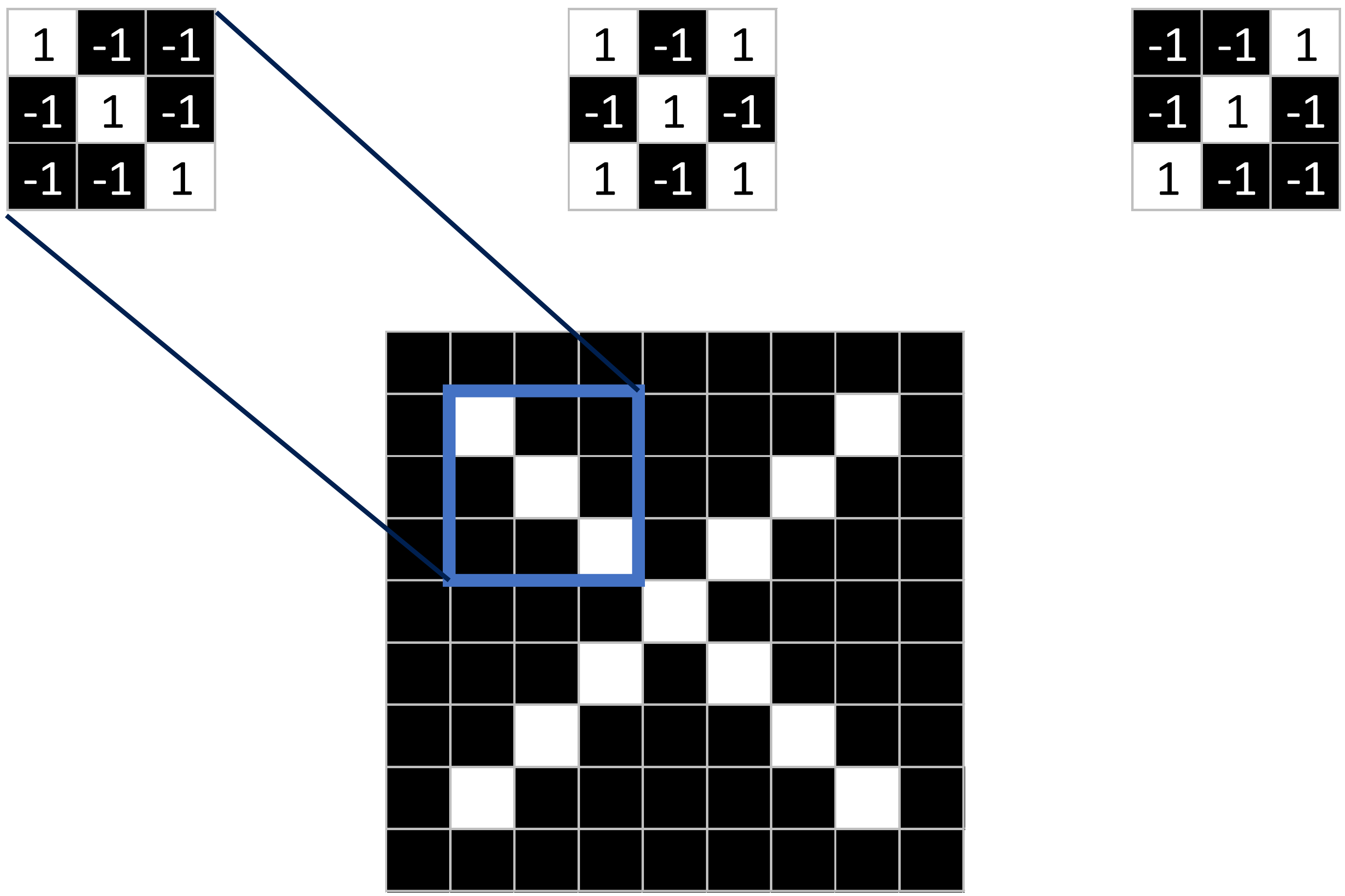


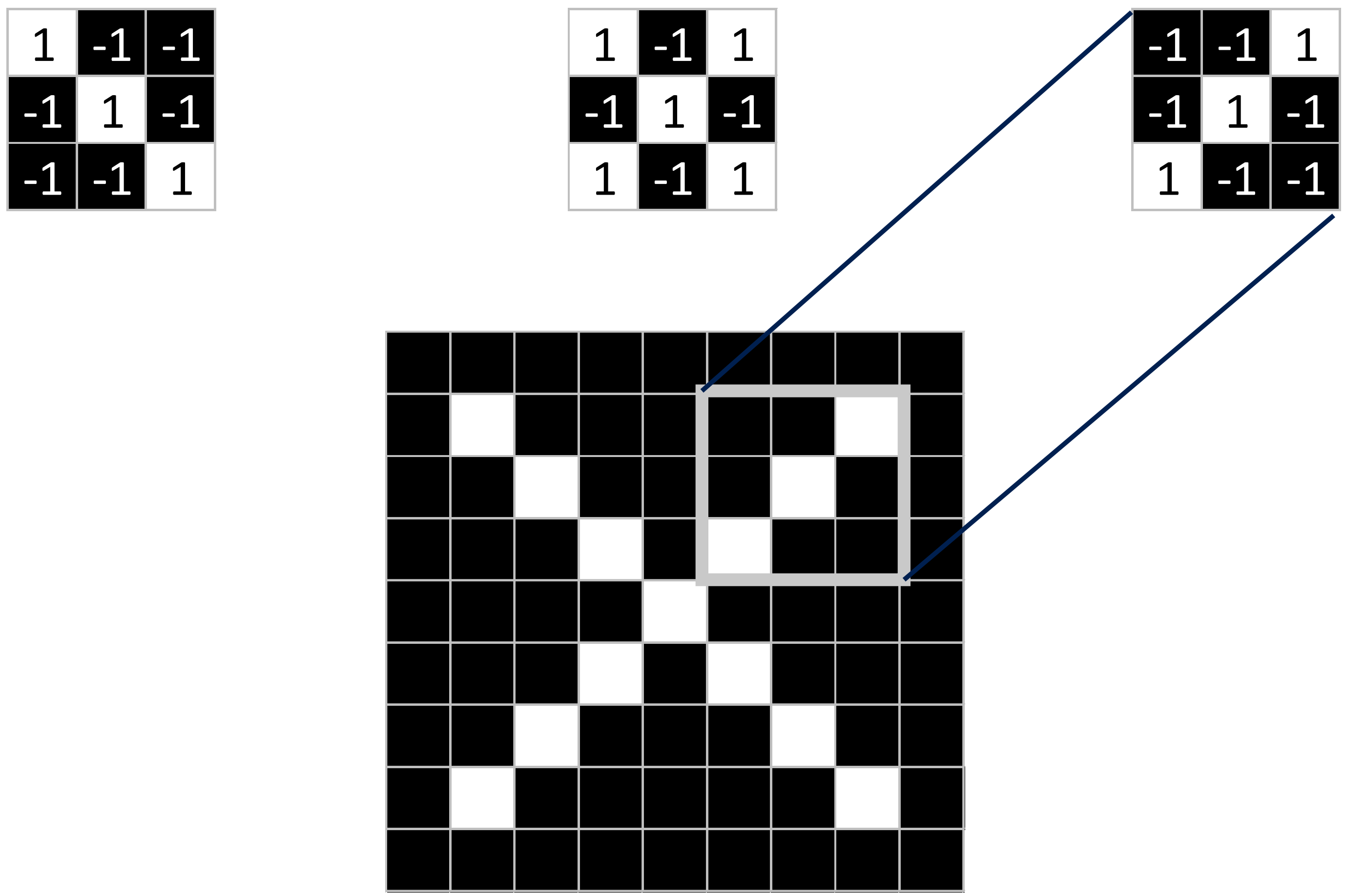
Features match pieces of the image

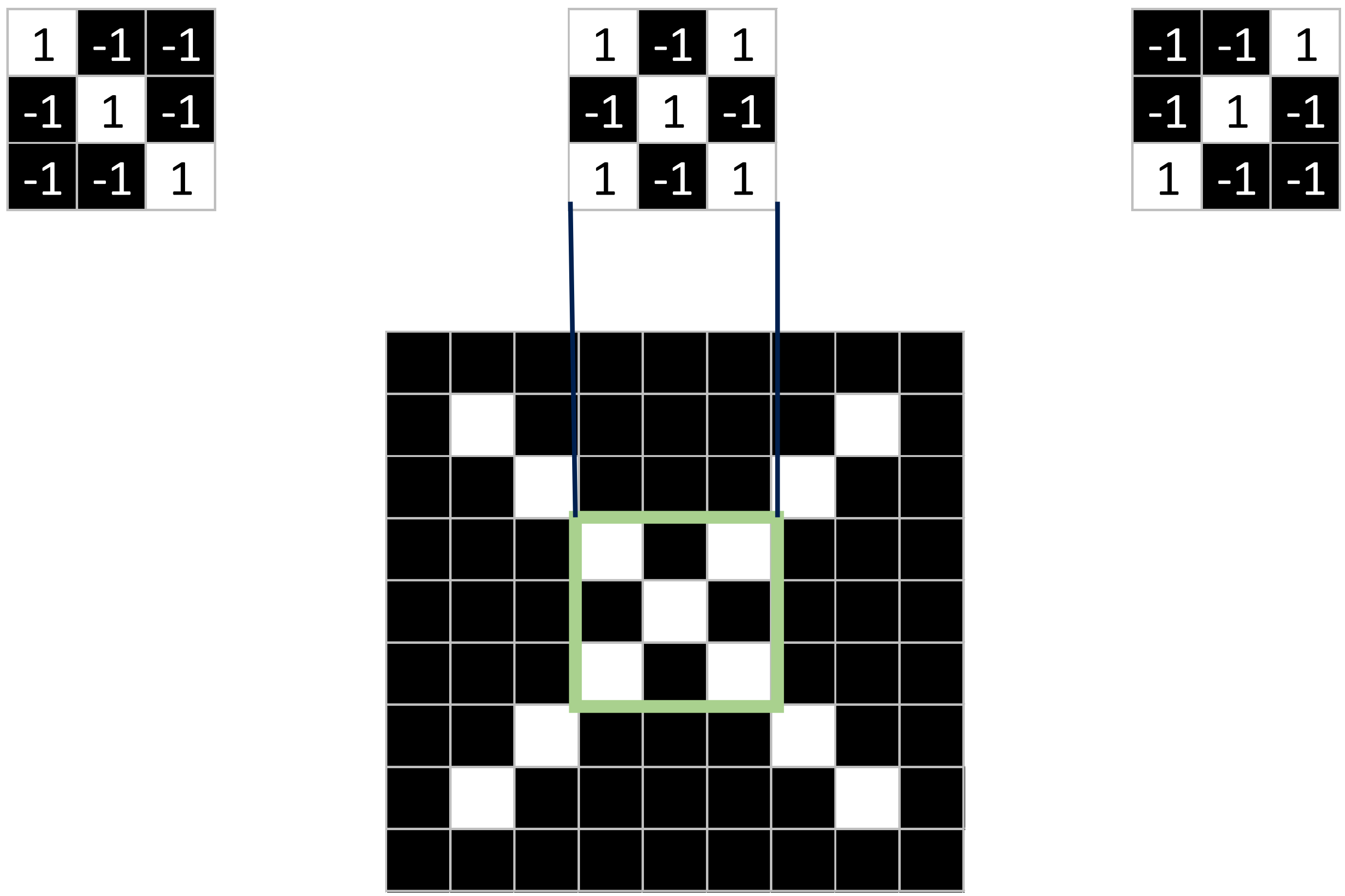
1	-1	-1
-1	1	-1
-1	-1	1

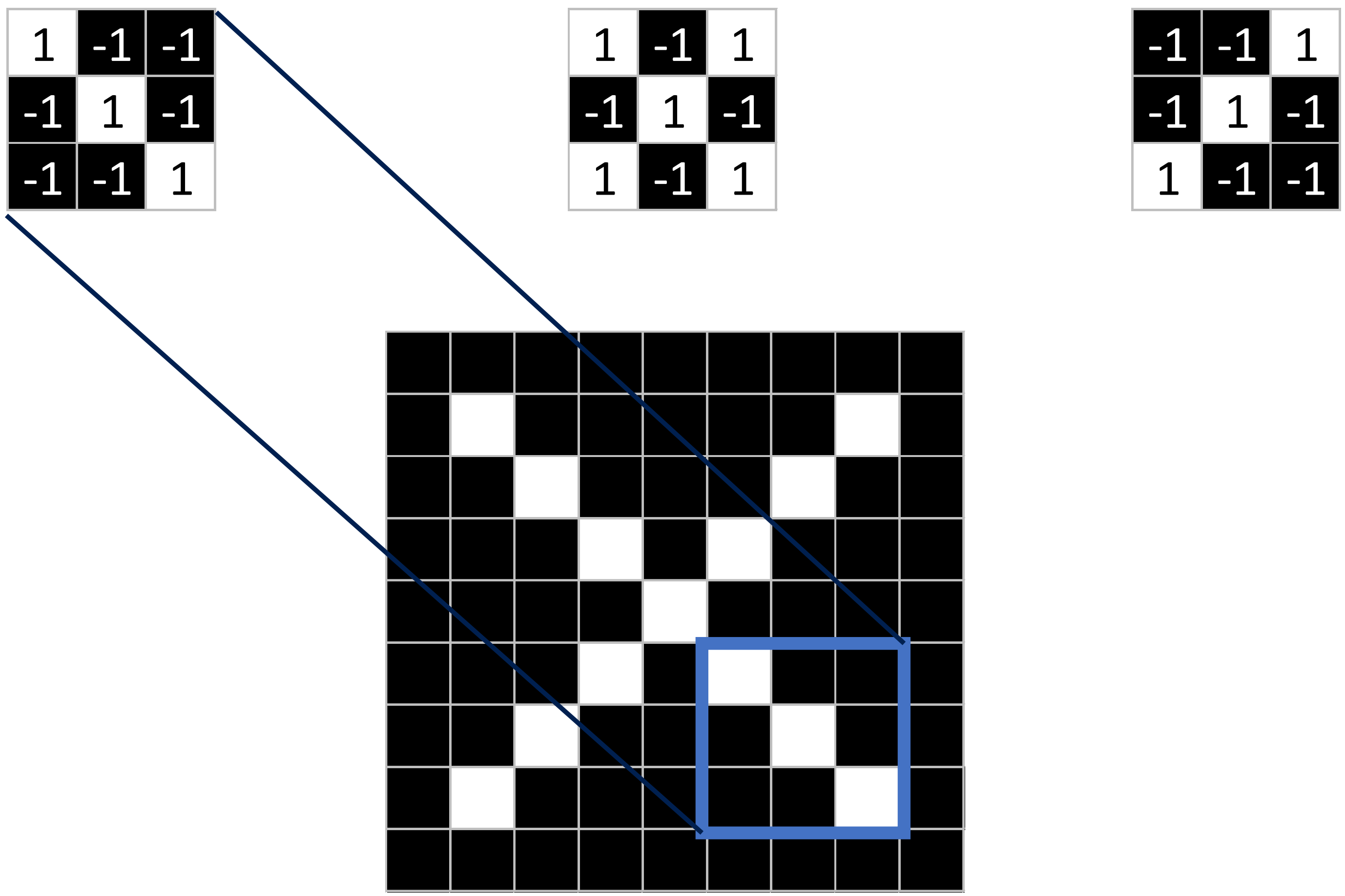
1	-1	1
-1	1	-1
1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

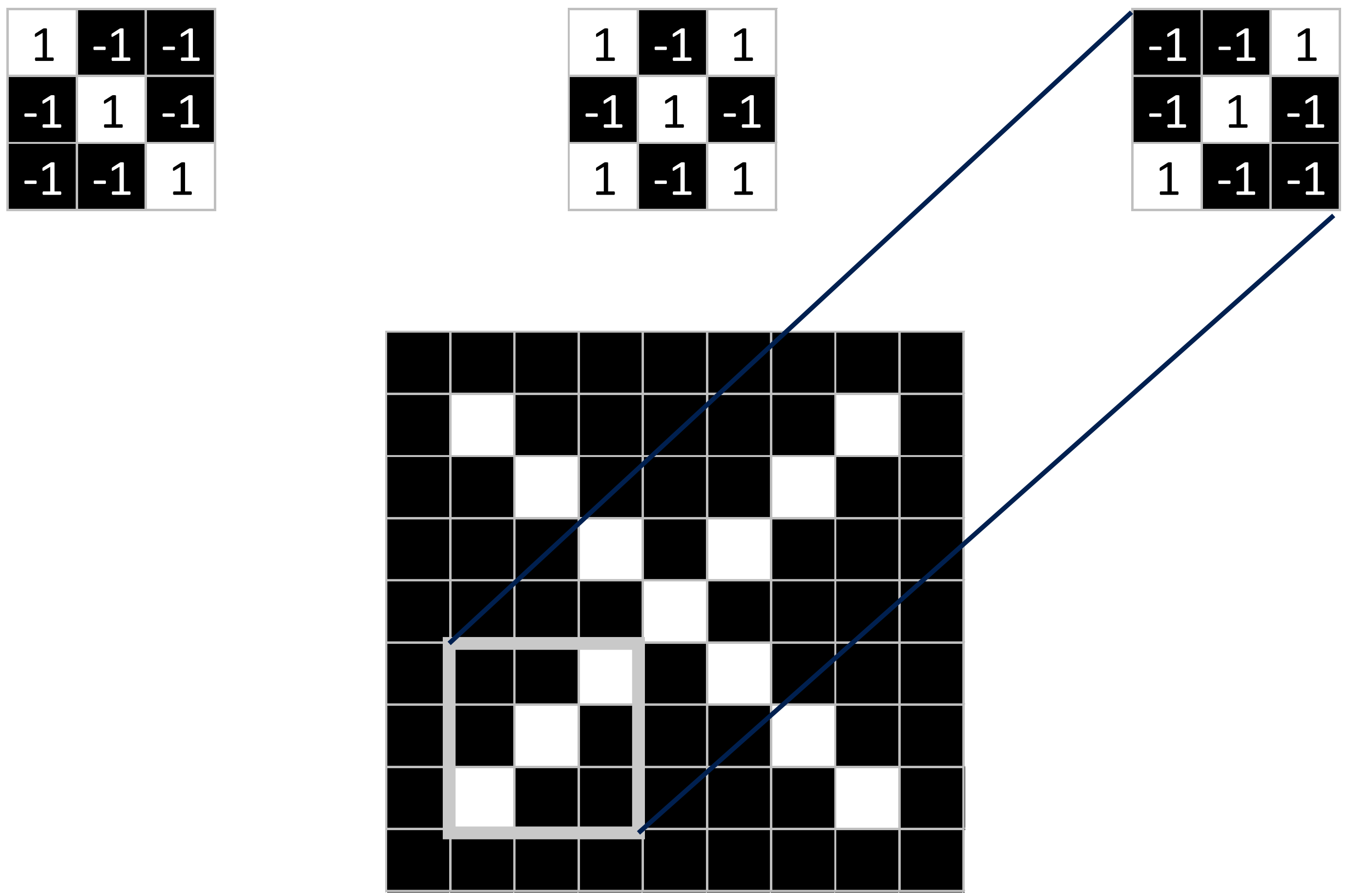

1 -1 -1
-1 1 -1
-1 -1 1
1 -1 1
-1 1 -1
1 -1 1
-1 -1 1
-1 1 -1
1 -1 -1


1 -1 -1
-1 1 -1
-1 -1 1
1 -1 1
-1 1 -1
1 -1 1
-1 -1 1
-1 1 -1
1 -1 -1


1 -1 -1
-1 1 -1
-1 -1 1
1 -1 1
-1 1 -1
1 -1 1
-1 -1 1
-1 1 -1
1 -1 -1



1 -1 -1
-1 1 -1
-1 -1 1
1 -1 1
-1 1 -1
1 -1 1
-1 -1 1
-1 1 -1
1 -1 -1


1 -1 -1
-1 1 -1
-1 -1 1
1 -1 1
-1 1 -1
1 -1 1
-1 -1 1
-1 1 -1
1 -1 -1

Filtering: The math behind the match

1	-1	-1
-1	1	-1
-1	-1	1

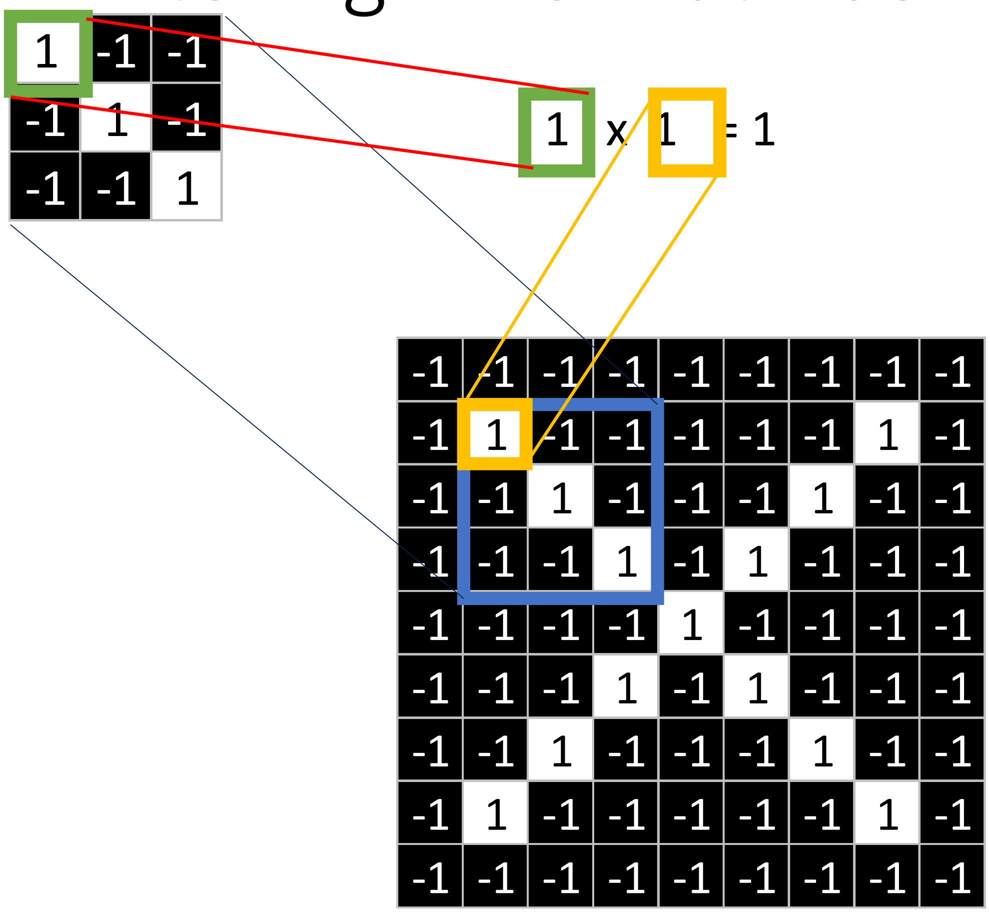
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

DON'T PANIC

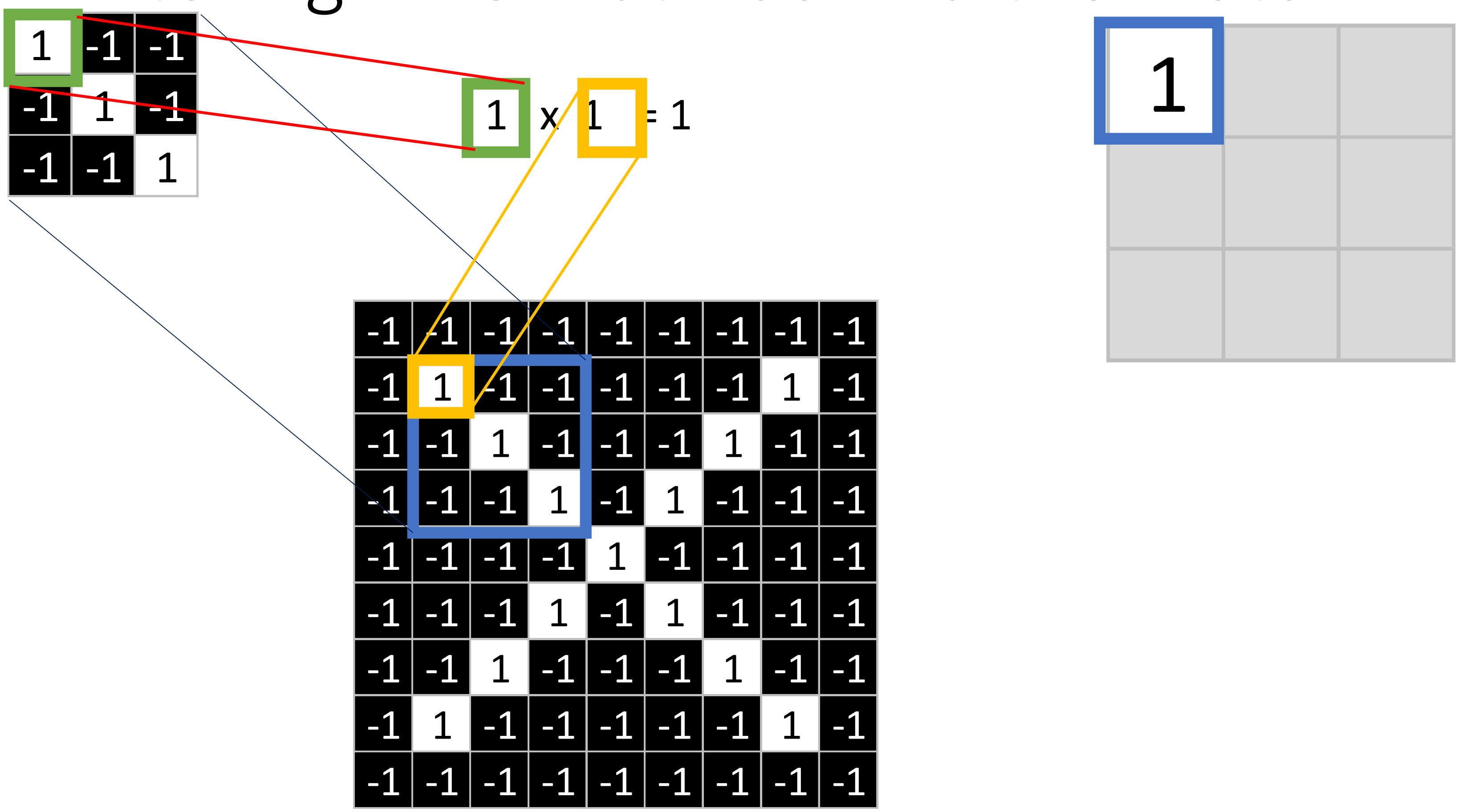
Filtering: The math behind the match

1. Line up the feature and the image patch.
2. Multiply each image pixel by the corresponding feature pixel.
3. Add them up.
4. Divide by the total number of pixels in the feature.

Filtering: The math behind the match



Filtering: The math behind the match



Filtering: The math behind the match

1	-1	-1
-1	1	-1
-1	-1	1

-1 x -1 = 1

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

1	1	

Filtering: The math behind the match

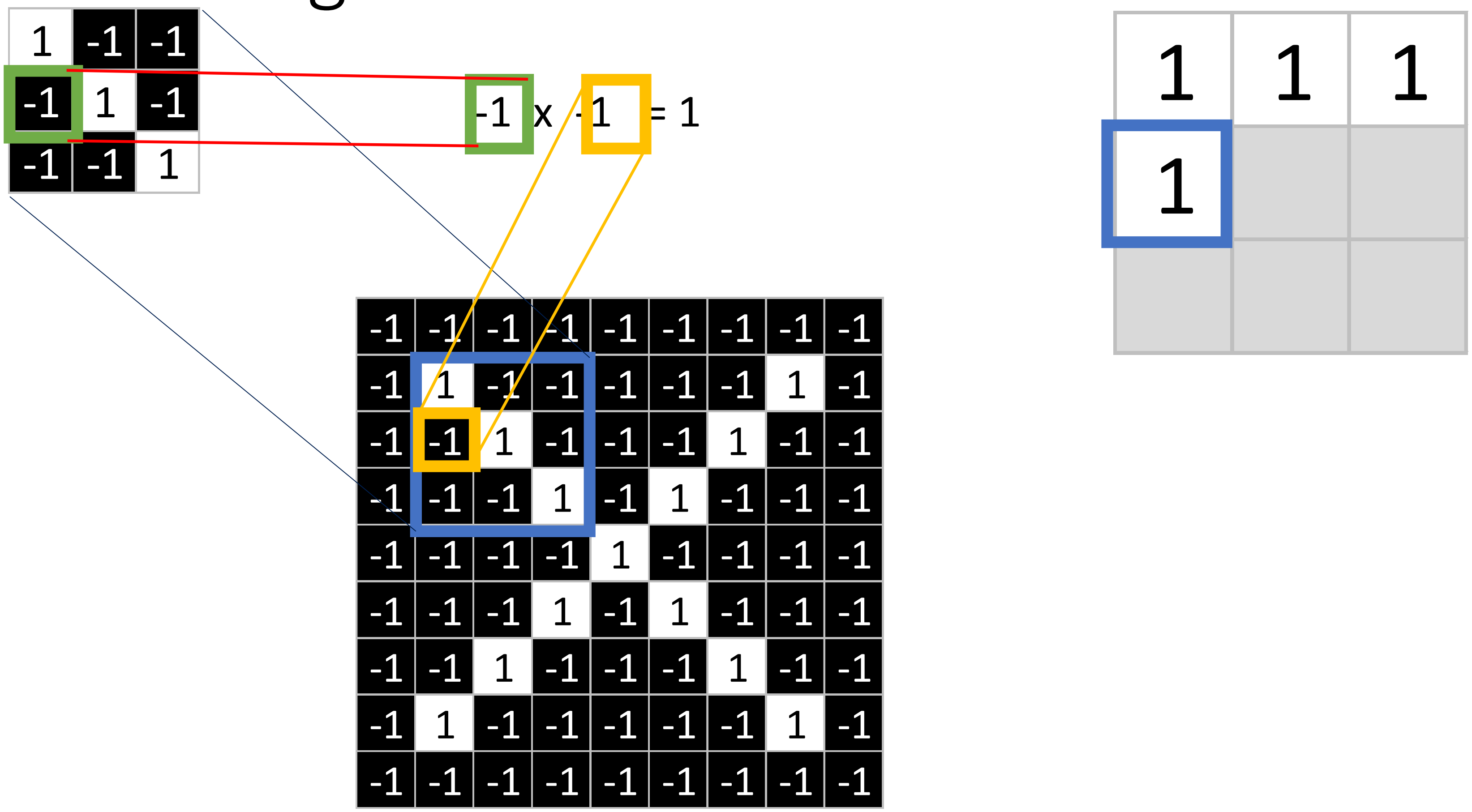
1	-1	-1
-1	1	-1
-1	-1	1

-1 x -1 = 1

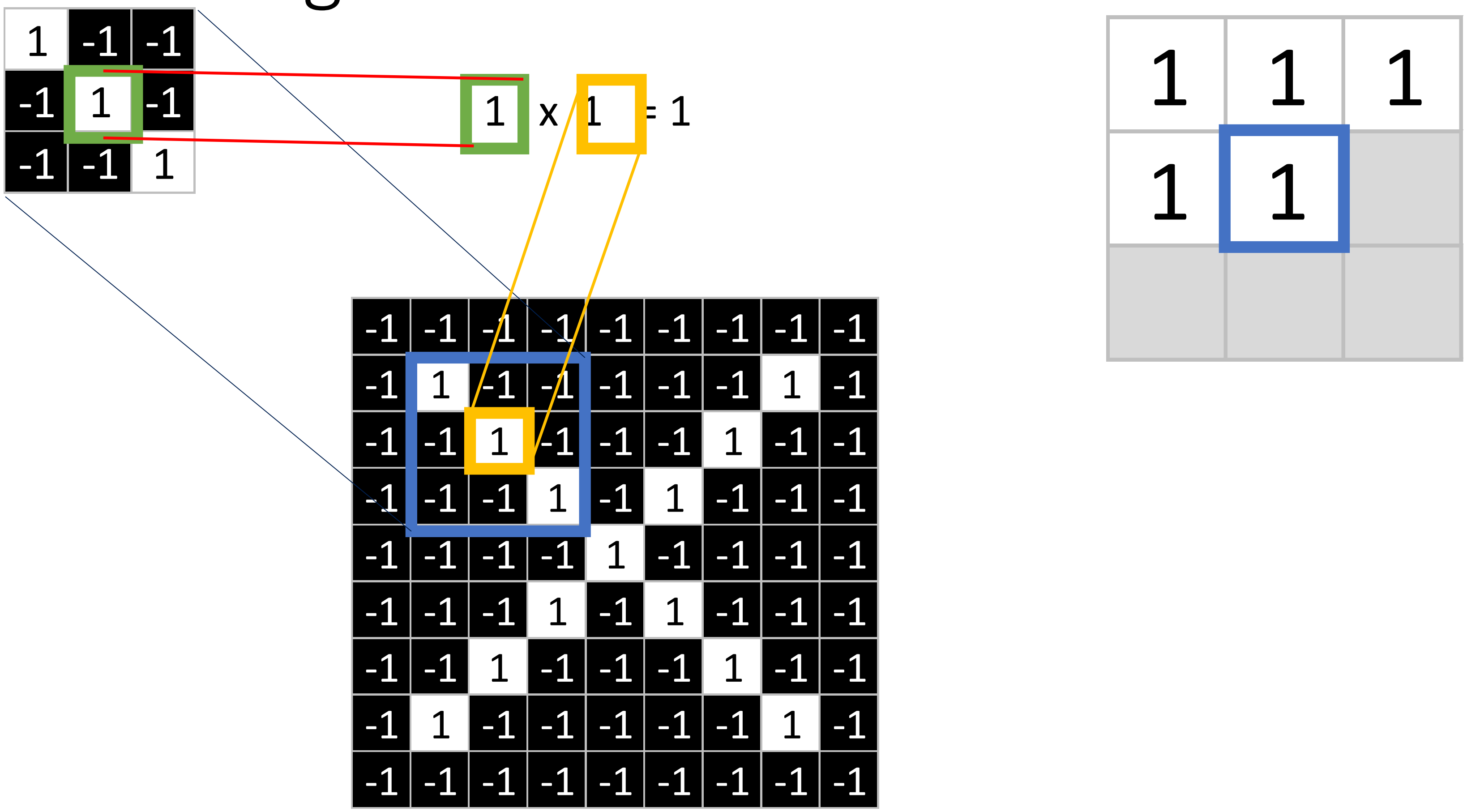
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

1	1	1

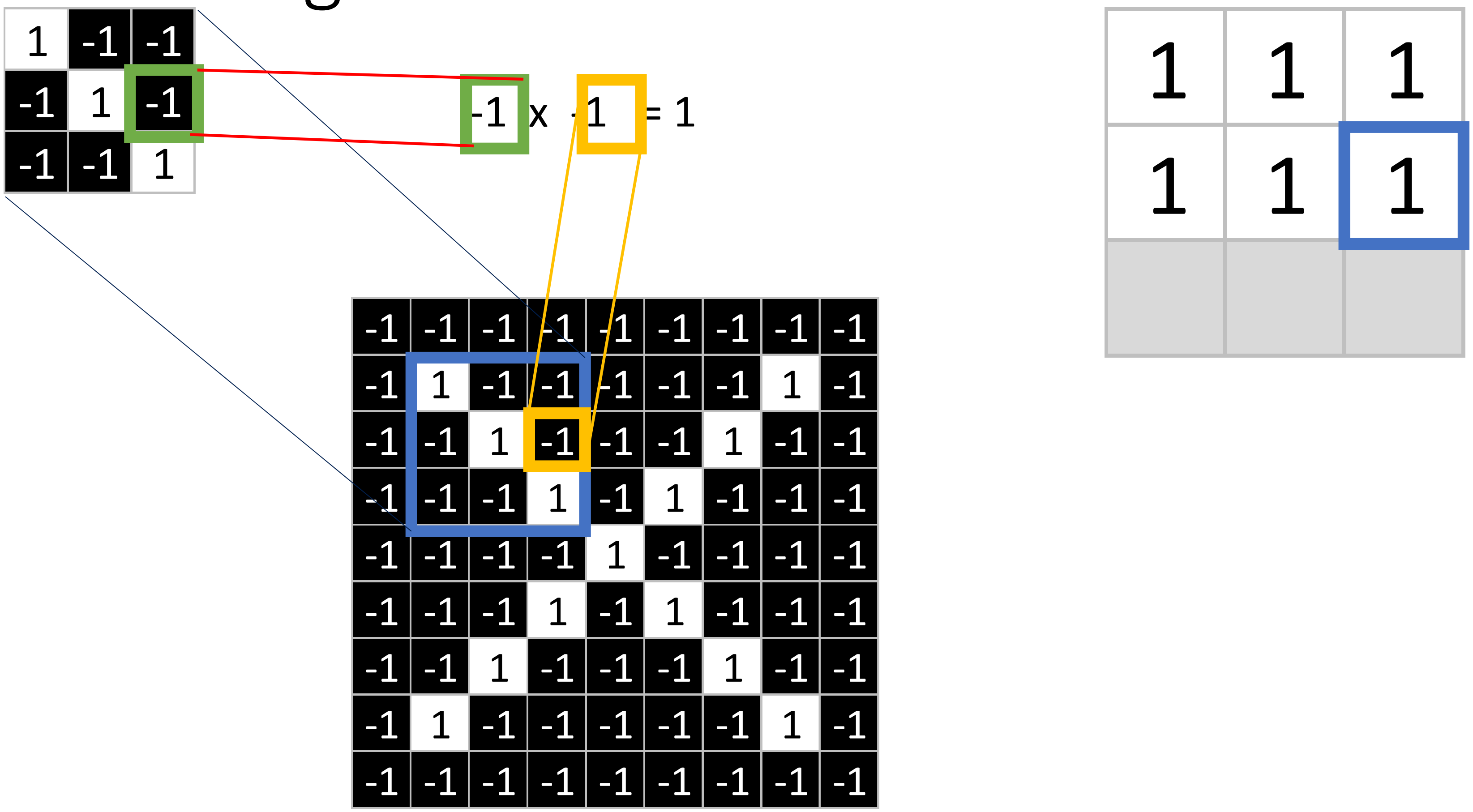
Filtering: The math behind the match



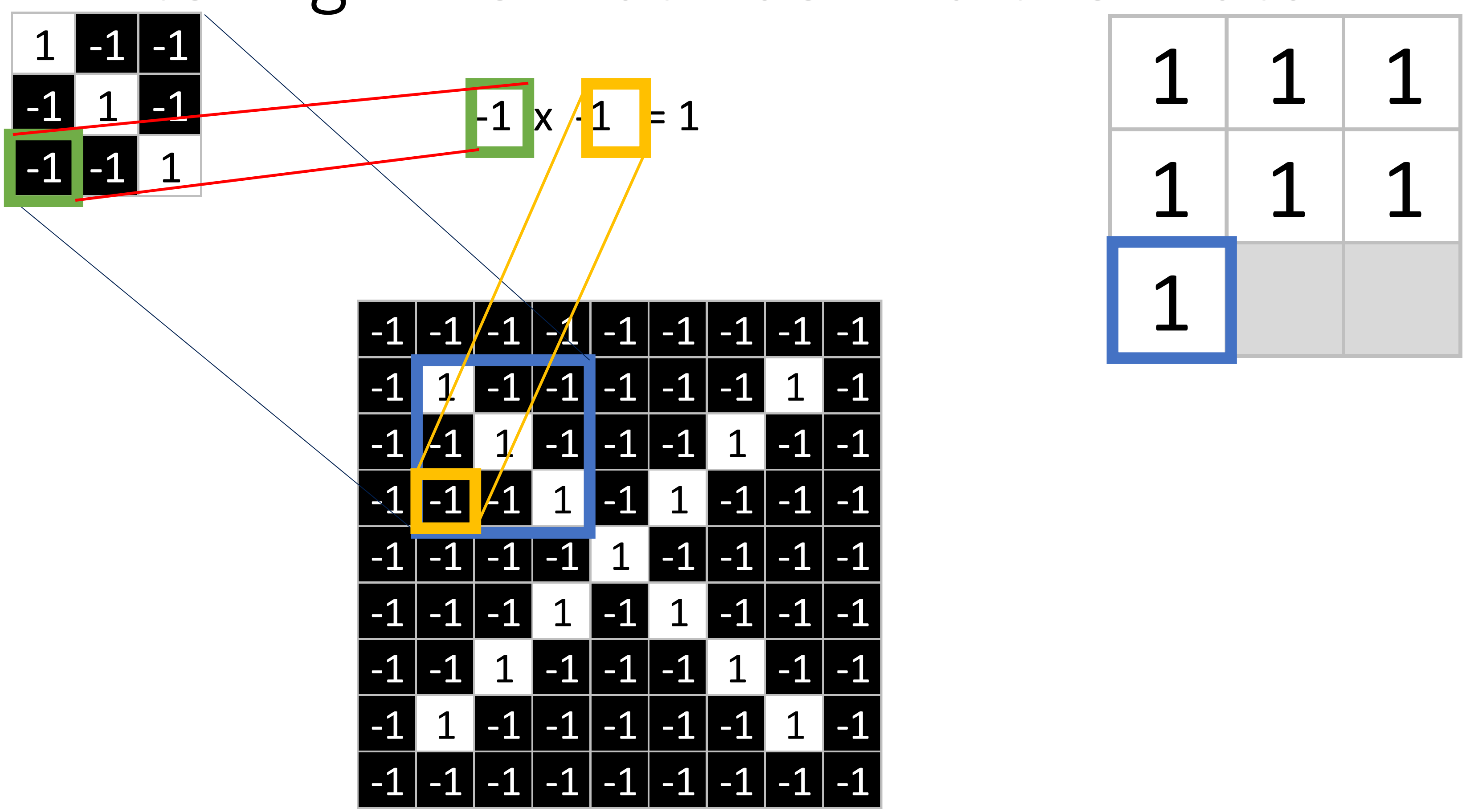
Filtering: The math behind the match



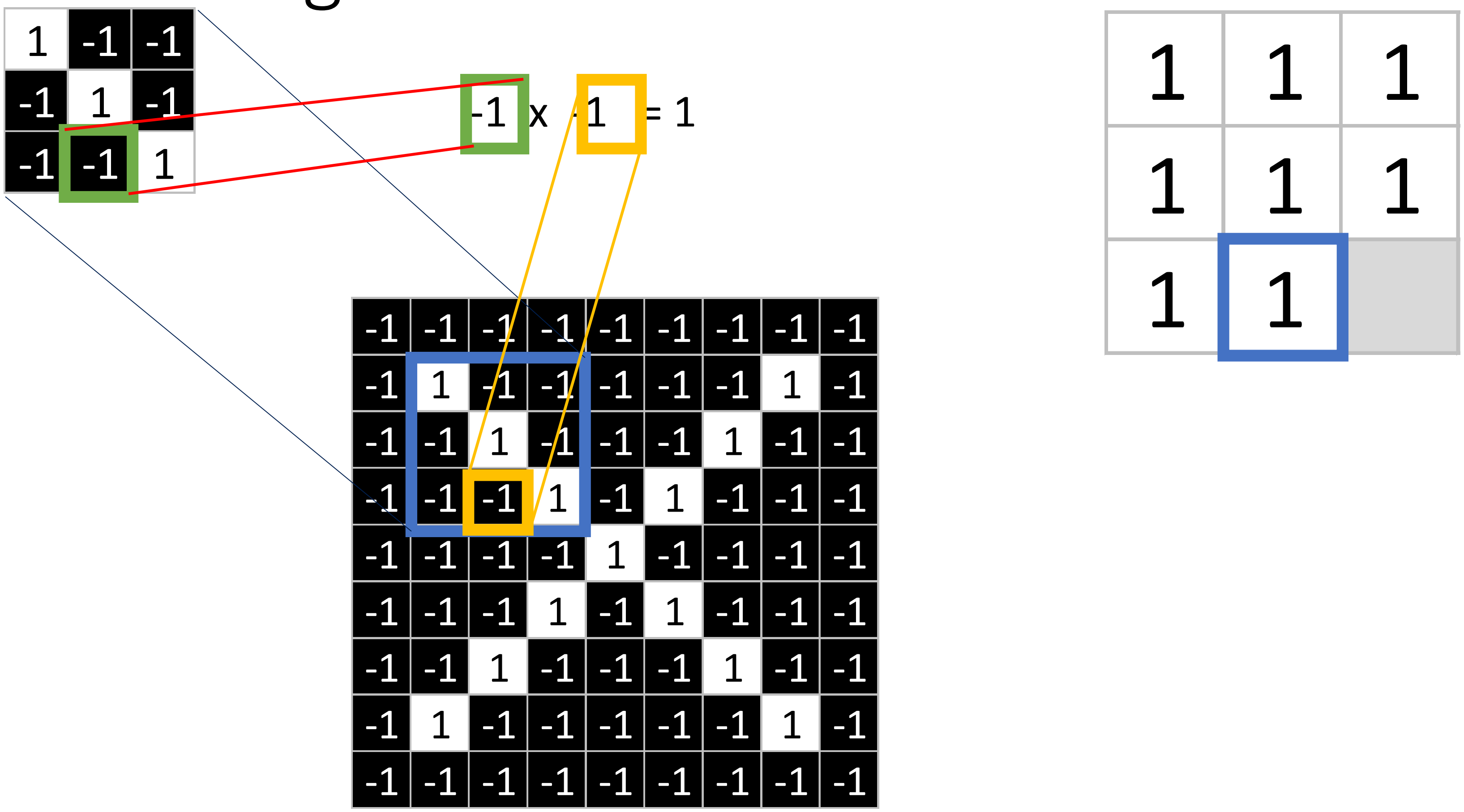
Filtering: The math behind the match



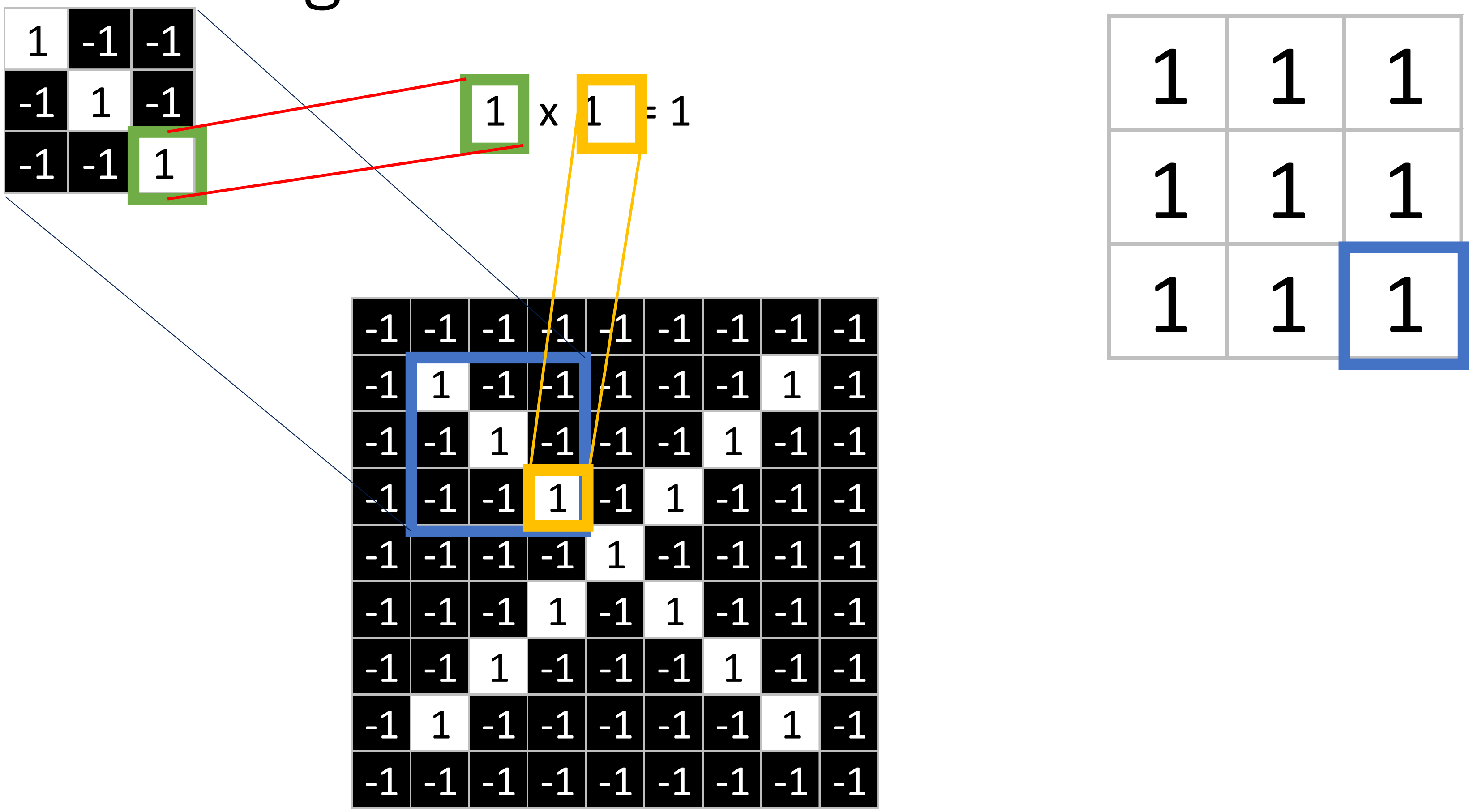
Filtering: The math behind the match



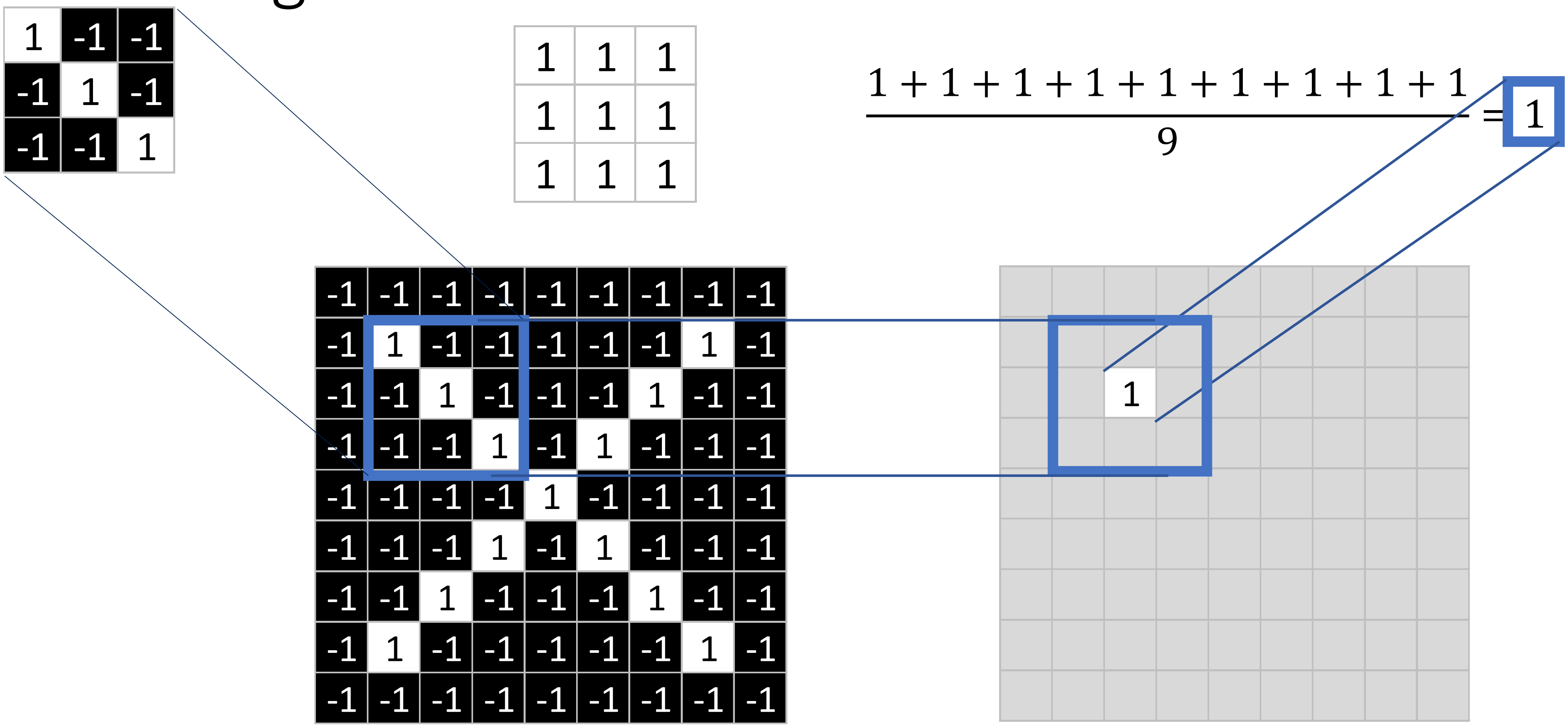
Filtering: The math behind the match



Filtering: The math behind the match

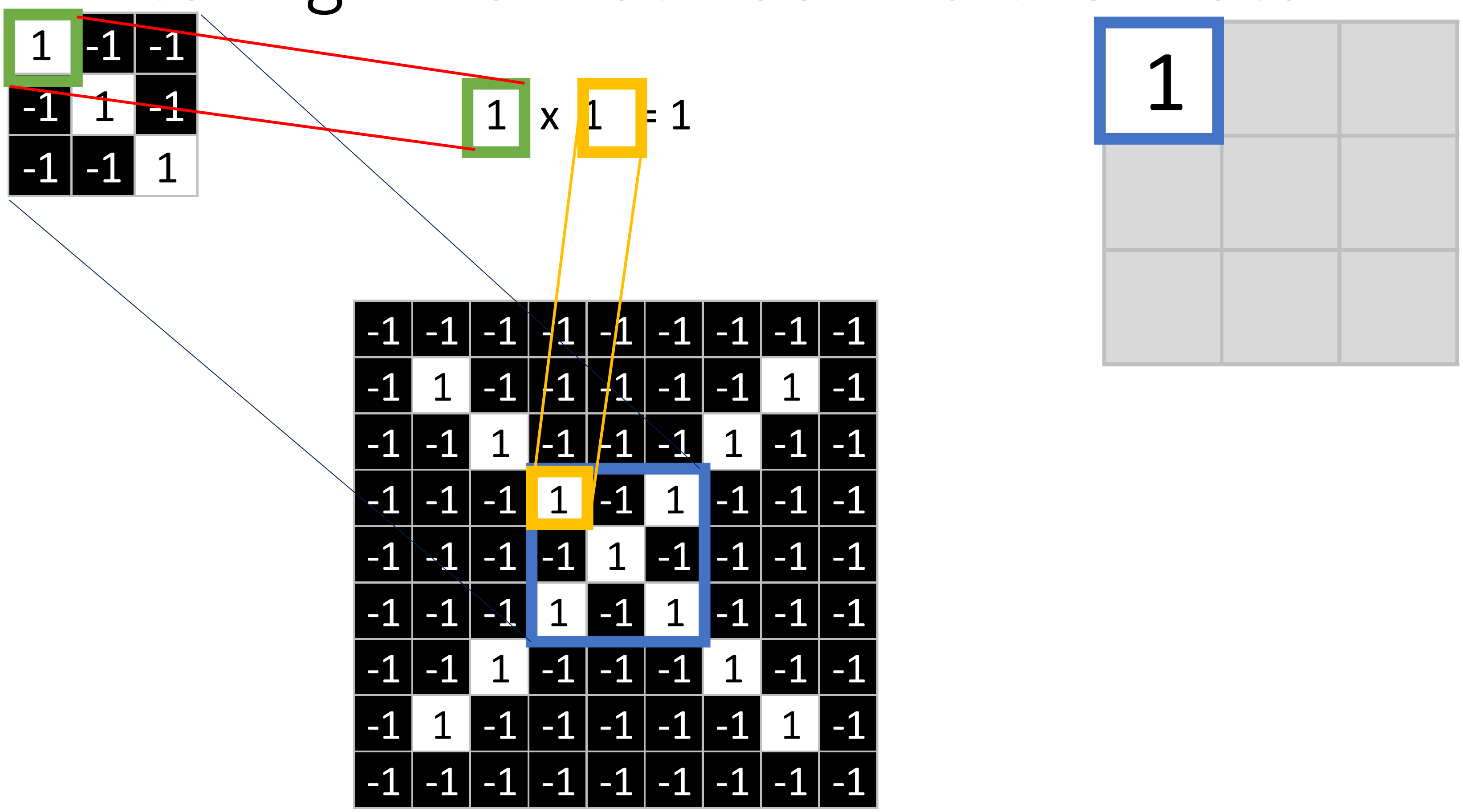


Filtering: The math behind the match

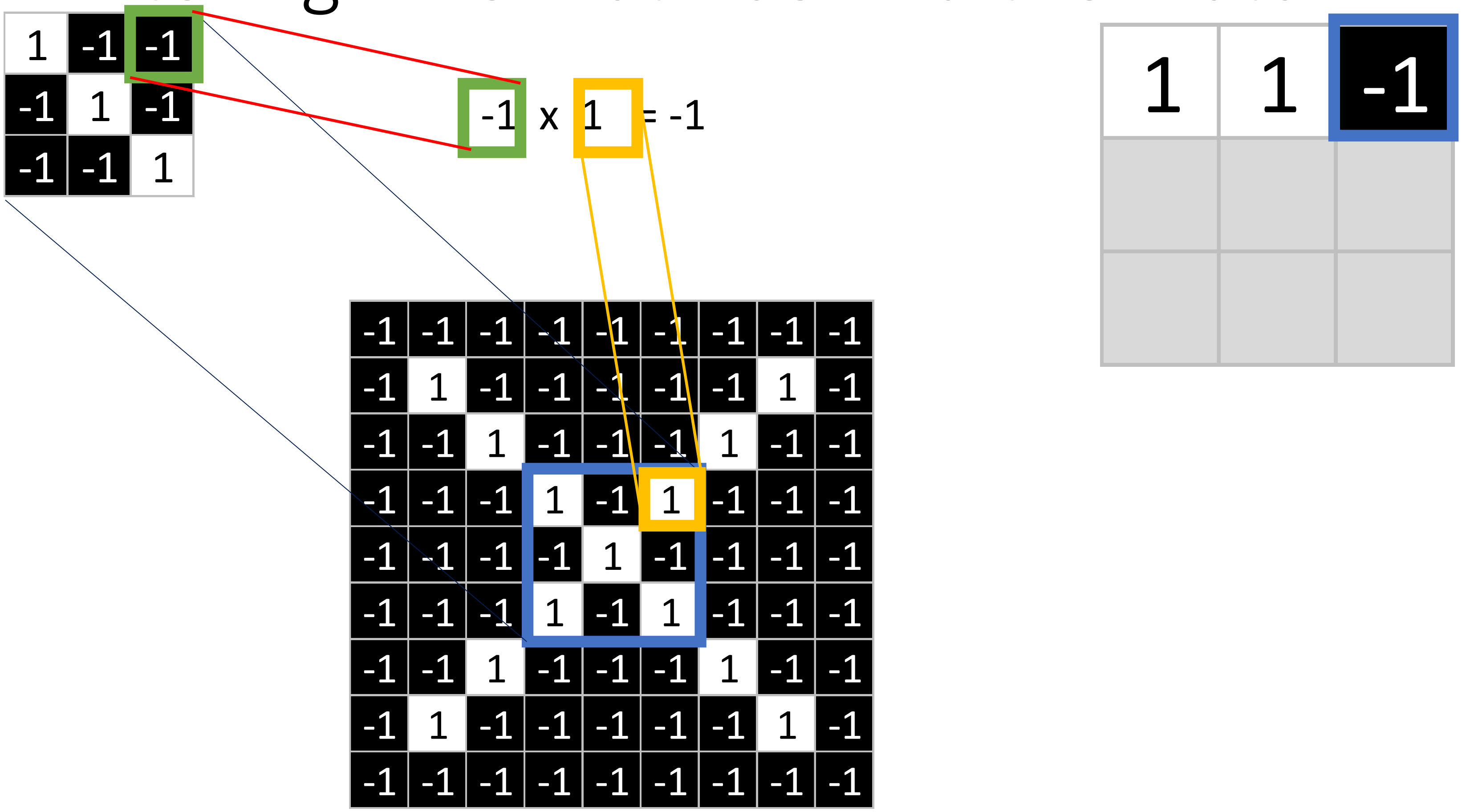


$$\frac{1+1+1+1+1+1+1+1+1}{9} = \boxed{1}$$

Filtering: The math behind the match



Filtering: The math behind the match



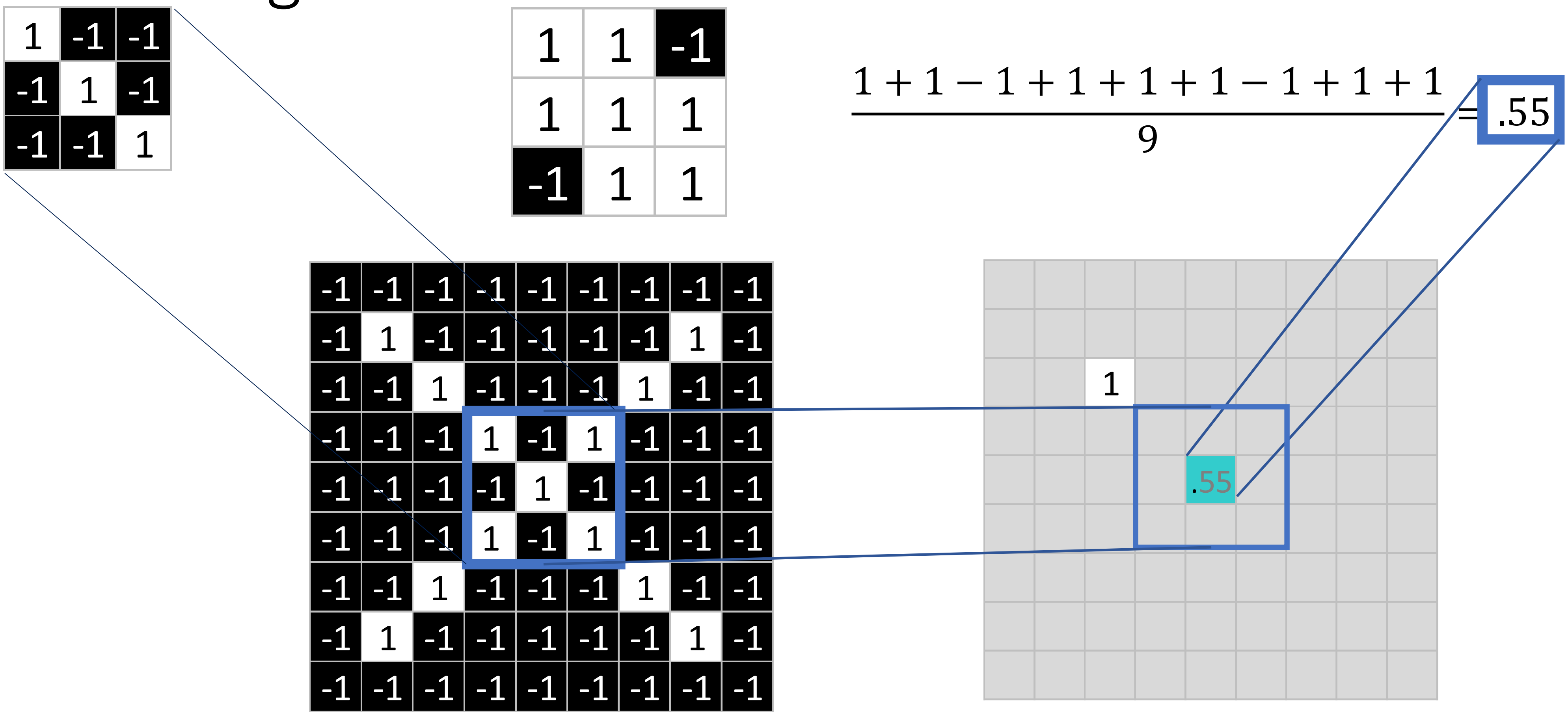
Filtering: The math behind the match

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

1	1	-1
1	1	1
-1	1	1

Filtering: The math behind the match



$$\frac{1+1-1+1+1+1-1+1+1}{9} = \boxed{.55}$$

Convolution: Trying every possible match

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

Convolution: Trying every possible match

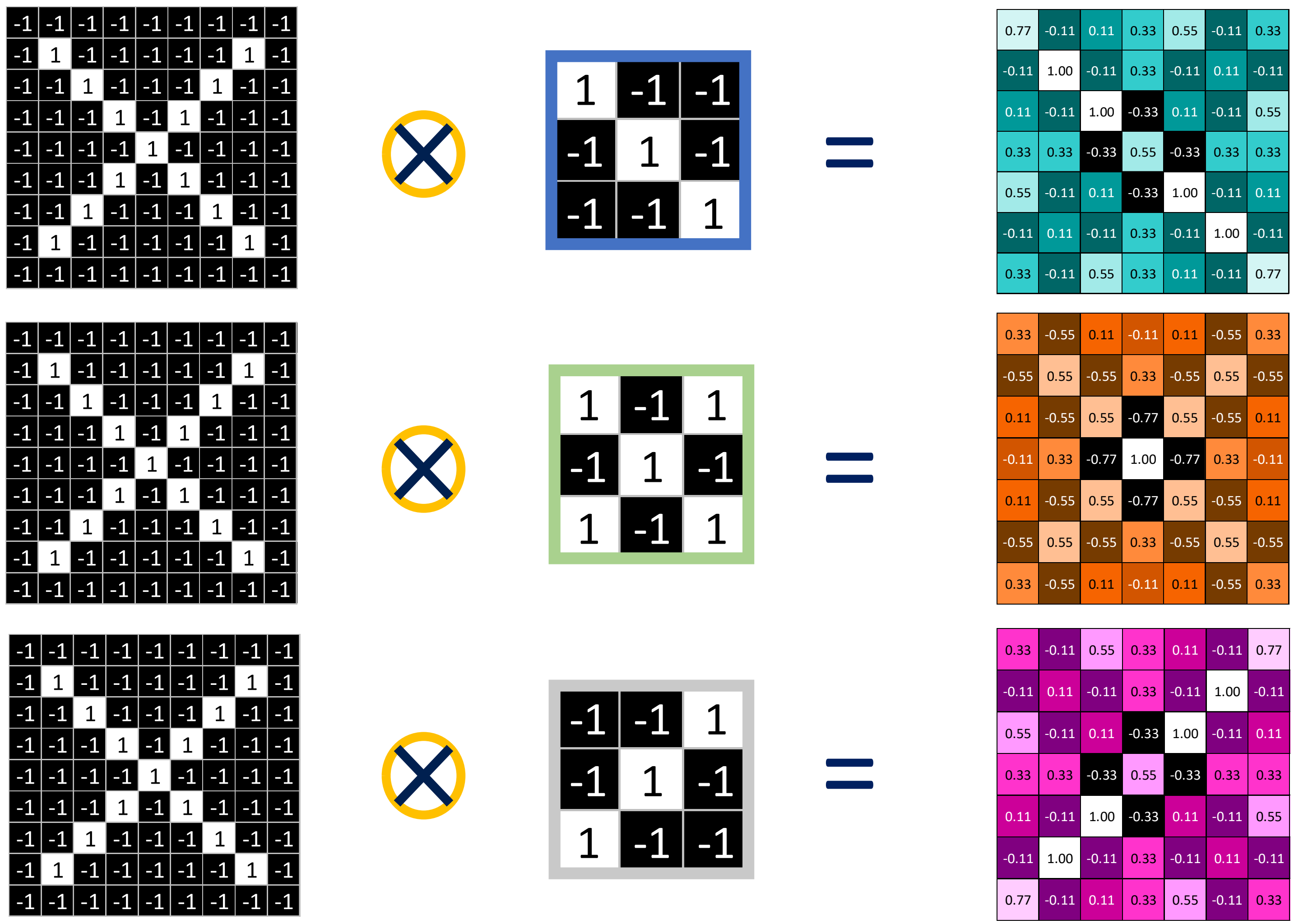
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

$\otimes$

1	-1	-1
-1	1	-1
-1	-1	1

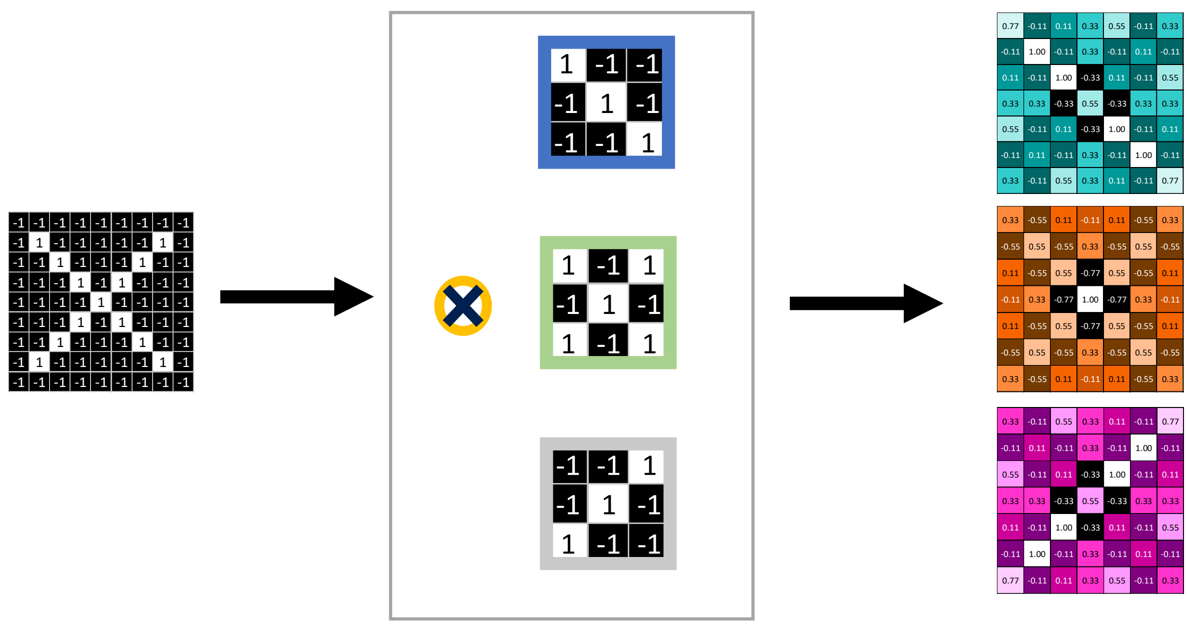
$=$

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77


-1 -1 -1 -1 -1 -1 -1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 1 -1 -1 -1 1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 -1 -1 1 -1 -1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 1 -1 -1 -1 1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1
1 -1 -1
-1 1 -1
-1 -1 1
0.77 -0.11 0.11 0.33 0.55 -0.11 0.33
-0.11 1.00 -0.11 0.33 -0.11 0.11 -0.11
0.11 -0.11 1.00 -0.33 0.11 -0.11 0.55
0.33 0.33 -0.33 0.55 -0.33 0.33 0.33
0.55 -0.11 0.11 -0.33 1.00 -0.11 0.11
-0.11 0.11 -0.11 0.33 -0.11 1.00 -0.11
0.33 -0.11 0.55 0.33 0.11 -0.11 0.77
-1 -1 -1 -1 -1 -1 -1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 1 -1 -1 -1 1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 -1 -1 1 -1 -1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 1 -1 -1 -1 1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1
1 -1 1
-1 1 -1
1 -1 1
0.33 -0.55 0.11 -0.11 0.11 -0.55 0.33
-0.55 0.55 -0.55 0.33 -0.55 0.55 -0.55
0.11 -0.55 0.55 -0.77 0.55 -0.55 0.11
-0.11 0.33 -0.77 1.00 -0.77 0.33 -0.11
0.11 -0.55 0.55 -0.77 0.55 -0.55 0.11
-0.55 0.55 -0.55 0.33 -0.55 0.55 -0.55
0.33 -0.55 0.11 -0.11 0.11 -0.55 0.33
-1 -1 -1 -1 -1 -1 -1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 1 -1 -1 -1 1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 -1 -1 1 -1 -1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 1 -1 -1 -1 1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 1
-1 1 -1
1 -1 -1
0.33 -0.11 0.55 0.33 0.11 -0.11 0.77
-0.11 0.11 -0.11 0.33 -0.11 1.00 -0.11
0.55 -0.11 0.11 -0.33 1.00 -0.11 0.11
0.33 0.33 -0.33 0.55 -0.33 0.33 0.33
0.11 -0.11 1.00 -0.33 0.11 -0.11 0.55
-0.11 1.00 -0.11 0.33 -0.11 0.11 -0.11
0.77 -0.11 0.11 0.33 0.55 -0.11 0.33

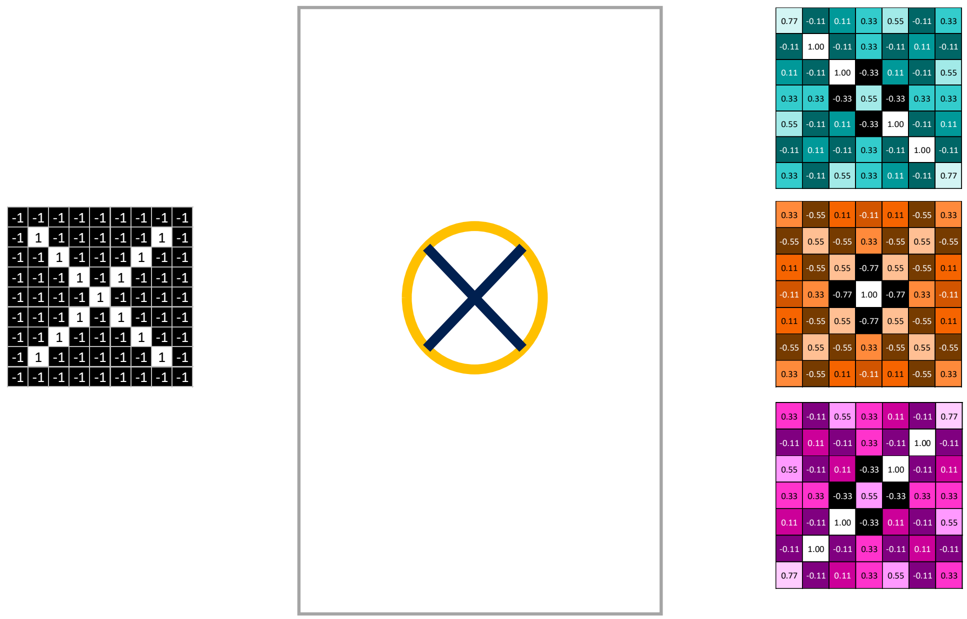
Convolution layer

One image becomes a stack of filtered images



Convolution layer

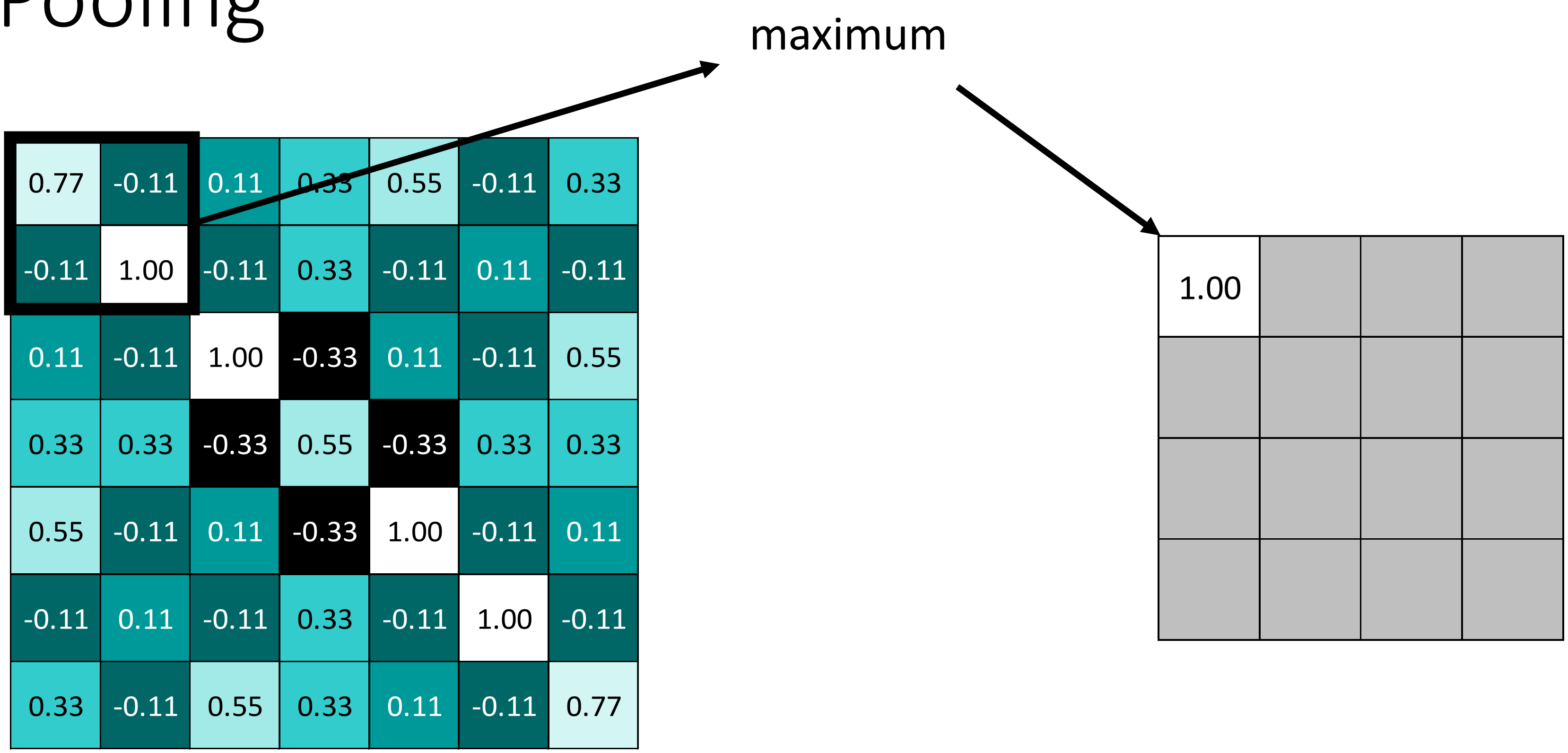
One image becomes a stack of filtered images



Pooling: Shrinking the image stack

1. Pick a window size (usually 2 or 3).
2. Pick a stride (usually 2).
3. Walk your window across your filtered images.
4. From each window, take the maximum value.

Pooling


0.77 -0.11 0.11 0.33 0.55 -0.11 0.33
-0.11 1.00 -0.11 0.33 -0.11 0.11 -0.11
0.11 -0.11 1.00 -0.33 0.11 -0.11 0.55
0.33 0.33 -0.33 0.55 -0.33 0.33 0.33
0.55 -0.11 0.11 -0.33 1.00 -0.11 0.11
-0.11 0.11 -0.11 0.33 -0.11 1.00 -0.11
0.33 -0.11 0.55 0.33 0.11 -0.11 0.77
maximum
1.00

Pooling

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

maximum

1.00	0.33		

Pooling

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

maximum

1.00	0.33	0.55	

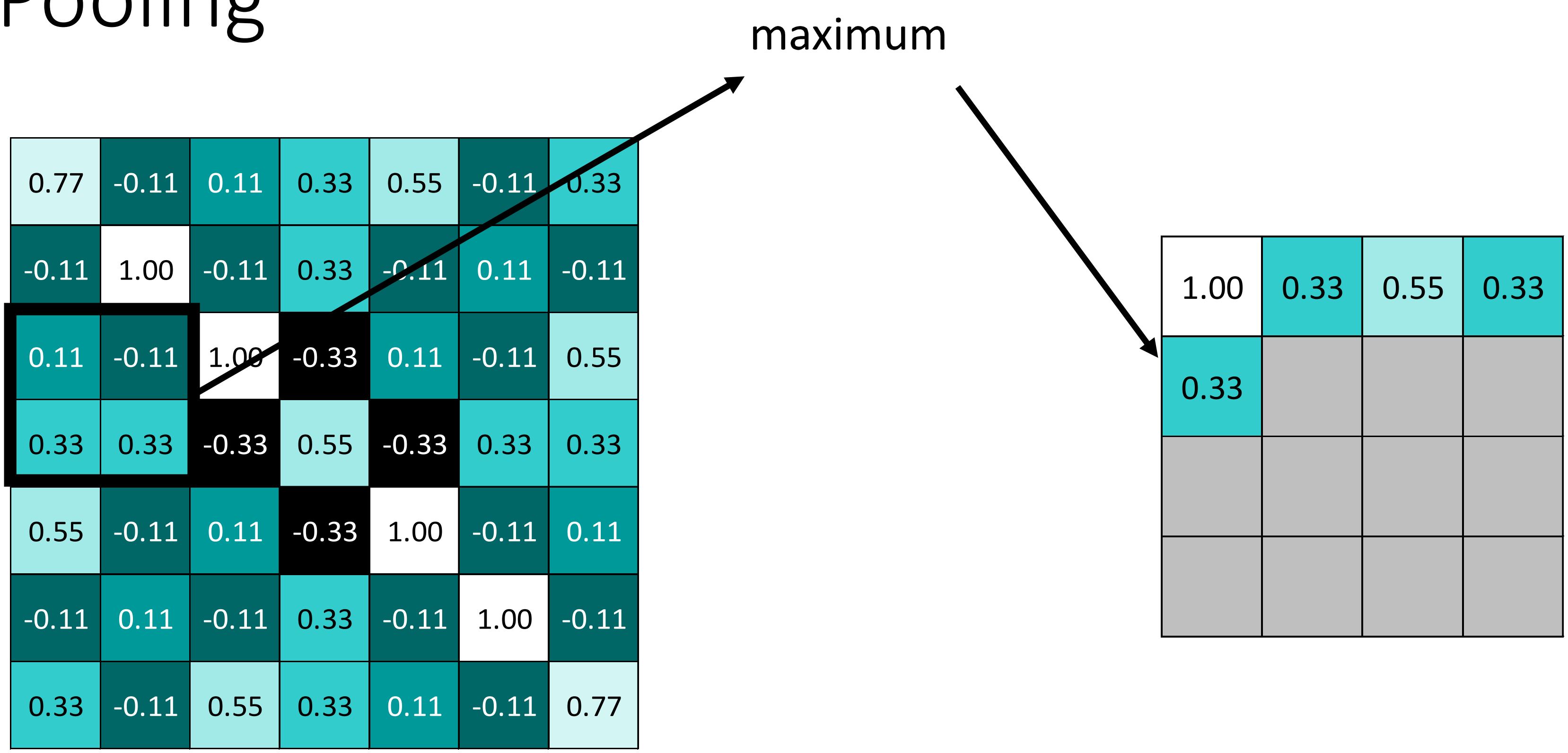
Pooling

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

maximum

1.00	0.33	0.55	0.33

Pooling


maximum
0.77 -0.11 0.11 0.33 0.55 -0.11 0.33
-0.11 1.00 -0.11 0.33 -0.11 0.11 -0.11
0.11 -0.11 1.00 -0.33 0.11 -0.11 0.55
0.33 0.33 -0.33 0.55 -0.33 0.33 0.33
0.55 -0.11 0.11 -0.33 1.00 -0.11 0.11
-0.11 0.11 -0.11 0.33 -0.11 1.00 -0.11
0.33 -0.11 0.55 0.33 0.11 -0.11 0.77
1.00 0.33 0.55 0.33
0.33

Pooling

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

max pooling

1.00	0.33	0.55	0.33
0.33	1.00	0.33	0.55
0.55	0.33	1.00	0.11
0.33	0.55	0.11	0.77

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

1.00	0.33	0.55	0.33
0.33	1.00	0.33	0.55
0.55	0.33	1.00	0.11
0.33	0.55	0.11	0.77

0.33	-0.55	0.11	-0.11	0.11	-0.55	0.33
-0.55	0.55	-0.55	0.33	-0.55	0.55	-0.55
0.11	-0.55	0.55	-0.77	0.55	-0.55	0.11
-0.11	0.33	-0.77	1.00	-0.77	0.33	-0.11
0.11	-0.55	0.55	-0.77	0.55	-0.55	0.11
-0.55	0.55	-0.55	0.33	-0.55	0.55	-0.55
0.33	-0.55	0.11	-0.11	0.11	-0.55	0.33

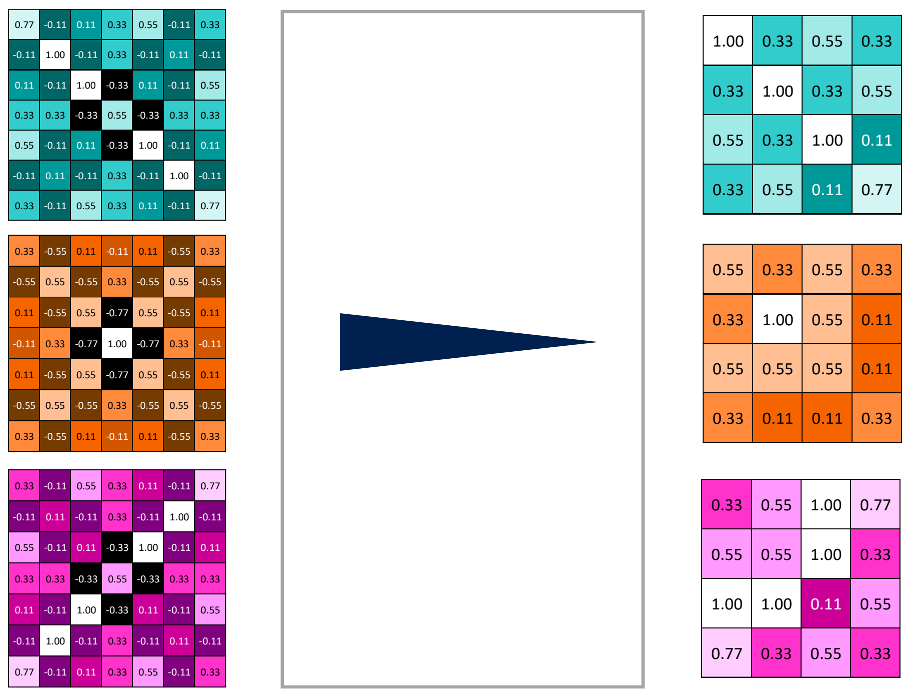
0.55	0.33	0.55	0.33
0.33	1.00	0.55	0.11
0.55	0.55	0.55	0.11
0.33	0.11	0.11	0.33

0.33	-0.11	0.55	0.33	0.11	-0.11	0.77
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.77	-0.11	0.11	0.33	0.55	-0.11	0.33

0.33	0.55	1.00	0.77
0.55	0.55	1.00	0.33
1.00	1.00	0.11	0.55
0.77	0.33	0.55	0.33

Pooling layer

A stack of images becomes a stack of smaller images.

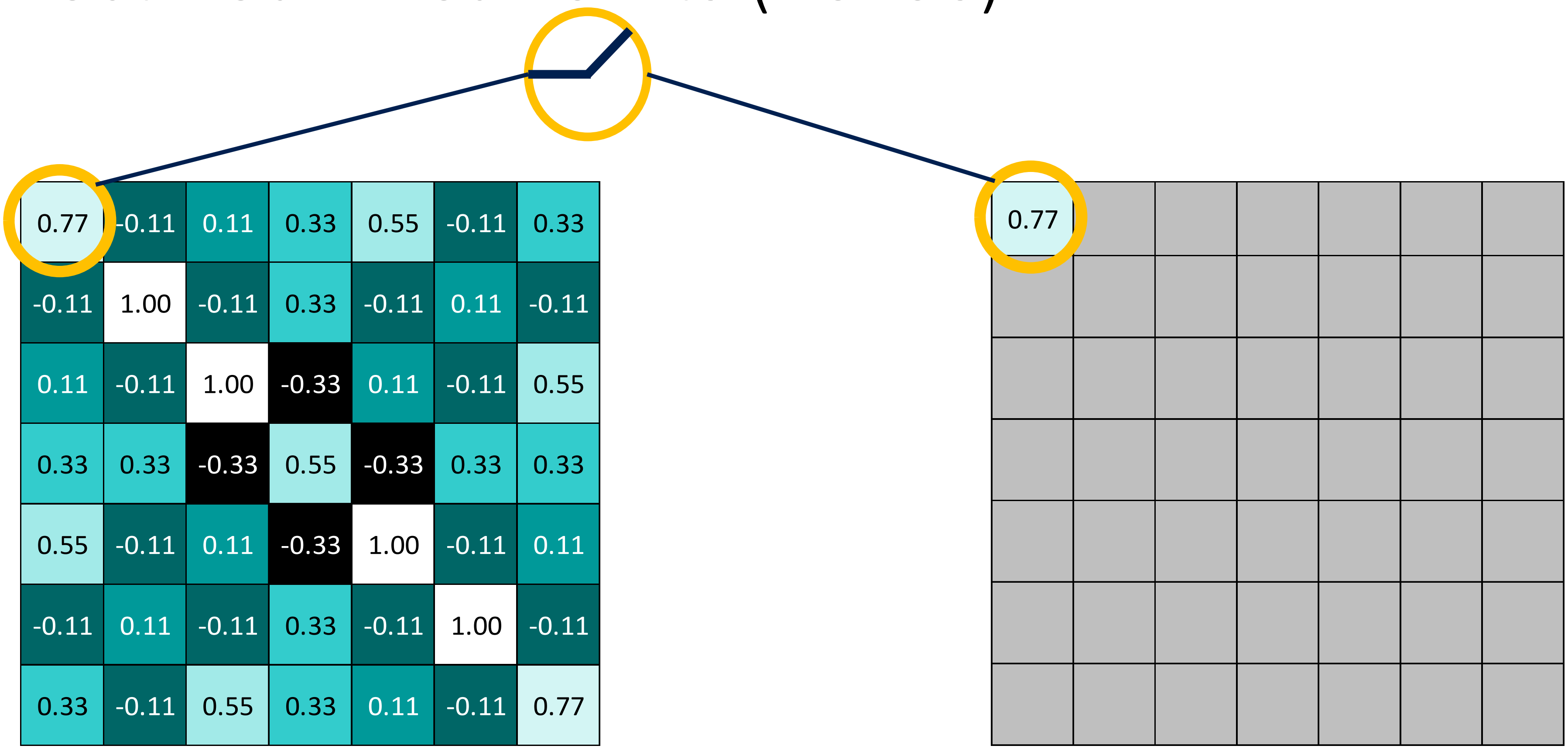


Normalization

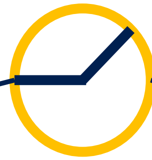
Keep the math from breaking by tweaking each of the values just a bit.

Change everything negative to zero.

Rectified Linear Units (ReLUs)



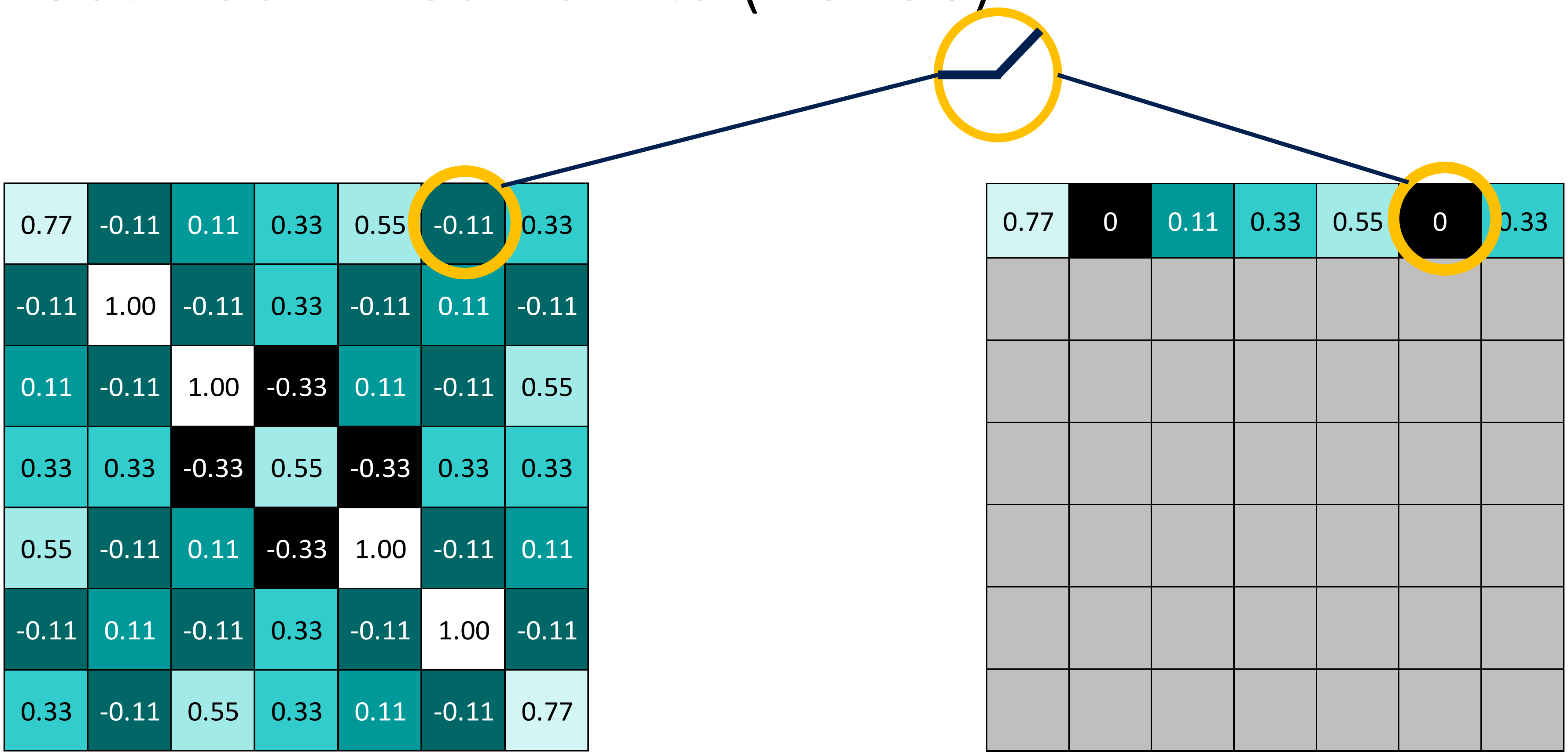
Rectified Linear Units (ReLUs)



0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

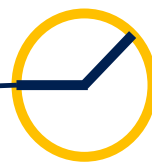
0.77	0					

Rectified Linear Units (ReLUs)



Rectified Linear Units (ReLUs)

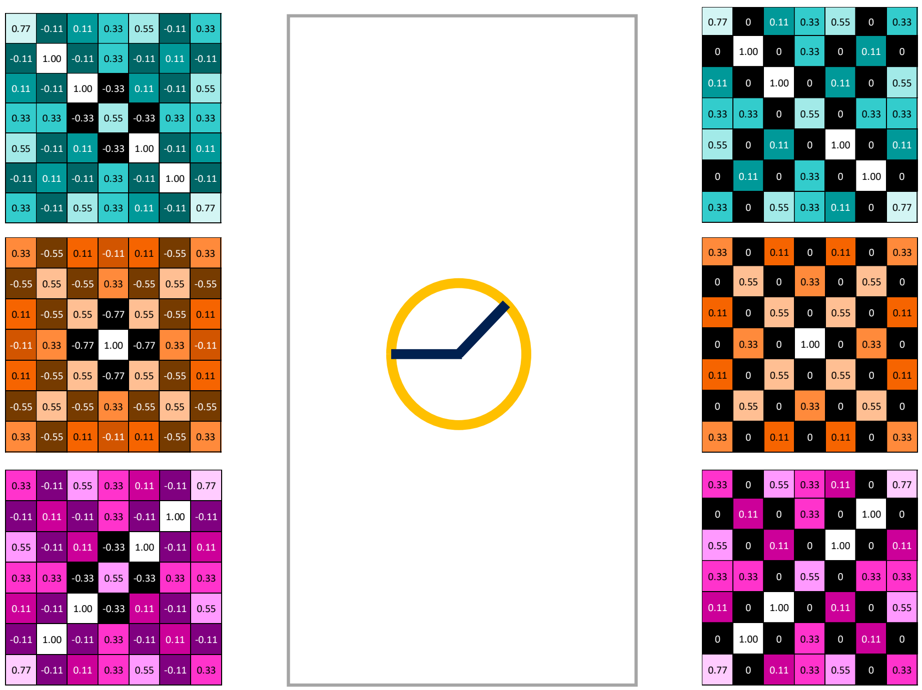
0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77



0.77	0	0.11	0.33	0.55	0	0.33
0	1.00	0	0.33	0	0.11	0
0.11	0	1.00	0	0.11	0	0.55
0.33	0.33	0	0.55	0	0.33	0.33
0.55	0	0.11	0	1.00	0	0.11
0	0.11	0	0.33	0	1.00	0
0.33	0	0.55	0.33	0.11	0	0.77

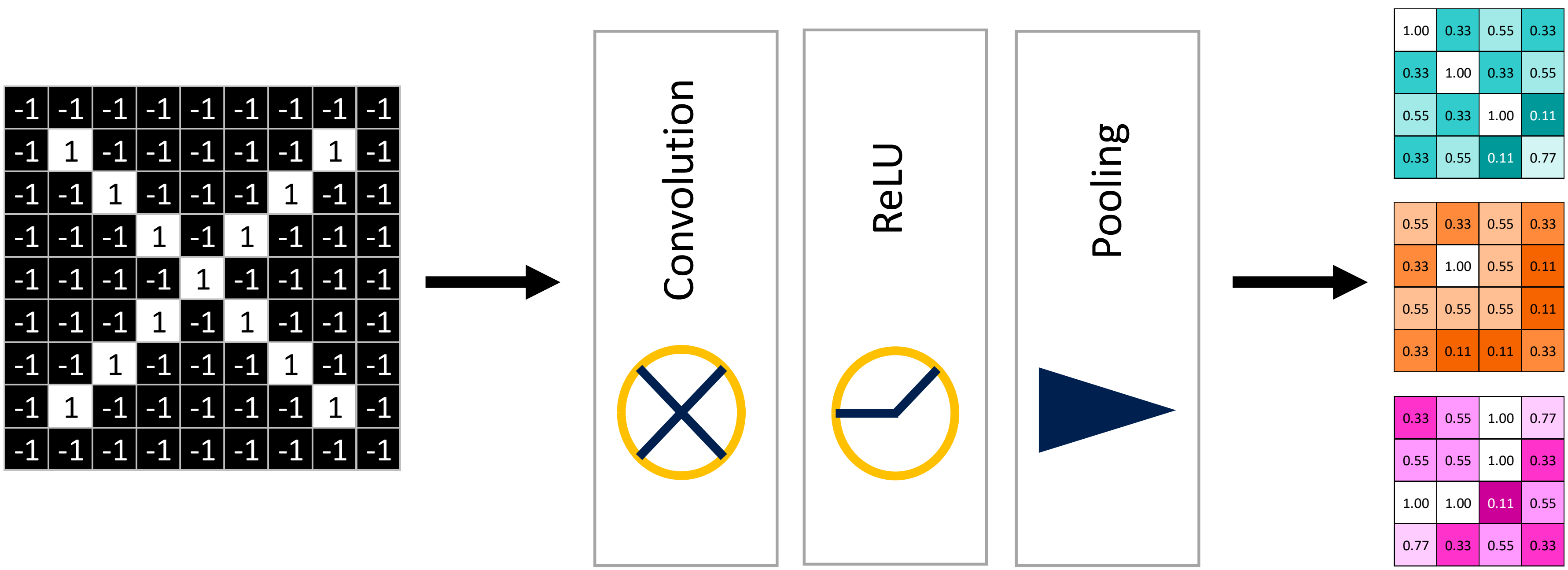
ReLU layer

A stack of images becomes a stack of images with no negative values.



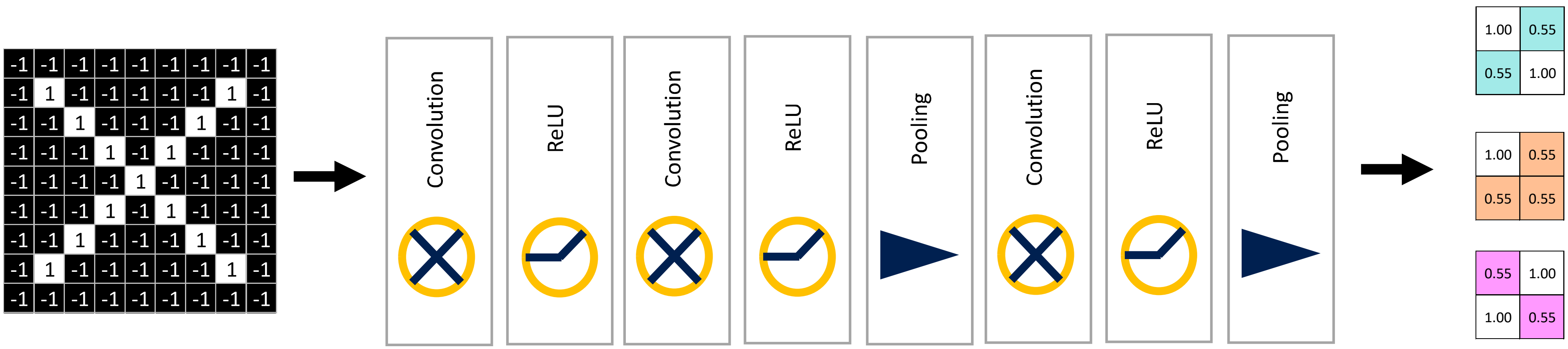
Layers get stacked

The output of one becomes the input of the next.



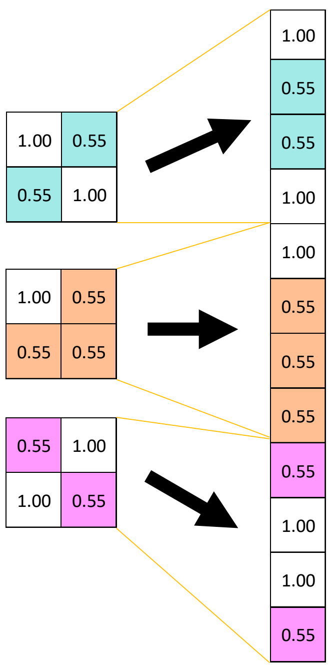
Deep stacking

Layers can be repeated several (or many) times.



Fully connected layer

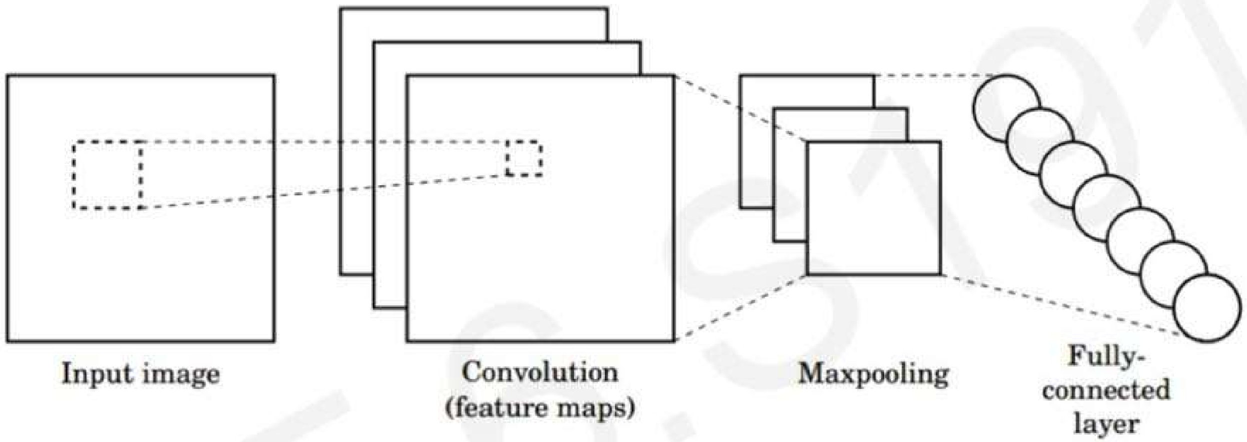
Every value gets a vote



Kerangka Bahasan

1. Pengantar
2. Pembelajaran Fitur Visual
3. Ekstraksi Fitur dan Convolution: Contoh Kasus Sederhana
4. CNN
5. Sebuah Arsitektur untuk Banyak Aplikasi
6. Ringkasan

CNN untuk Klasifikasi

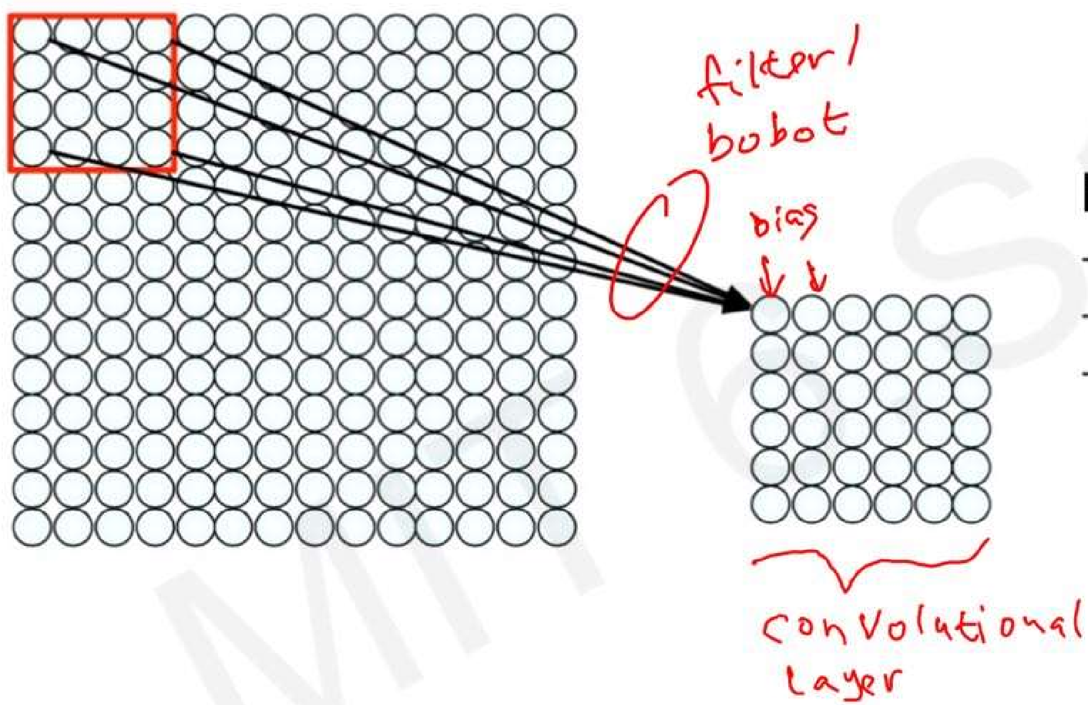


1. **Convolution**: Apply filters to generate feature maps.
2. **Non-linearity**: Often ReLU.
3. **Pooling**: Downsampling operation on each feature map.

```
tf.keras.layers.Conv2D
tf.keras.activations.*
tf.keras.layers.MaxPool2D
```

Train model with image data.
Learn weights of filters in convolutional layers.

Convolutioanl Layers: Local Connectivity



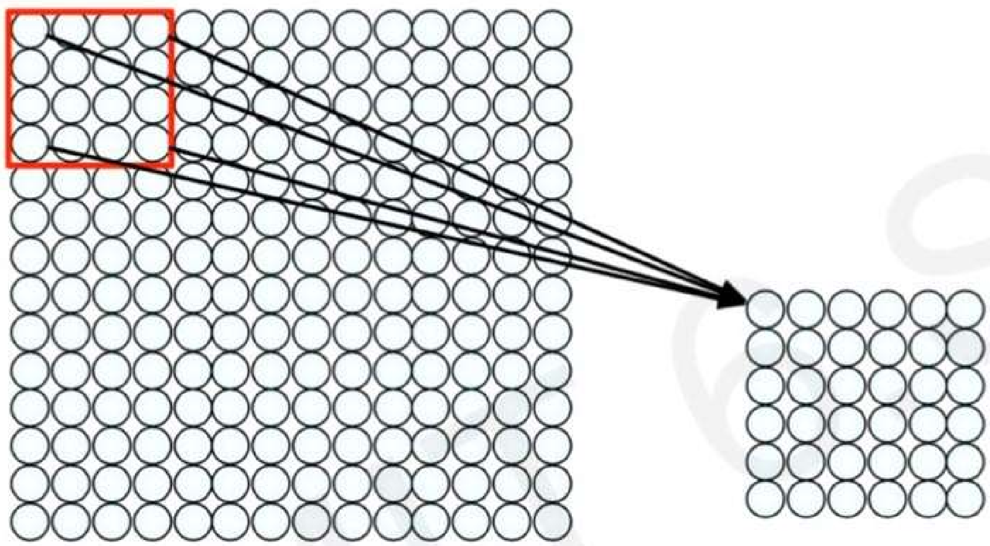
tf.keras.layers.Conv2D

For a neuron in hidden layer:

- Take inputs from patch
- Compute weighted sum
- Apply bias

Convolutioanl Layers: Local Connectivity

`tf.keras.layers.Conv2D`



For a neuron in hidden layer:

- Take inputs from patch
- Compute weighted sum
- Apply bias

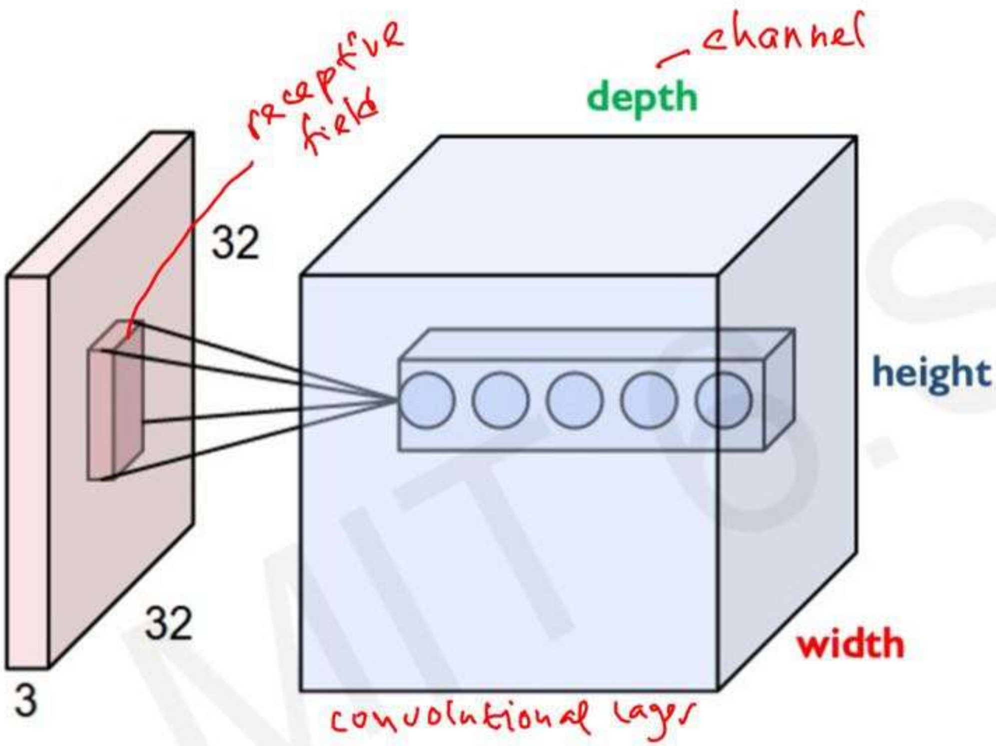
4x4 filter: matrix of weights w_{ij}

$$\sum_{i=1}^{4}\sum_{j=1}^{4} w_{ij}\, x_{i+p,j+q} + b$$

for neuron (p,q) in hidden layer

1) applying a window of weights
2) computing linear combinations
3) activating with non-linear function

CNN: Volume pada Hidden Layer



Layer Dimensions:

$h \; x \; w \; x \; d$

where h and w are spatial dimensions
d (depth) = number of filters

Stride:

Filter step size

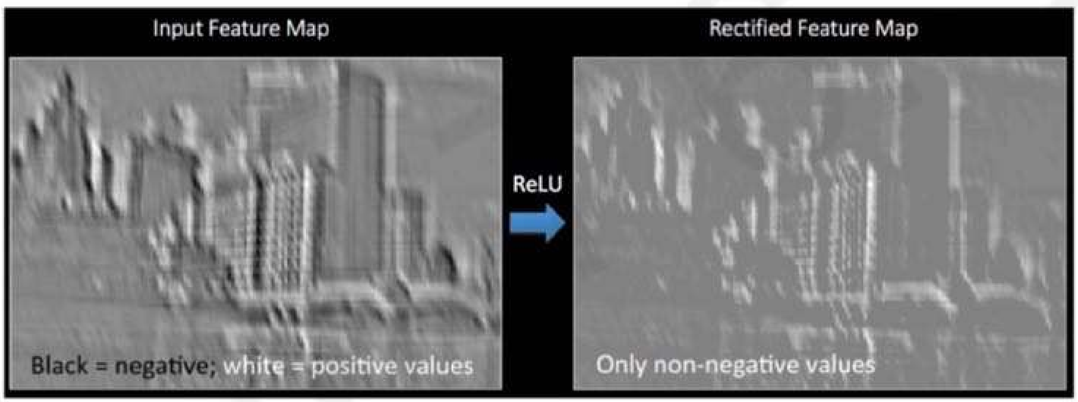
Receptive Field:

Locations in input image that
a node is path connected to

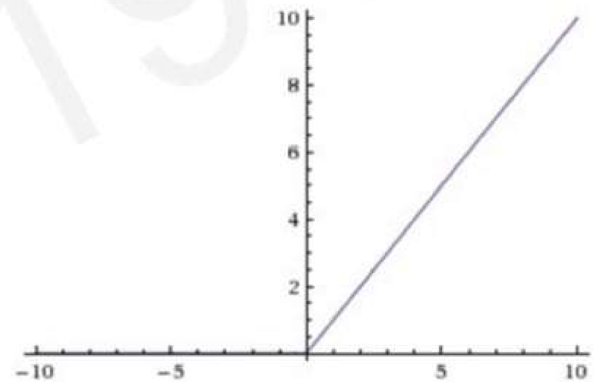
```
tf.keras.layers.Conv2D( filters=d, kernel_size=(h,w), strides=s )
```

Non-Linearity

- Apply after every convolution operation (i.e., after convolutional layers)
- ReLU: pixel-by-pixel operation that replaces all negative values by zero. **Non-linear operation!**



Rectified Linear Unit (ReLU)

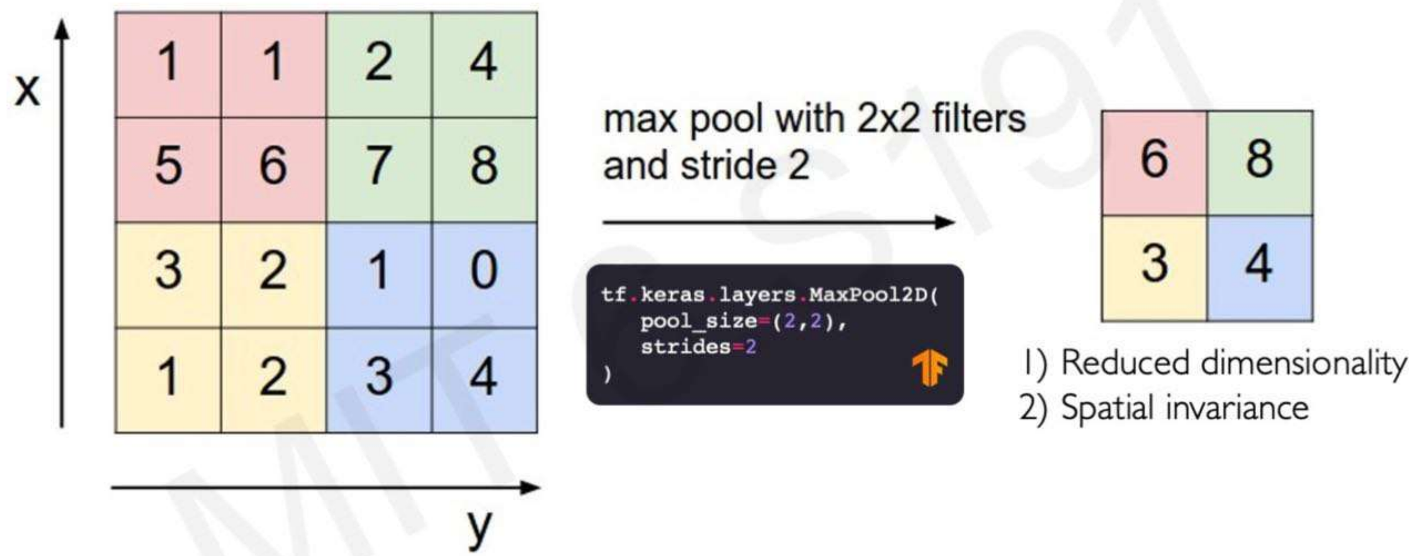


$$g(z) = \max(0, z)$$

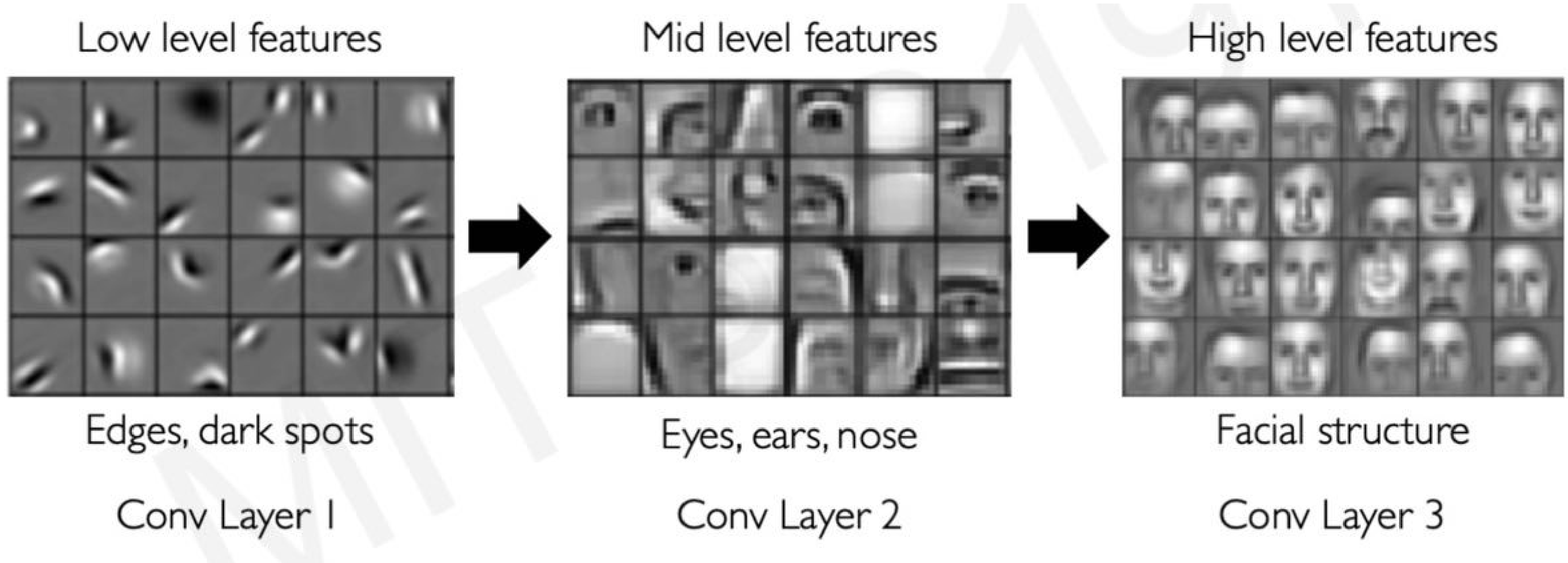
```
tf.keras.layers.ReLU
```

Pooling (pengumpulan)

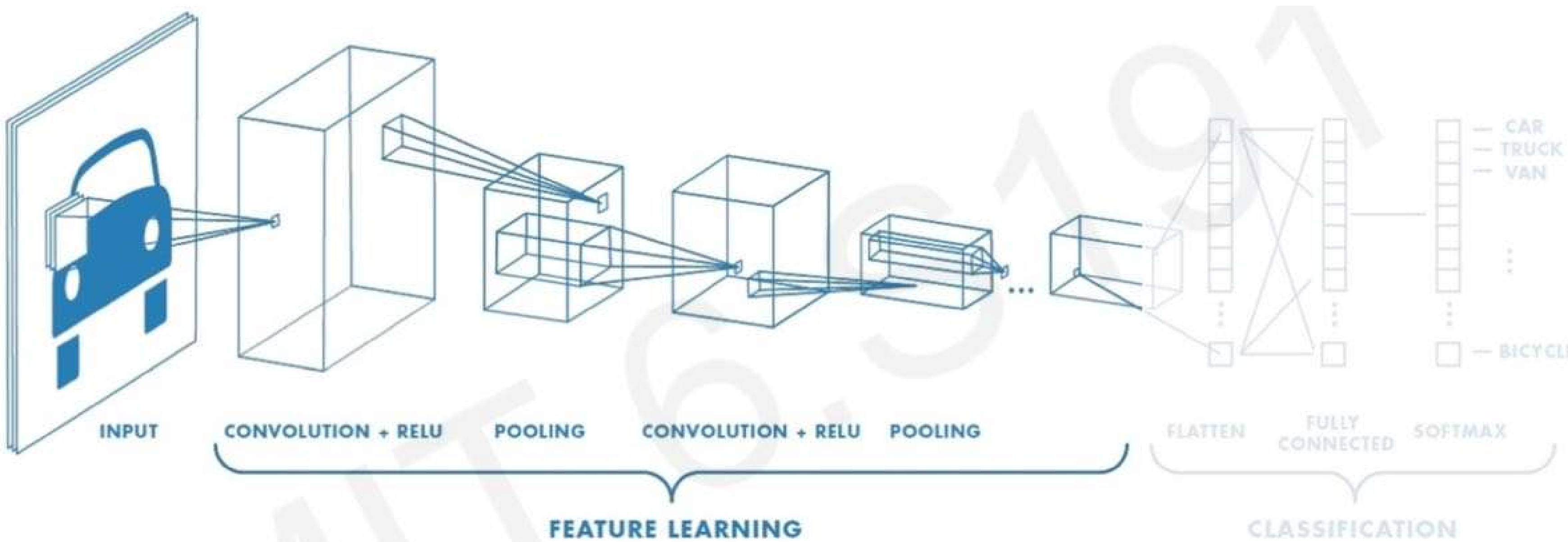
- Perlu memperkecil ukuran (downsample), namun bagaimana agar tetap menjaga informasi spatial?
- Pooling tdk ada parameter / bobot (filter sederhana tanpa bobot).



Representation Learning pada Deep CNN

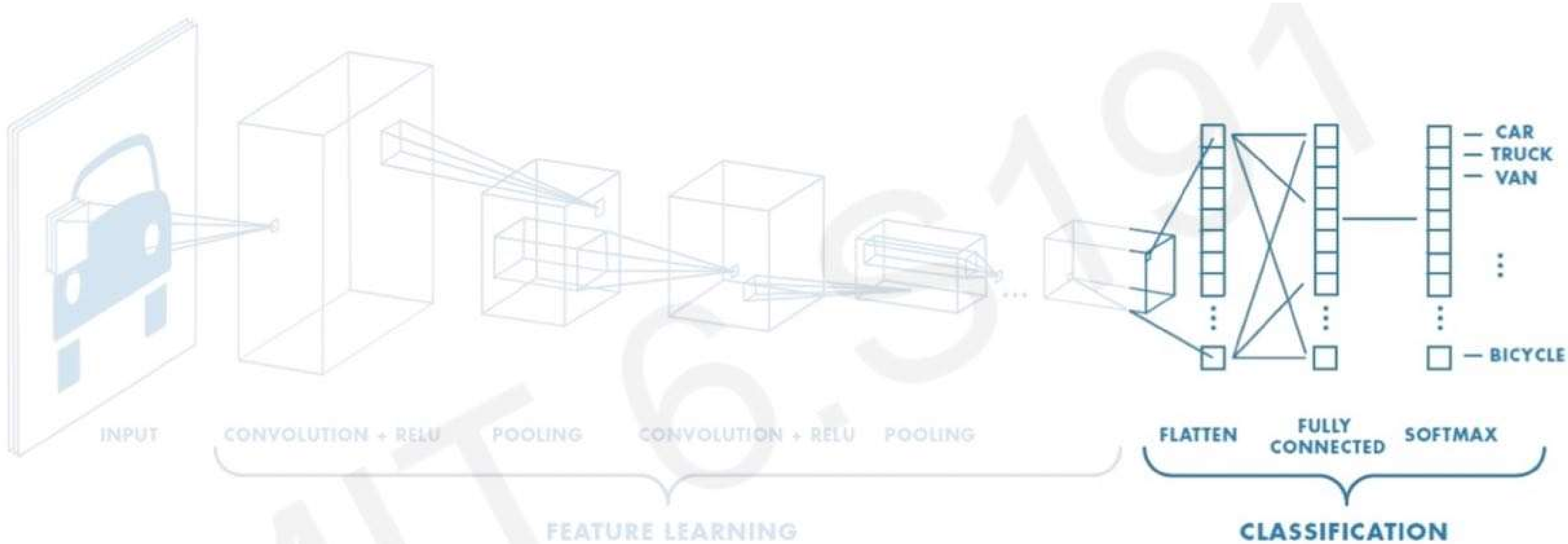


CNN untuk Classification: Feature Learning



1. Learn features in input image through **convolution**
2. Introduce **non-linearity** through activation function (real-world data is non-linear!)
3. Reduce dimensionality and preserve spatial invariance with **pooling**

CNN untuk Classification: Class Probabilities



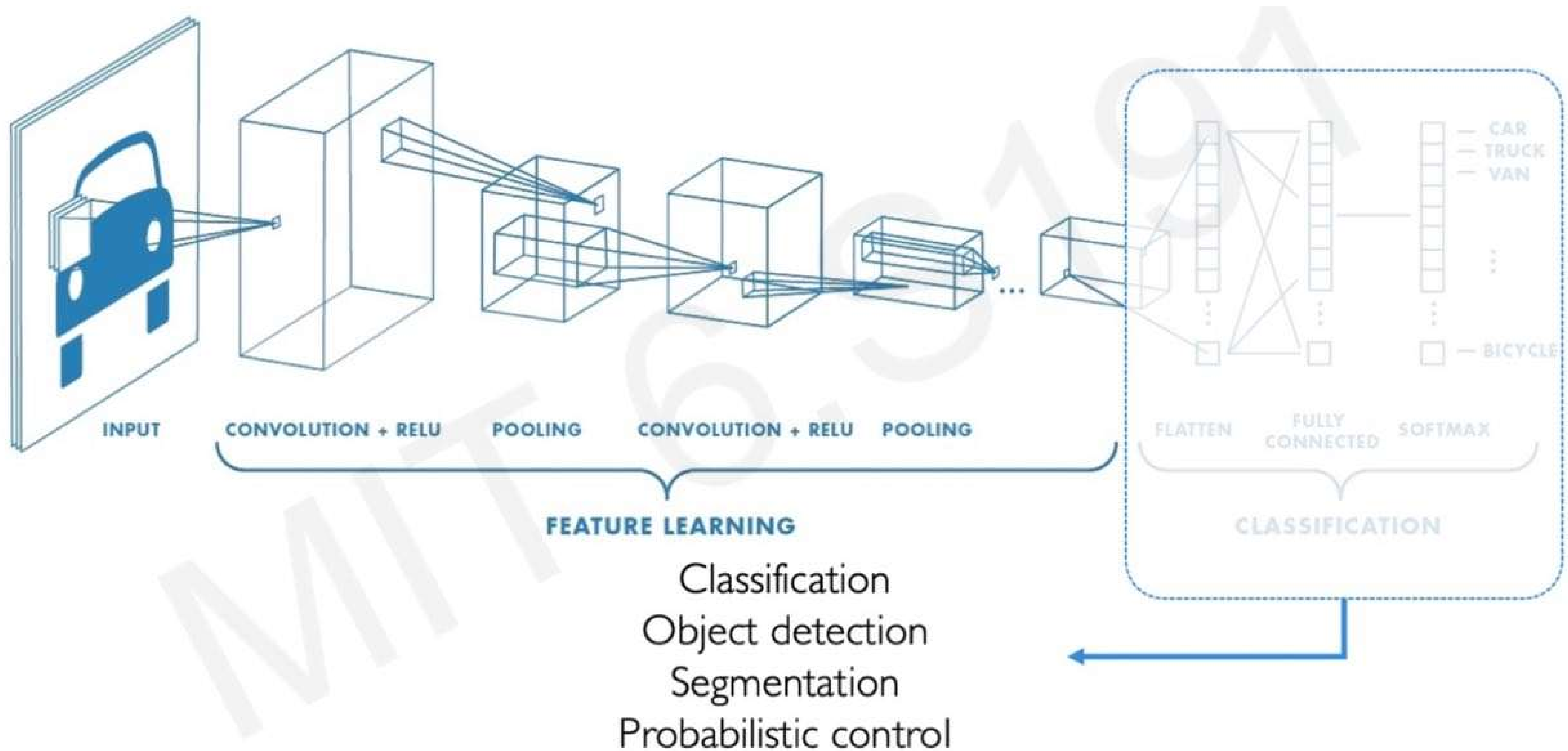
- CONV and POOL layers output high-level features of input
- Fully connected layer uses these features for classifying input image
- Express output as **probability** of image belonging to a particular class

$$\text{softmax}(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$

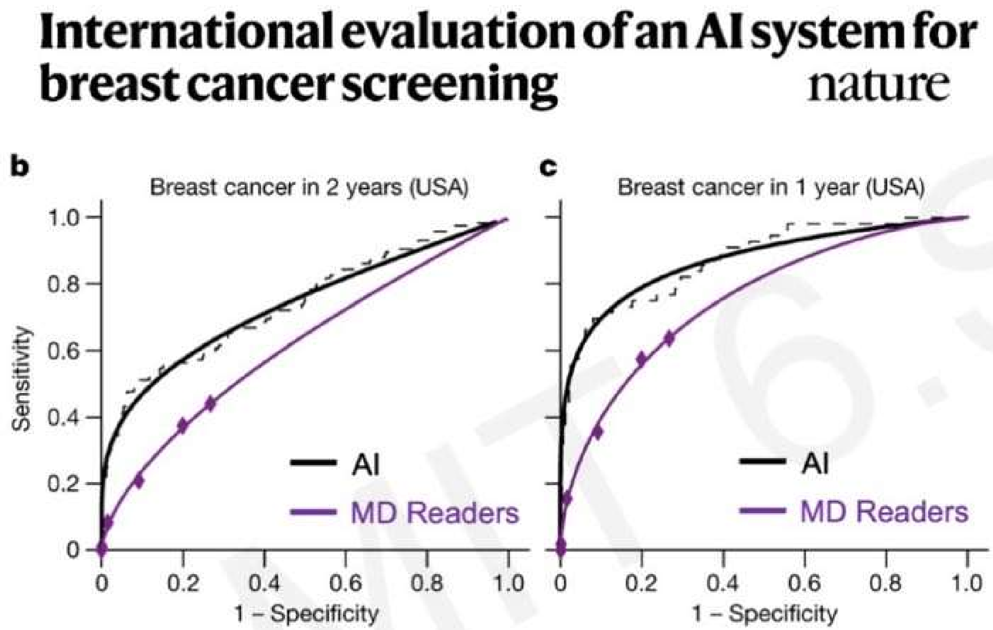
Kerangka Bahasan

1. Pengantar
2. Pembelajaran Fitur Visual
3. Ekstraksi Fitur dan Convolution: Contoh Kasus Sederhana
4. CNN
5. Sebuah Arsitektur untuk Banyak Aplikasi
6. Ringkasan

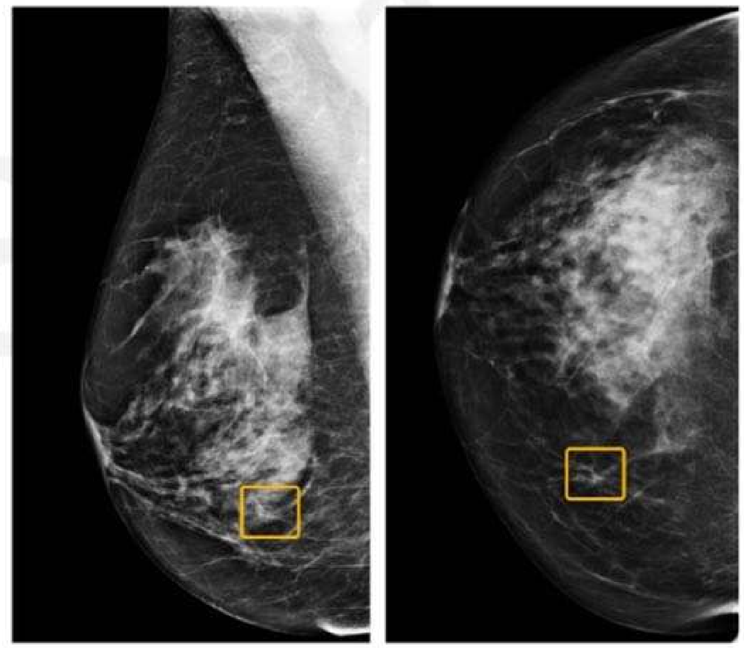
Sebuah Arsitektur untuk Banyak Aplikasi



Classification: Breast Cancer Screening

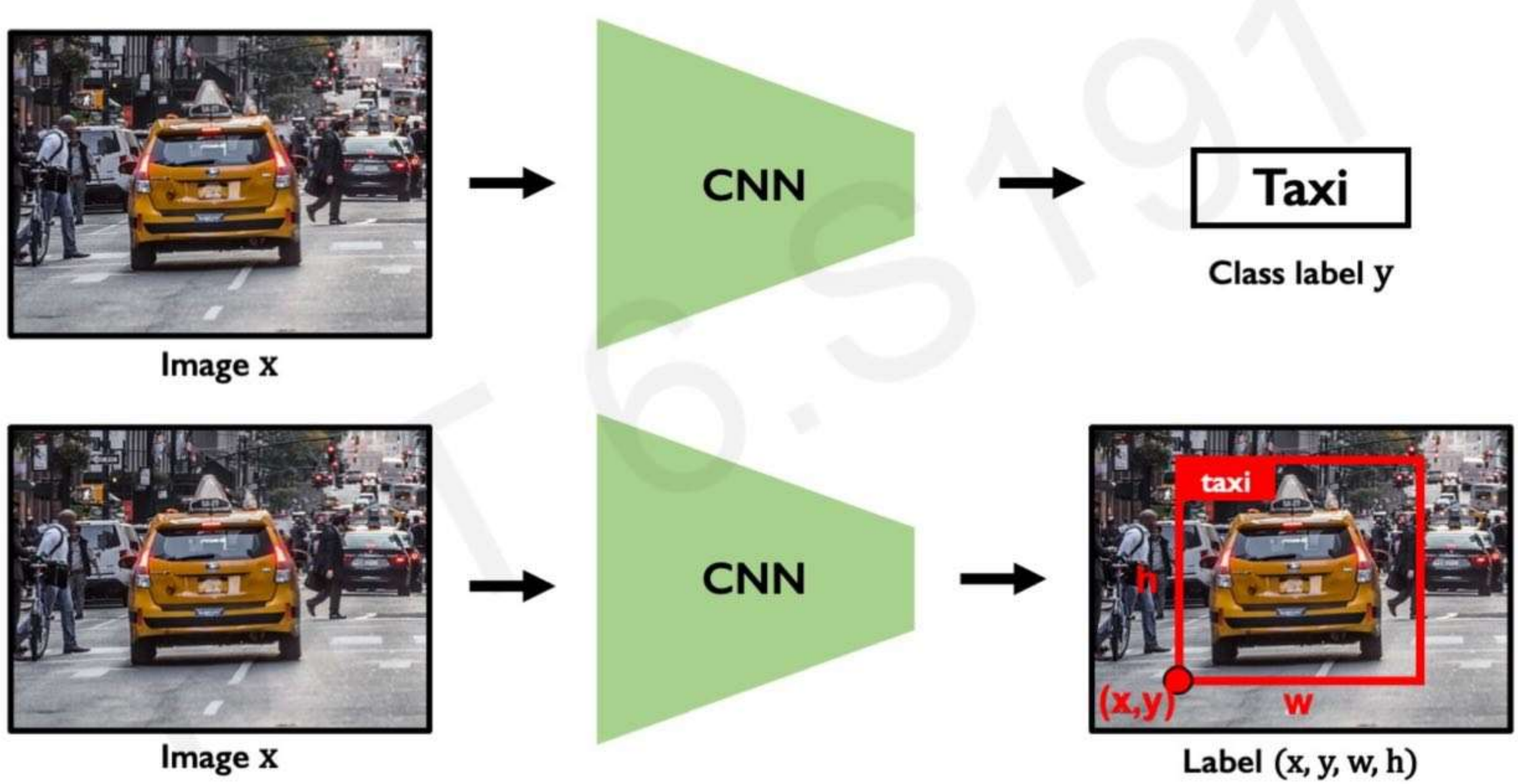


CNN-based system outperformed expert radiologists at detecting breast cancer from mammograms



Breast cancer case missed by radiologist but detected by AI

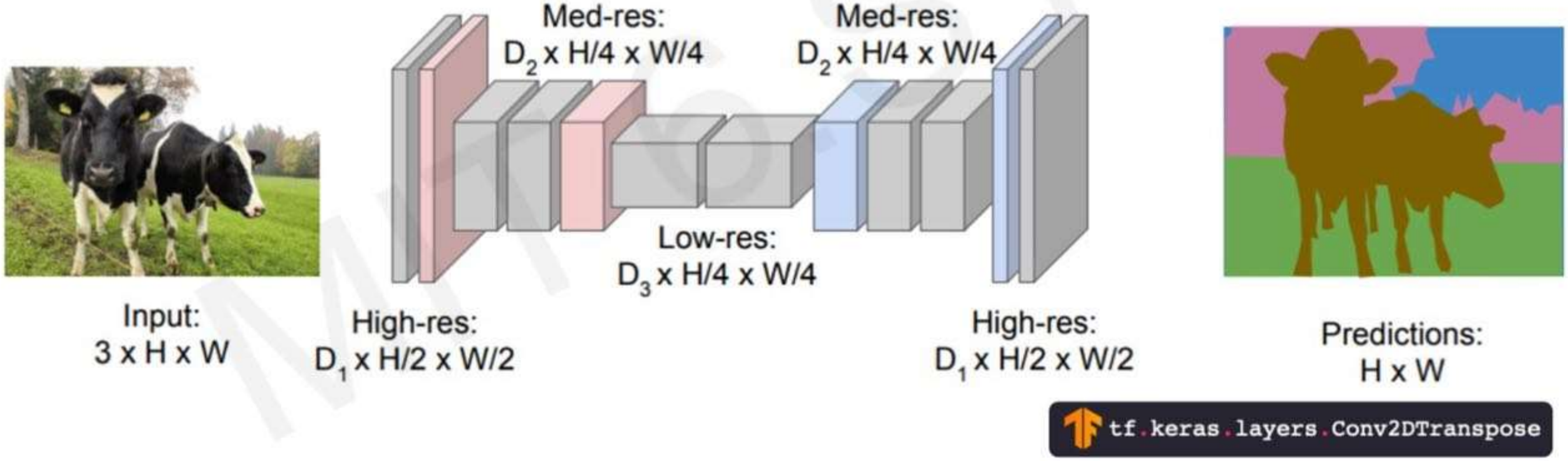
Object Detection



Object Detection (cont)

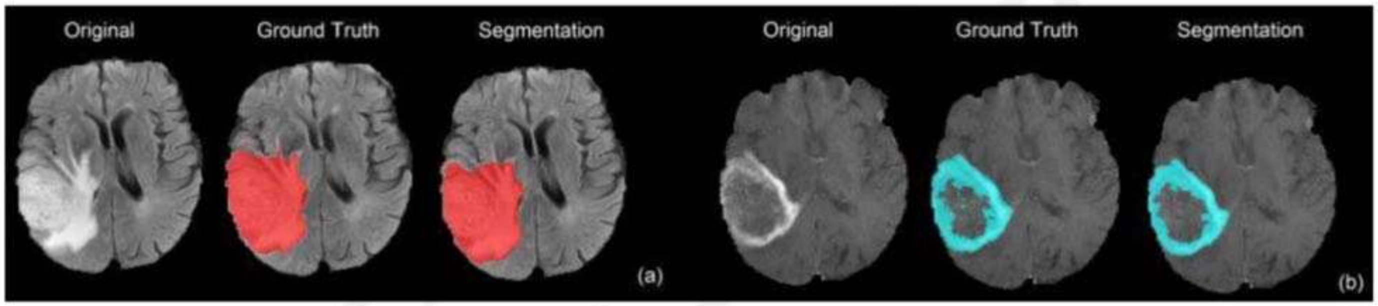
Semantic Segmentation: Fully Convolutional Networks

FCN: Fully Convolutional Network.
Network designed with all convolutional layers, with **downsampling** and **upsampling** operations

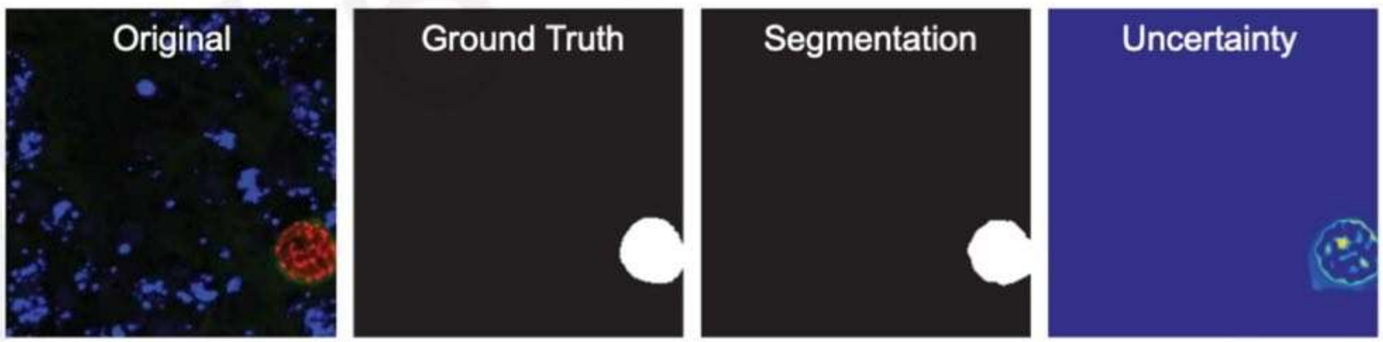


Semanric Segmentation: Biomedical Image Analysis

Brain Tumors
Dong+ *MIUA* 2017.



Malaria Infection
Soleimany+ *arXiv* 2019.



Kerangka Bahasan

1. Pengantar
2. Pembelajaran Fitur Visual
3. Ekstraksi Fitur dan Convolution: Contoh Kasus Sederhana
4. CNN
5. Sebuah Arsitektur untuk Banyak Aplikasi
6. Ringkasan

Ringkasan

- Pembelajaran fitur perlu memperhatikan properti obyek aslinya
- Pada image 2D informasi spatial perlu diakomodir
- MLP tdk efektif untuk pembelajaran learning yg mempunyai info spatial
- CNN dengan proses convolutional dan filternya merupakan pembelajaran fitur yg efektif untuk obyek 2D
- Parameter sharing pada filtering dan didukung proses pooling membuat jumlah parameter dapat diminimalisir
- Banyak aplikasi riil yang menggunakan CNN

Beberapa Istilah

- Beberapa istilah yang perlu diketahui
  - Proses convolution, convolution layer
  - Filter/kernel
  - Receptive field
  - Spatial information
  - Parameter sharing
  - Pooling
  - Channel/dimensi
  - Pembelajaran fitur
  - Fungsi aktivasi non-linear
  - Unit penyearah linear (rectified liear unit), disingkat RelU