Machine Learning in Biomedical Sciences and Bioengineering

Lecture 7
Convolutional Neural Network (CNN)

2025 version 1.00

James Choi

Convolutional neural networks (CNN)

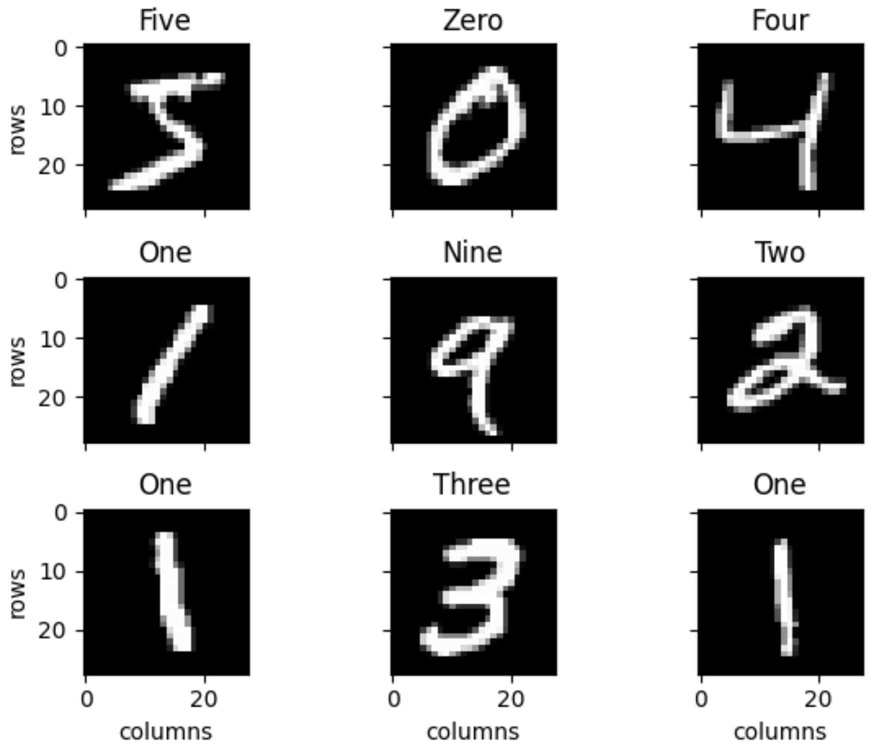
- **Convolutional neural networks (CNN)** are a specialised neural network architecture for processing data have a grid-like topology.
  - Time-series data
  - 2D images
  - Videos (2D space, 1D time)

- Many applications, including tasks in **computer vision**.

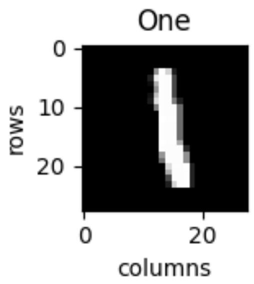
- **Classification:** classify what an image is
- **Object detection:** detect and locate objects in images
- **Segmentation:** segment images into different regions and tag each region with a semantic class.

Example: handwritten digits

- The ‘hello world’ of machine learning is to take in handwritten digits and classify them as being 0 to 9.
- The **MNIST database** stores a very large number of images of handwritten digits with labels.
  - MNIST: Modified National Institute of Standards and Technology
  - Widely available.



Fully connected neural network



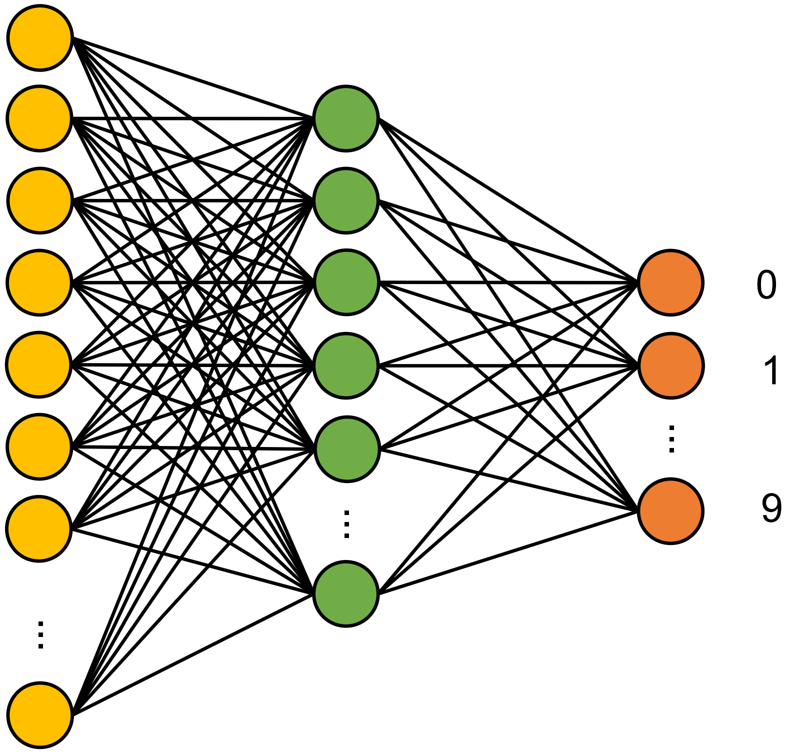
28 by 28 images

60,000 labelled images
(MNIST)

Input layer
(N = 784)

Hidden layer
(128)

Output layer
(N = 10)



Predict the correct label

Spatial information is lost by flattening

Convolutions

Input: x
4 by 4

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

Kernel: w
3 by 3

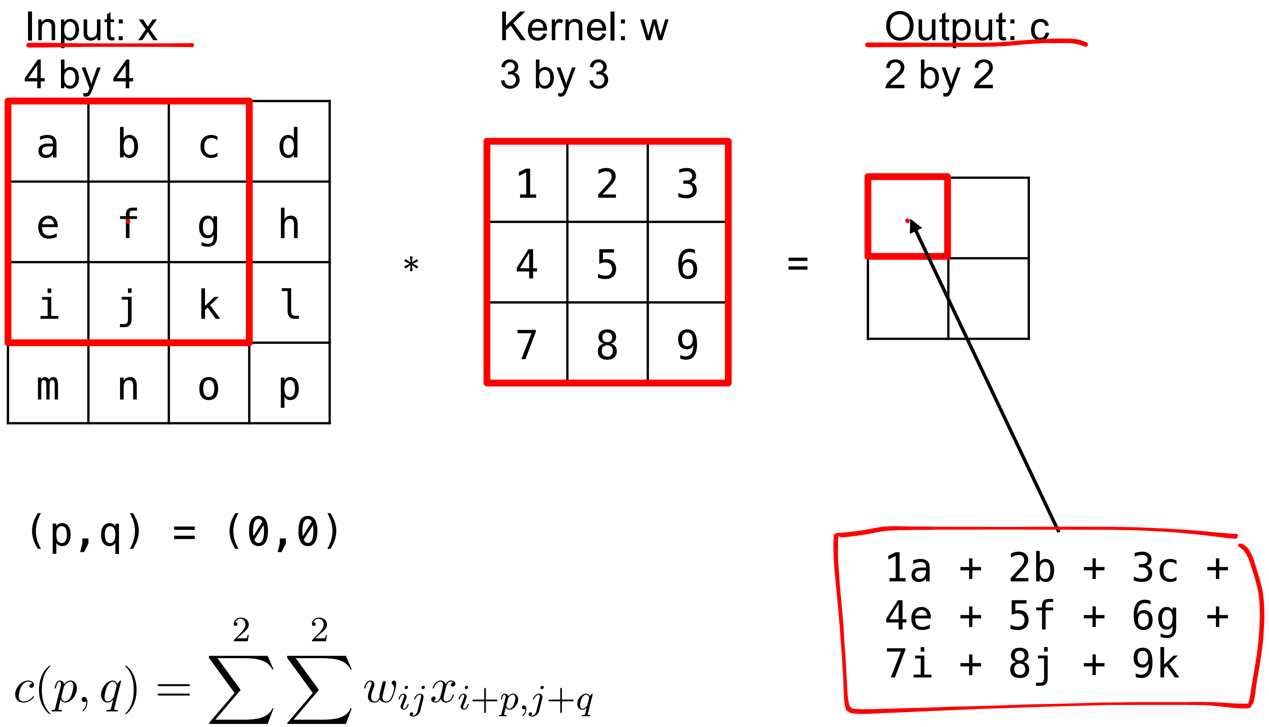
1	2	3
4	5	6
7	8	9

=

Output: c
2 by 2

$$c(p,q) = \sum_{i=0}^{2} \sum_{j=0}^{2} w_{ij} x_{i+p,j+q}$$

Convolutions



Input: x
4 by 4

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

Kernel: w
3 by 3

1	2	3
4	5	6
7	8	9

=

Output: c
2 by 2

1a + 2b + 3c +
4e + 5f + 6g +
7i + 8j + 9k

(p,q) = (0,0)

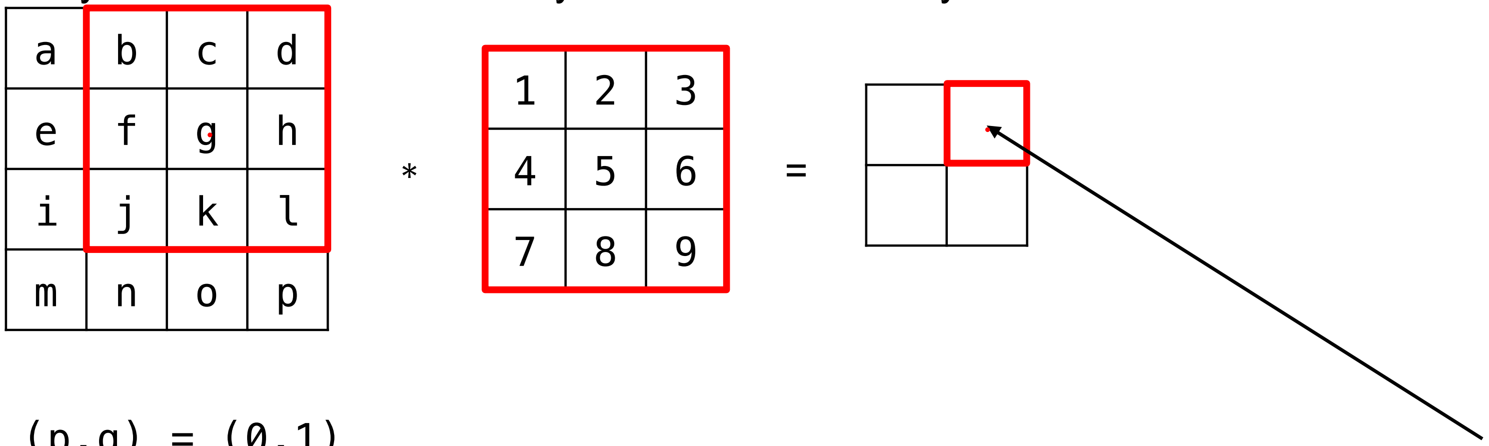
$$c(p,q) = \sum_{i=0}^{2} \sum_{j=0}^{2} w_{ij} x_{i+p,j+q}$$

Convolutions

Input: x
4 by 4

Kernel: w
3 by 3

Output: c
2 by 2



a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

1	2	3
4	5	6
7	8	9

=

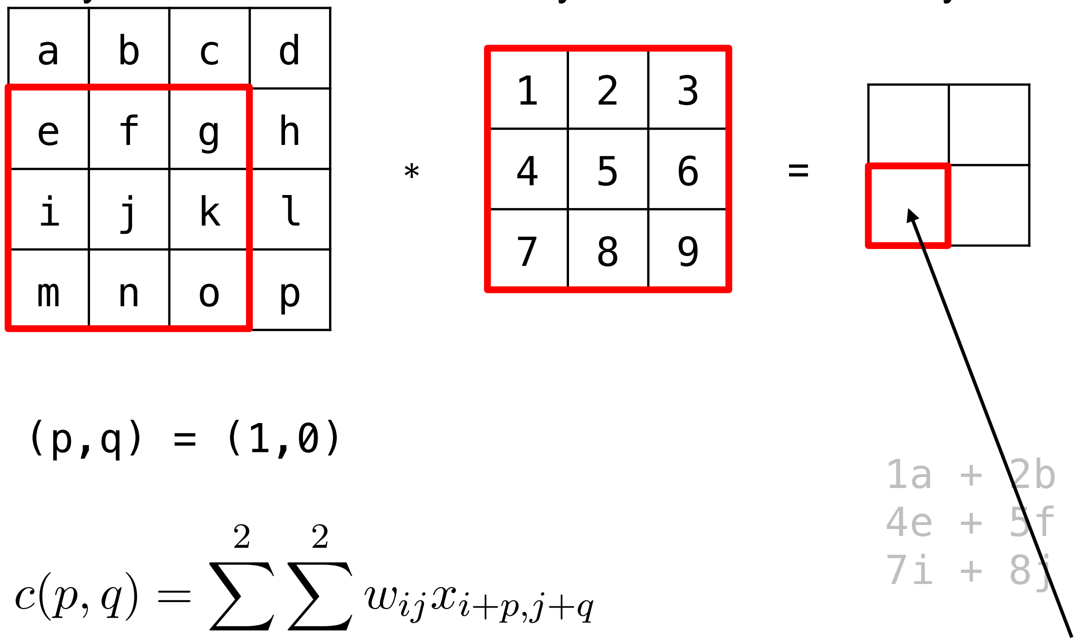
(p,q) = (0,1)

$$c(p,q) = \sum_{i=0}^{2} \sum_{j=0}^{2} w_{ij} x_{i+p,j+q}$$

1a + 2b + 3c +
4e + 5f + 6g +
7i + 8j + 9k

1b + 2c + 3d
4f + 5g + 6h
7j + 8k + 9l

Convolutions



Input: x
4 by 4

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

Kernel: w
3 by 3

1	2	3
4	5	6
7	8	9

=

Output: c
2 by 2


```
(p,q) = (1,0)
```

$$c(p,q) = \sum_{i=0}^{2}\sum_{j=0}^{2} w_{ij} x_{i+p,j+q}$$

```
1a + 2b + 3c +
4e + 5f + 6g +
7i + 8j + 9k
```

```
1b + 2c + 3d
4f + 5g + 6h
7j + 8k + 9l
```

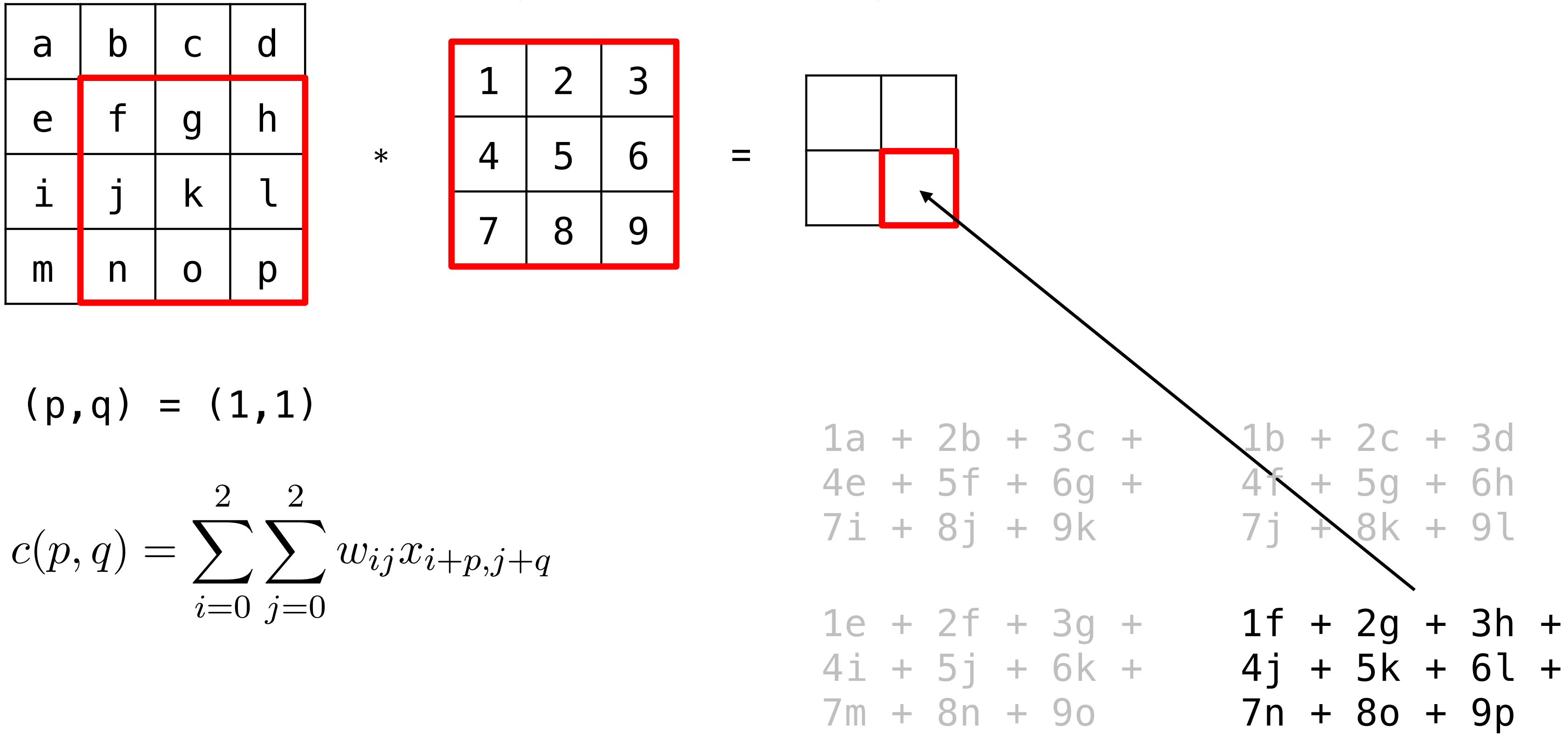
```
1e + 2f + 3g +
4i + 5j + 6k +
7m + 8n + 9o
```

Convolutions

Input: x
4 by 4

Kernel: w
3 by 3

Output: c
2 by 2



a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

1	2	3
4	5	6
7	8	9

=


```
(p,q) = (1,1)
```

$$c(p,q) = \sum_{i=0}^{2} \sum_{j=0}^{2} w_{ij} x_{i+p,j+q}$$

```
1a + 2b + 3c +      1b + 2c + 3d
4e + 5f + 6g +      4f + 5g + 6h
7i + 8j + 9k        7j + 8k + 9l

1e + 2f + 3g +      1f + 2g + 3h +
4i + 5j + 6k +      4j + 5k + 6l +
7m + 8n + 9o        7n + 8o + 9p
```

Kernels

No effect

0	0	0
0	1	0
0	0	0

Prewitt operator
edge detection

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

<= x direction =>

<= y direction =>

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

... and many more!

Mean filter
blurring effect

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Gaussian blur
blurring effect

$\frac{1}{16}$

1	2	1
2	4	2
1	2	1

Sobel operator
edge detection

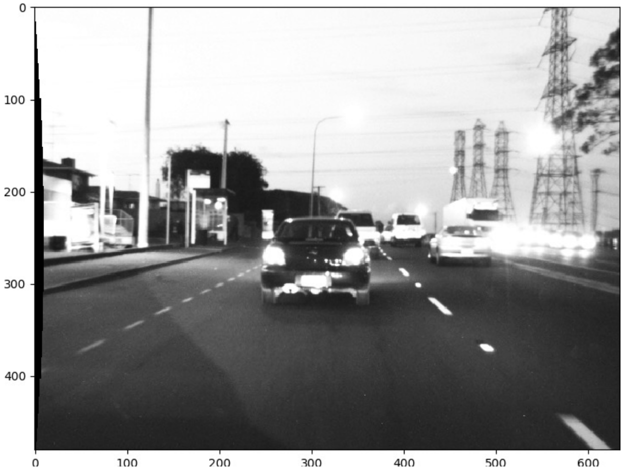
-1	0	1
-2	0	2
-1	0	1

<= x direction =>

<= y direction =>

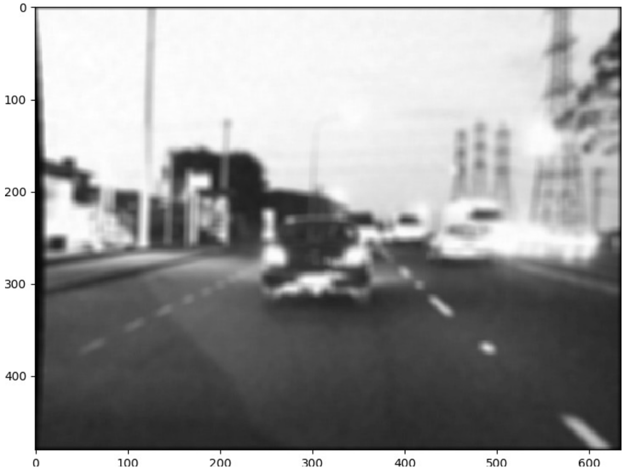
-1	-2	-1
0	0	0
1	2	1

Kernels



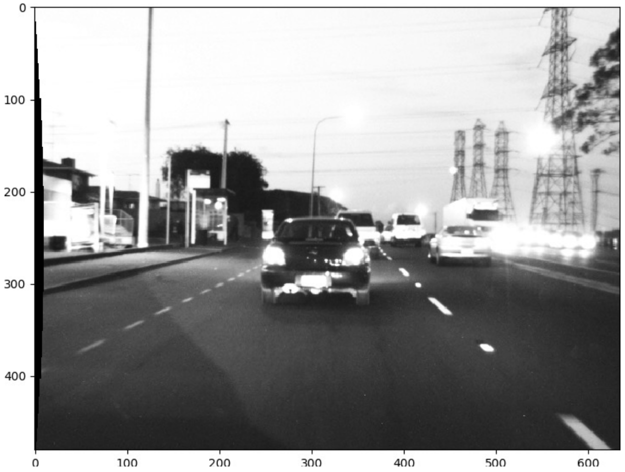
Mean filter
blurring effect

$$* \quad \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad =$$



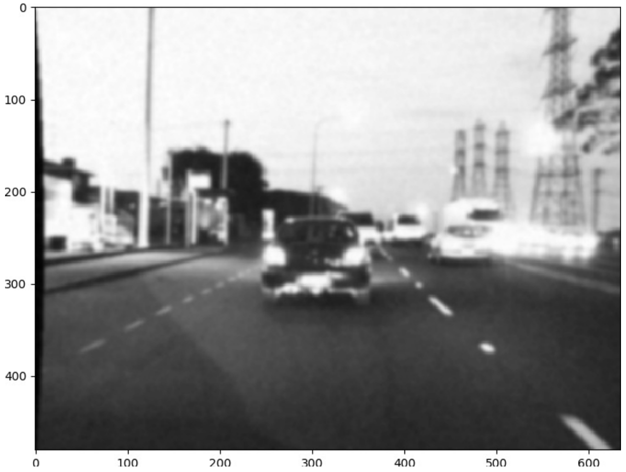
https://sbme-tutorials.github.io/2018/cv/notes/4_week4.html

Kernels



Gaussian blur
blurring effect

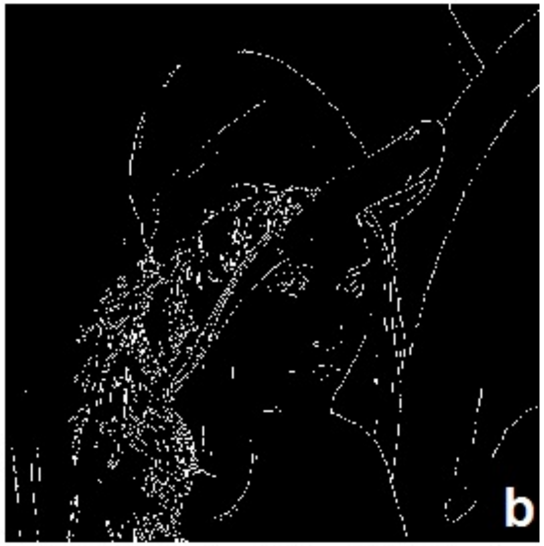
$$* \quad \frac{1}{16} \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array} \quad =$$



https://sbme-tutorials.github.io/2018/cv/notes/4_week4.html

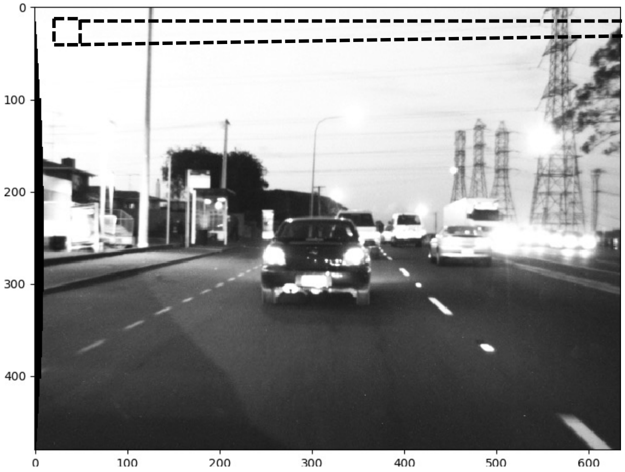
Kernels

$*$ **sobel operators** =



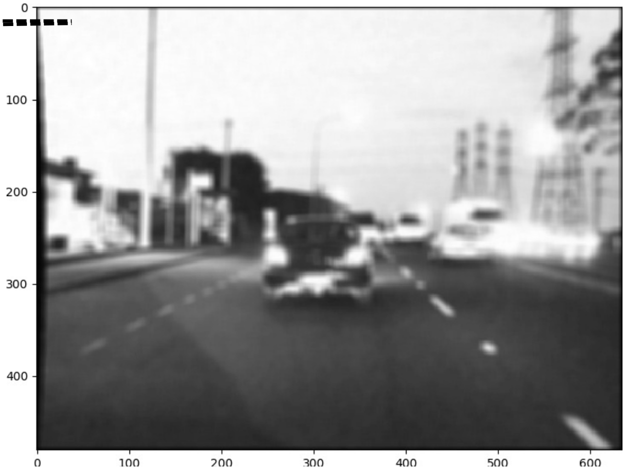
Tiwari et al. Circuits Systems and Signal Processing (2018) 37(2)

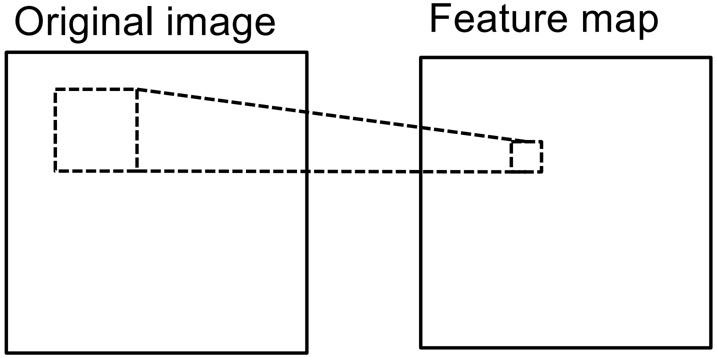
Kernels



Mean filter
blurring effect

$$* \quad \frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} \quad =$$





https://sbme-tutorials.github.io/2018/cv/notes/4_week4.html

Pooling

- **Pooling** is an operation that reduces the spatial size of feature maps.
  - It summarises regions (patches) of the input
  - Commonly done by taking the maximum (max pooling) or the average (average pooling) value within each patch.
- The main purposes of pooling are:
  - **Downsampling**: reduces the number of parameters and computations in the network.
  - **Focus on key features**: keeps the most important information while discarding less relevant details.
  - **Adds robustness**: provides some invariance to small translations or distortions in the input.

Max pooling

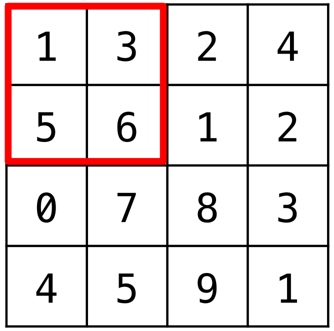
Input: x
4 by 4

1	3	2	4
5	6	1	2
0	7	8	3
4	5	9	1

Output: p
2 by 2

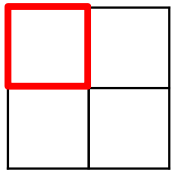
Max pooling

Input: x
4 by 4



1	3	2	4
5	6	1	2
0	7	8	3
4	5	9	1

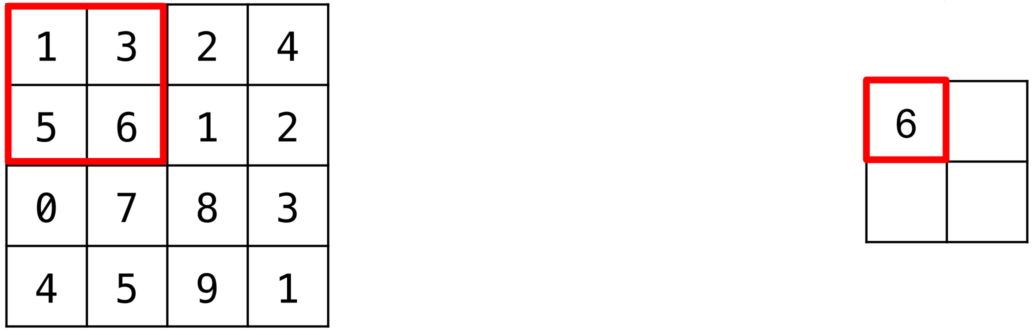
Output: p
2 by 2



Max pooling

Input: x
4 by 4

Output: p
2 by 2



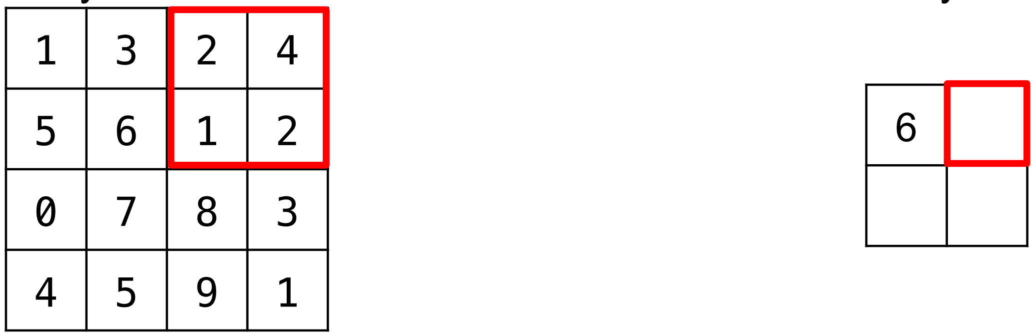
1	3	2	4
5	6	1	2
0	7	8	3
4	5	9	1

6	

Max pooling

Input: x
4 by 4

Output: p
2 by 2



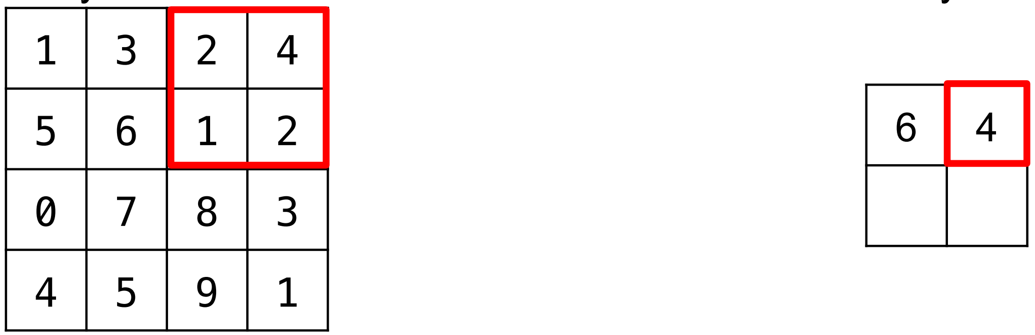
1	3	2	4
5	6	1	2
0	7	8	3
4	5	9	1

6	

Max pooling

Input: x
4 by 4

Output: p
2 by 2



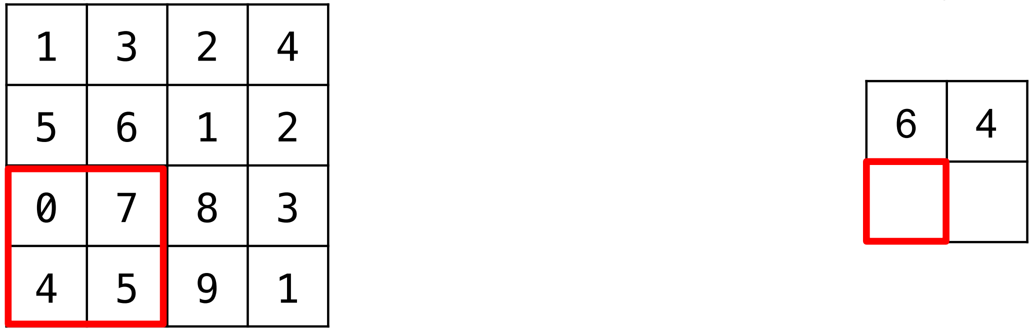
1	3	2	4
5	6	1	2
0	7	8	3
4	5	9	1

6	4

Max pooling

Input: x
4 by 4

Output: p
2 by 2



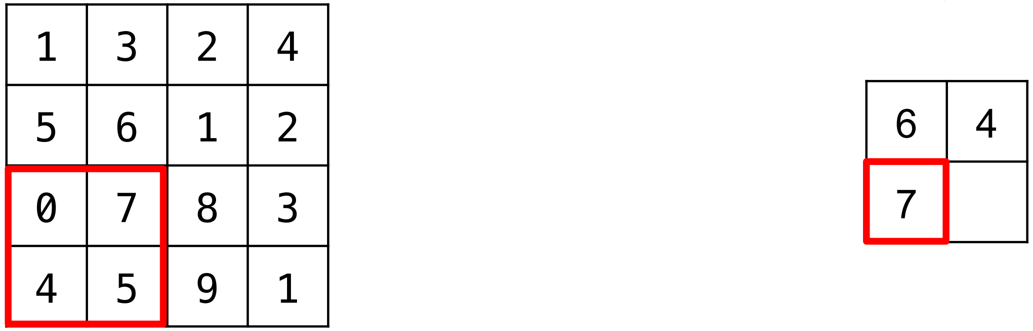
1	3	2	4
5	6	1	2
0	7	8	3
4	5	9	1

6	4

Max pooling

Input: x
4 by 4

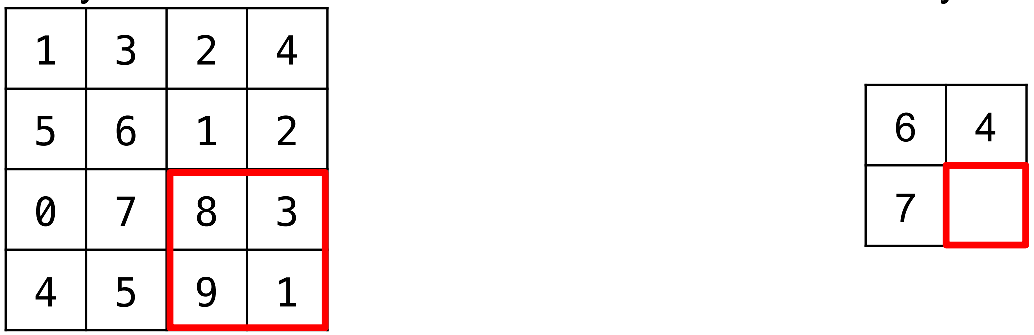
Output: p
2 by 2



1	3	2	4
5	6	1	2
0	7	8	3
4	5	9	1

6	4
7	

Max pooling



Input: x
4 by 4

1	3	2	4
5	6	1	2
0	7	8	3
4	5	9	1

Output: p
2 by 2

6	4
7	

Max pooling

Input: x
4 by 4

1	3	2	4
5	6	1	2
0	7	8	3
4	5	9	1

Output: p
2 by 2

6	4
7	9

Max pooling

Input: x
4 by 4

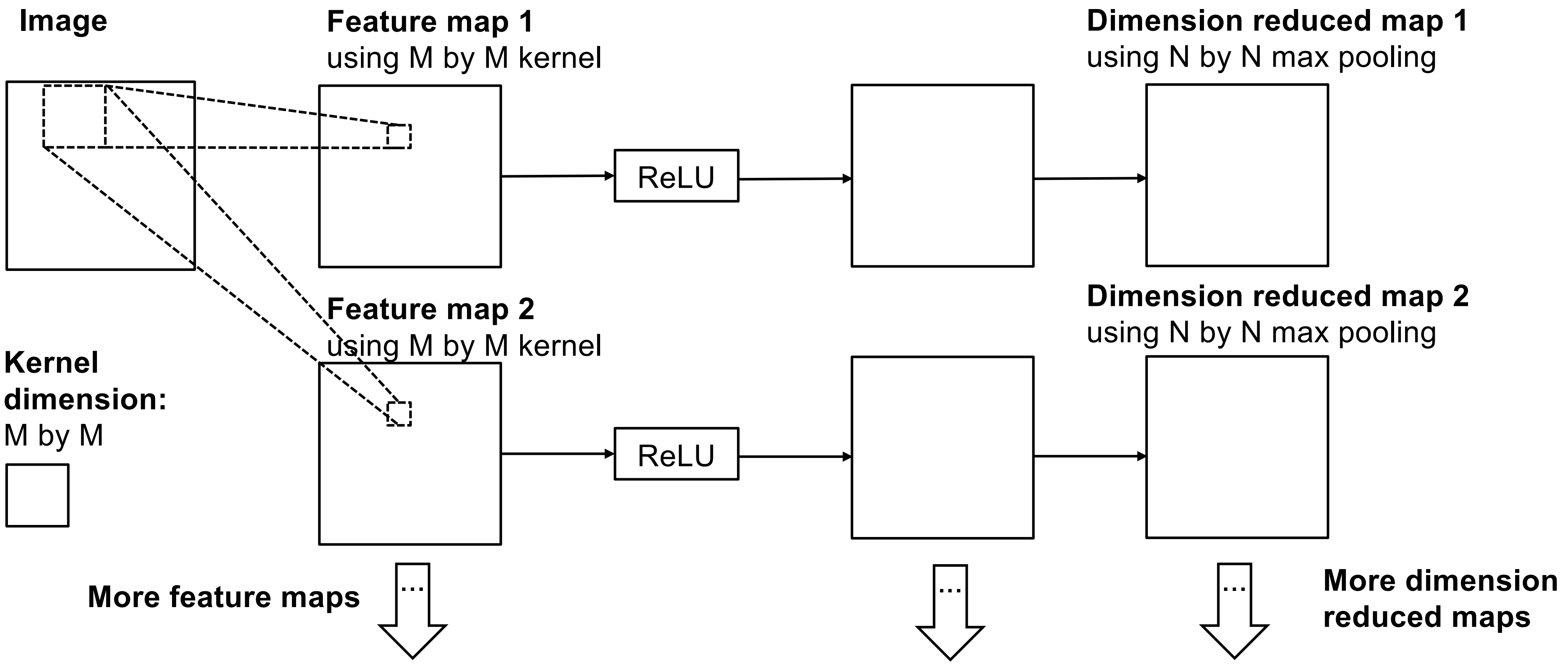
1	3	2	4
5	6	1	2
0	7	8	3
4	5	9	1

Output: p
2 by 2

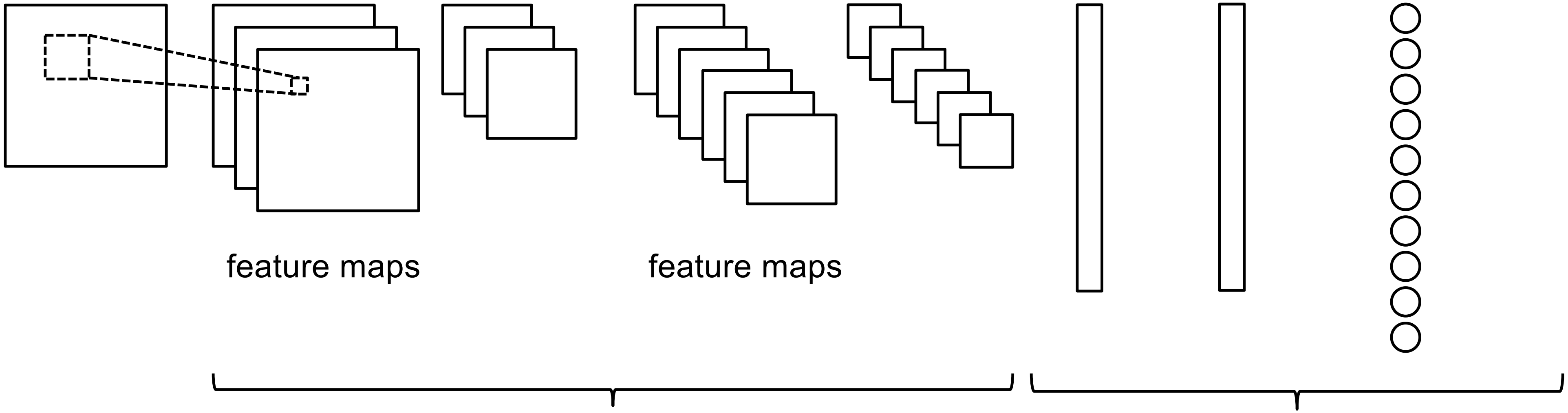
6	4
7	9

Convolution

- A convolutional layers consists of many feature maps plus activation function



CNN architecture



=32*5*5

28 x 28

Input:
(image)

24 x 24

Convolution
+ ReLU

12 x 12

Max
pool

10 x 10

Convolution
+ ReLU

5 x 5

Max
pool

256

Hidden
layer 1

10

Hidden
layer 2

Output:
Classifications

feature maps

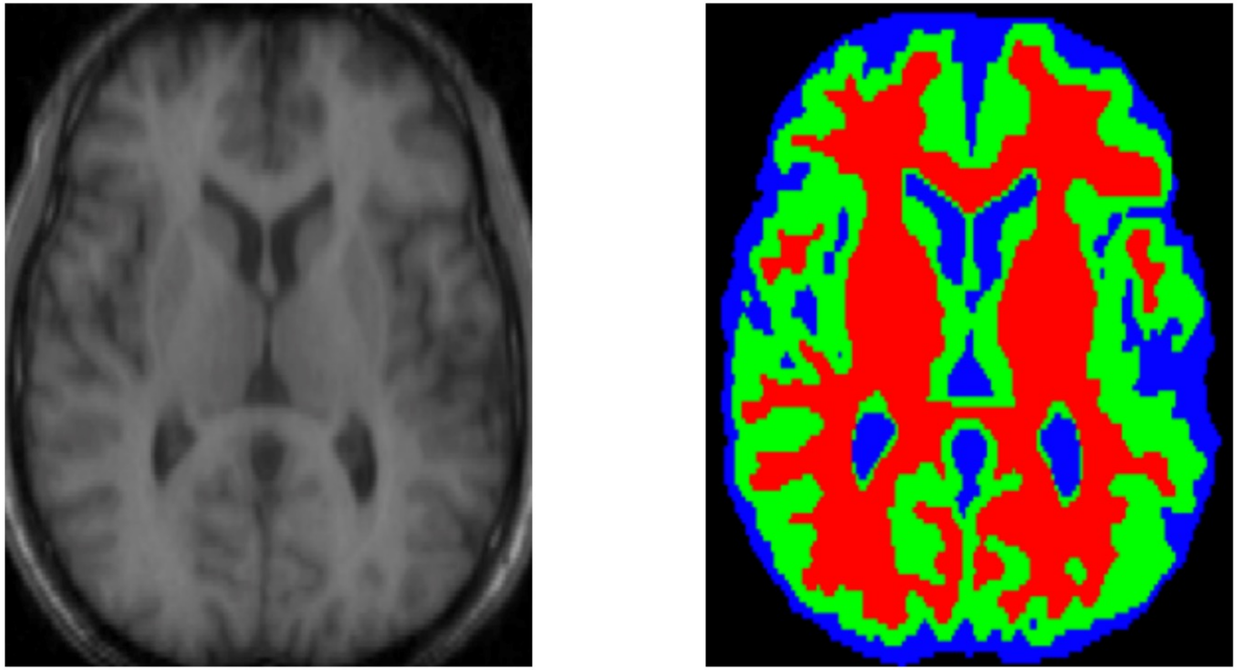
feature maps

feature learning

classification

object detection
segmentation
...

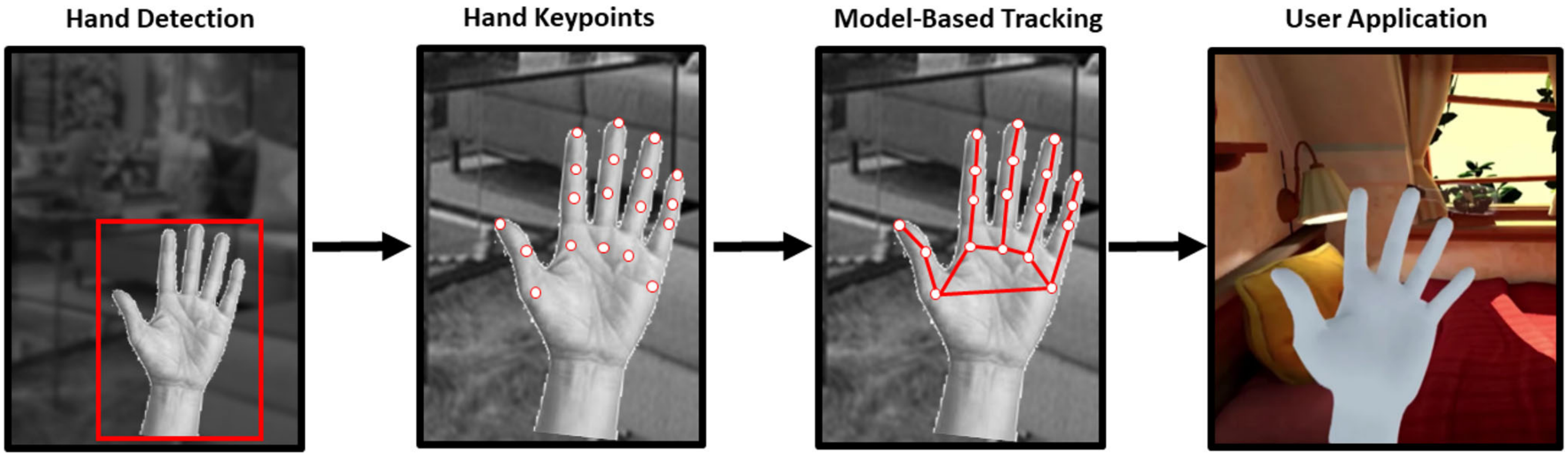
Medical Image Segmentation



(a) Axial slice

(b) Tissue segmentation

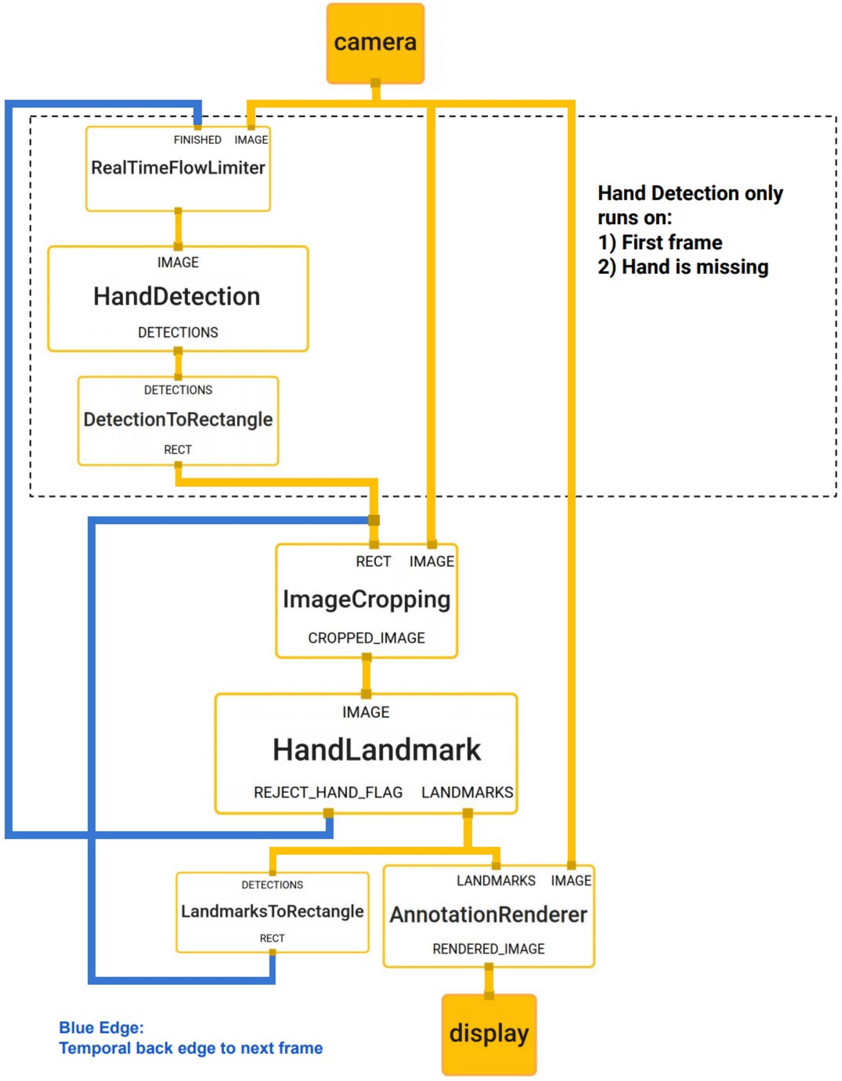
Hand tracking



Abdlkarim D et al. Behavior Research Methods (2024) 56:1052–1063

Hand tracking

- https://developers.google.com/mediapipe/solutions/vision/hand_landmarker



Live coding demonstration

- Dataset is from Kaggle:
  - https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset