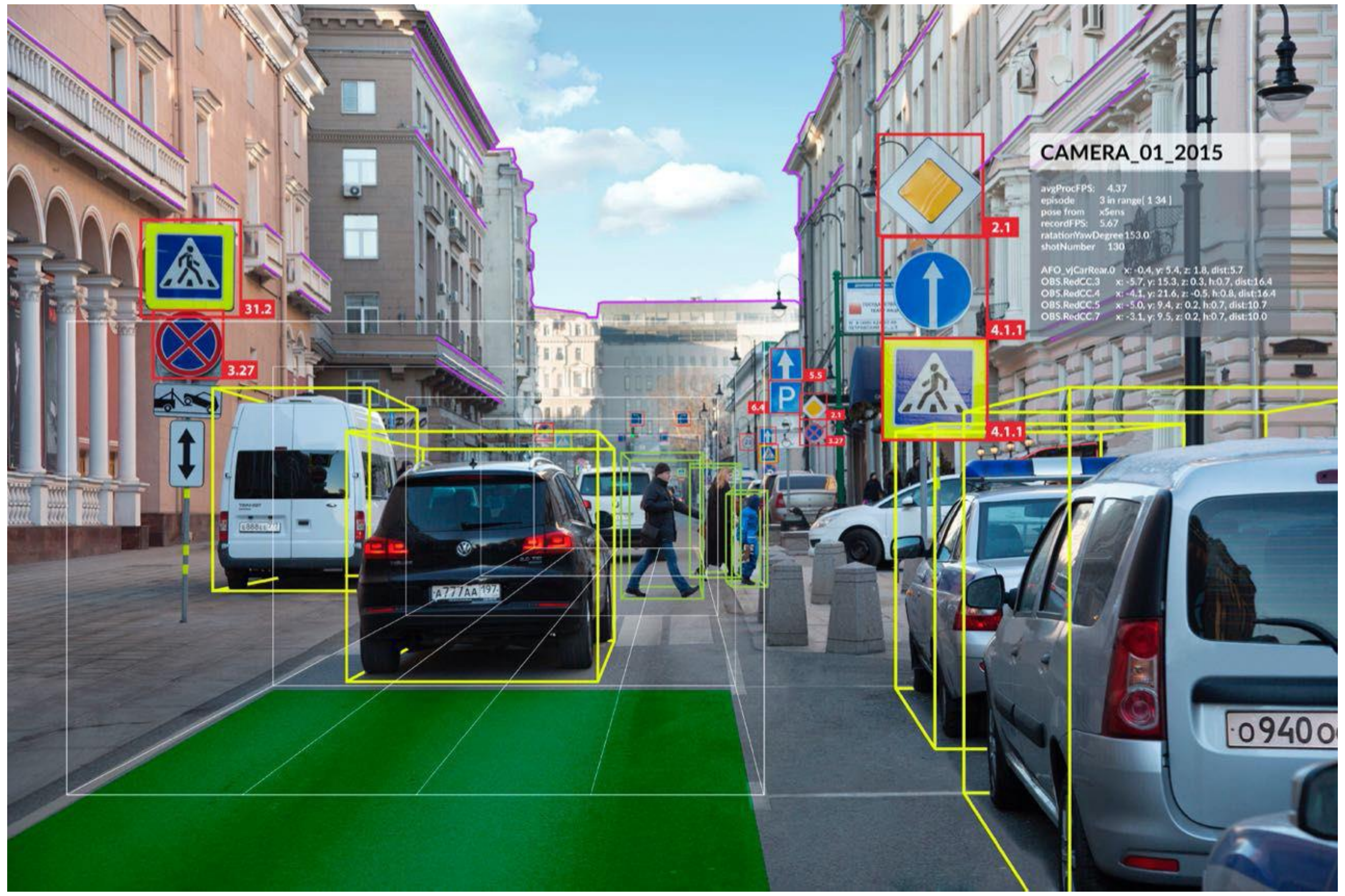
Computer Vision

Michiel Bontenbal
Maarten Post

HvA Minor Applied AI
6 september 2024



Hogeschool van Amsterdam

Creating Tomorrow

Agenda vandaag

Intro to Computer Vision

Classic CV: Convolutional Neural Networks

Modern CV : Image embeddings met CLIP
Vision Language Models

Computer Vision lessen

- **Computer Vision 1: (vandaag)**
  - Introductie
  - Convolutional Neural Networks
  - Image embeddings: similarity and clustering
  - Vision Language Models with ollama

- **Computer Vision 2: (volgende week)**
  - OpenCV -> pre-processing van images & gezichtsdetectie
  - Numpy -> images omzetten naar data

- **Computer Vision 3 + 4: (oktober)**
  - Generative AI voor Images
  - Object detectie en segmentatie
  - Foundation models

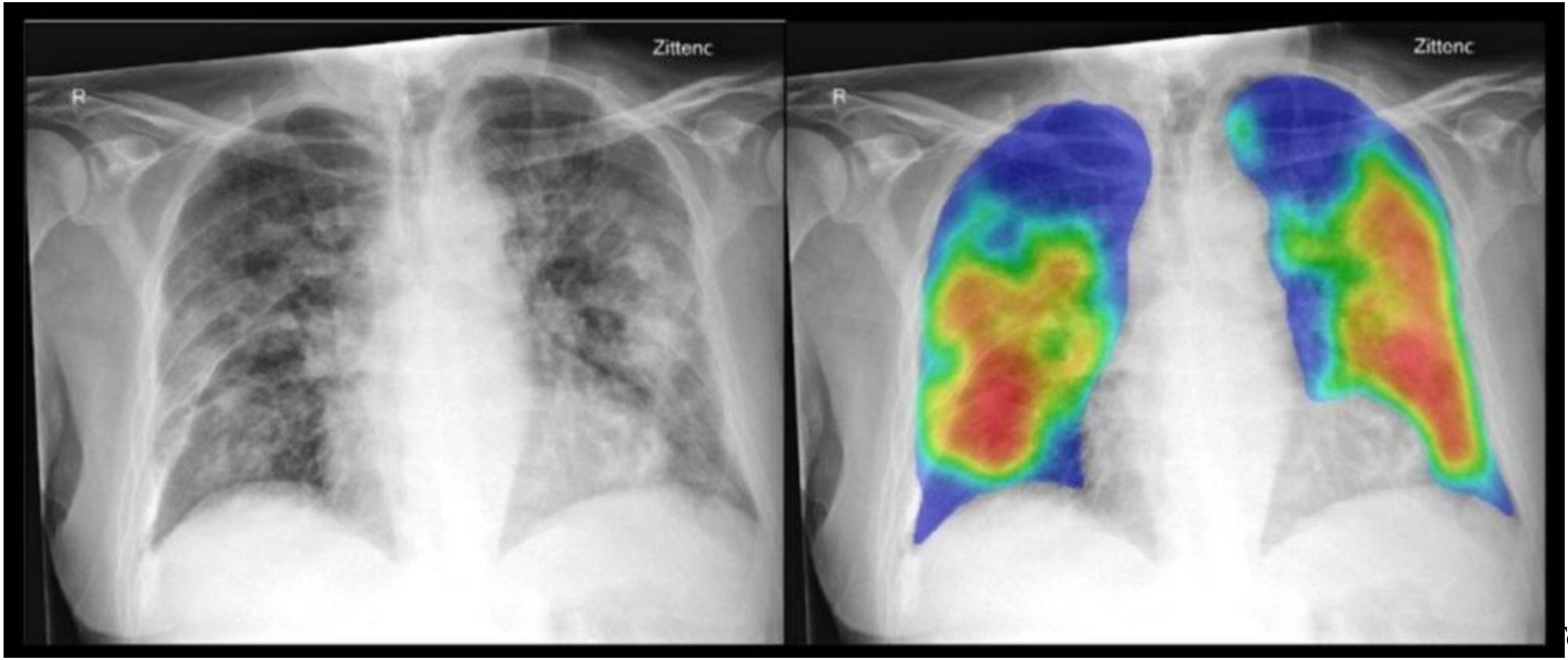
- **Computer Vision 5:**
- Data Centric AI

Some questions… raise your hand!

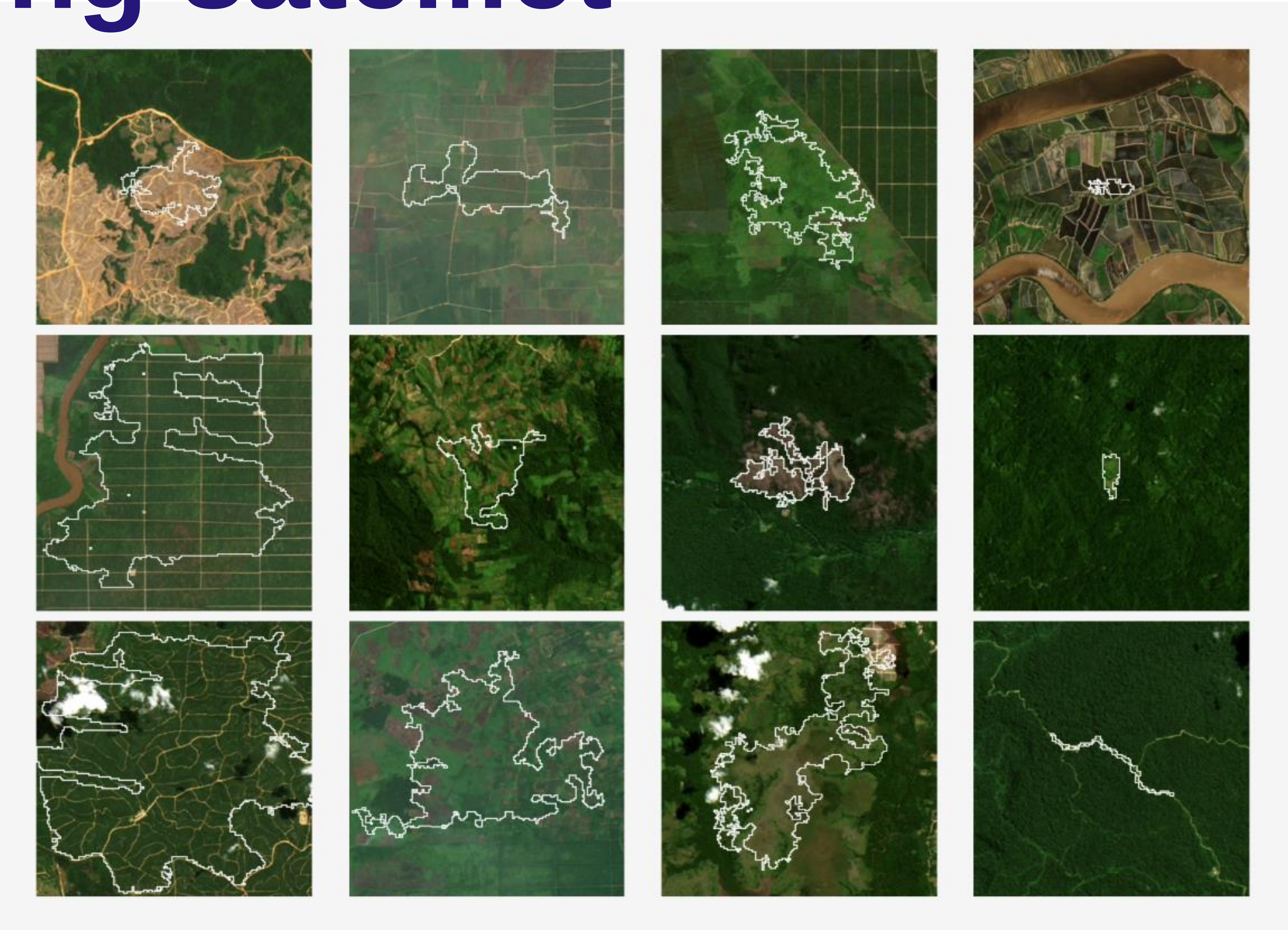
- Who has worked with:
  - Computer Vision
  - CNN
  - Vision Language Models
  - Ollama?

Intro to Computer Vision

Medisch

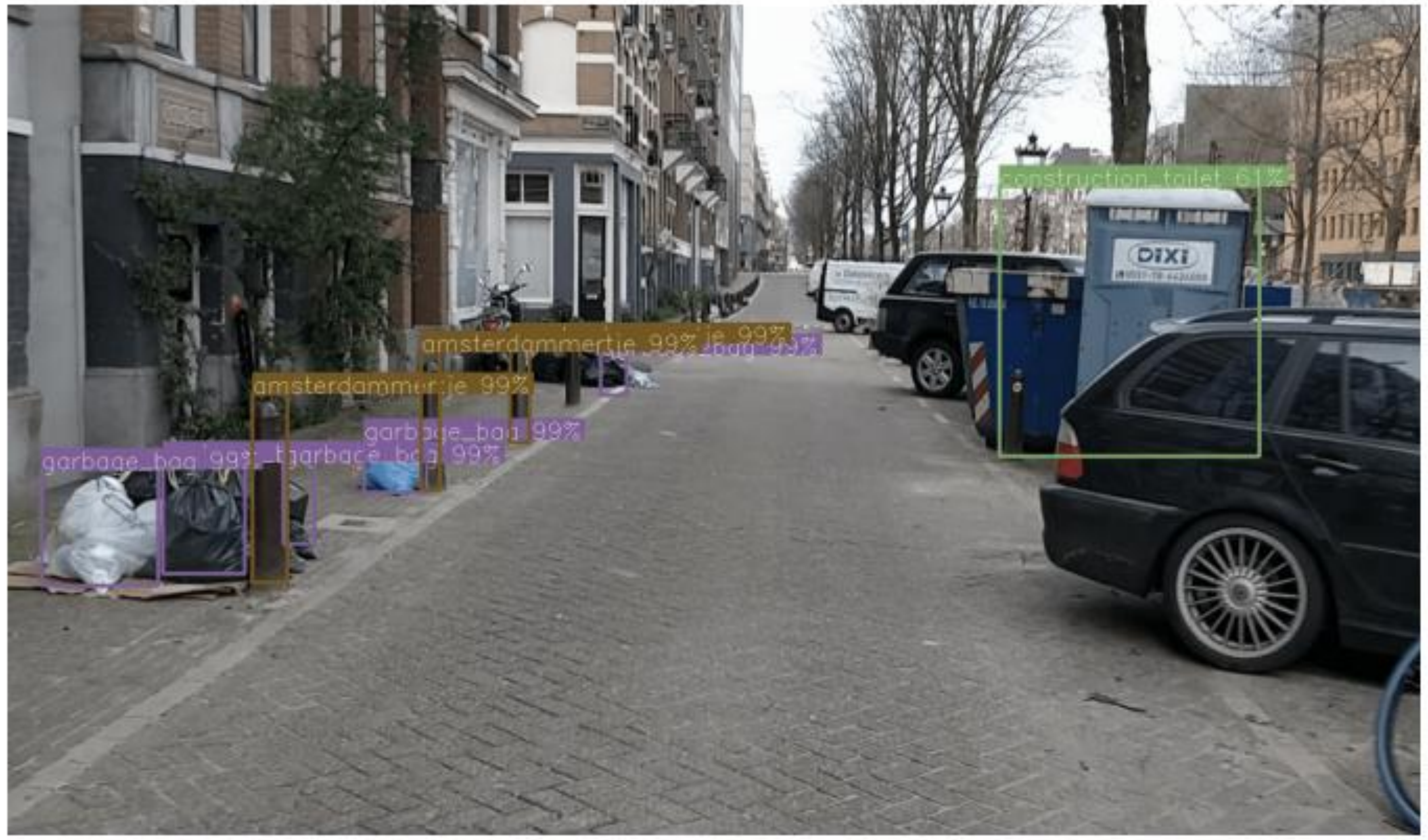


Ontbossing satelliet



https://stanfordmlgroup.github.io/projects/forestnet/

Amsterdam gebruikt actief AI

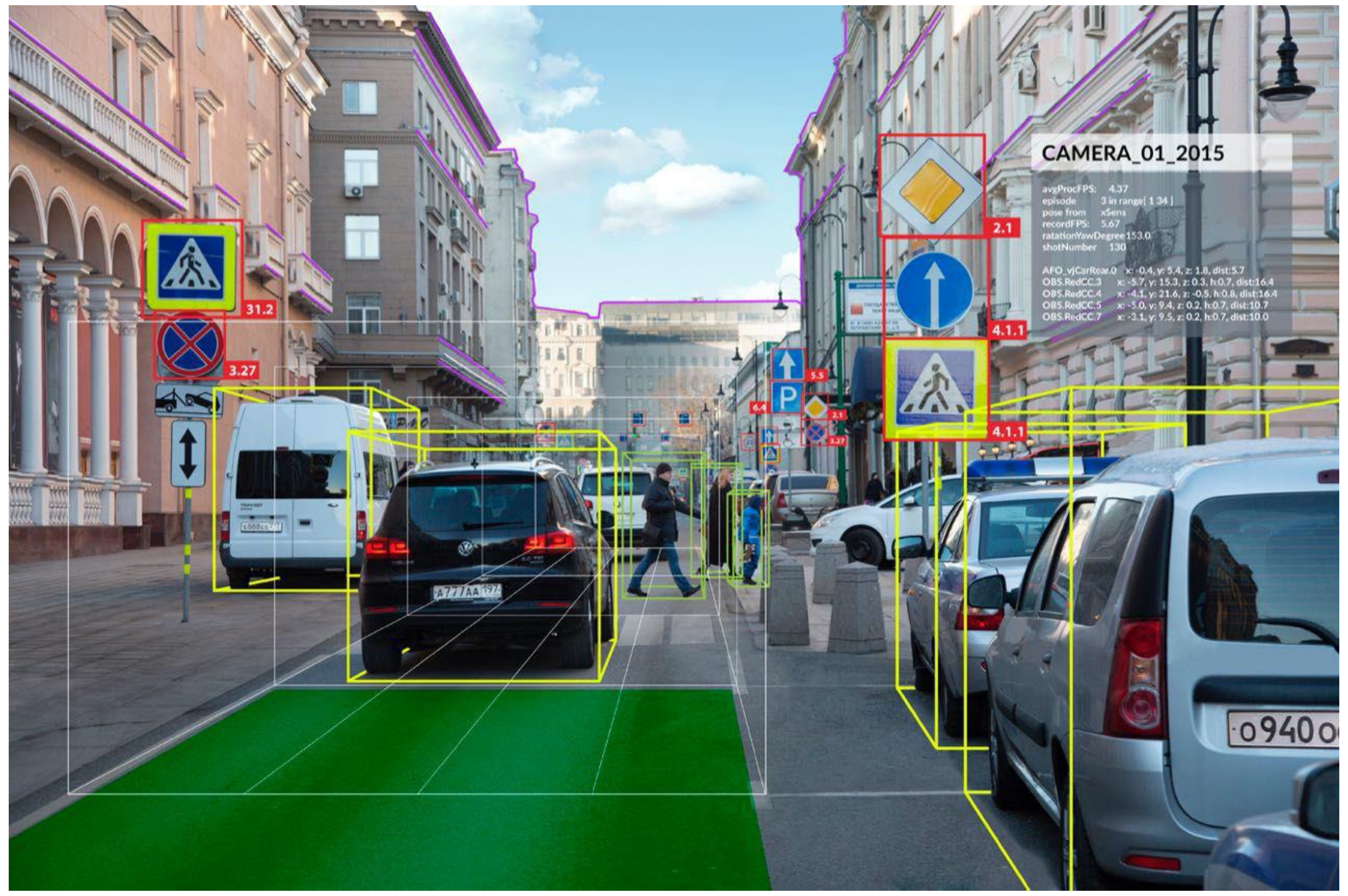


https://openresearch.amsterdam/nl/page/63153/urban-object-detection-kit-a-system-for-collection-and-analysis-of

Generative AI for vision

What is computer vision?

- Computers + camera's that can "see"
  - Retrieve information from picture or video
  - Take an action based on this information

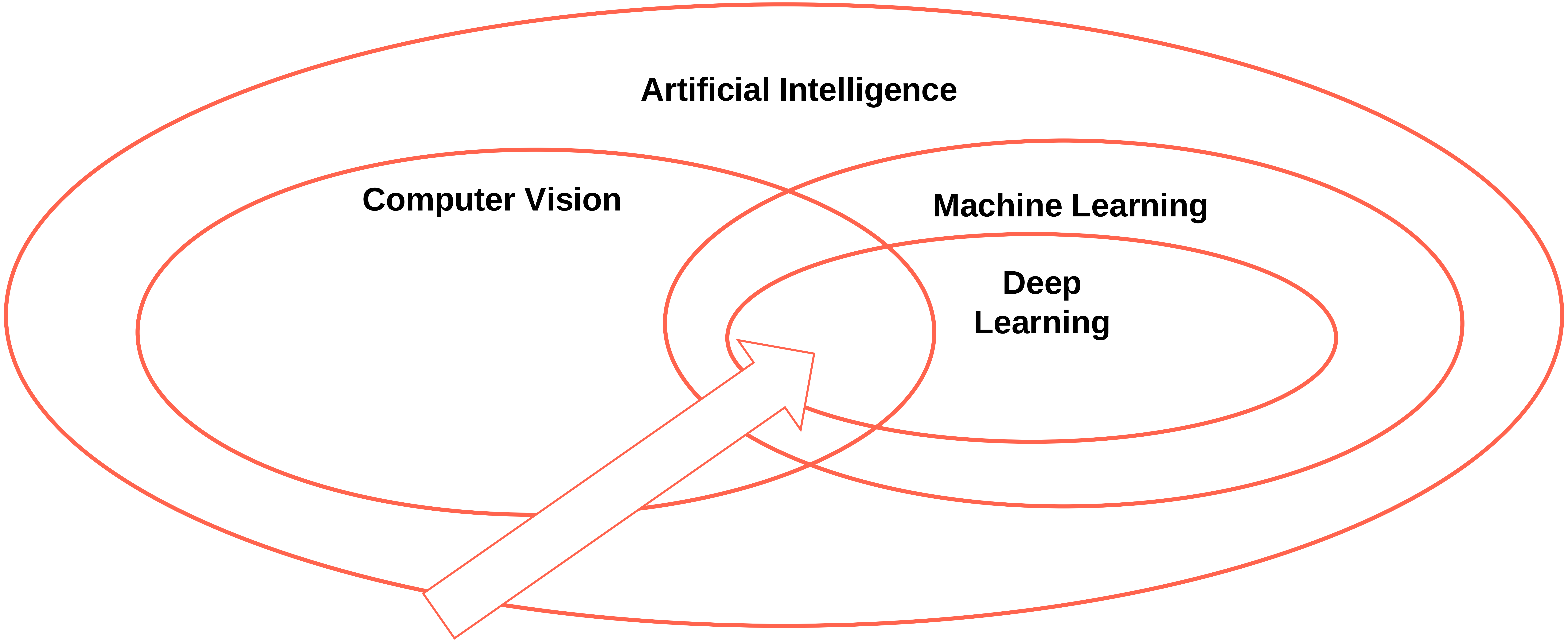


Je gebruikt computer vision al elke dag

- Wat kun je met de **computer vision** in je telefoon?

- **Selfie door hand opsteken**
- **Gezichtsdetectie**
- **Filters voor achtergrond blurren**
- **QR codes**
- **Stickers maken met je foto's**
- **…**

Computer Vision in AI



This lesson is about Computer Vision using Deep Learning.

What is the ‘deep’ in deep learning?

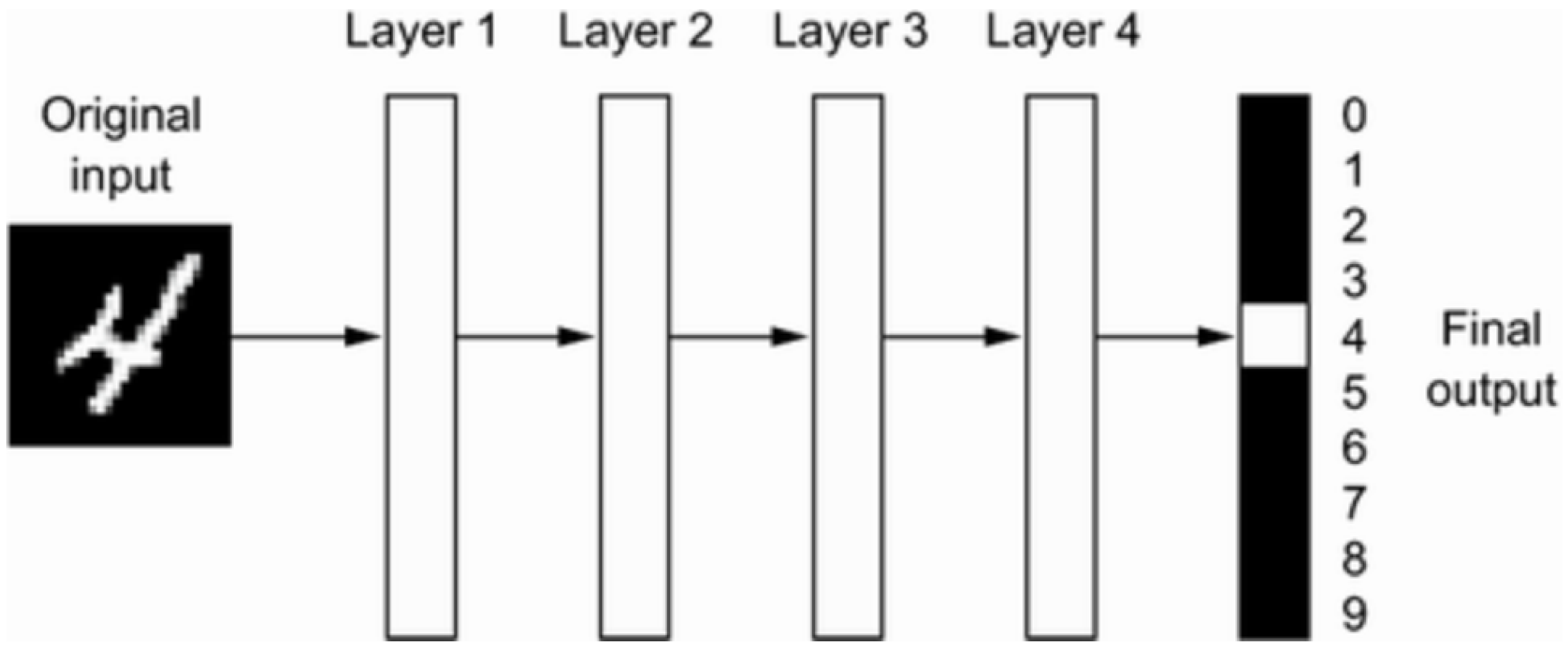
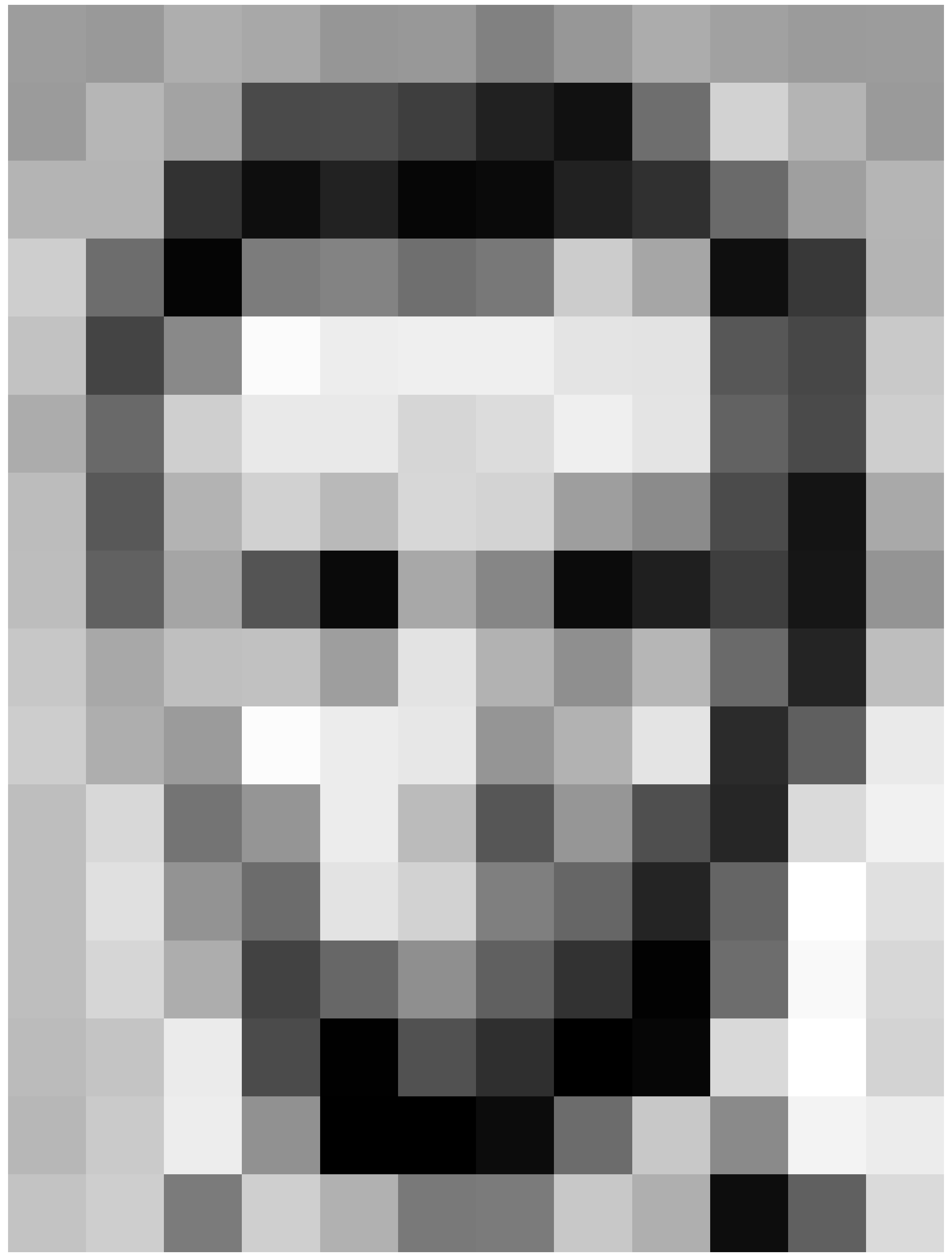
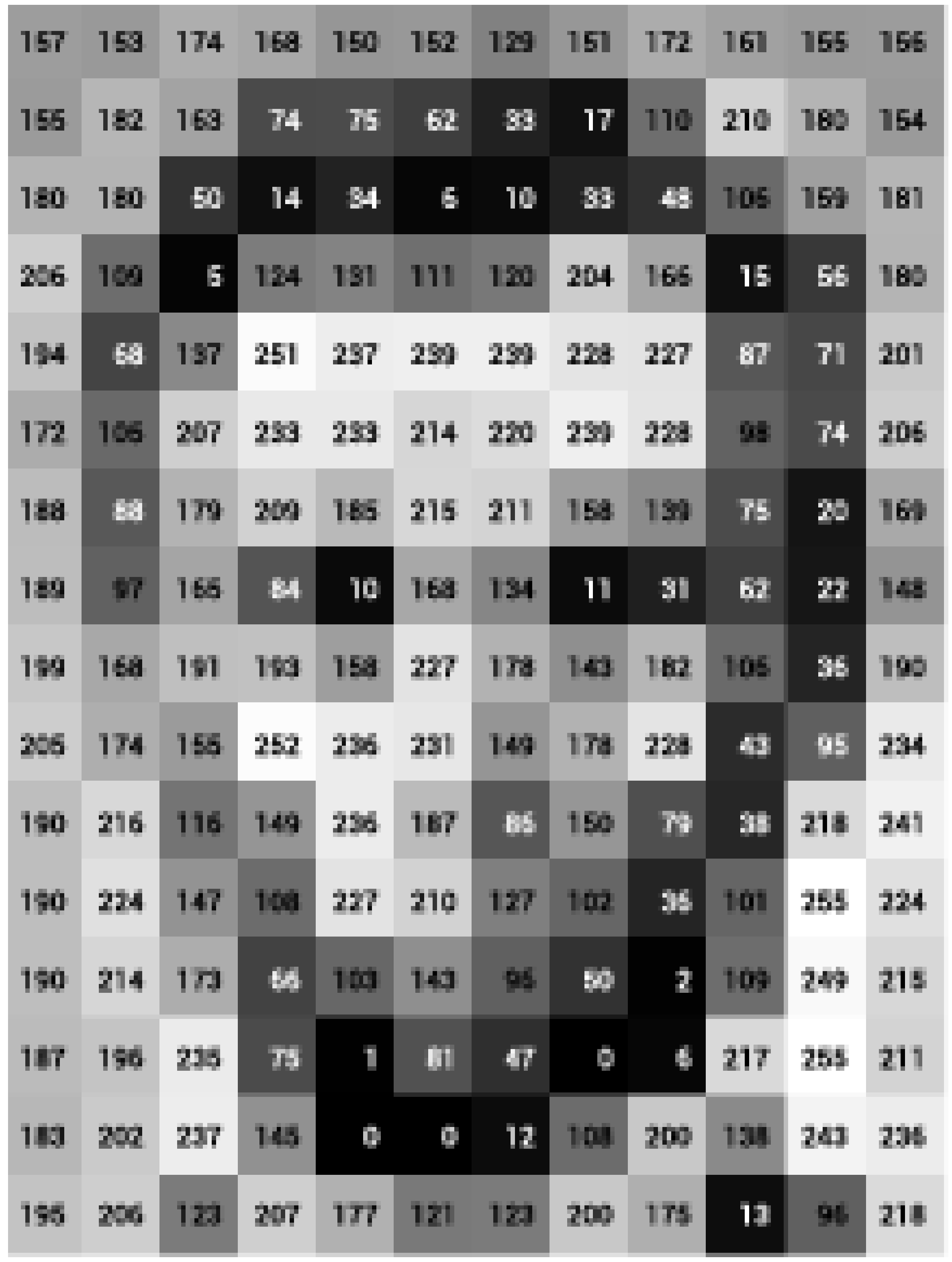


Figure 1.5 A deep neural network for digit classification

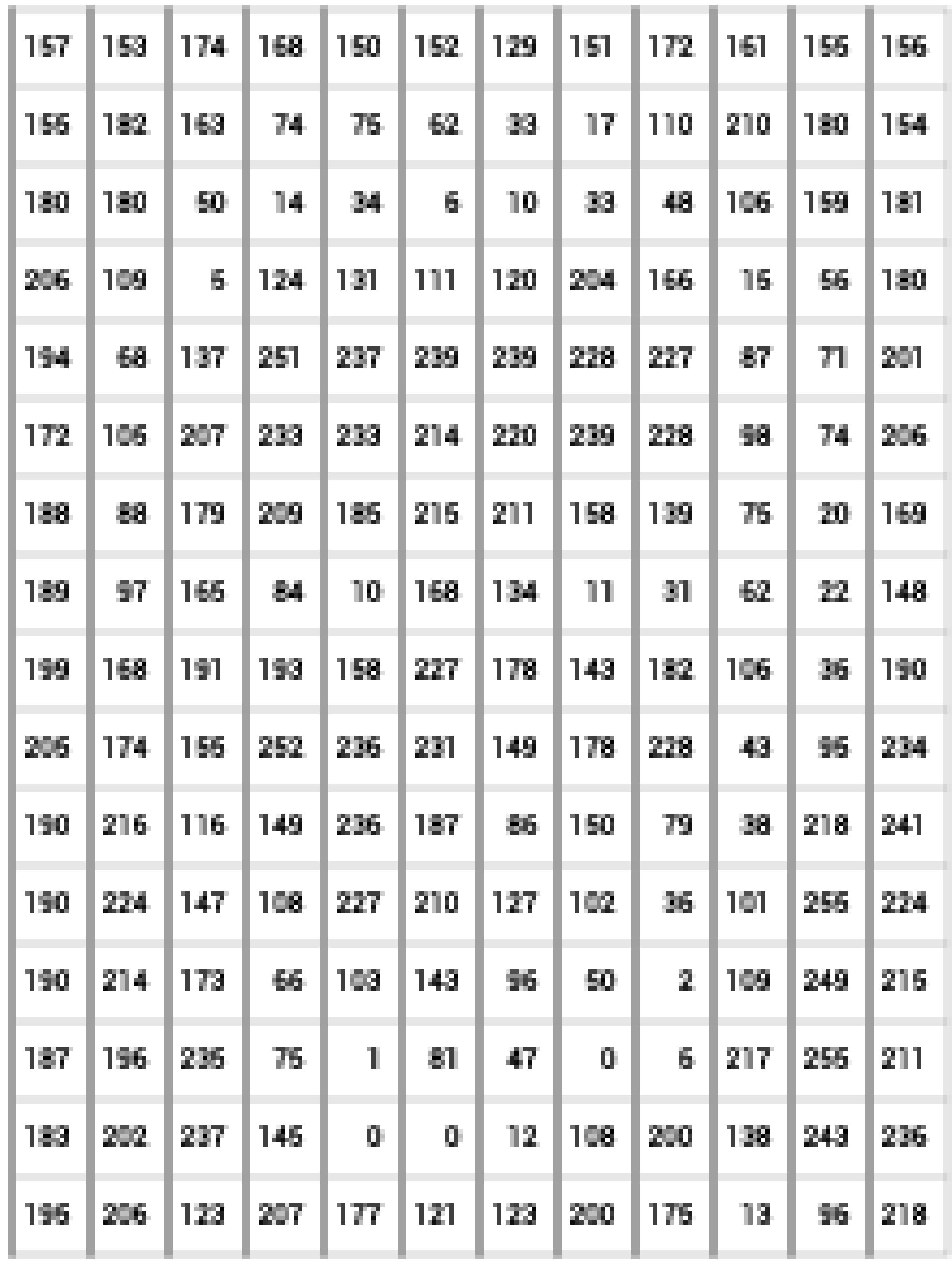
Source: Deep Learning with Python, Francois Chollet, 1.1.4 The ‘deep’ in deep learning

We see images, computers ‘see’ numbers



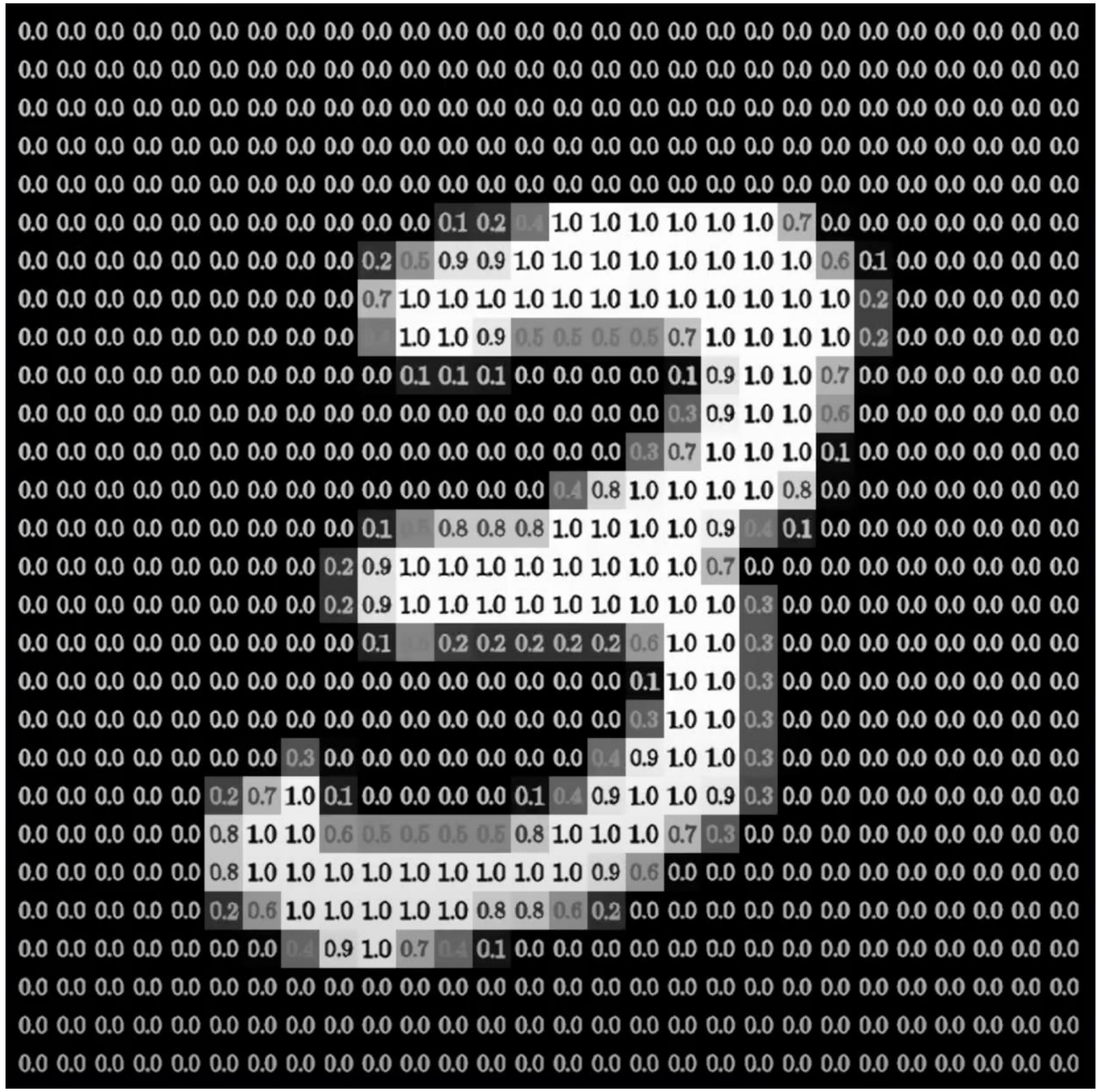


157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218



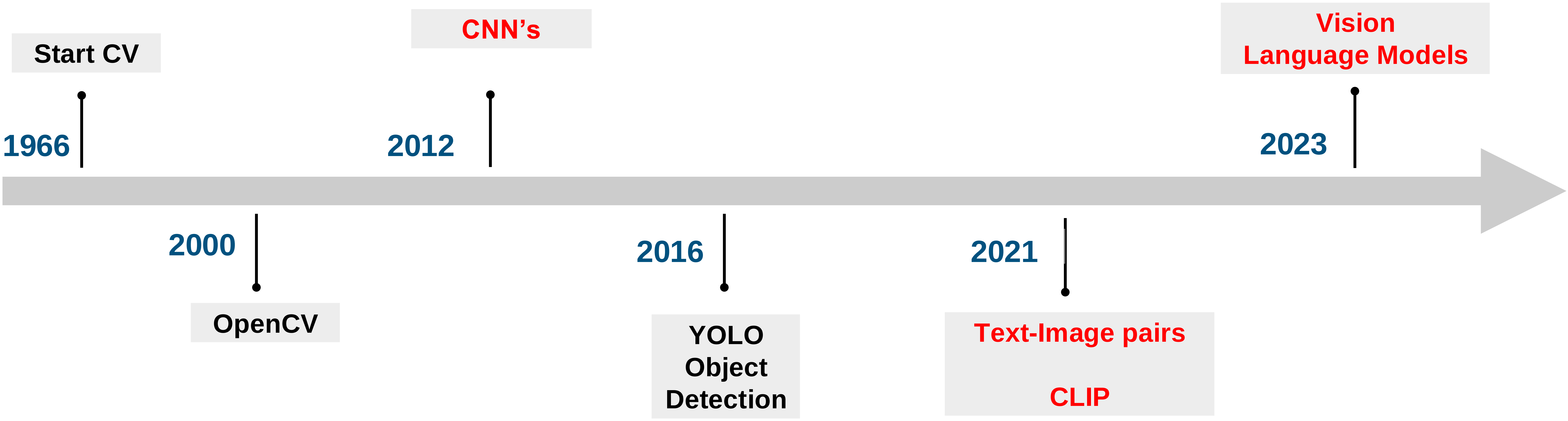
157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

Remember, AI systems do calculations



0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.2	0.4	1.0	1.0	1.0	1.0	1.0	1.0	0.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.5	0.9	0.9	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.6	0.1	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.7	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.2	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.4	1.0	1.0	0.9	0.5	0.5	0.5	0.5	0.7	1.0	1.0	1.0	1.0	0.2	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.1	0.1	0.0	0.0	0.0	0.0	0.1	0.9	1.0	1.0	0.7	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.9	1.0	1.0	0.6	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.7	1.0	1.0	1.0	0.1	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.4	0.8	1.0	1.0	1.0	1.0	0.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.5	0.8	0.8	0.8	1.0	1.0	1.0	1.0	0.9	0.4	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.9	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.9	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	[illegible]	0.2	0.2	0.2	0.2	0.2	0.6	1.0	1.0	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	1.0	1.0	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	1.0	1.0	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.4	0.9	1.0	1.0	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.2	0.7	1.0	0.1	0.0	0.0	0.0	0.0	0.1	0.4	0.9	1.0	1.0	0.9	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.8	1.0	1.0	0.6	0.5	0.5	0.5	0.5	0.8	1.0	1.0	1.0	0.7	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.8	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.9	0.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.2	0.6	1.0	1.0	1.0	1.0	1.0	0.8	0.8	0.6	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.4	0.9	1.0	0.7	0.4	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Tijdslijn Computer Vision (CV)



1966 Start CV

2000 OpenCV

2012 CNN’s

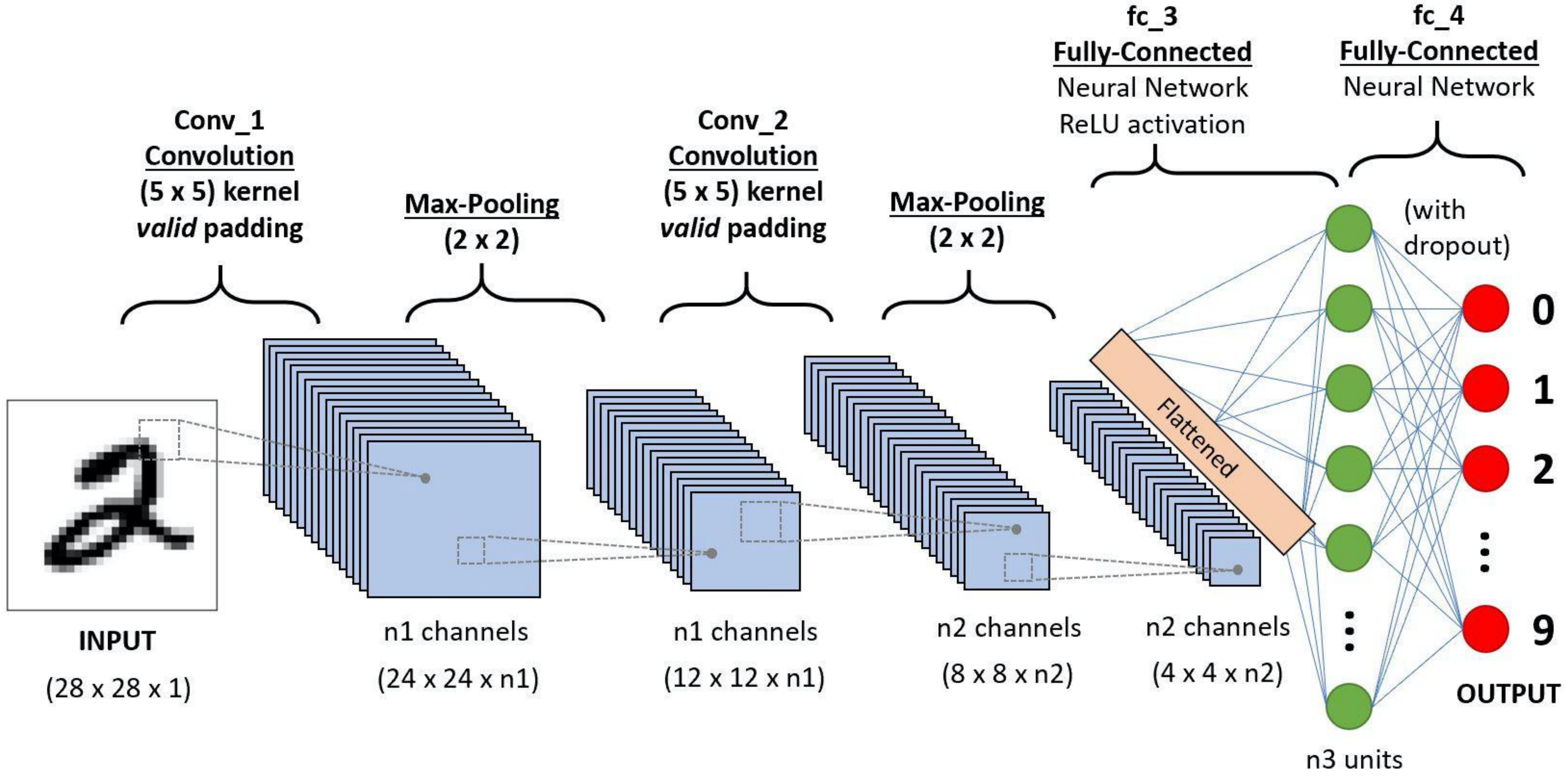
2016 YOLO Object Detection

2021 Text-Image pairs

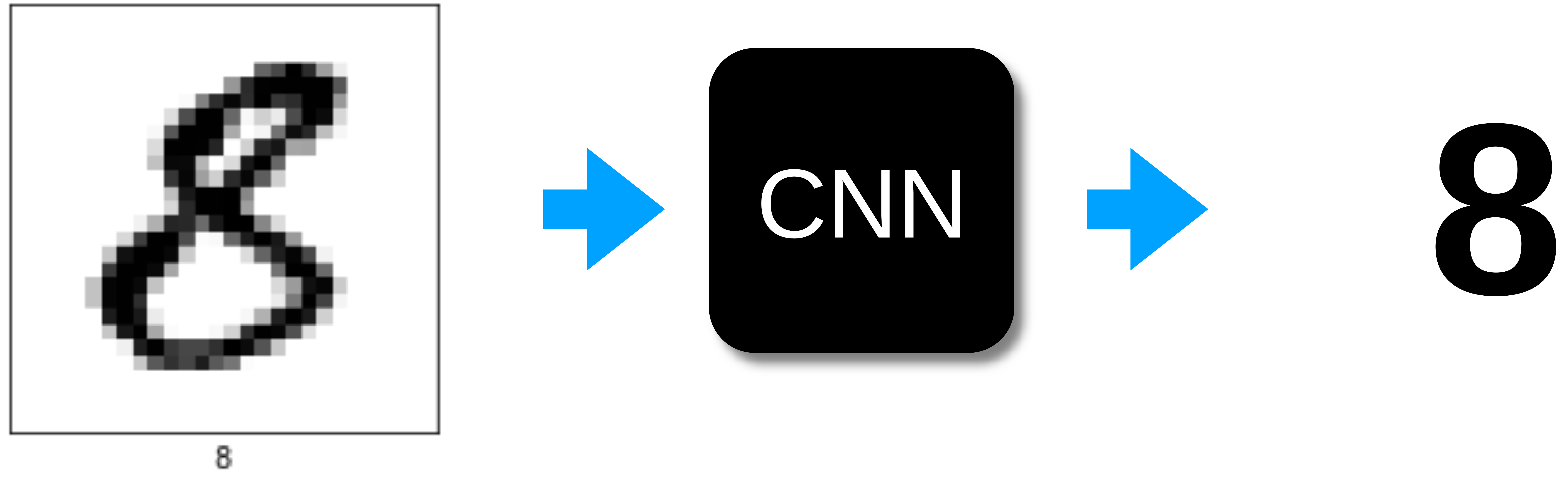
CLIP

2023 Vision Language Models

Convolutional Neural Netwerks



Handwritten digits



8

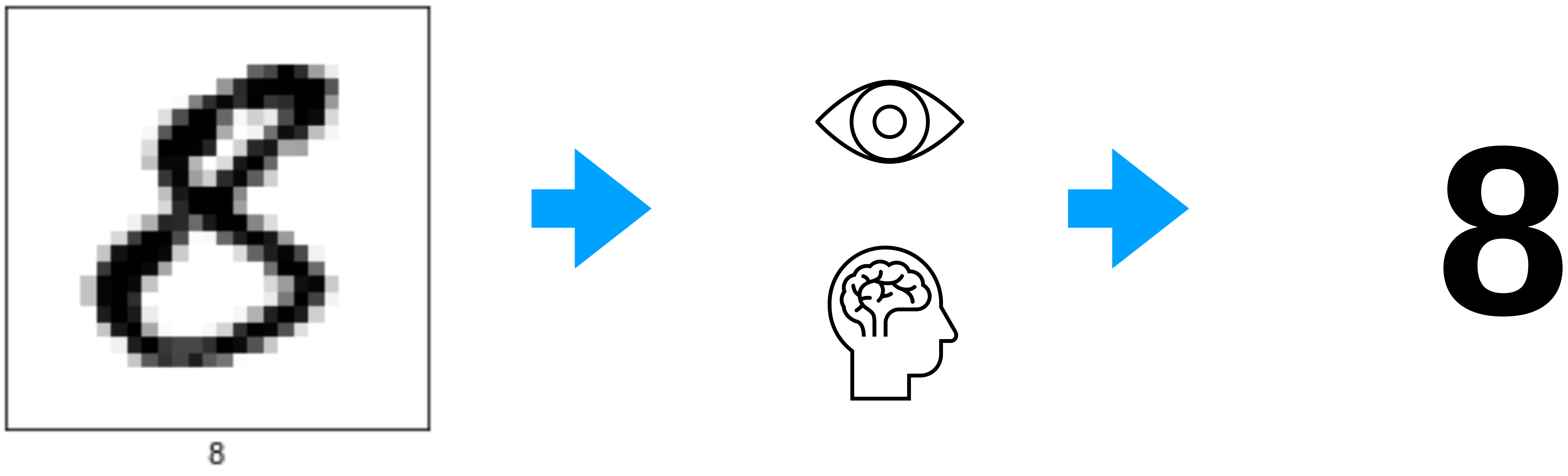
CNN

8

Convolutional Neural Network (CNN)

1. What is a convolution?
2. What’s the difference with ‘normal’ Neural Nets?
3. What are the layers in a CNN?
4. How can a CNN learn?

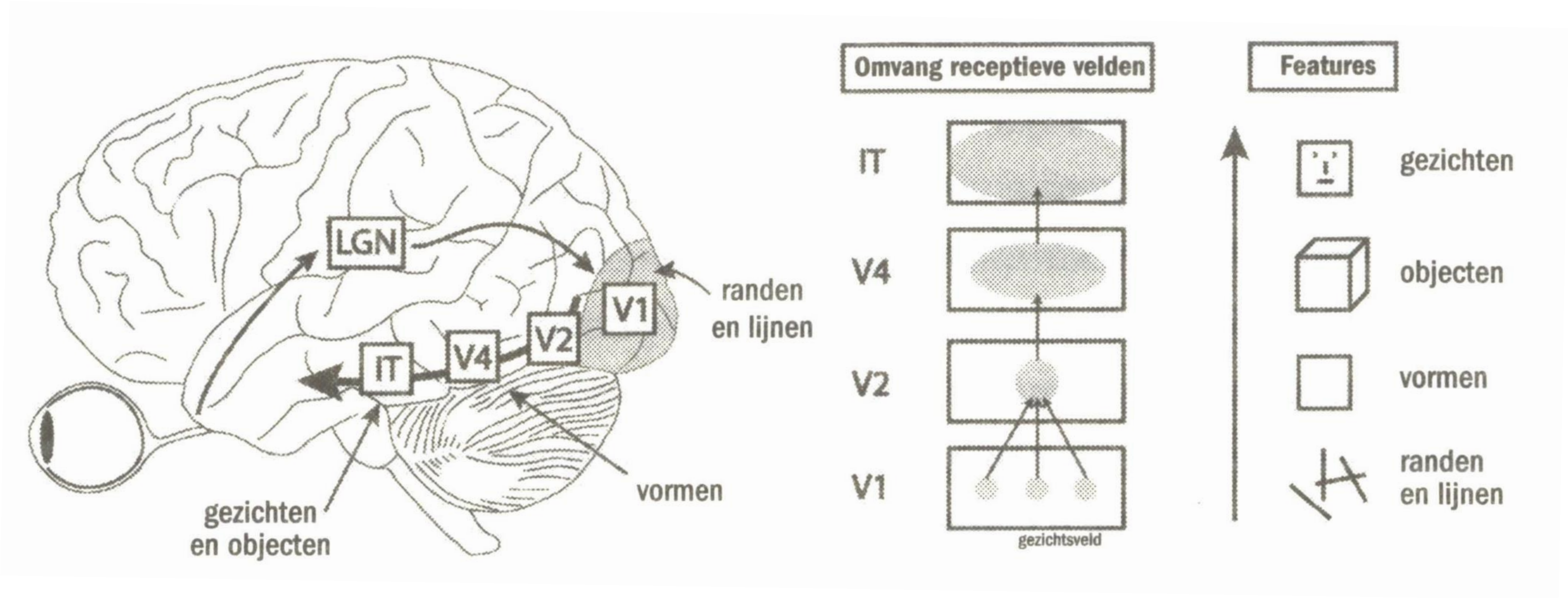
How do we as humans recognize this?



8

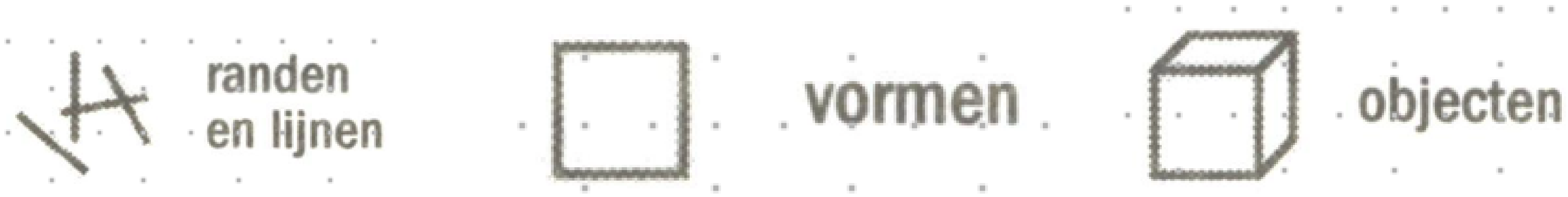
Hoe wij dingen zien en herkennen

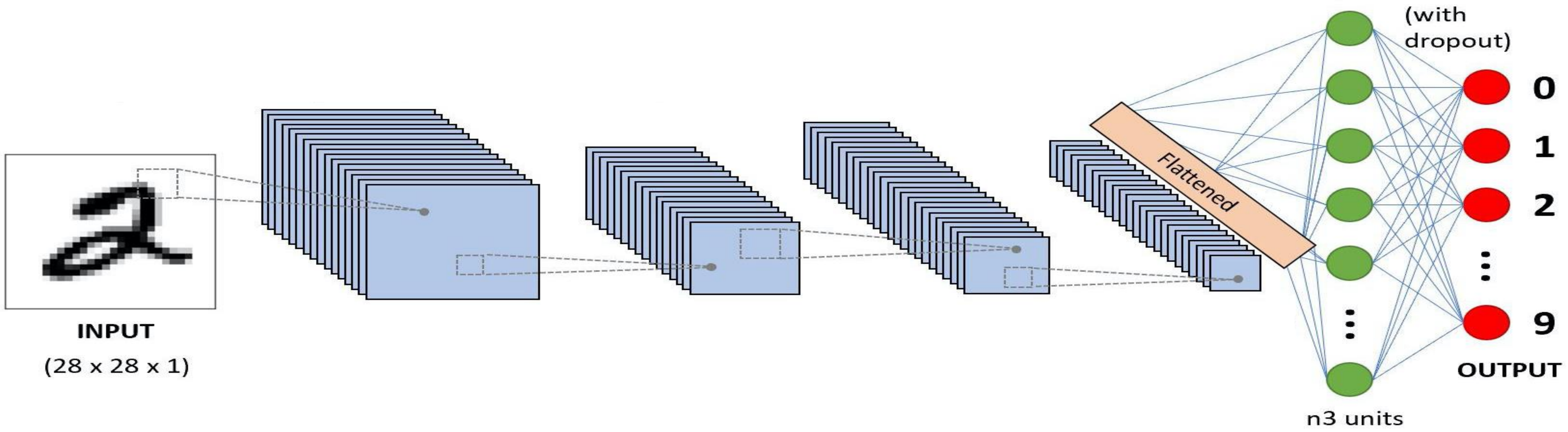
(Hubel & Wiesel, Nobelprijs ‘81)



Source: A Brief history of Intelligence - Max Bennett

Features herkennen + Neural Net = CNN

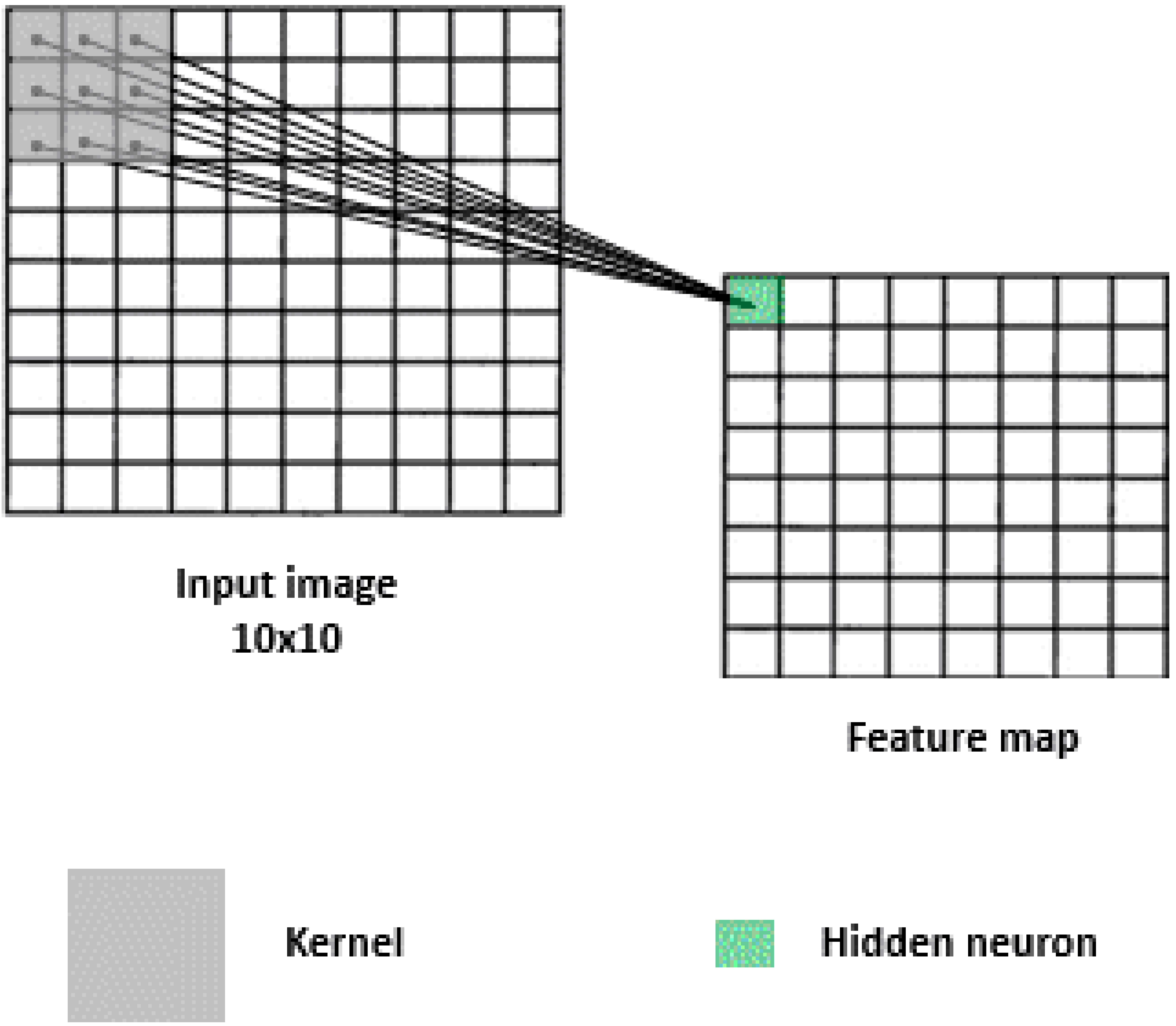




Features herkennen

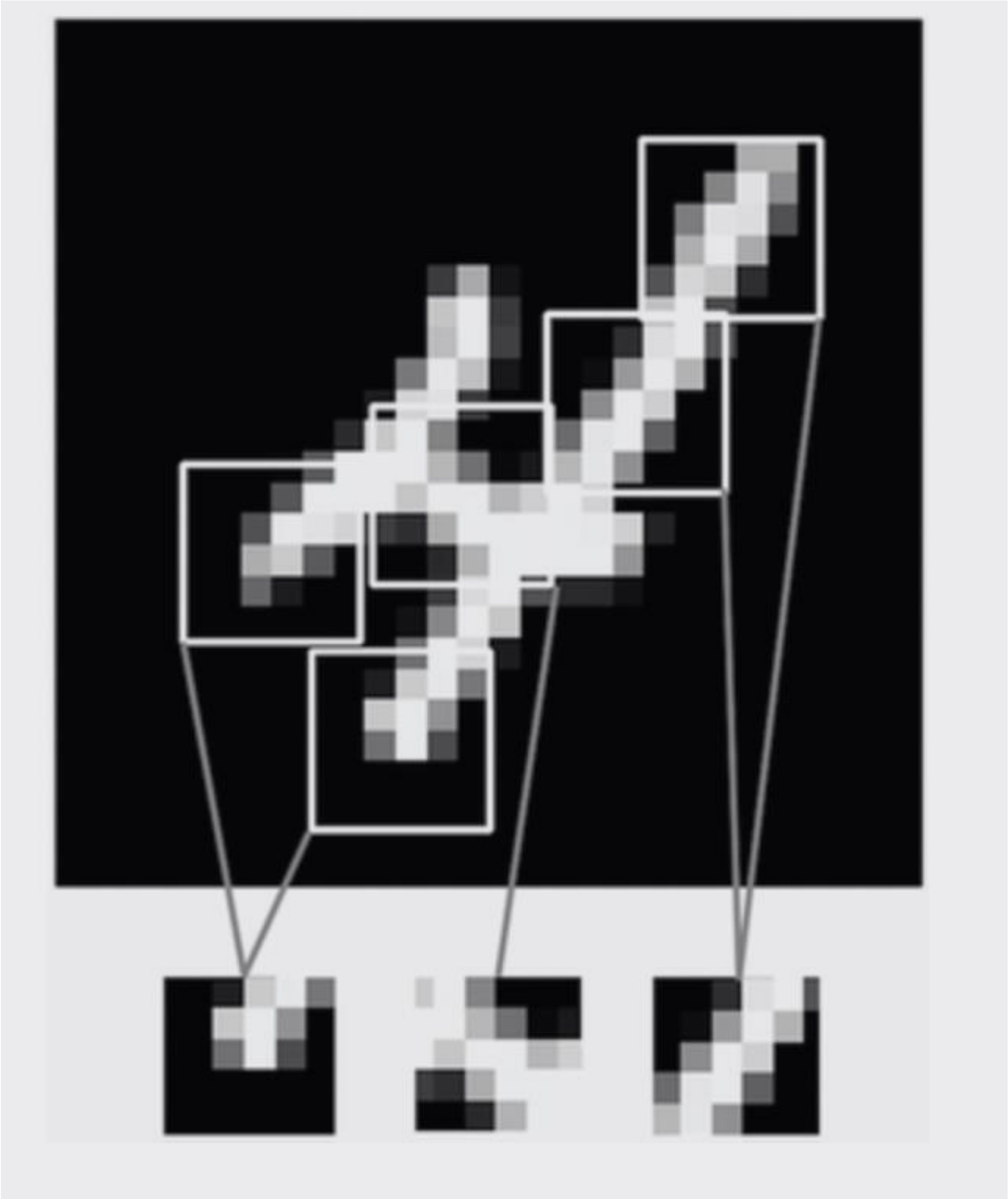
Neural net
voor classificatie

Convolution2D Layer



(Ctrl)

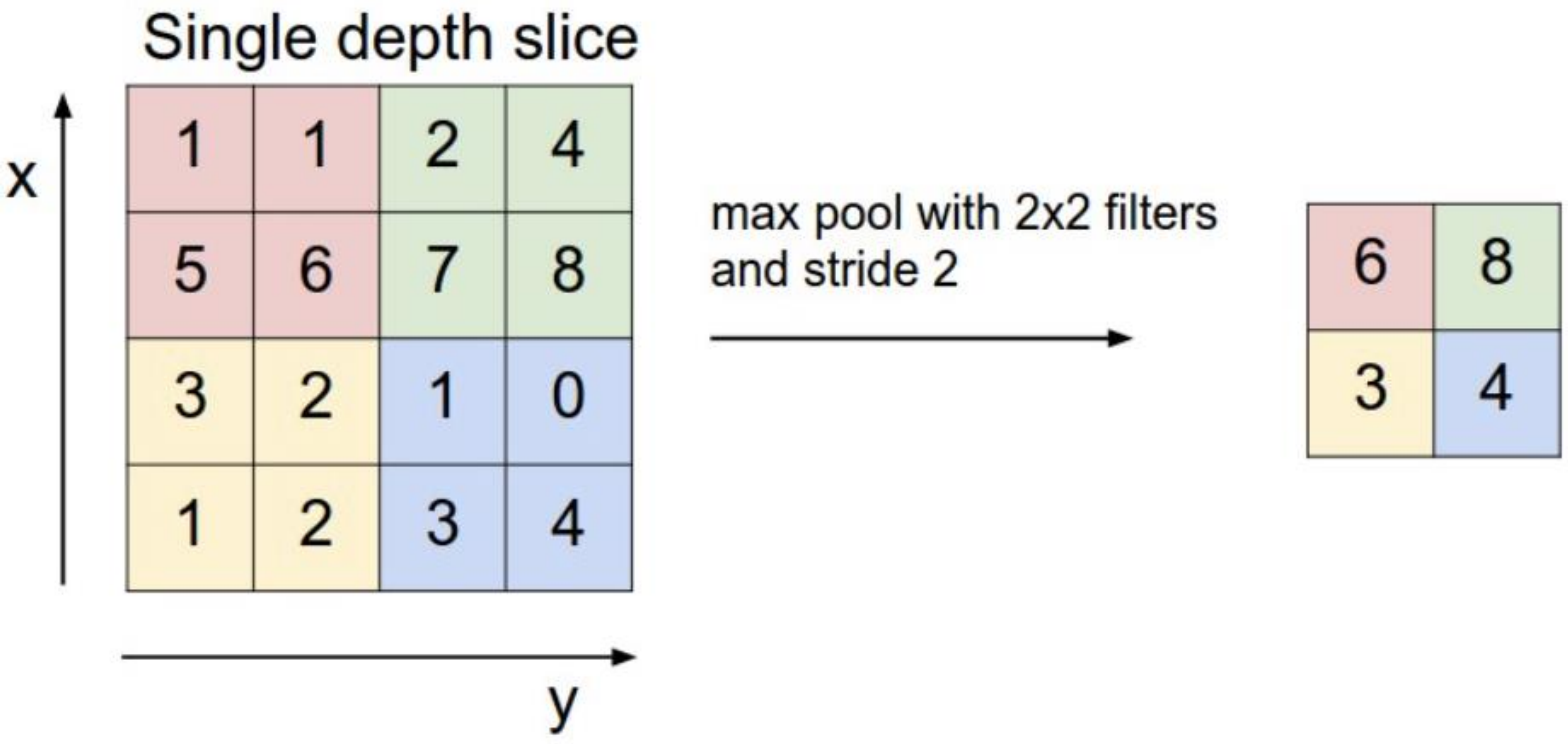
1e convolution layer herkent lijnen en randen



Source: Deep Learning with Python, Francois Chollet, 8.1 Intro to Convnets

Pooling layer

max pooling, average pooling

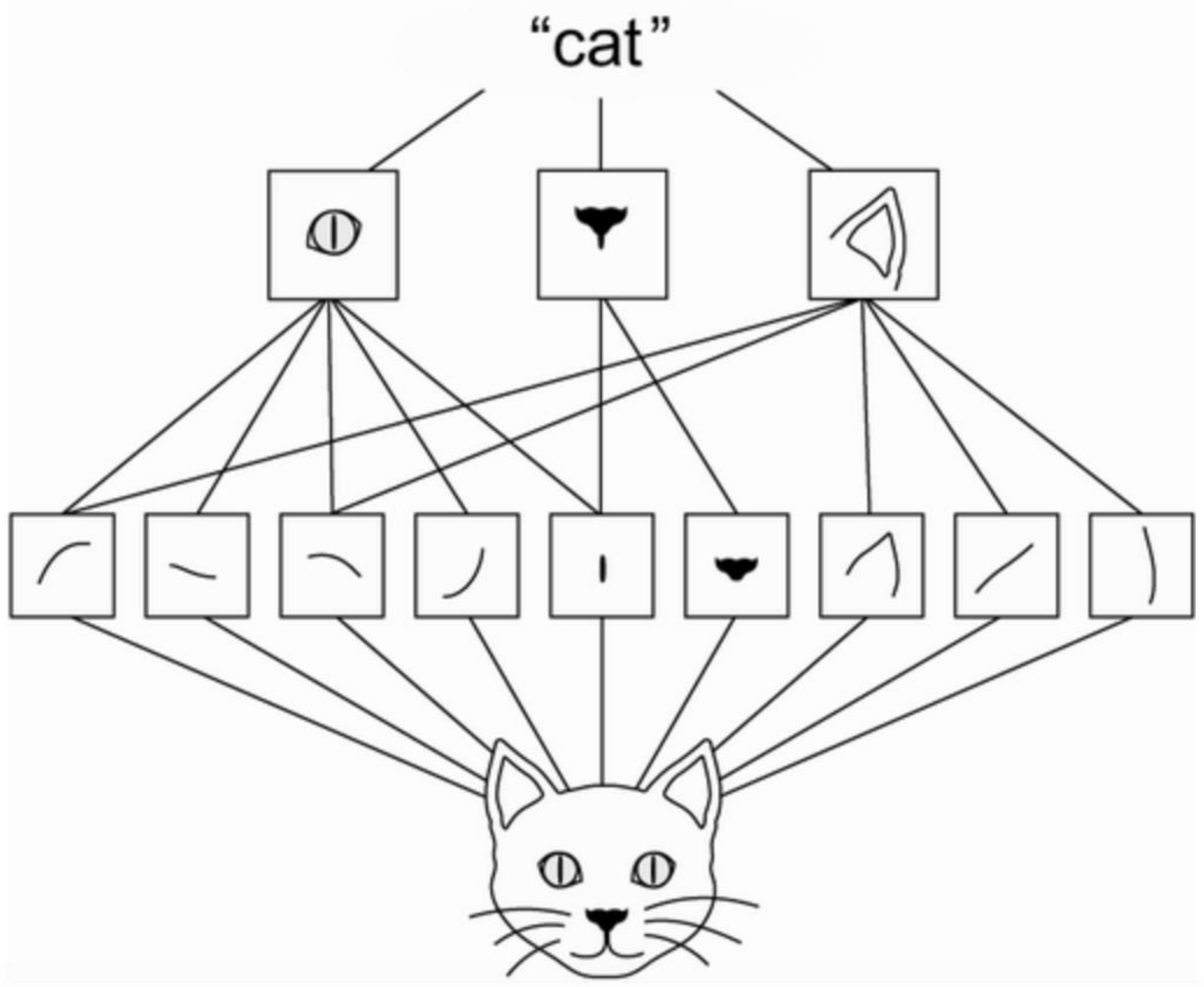


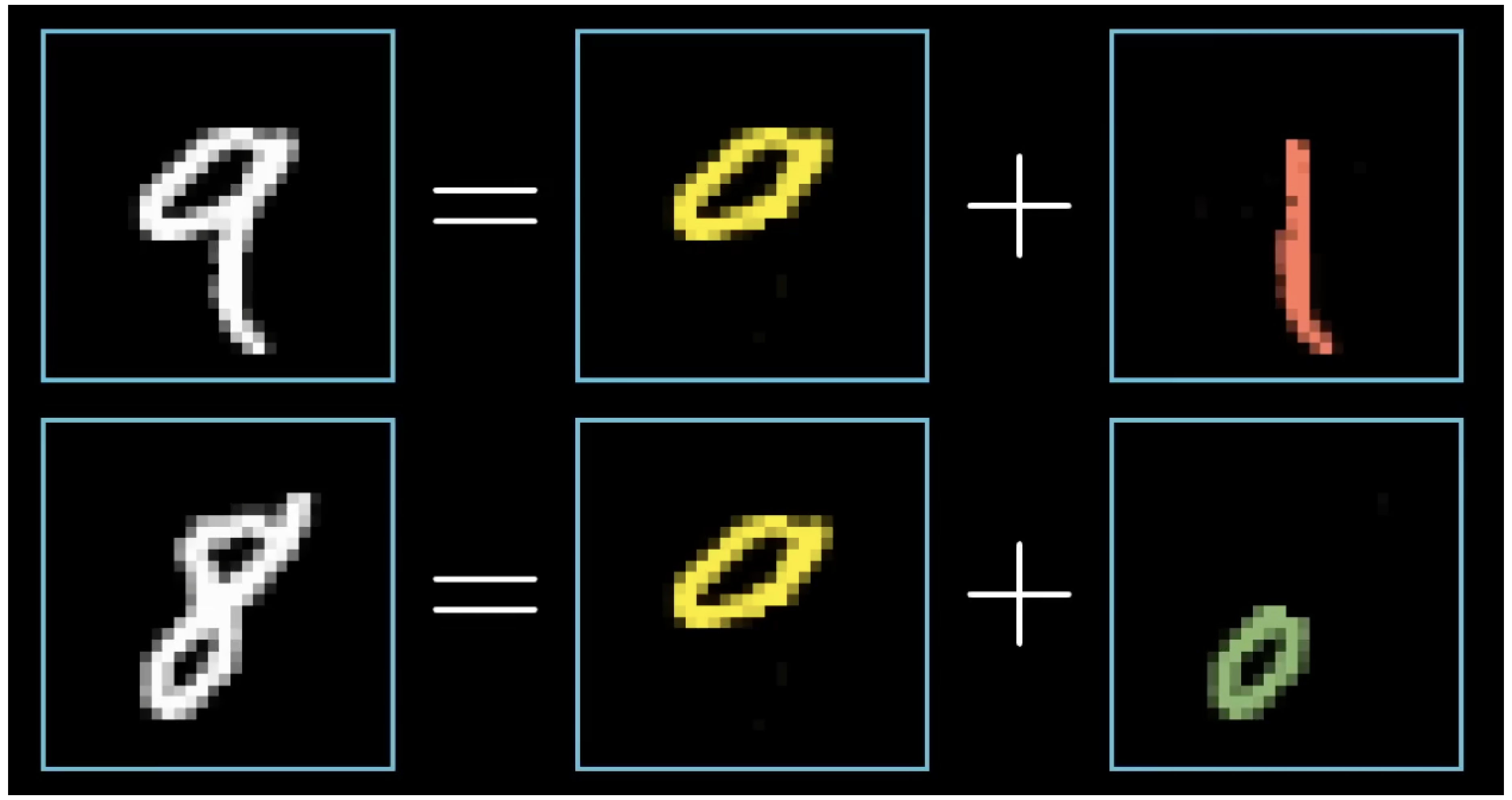
1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

6	8
3	4

Source: https://cs231n.github.io/

By repeating conv2d + pooling, the model can recognize larger features

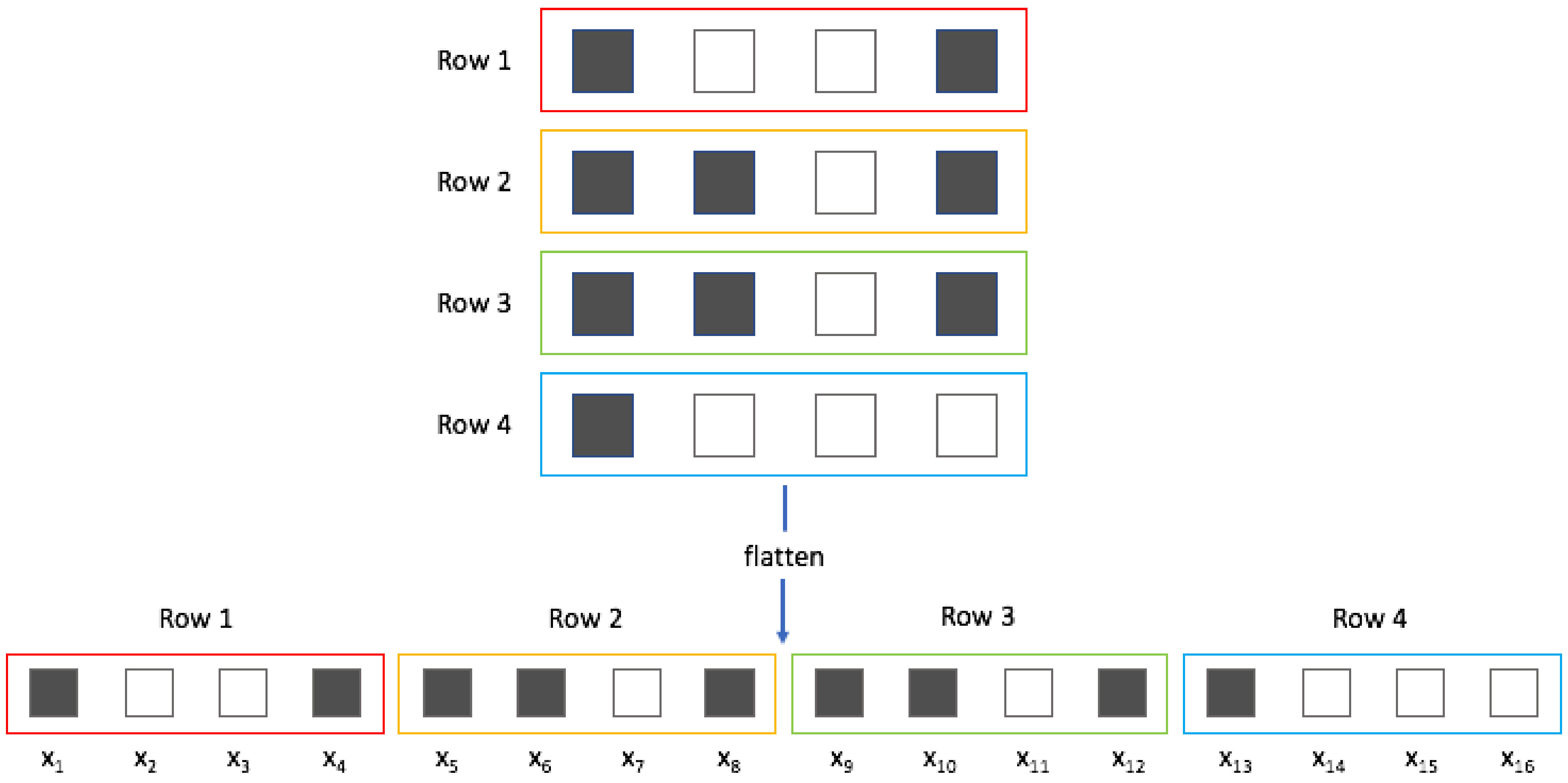




Source: Deep Learning with Python, Francois Chollet, 8.1 Intro to Convnets

Creating Tomorrow

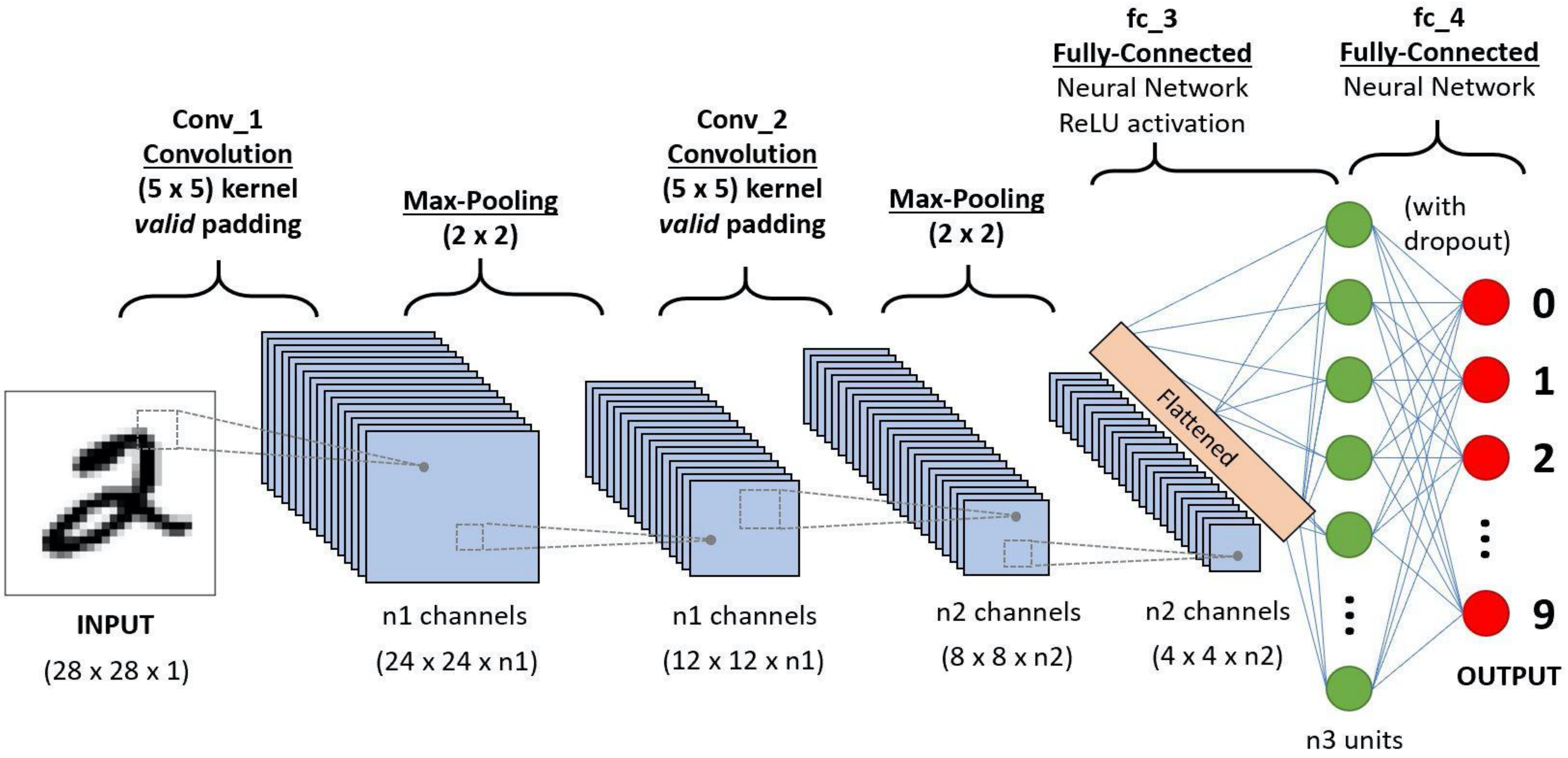
Flatten(): 2D image => 1D input layer



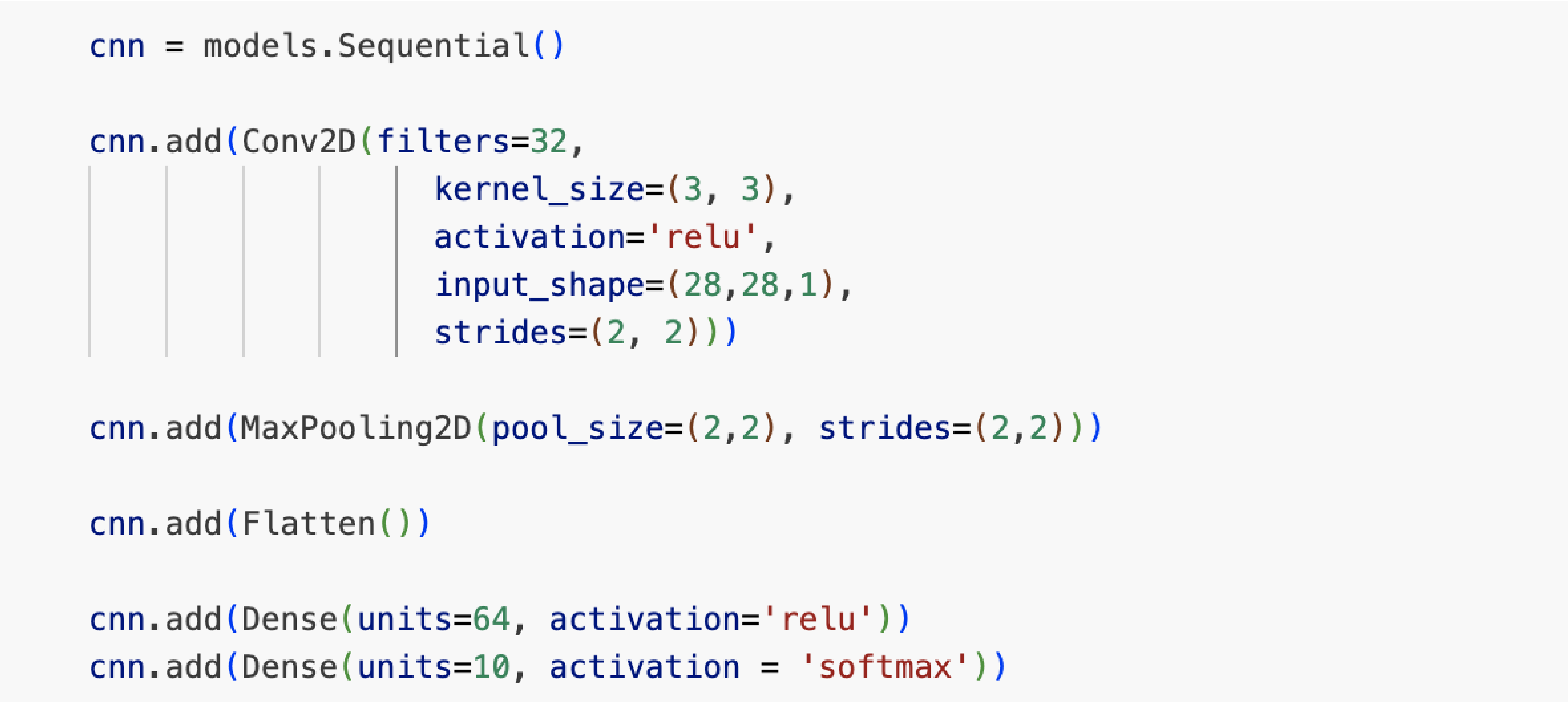
“flatten()”

Convert 1 pixel => 1 node of input layer.
For MNIST 28 x 28 = 784 node

Convolutional Neural Network



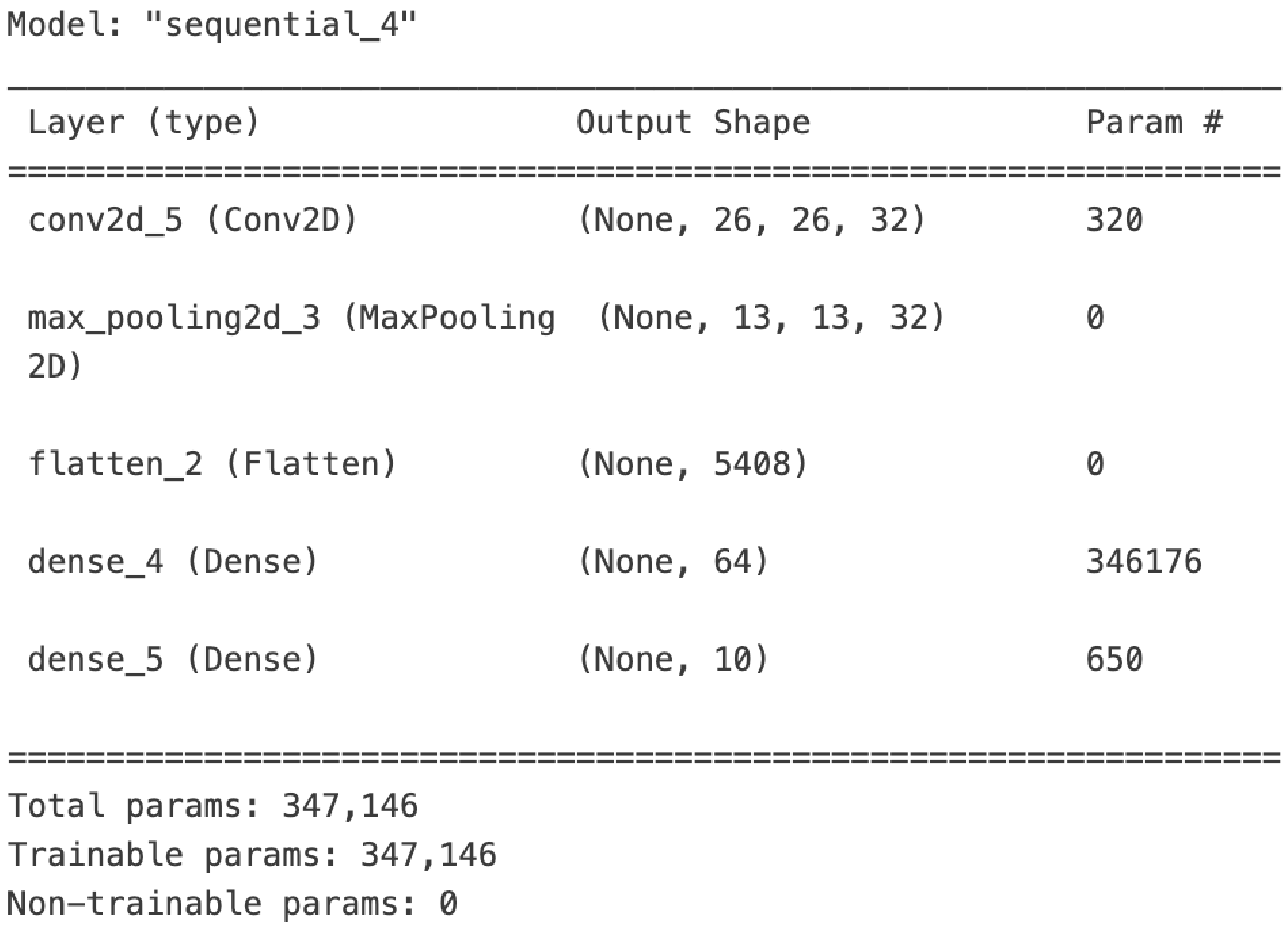
CNN in Keras



```
cnn = models.Sequential()

cnn.add(Conv2D(filters=32,
                  kernel_size=(3, 3),
                  activation='relu',
                  input_shape=(28,28,1),
                  strides=(2, 2)))

cnn.add(MaxPooling2D(pool_size=(2,2), strides=(2,2)))

cnn.add(Flatten())

cnn.add(Dense(units=64, activation='relu'))
cnn.add(Dense(units=10, activation = 'softmax'))
```

CNN in Keras

model.summary()



```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_5 (Conv2D)           (None, 26, 26, 32)        320

 max_pooling2d_3 (MaxPooling  (None, 13, 13, 32)       0
 2D)

 flatten_2 (Flatten)         (None, 5408)              0

 dense_4 (Dense)             (None, 64)                346176

 dense_5 (Dense)             (None, 10)                650

=================================================================
Total params: 347,146
Trainable params: 347,146
Non-trainable params: 0
```

Calculate the number of parameters

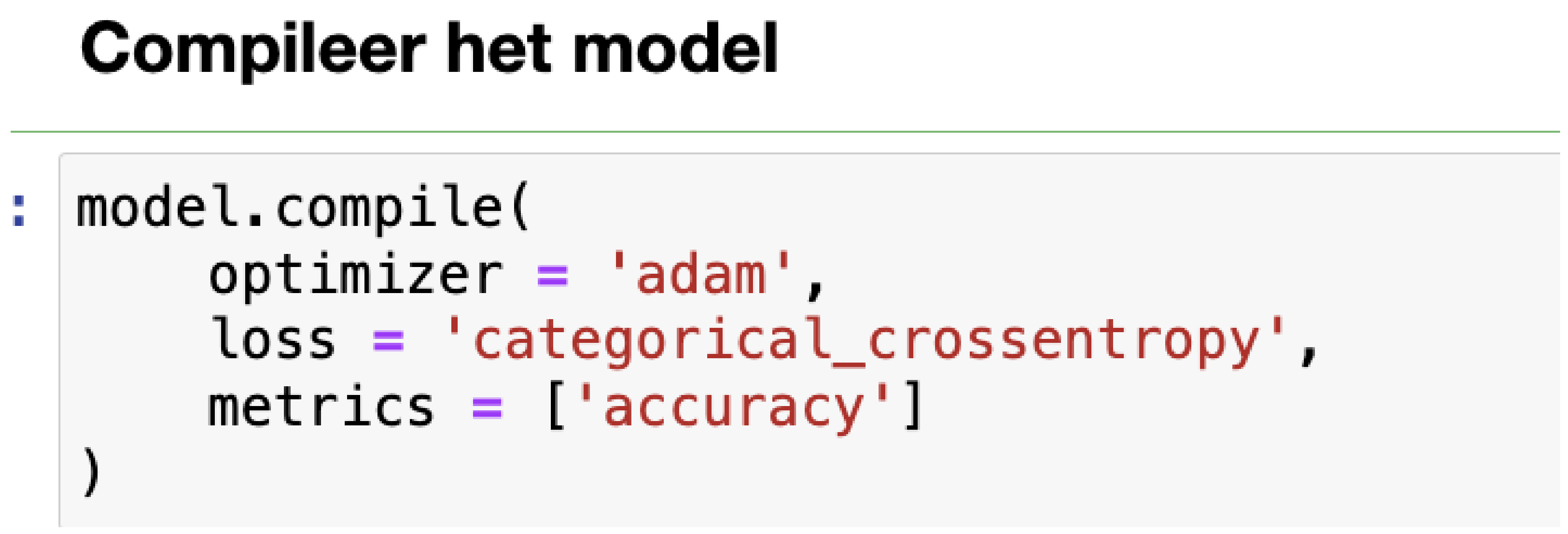
Conv2D: = (Kernel Height * Kernel Width * Input Channels + 1) * Number of Filters

Pooling: always zero

Flatten: always zero

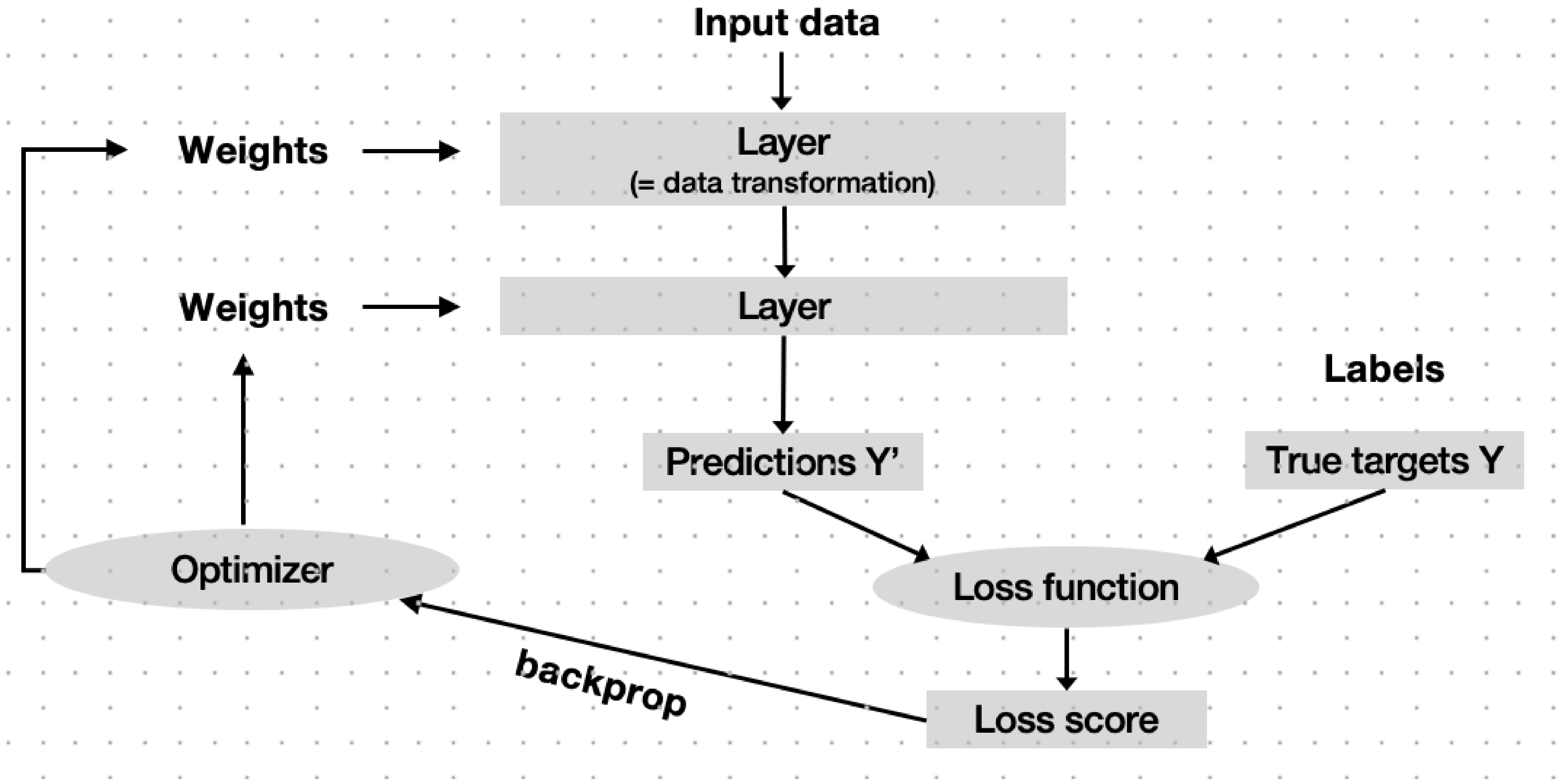
Dense: (Input units * Output units) + Biases

Model compileren met Keras



Compileer het model

```
model.compile(
    optimizer = 'adam',
    loss = 'categorical_crossentropy',
    metrics = ['accuracy']
)
```



Zie www.keras.io/api

Creating Tomorrow

Keras – www.keras.io

- Python library for deep learning.
- Developed by Francois Chollet at Google.
- Current version 3.0
- High level library: you just define the layers
- Uses Tensorflow, JAX or Pytorch as backend.

Summary CNN

1. **What is a “convolution”?**
   - A ‘convolution kernel’ is a matrix that goes step-by-step over the image and calculates pixel values.

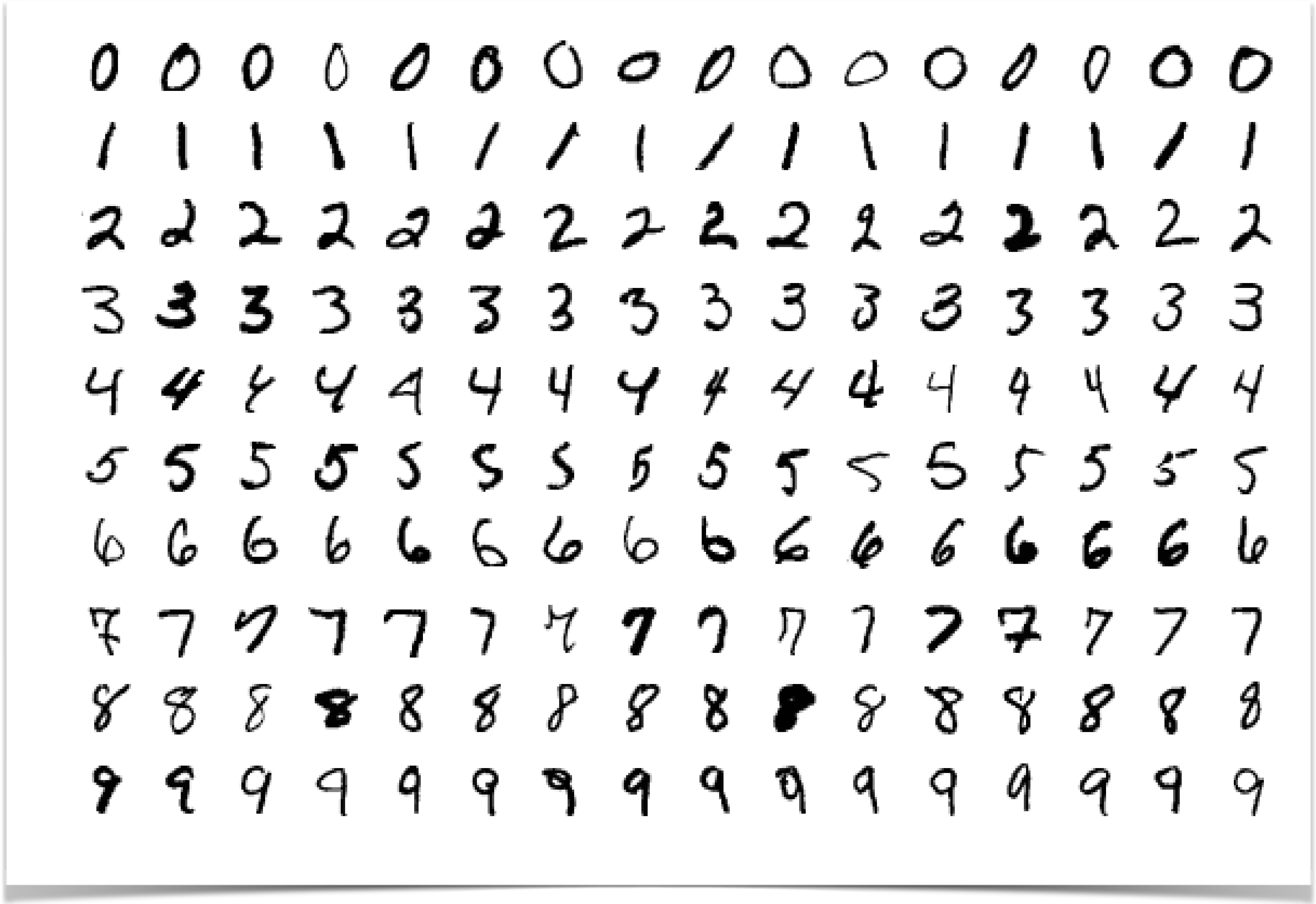
2. **What is difference with ‘normal’ NN’s?**
   - Conv2D + Pooling Layers

Summary CNN

3. **What are the layers of a CNN?**
   - Conv2D, Pooling, Flatten, Dense

4. **Summary: how can a CNN ‘learn’**
   - from small features to larger, more complex features

You will use the MNIST dataset



- Handwritten digits 0 - 9
- 1994
- 70.000 images
- 28 x 28 pixels
- Yann LeCunn

Notebook CNN

`2024_09_06_MNIST_CNN.ipynb`

NN	**ca. 0.94**
CNN	**> 0.99**

World record: **0.9983**

12:30 Bespreking

Name	Score CNN
Valentijn	0.992
WInston	0.9926
Sun	DQ
Ahmed	0.9971

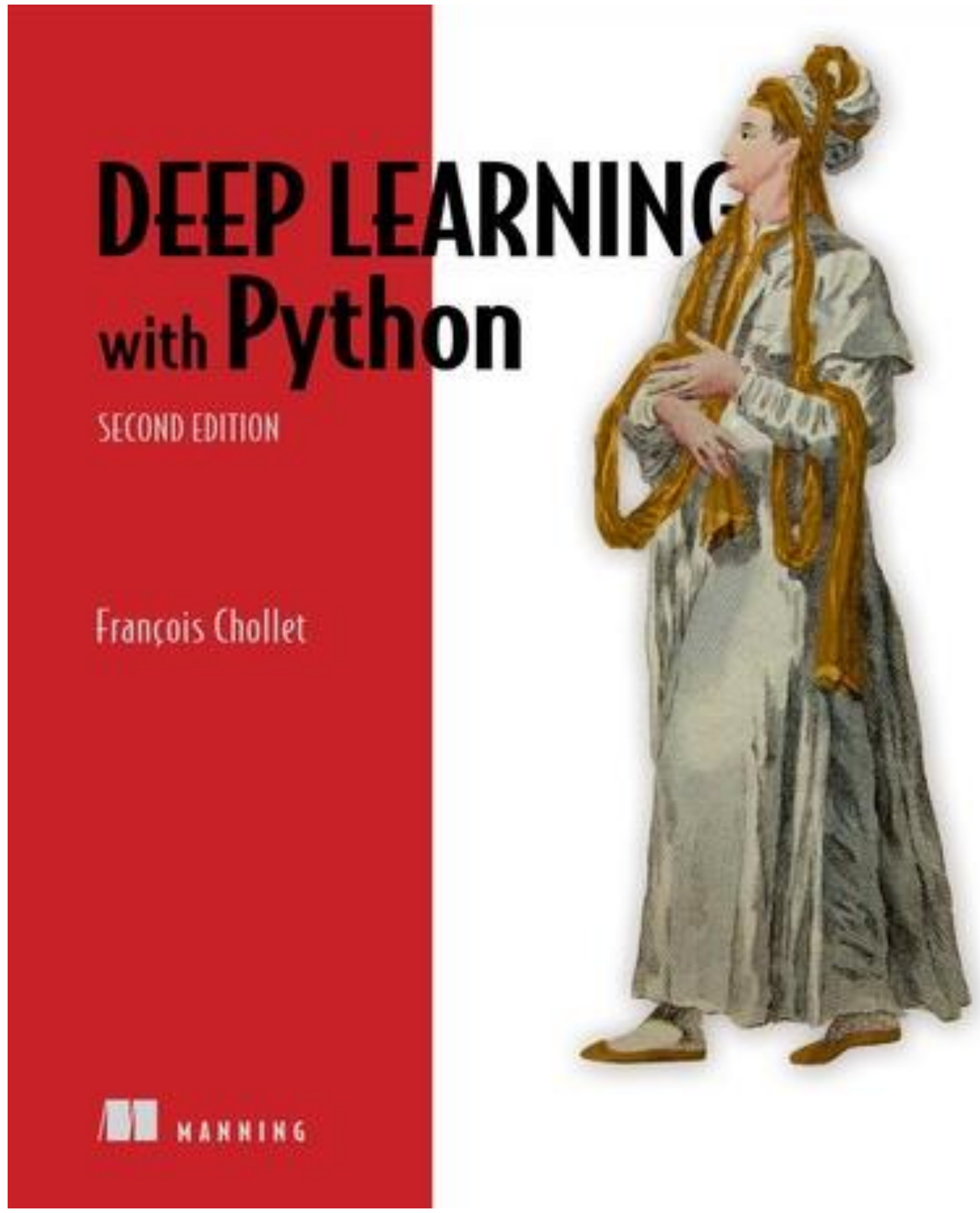
Read & view more

Book
‘Deep learning with python’ – Francois Chollet
https://learning-oreilly-com.rps.hva.nl/library/view/deep-learning-with/9781617296864/

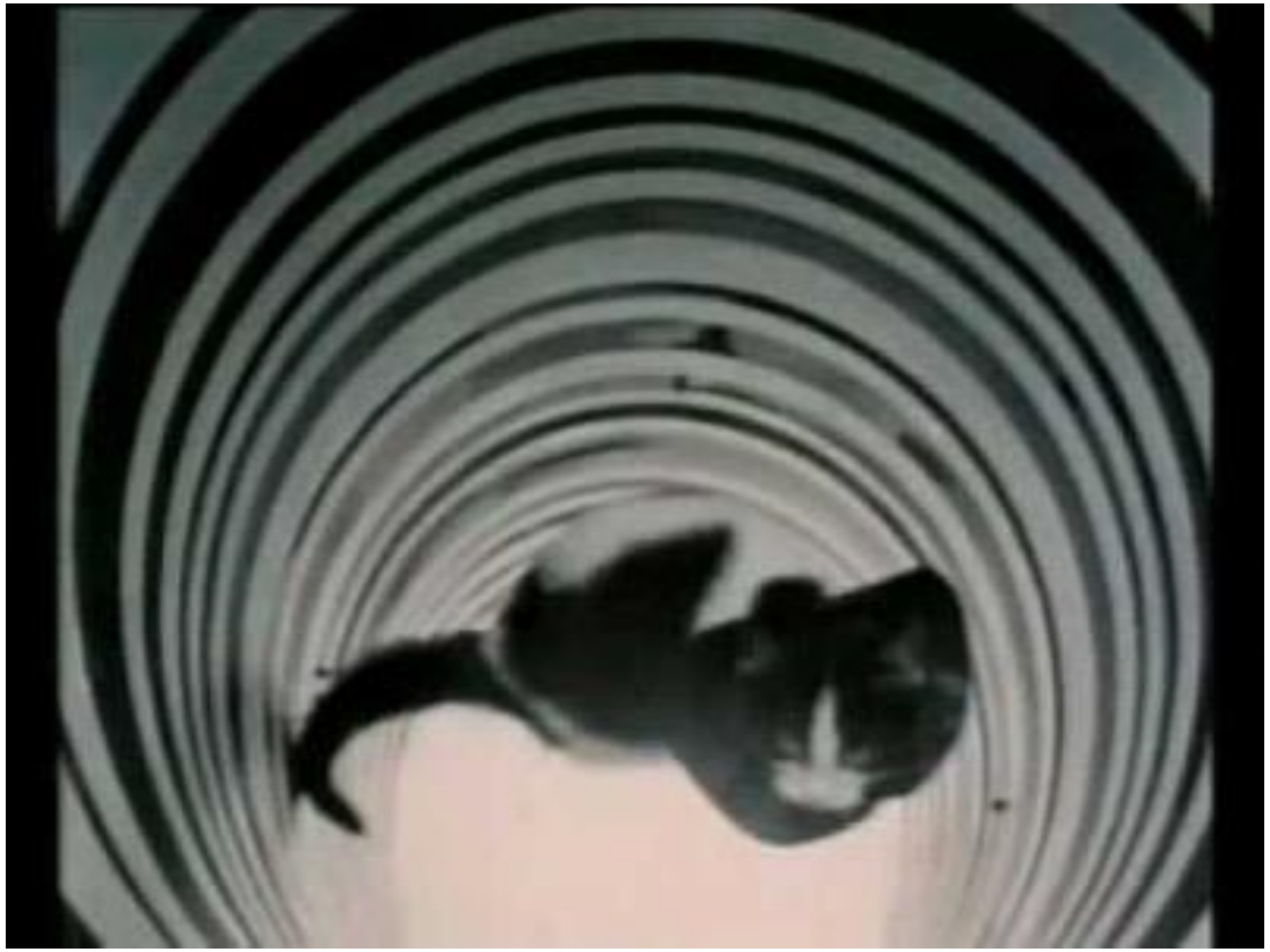
www.keras.io
Tutorials & code examples op www.keras.io

Neural network in 60 seconds - Youtube
https://www.youtube.com/shorts/LCE3LY-iSac

But what is a neural network? Youtube
https://www.youtube.com/watch?v=aircAruvnKk&



Bonus: Can cats see lines from birth? Or do they develop their vision by 'training' their brain?



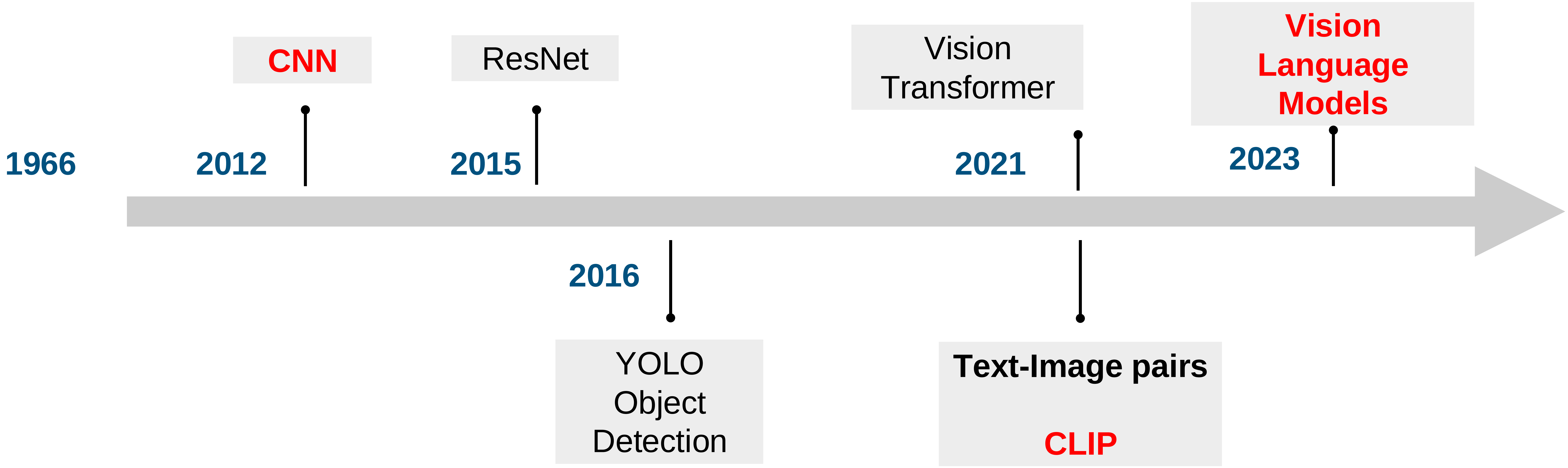
- https://youtu.be/QzkMo45pcUo?t=198

Modern CV

- Image embeddings with CLIP

- Vision Language Models

Timeline Computer Vision



1966

2012 CNN

2015 ResNet

2016 YOLO Object Detection

2021 Vision Transformer

Text-Image pairs

CLIP

2023 Vision Language Models

Creating Tomorrow

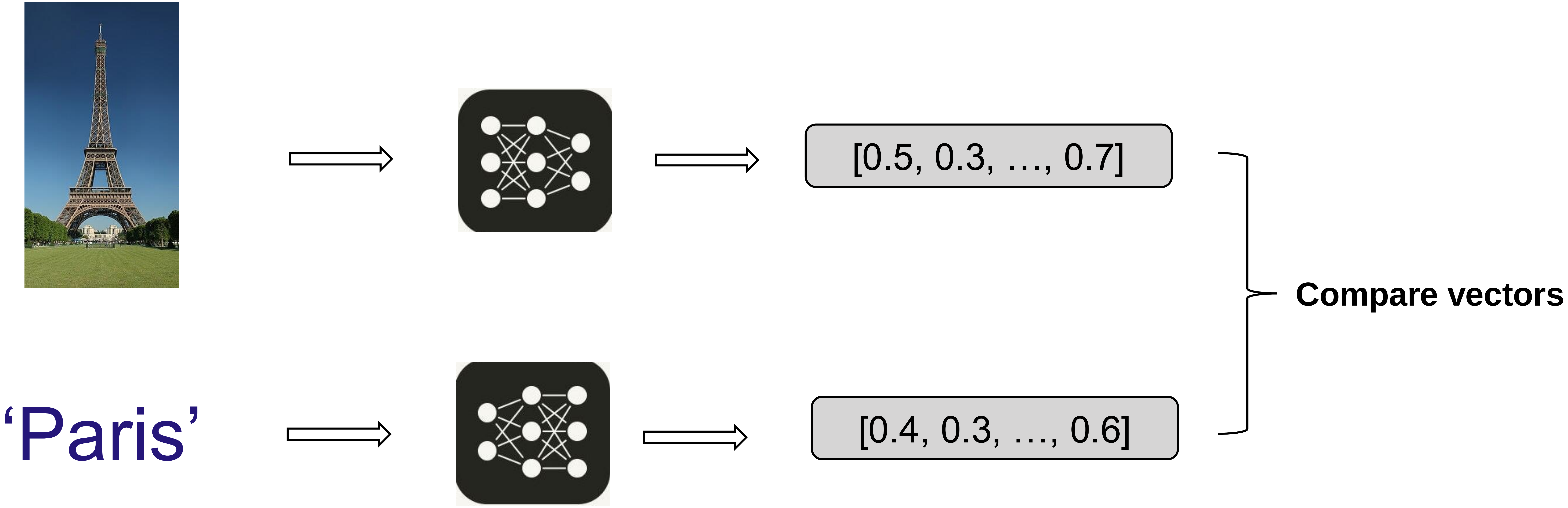
How can computers find this relationship?

‘Paris’

‘vector embeddings’

Read more: https://www.geeksforgeeks.org/what-are-embeddings-in-machine-learning/

We use an ‘embedding model’ and compare the vectors. Vectors capture the meaning.



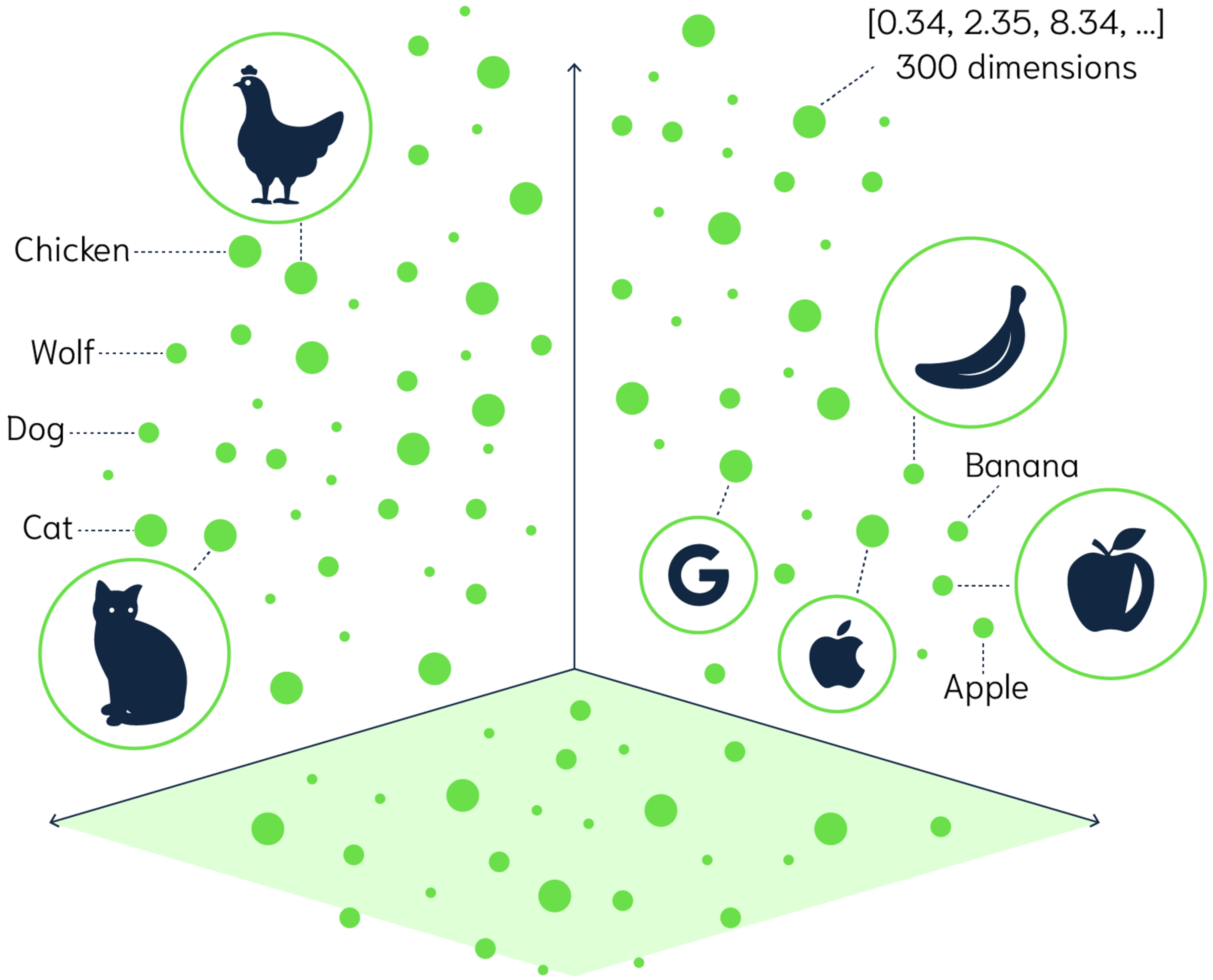
‘Paris’

[0.5, 0.3, …, 0.7]

[0.4, 0.3, …, 0.6]

Compare vectors

Texts and images in vector space



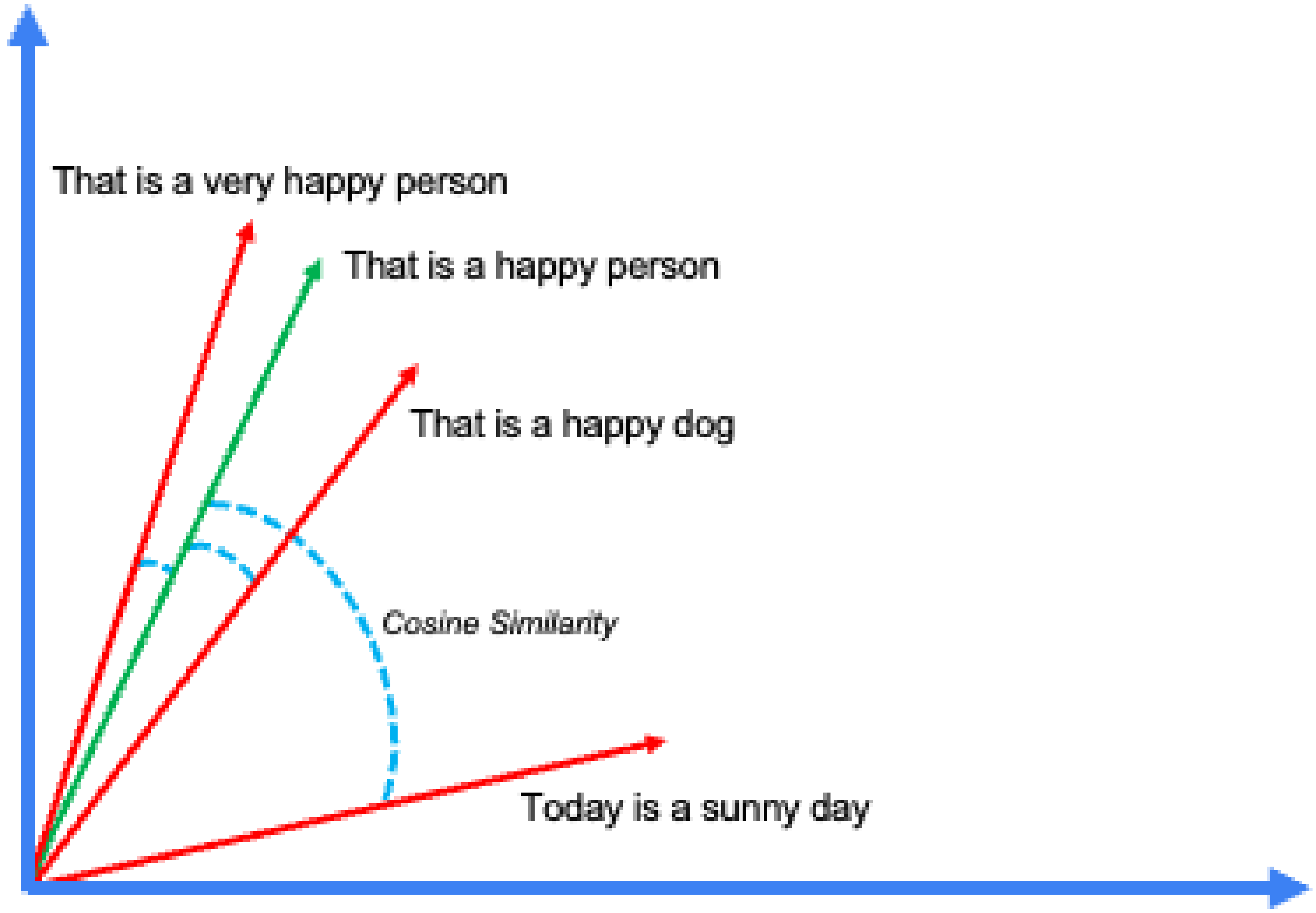
Words and images with same meaning are close in vector space.

Compare vectors to find most similar

- Comparing vector embeddings is known as ‘similarity search’

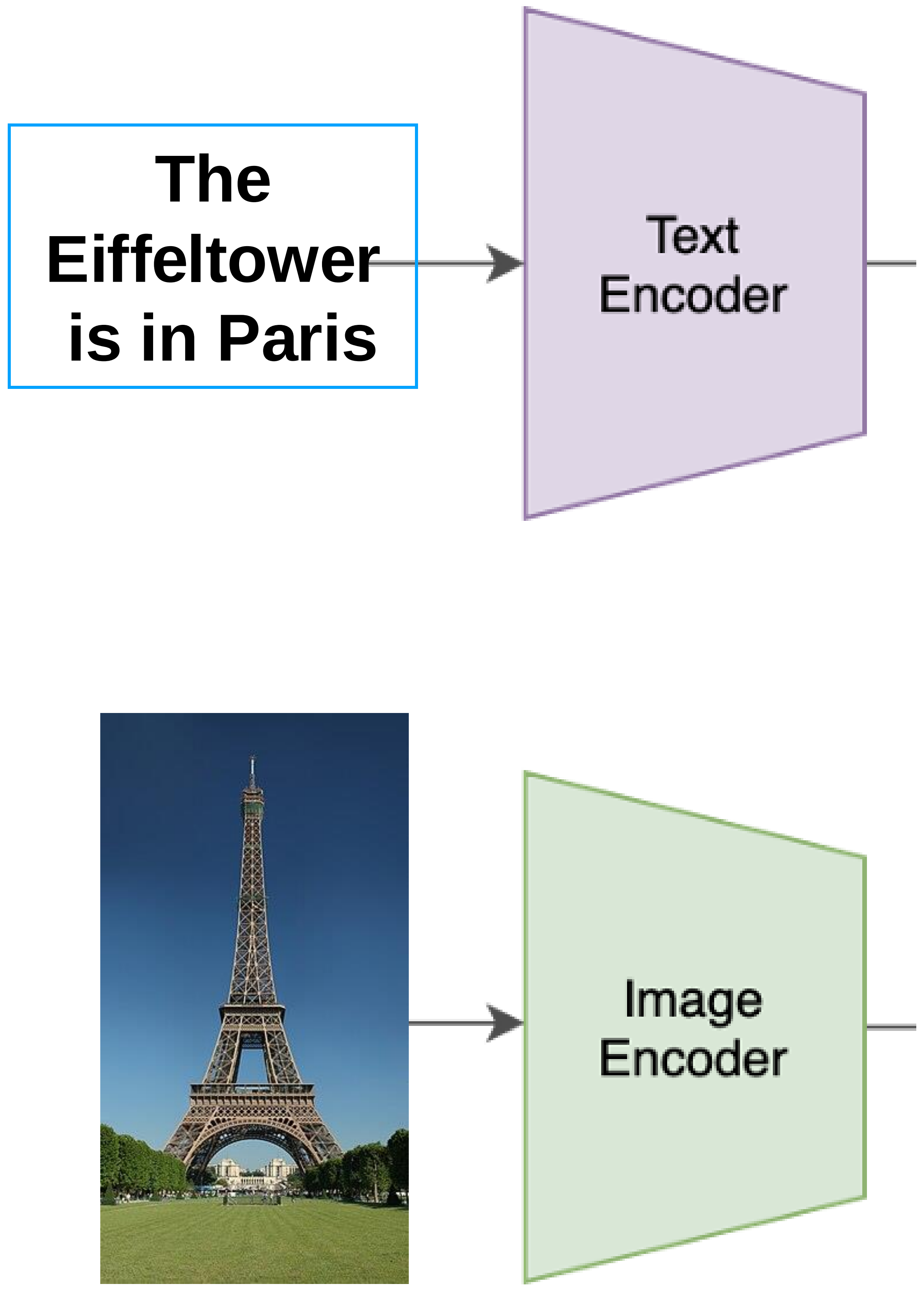
- Most often we use ‘cosine similarity’.
  - See notebook.

- It gives meaning to search – not just ‘strings’.



We will use OpenAI’s CLIP with text-image pairs

Contrastive Language-Image Pre-training



The Eiffeltower is in Paris

Text Encoder

Image Encoder

- Pairs of images with captions
- Trained on 400 million pairs. Published in 2021.

- Use CLIP to describe images => ‘**image2text**’
- Dall-e is the reverse => **text2image**

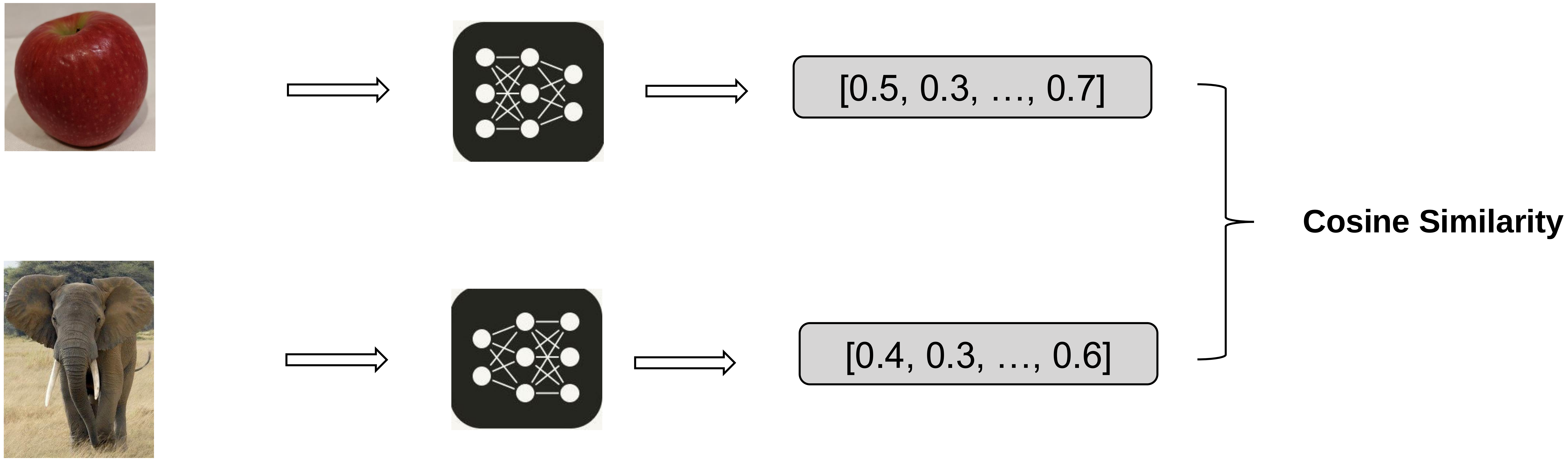
Sources:

- https://github.com/OpenAI/CLIP
- https://openai.com/research/clip

Notebook similarity and clustering.

- Image similarity and clustering.ipynb

Notebook similarity search + clustering



[0.5, 0.3, …, 0.7]

[0.4, 0.3, …, 0.6]

Cosine Similarity

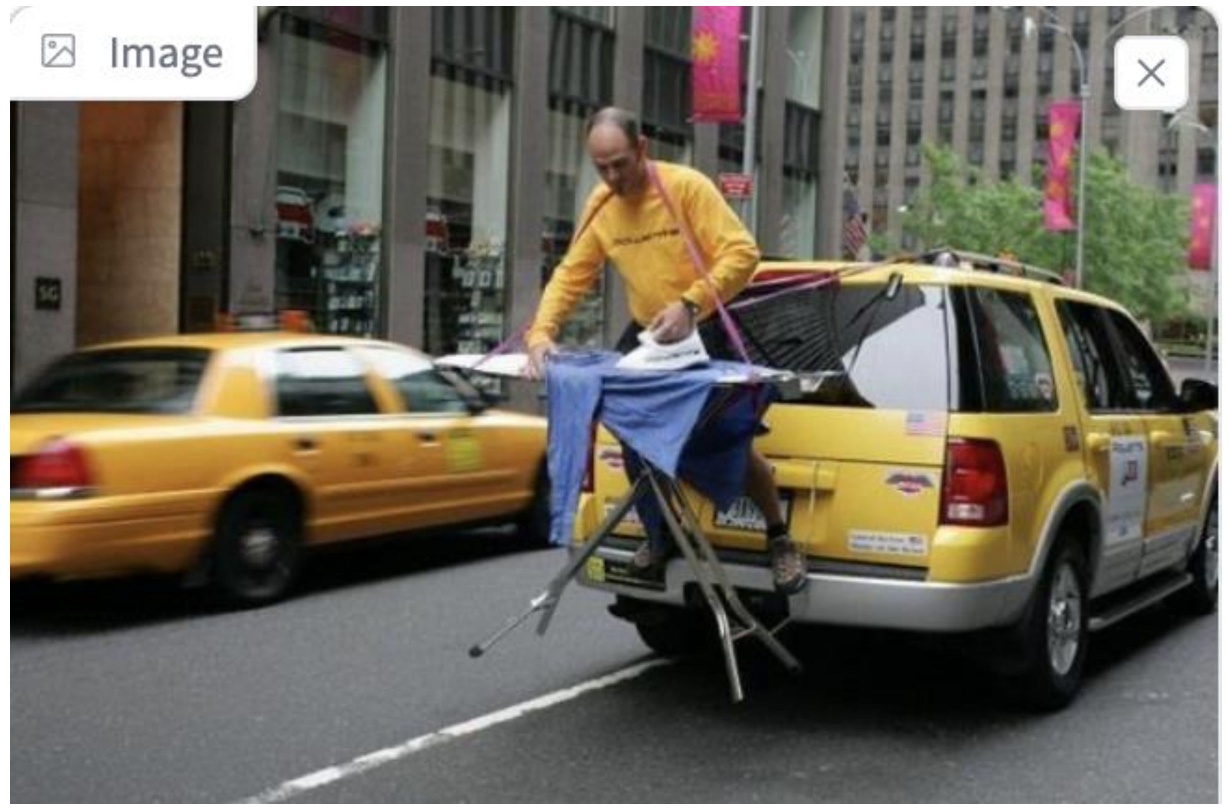
Modern CV

- Vision Language Models

www.ollama.com

Creating Tomorrow

What’s happening here?

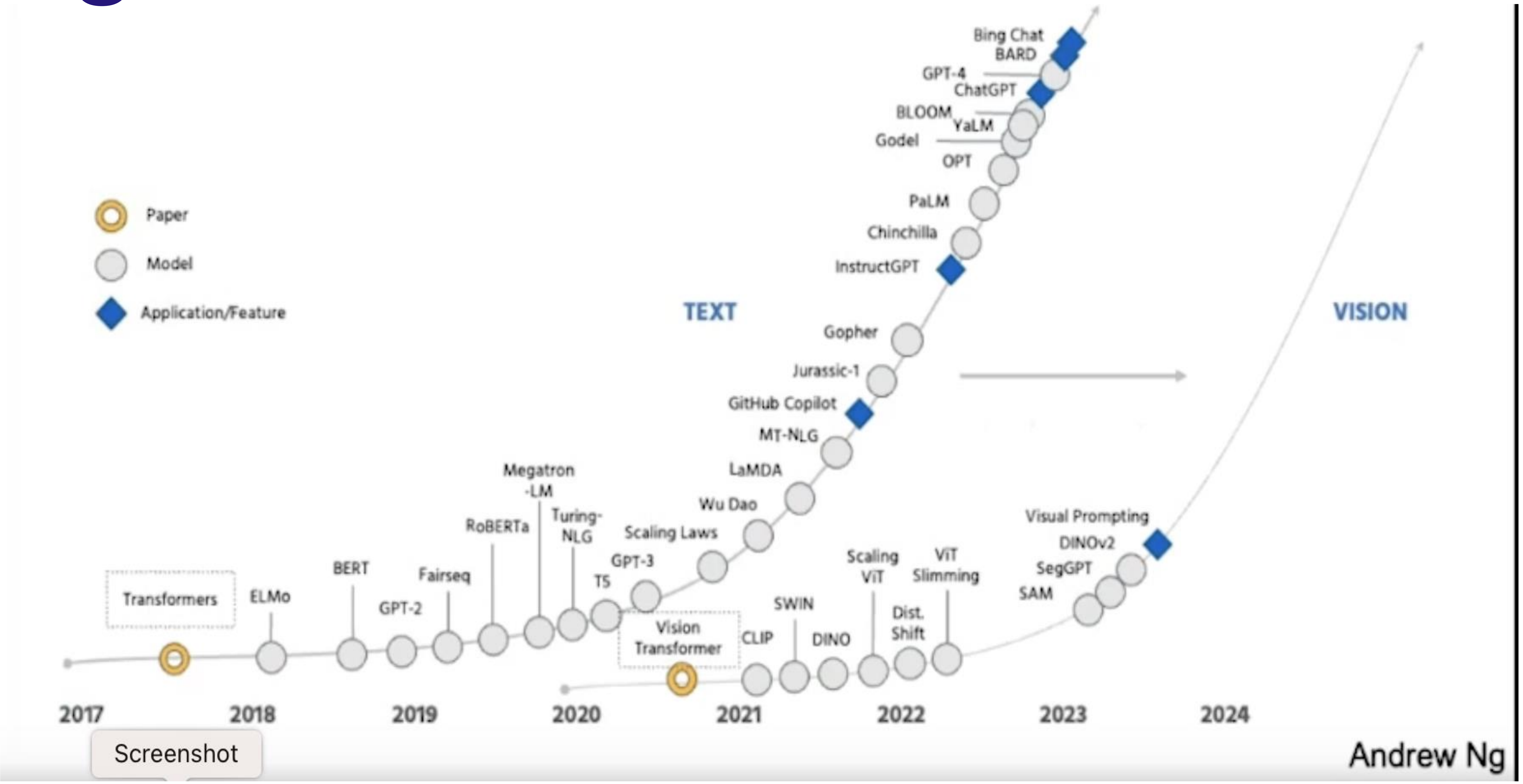


“What is unusual about this image?”

???

“The image shows a person ironing clothes on the back of a moving vehicle,…”

The ChatGPT revolution is coming to vision!



Large Vision Models =. Large Multimodal Models

Creating Tomorrow

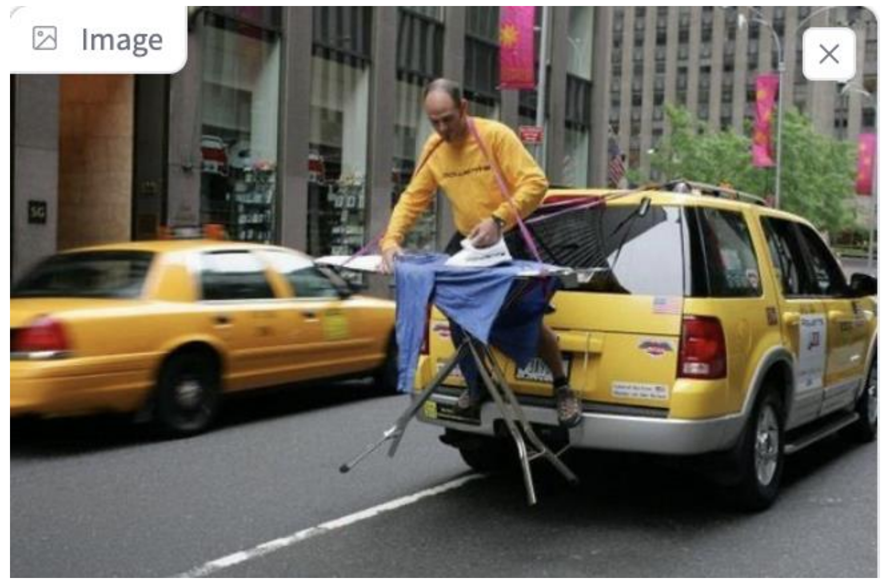
What you can do with Vision models

Talk with them using LLaVA or any other Vision – Language Model!

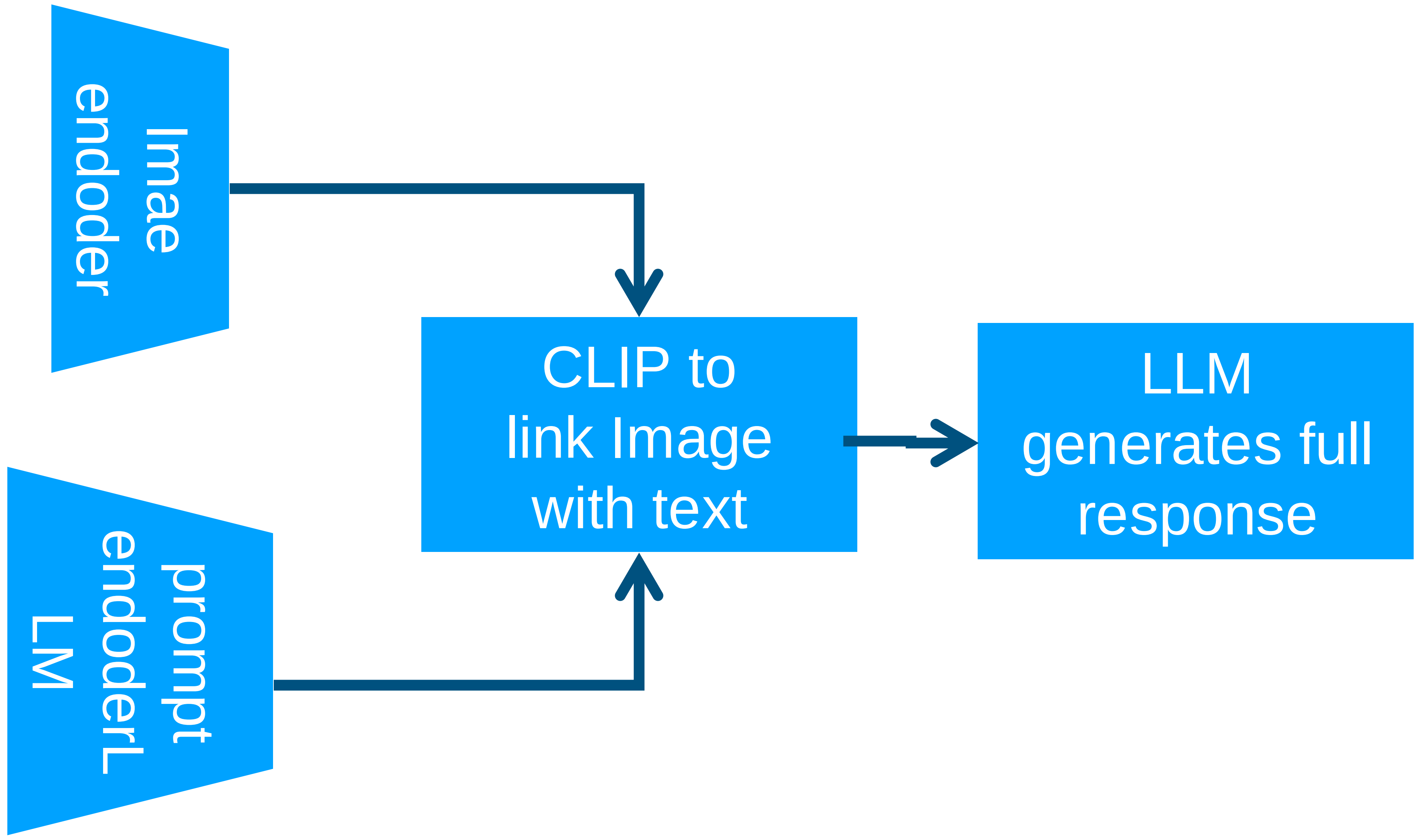
1. Human input with speech-2-text
2. Visual Question Answering
3. Text-2-Speech

LLaVA: Vision Language Model

Combines CLIP with a Large Language Model



“What is unusual about this image?”



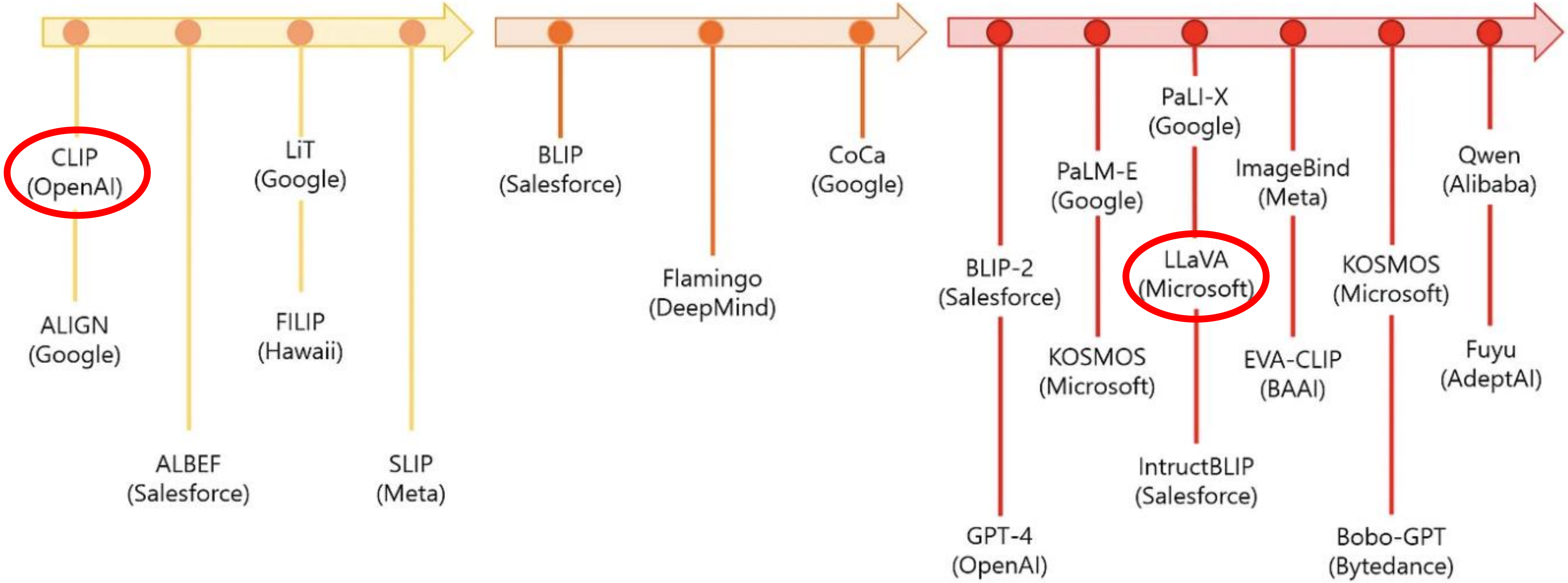
“The image shows a person ironing clothes on the back of a moving vehicle,…”

This is also called a ‘neck & head architecture’.

Try LLaVA on Huggingface

https://huggingface.co/spaces/merve/llava-next

Vision Language Models



Run models locally with ollama

- Download and install via www.ollama.com (Mac / Windows / Linux)

! Check your laptop

- Min. 8 Gb RAM +
- 20 Gb free disk space

Why ollama?
Why running LLM’s on your laptop?

1. Data stays on your device
   - Privacy / no leakage of sensitive data
2. No longer dependent on internet connection
3. Lower costs with less data usage
4. Less energy usage
5. Bring your own model

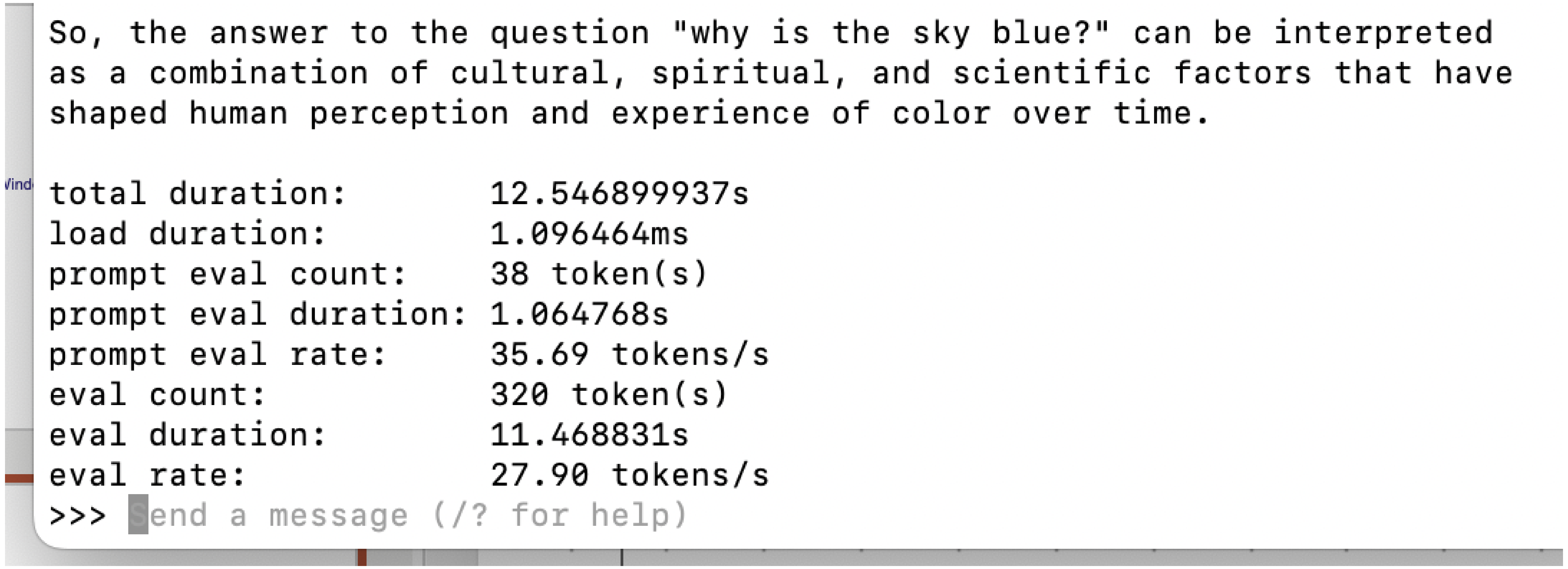
run ollama

- Download and install via www.ollama.com (Mac / Windows / Linux)
- Start from de terminal (CLI):
  `ollama run llava` (or any other model from ollama.com)
- Start chatting with the model!
- End ollama with `CTRL+C` or `/bye` or `/exit`
- You can then start another model.

Creating Tomorrow

Evaluate performance

```
ollama run mistral --verbose
```



```
So, the answer to the question "why is the sky blue?" can be interpreted
as a combination of cultural, spiritual, and scientific factors that have
shaped human perception and experience of color over time.

total duration:       12.546899937s
load duration:        1.096464ms
prompt eval count:    38 token(s)
prompt eval duration: 1.064768s
prompt eval rate:     35.69 tokens/s
eval count:           320 token(s)
eval duration:        11.468831s
eval rate:            27.90 tokens/s
>>> Send a message (/? for help)
```

Ollama has a python package

- `pip install ollama`
- Use a Jupyter notebook to program with ollama.
- Create a simple front-end with Gradio. www.gradio.app
- Ollama is a wrapper around a project called llama.cpp: https://github.com/ggerganov/llama.cpp

Notebook VLM’s with ollama

Describe images with llava + ollama

`ollama_llava_challenges.ipynb`