# SpotOn

## Progress #2

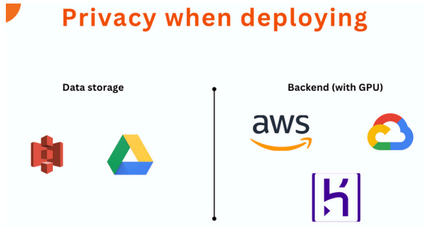
# Agenda

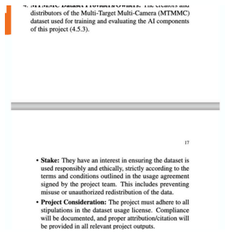
- Recap
- Architecture
  - CNN
    - One-stage v.s. Two-stage CNN
  - ViT
- Update the model’s latency when using GPU
- Re-Identification
  - Re-Identification model selection
  - Re-Identification model experimental
- Camera view selection

SpotOn

# Recap

# Feedback & Solutions

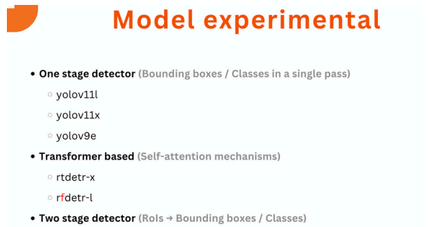



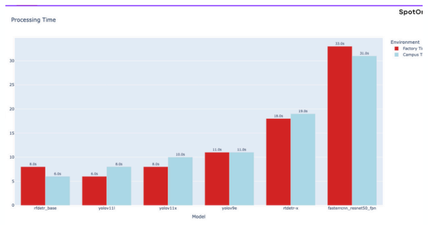


distributors of the Multi-Target Multi-Camera (MTMMC) dataset used for training and evaluating the AI components of this project (4.5.3).

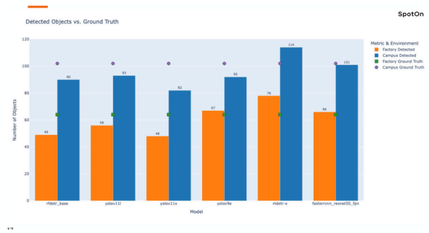
17

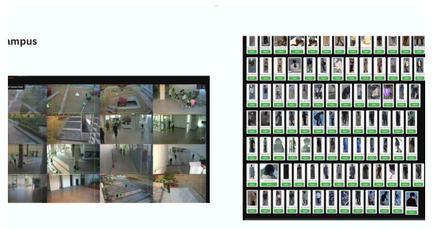
- **Stake:** They have an interest in ensuring the dataset is used responsibly and ethically, strictly according to the terms and conditions outlined in the usage agreement signed by the project team. This includes preventing misuse or unauthorized redistribution of the data.
- **Project Consideration:** The project must adhere to all stipulations in the dataset usage license. Compliance will be documented, and proper attribution/citation will be provided in all relevant project outputs.


# Detection models experimental



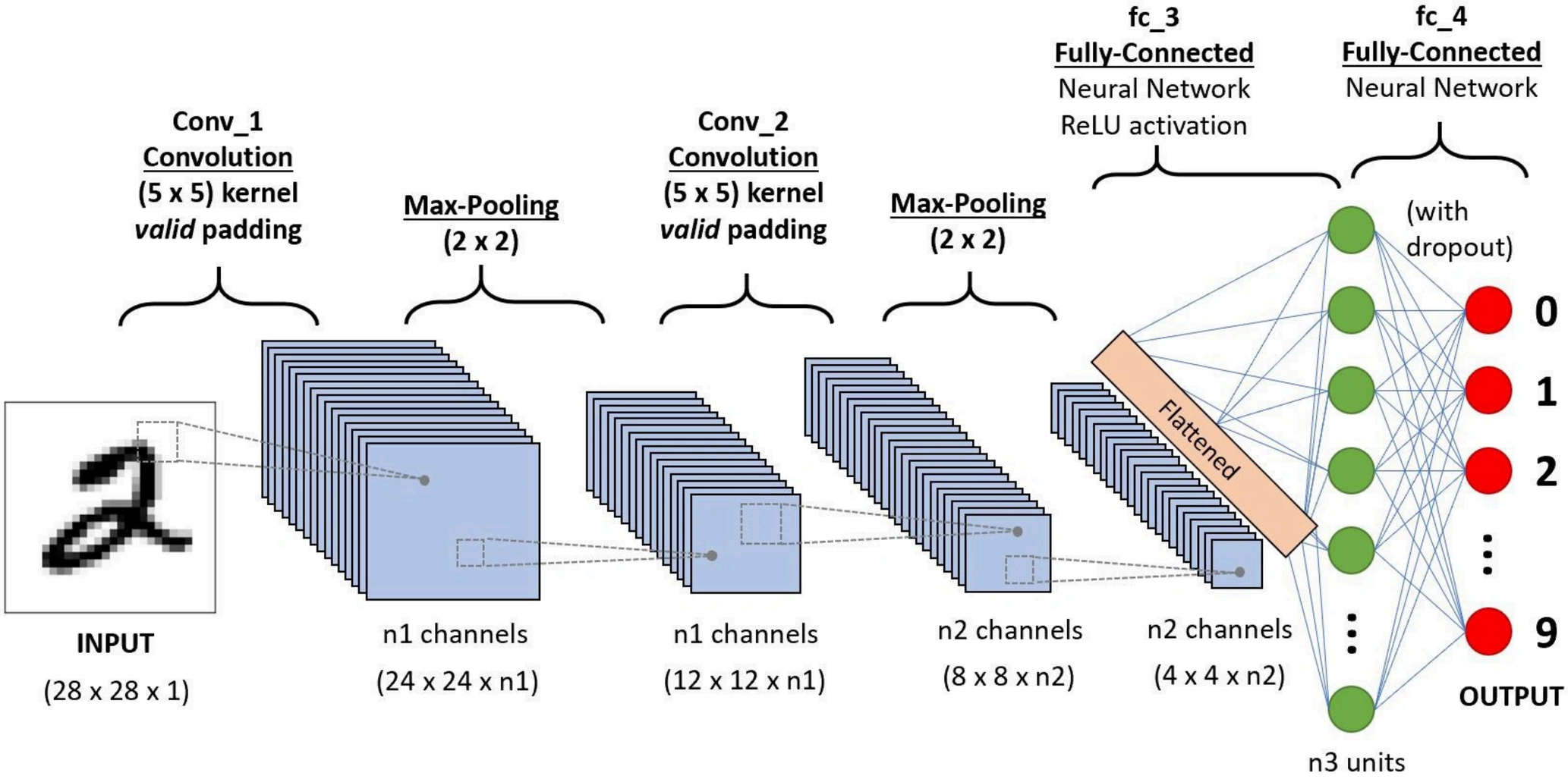



# Architecture (Simple term)

# CNN (1989)

SpotOn

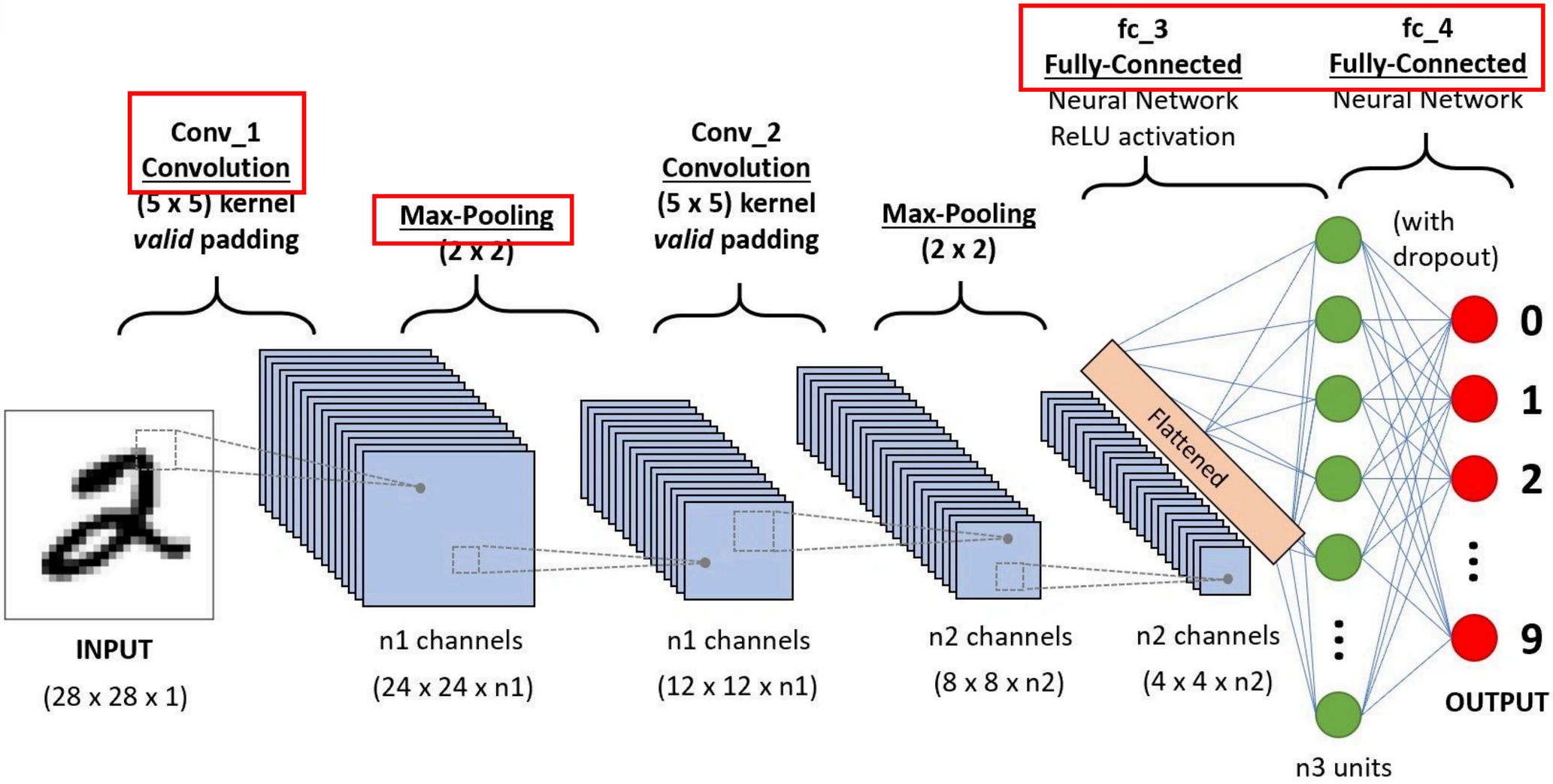





https://www.analyticsvidhya.com/blog/2020/10/what-is-the-convolutional-neural-network-architecture/

# CNN (1989)

SpotOn

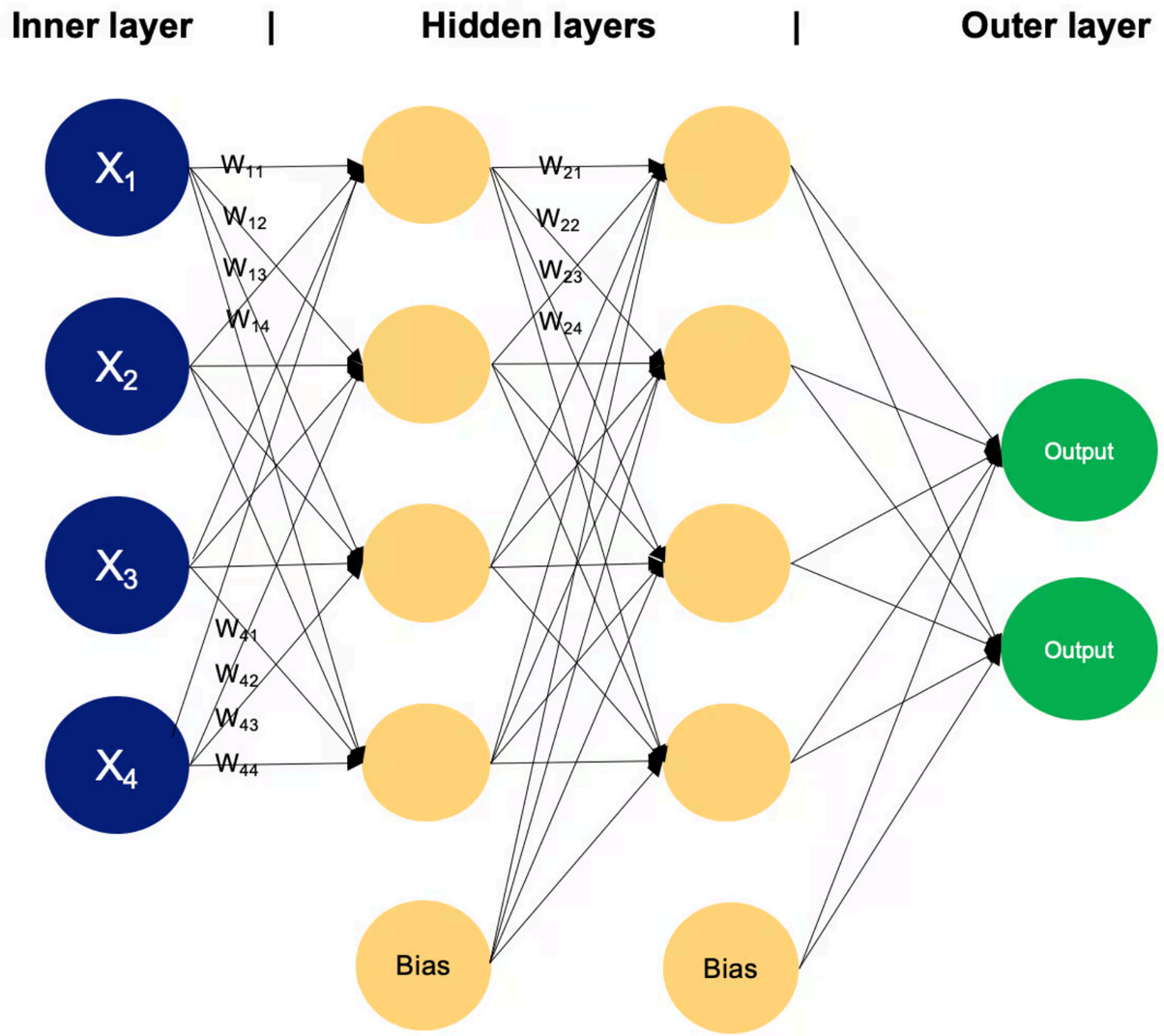





https://www.analyticsvidhya.com/blog/2020/10/what-is-the-convolutional-neural-network-architecture/

# MLP

SpotOn

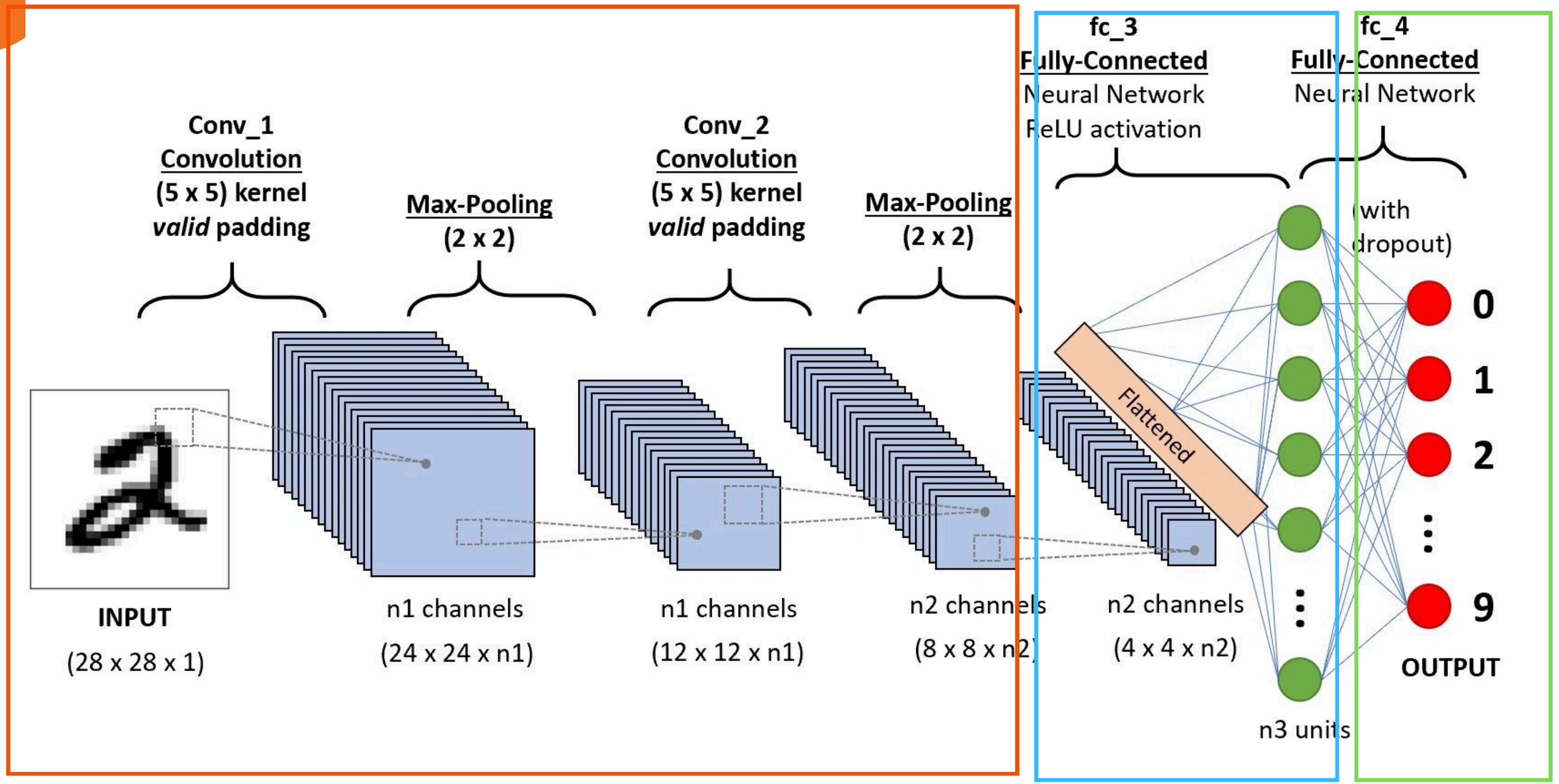




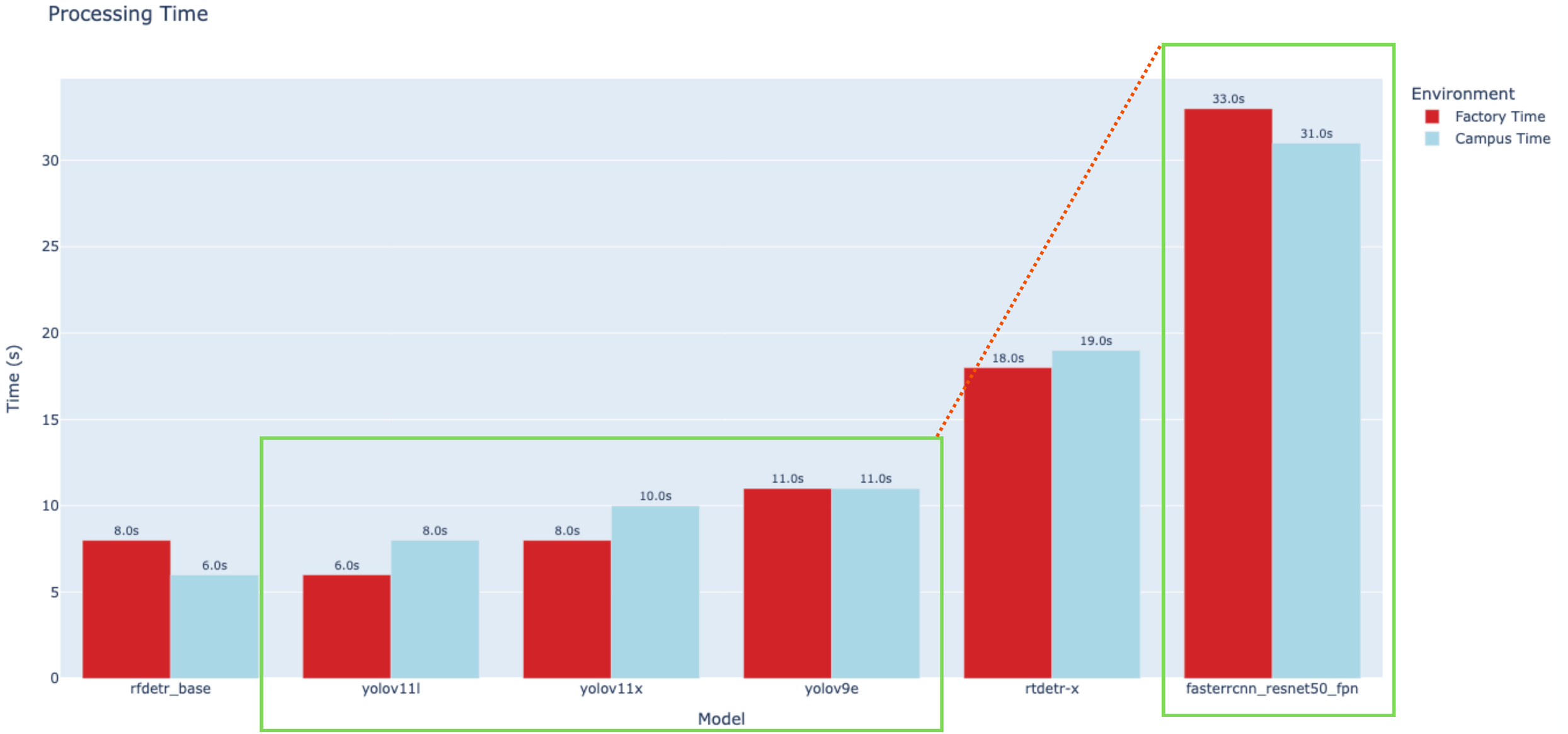

https://www.datacamp.com/tutorial/multilayer-perceptrons-in-machine-learning

# MLP

SpotOn



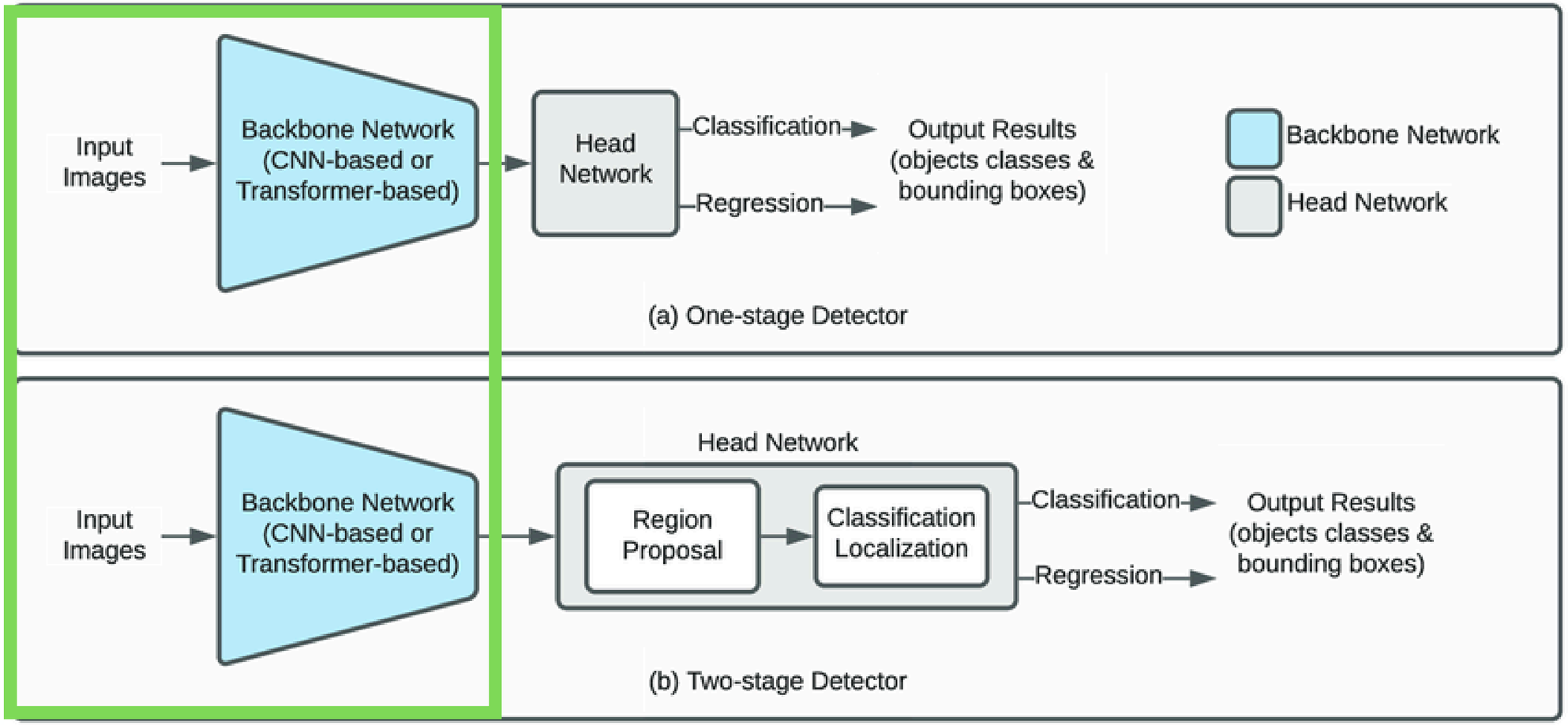

https://www.analyticsvidhya.com/blog/2020/10/what-is-the-convolutional-neural-network-architecture/






https://www.analyticsvidhya.com/blog/2020/10/what-is-the-convolutional-neural-network-architecture/

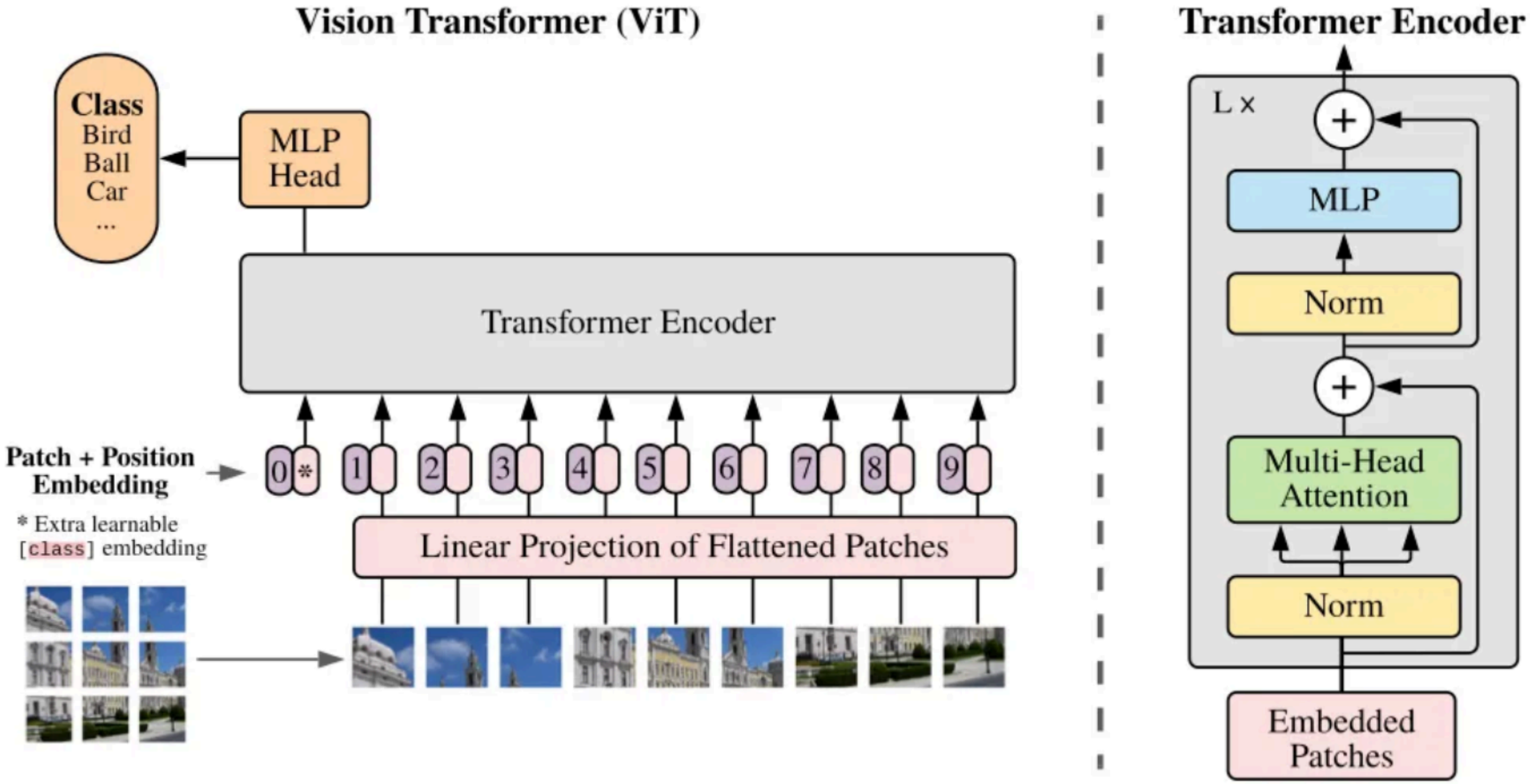
# One/Two Stage




 https://www.researchgate.net/figure/Basic-deep-learning-based-one-stage-vs-two-stage-object-detection-model-architectures_fig2_358362847

# ViT (2020)

SpotOn

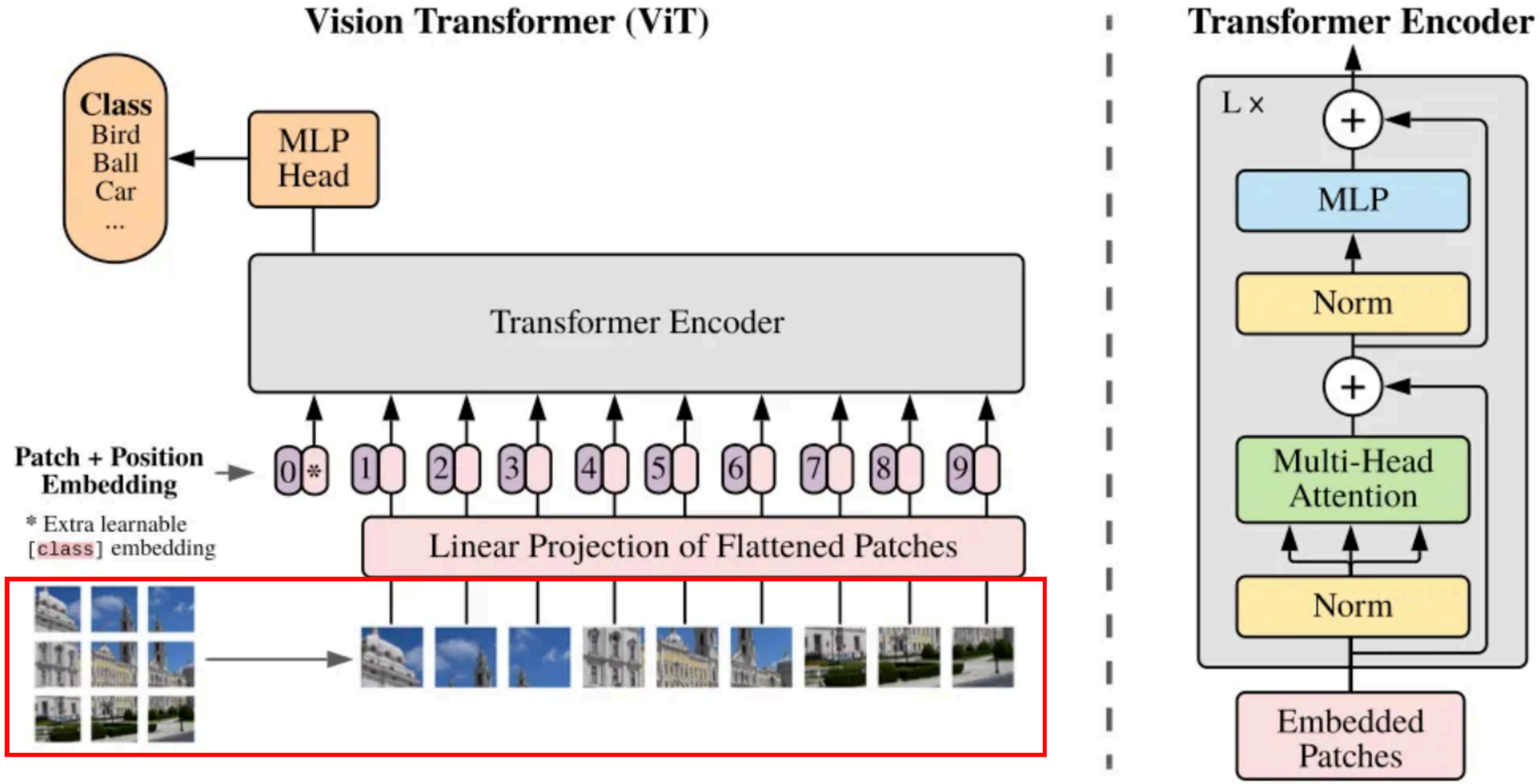





https://medium.com/@RTae/hello-transformer-and-goodbye-cnn-4ebda6e4aa5e

# ViT (2020)

SpotOn

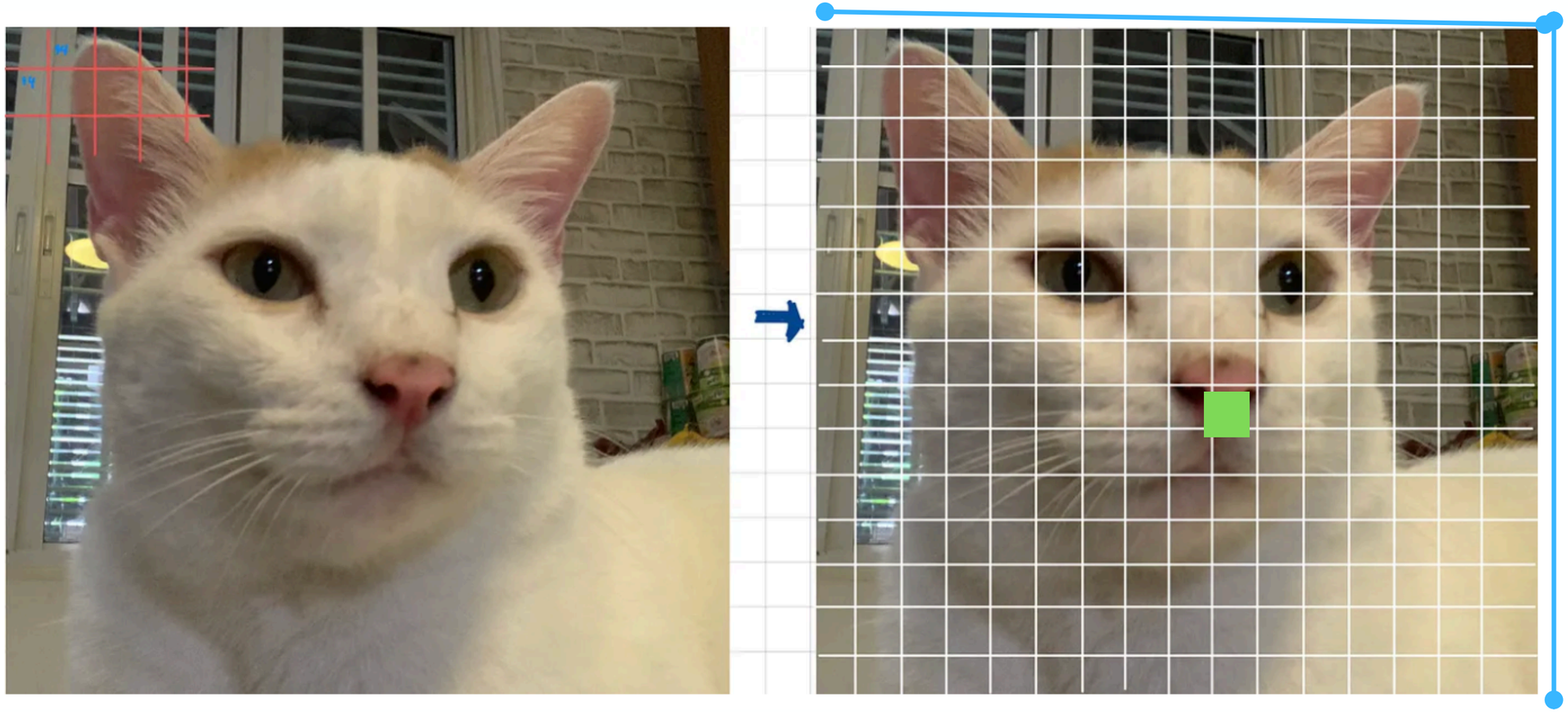





https://medium.com/@RTae/hello-transformer-and-goodbye-cnn-4ebda6e4aa5e

# 1. Split an image into patches

SpotOn



224 x 224 x 3 (RGB)

(H x W x C)

→

N (number ) = 16 x 16 = 256

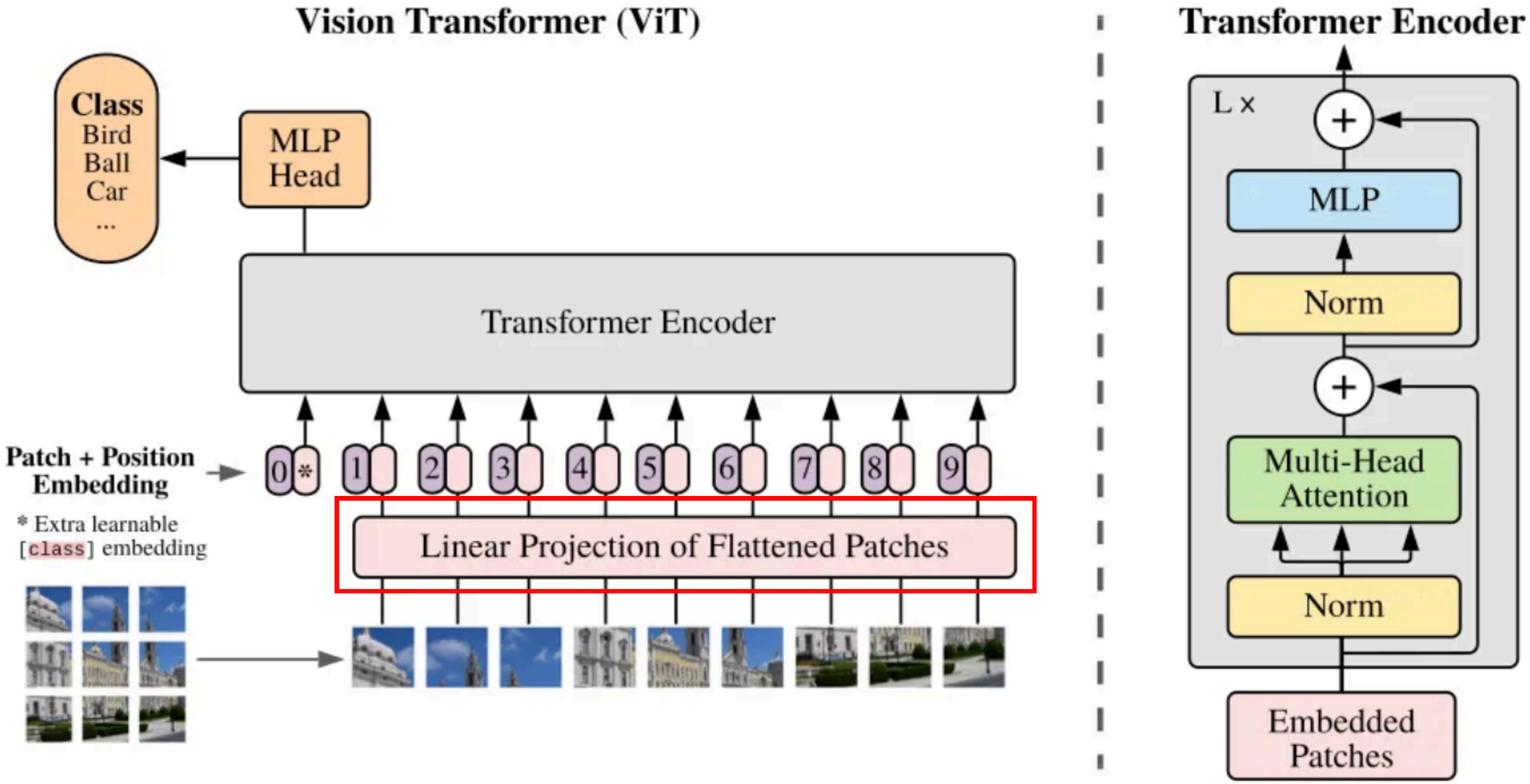
P (patch size) = 14 x 14 = 196



https://medium.com/@RTae/hello-transformer-and-goodbye-cnn-4ebda6e4aa5e

# ViT (2020)

SpotOn






https://medium.com/@RTae/hello-transformer-and-goodbye-cnn-4ebda6e4aa5e

# 2. Flattening Each Patch

SpotOn

2 (height) x 2 (width) x 3 (channels - R, G, B)

- **Red Channel (Layer 0):**

```
[[ R1, R2 ],
 [ R3, R4 ]]
```

Example values: `[[10, 20], [30, 40]]`

- **Green Channel (Layer 1):**

```
[[ G1, G2 ],
 [ G3, G4 ]]
```

Example values: `[[51, 61], [71, 81]]`

- **Blue Channel (Layer 2):**

```
[[ B1, B2 ],
 [ B3, B4 ]]
```

Example values: `[[92, 82], [72, 62]]`

Flattened Vector = `[ [R1,G1,B1], [R2,G2,B2], [R3,G3,B3], [R4,G4,B4] ]`

Flattened Vector = `[10, 51, 92, 20, 61, 82, 30, 71, 72, 40, 81, 62]`

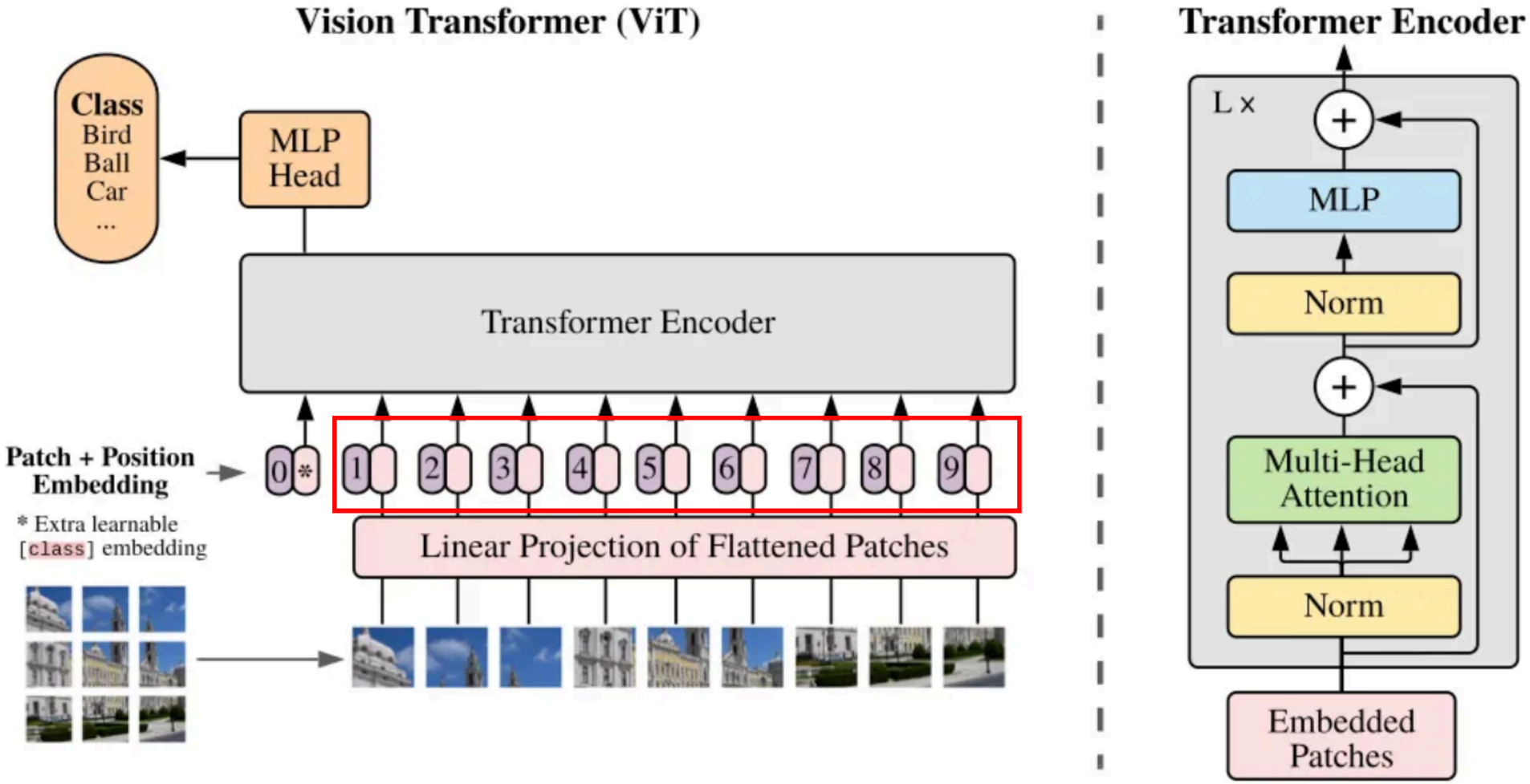
```
[
  [vector_for_patch_1],  // e.g., [10, 51, 92, ..., 62]
  [vector_for_patch_2],  // e.g., [15, 55, 95, ..., 65]
  [vector_for_patch_3],  // e.g., [12, 52, 92, ..., 60]
  ...
  ... N times (e.g., 256 times) ...
  ...
  [vector_for_patch_N]   // e.g., [18, 58, 98, ..., 68]
]
```



https://medium.com/@RTae/hello-transformer-and-goodbye-cnn-4ebda6e4aa5e

# ViT (2020)

SpotOn

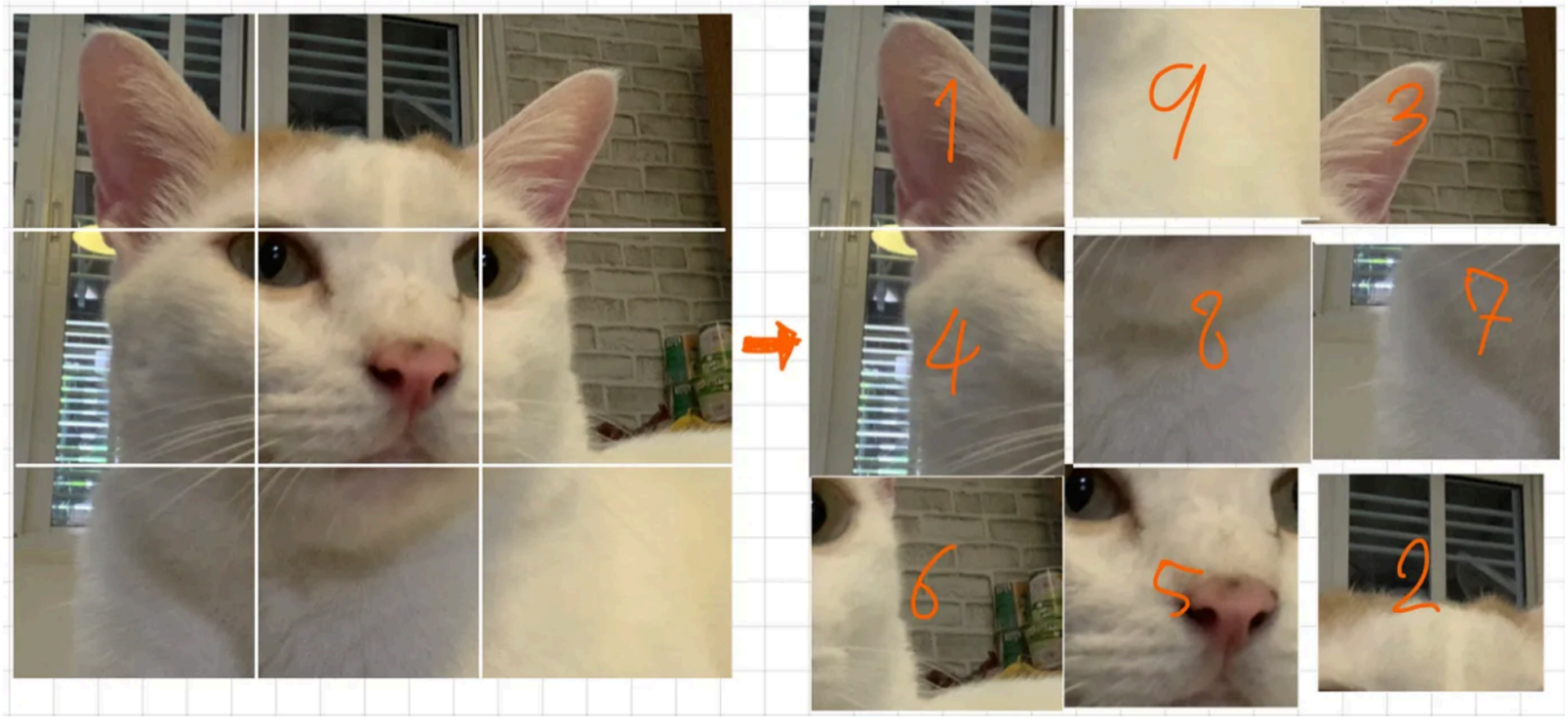





https://medium.com/@RTae/hello-transformer-and-goodbye-cnn-4ebda6e4aa5e

# 3. Add positional embeddings

SpotOn

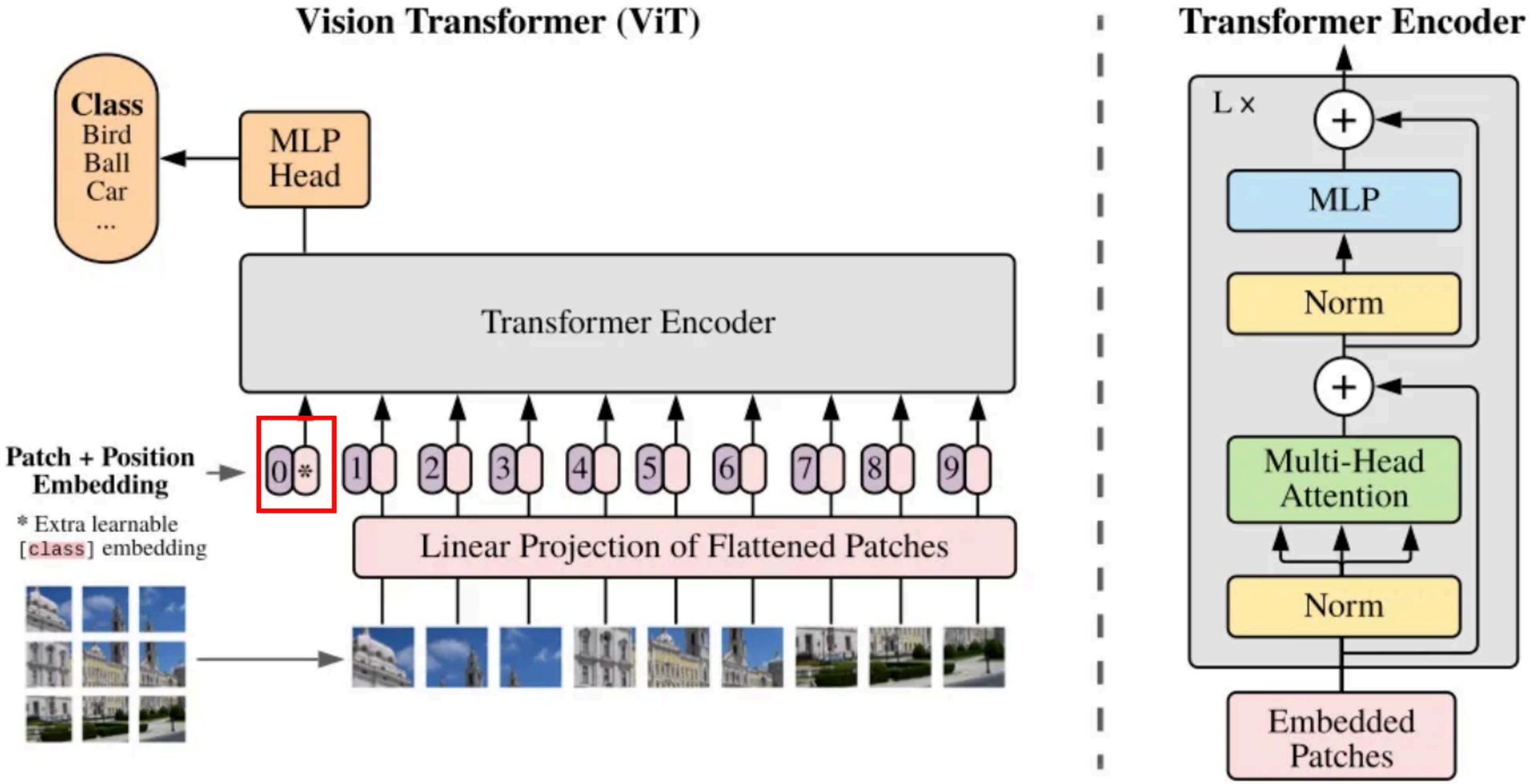





https://medium.com/@RTae/hello-transformer-and-goodbye-cnn-4ebda6e4aa5e

# 4. Add Class token for classification

SpotOn

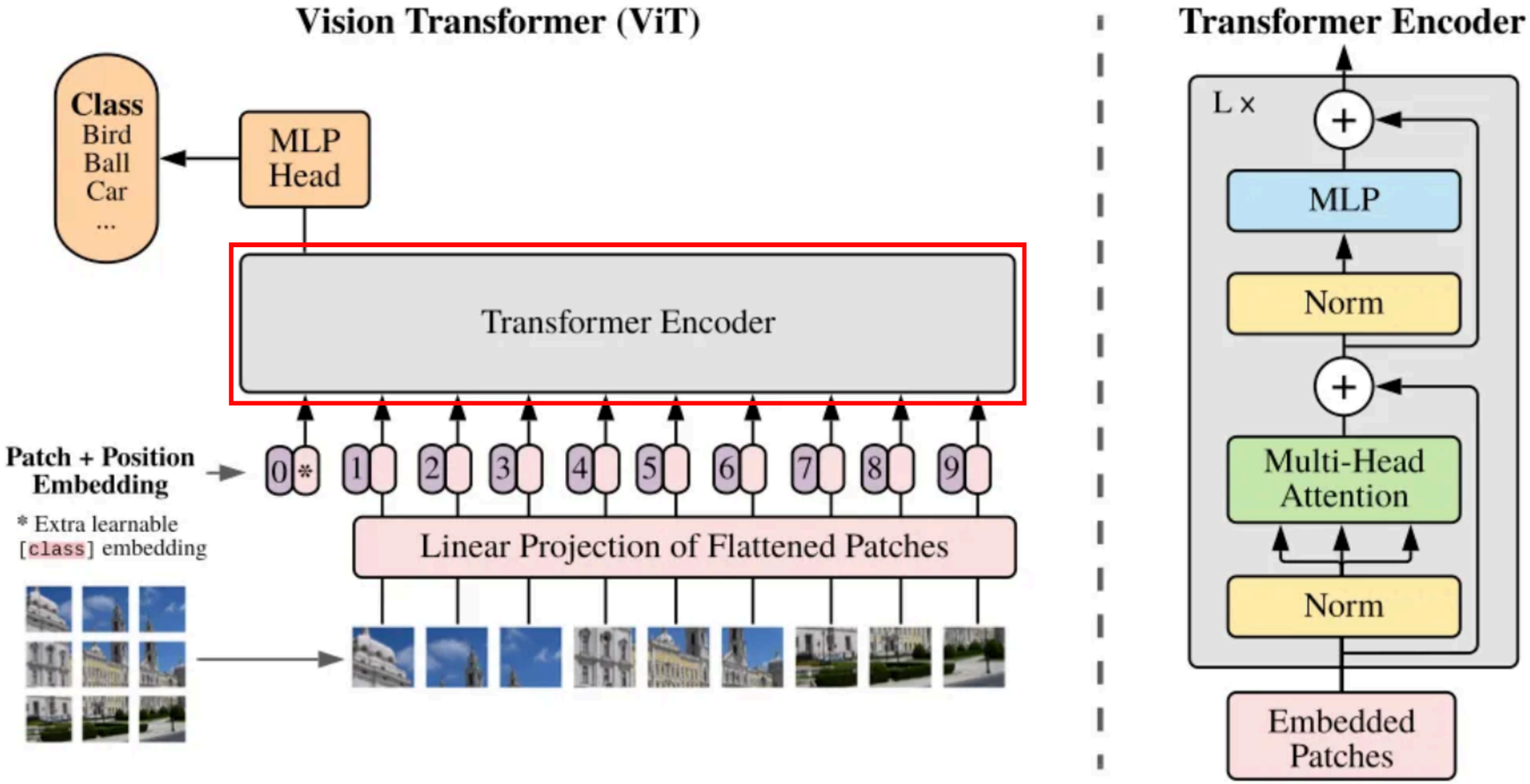





https://medium.com/@RTae/hello-transformer-and-goodbye-cnn-4ebda6e4aa5e

# ViT (2020)

SpotOn

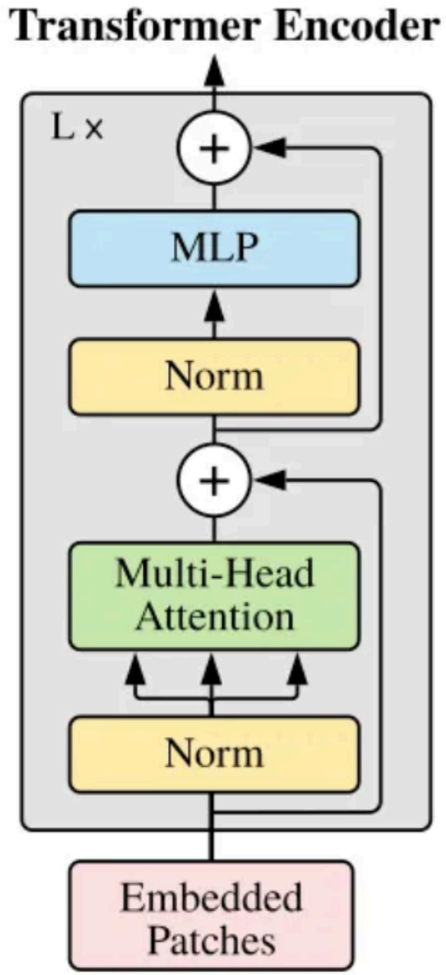





https://medium.com/@RTae/hello-transformer-and-goodbye-cnn-4ebda6e4aa5e

# 5. Feed the sequence as an input to standard transformer encoder

SpotOn




## Multi-Head Attention

1. Look at all patches
2. Calculate attention score (important)
3. Repeat multiple times in parallel

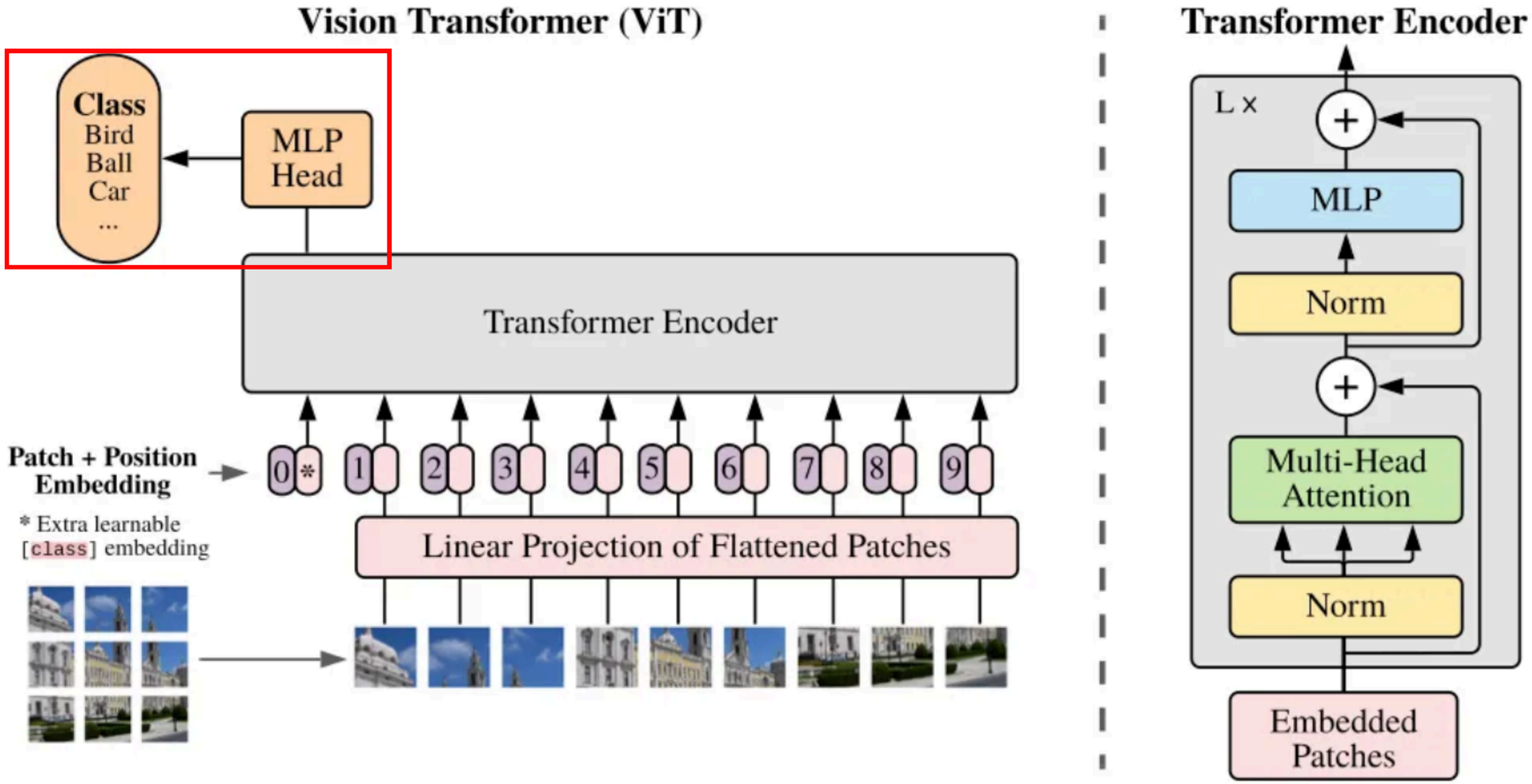
This allows the model to understand the context of the whole image e.g.  A patch of a cat's ear can learn it's related to a patch of the cat's eye



https://medium.com/@RTae/hello-transformer-and-goodbye-cnn-4ebda6e4aa5e

# Classify

SpotOn

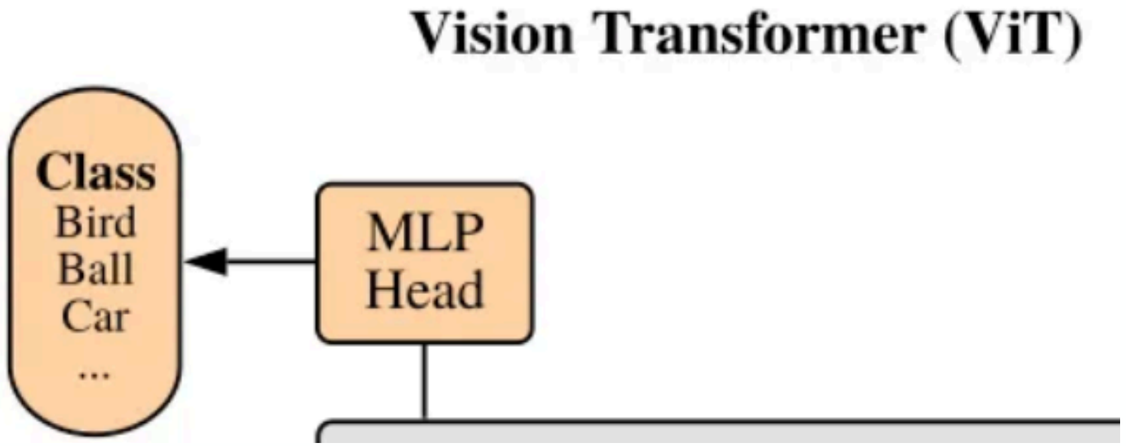


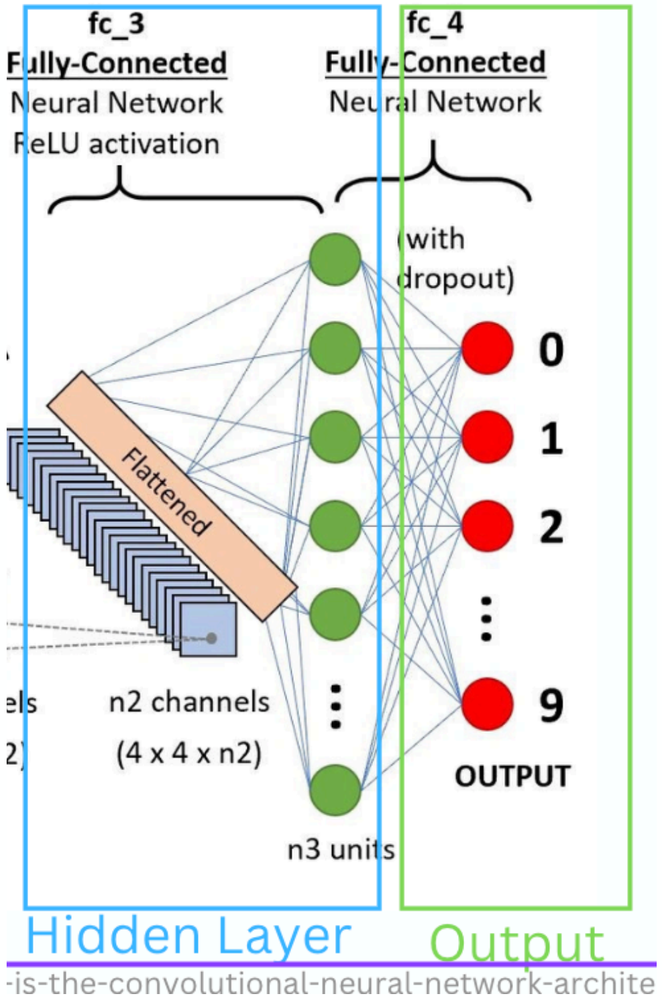



https://medium.com/@RTae/hello-transformer-and-goodbye-cnn-4ebda6e4aa5e

# Classify

SpotOn






https://medium.com/@RTae/hello-transformer-and-goodbye-cnn-4ebda6e4aa5e

# Update the model’s latency when using GPU

# 6s

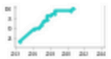
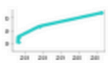
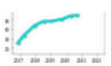
# Re-Identification

# Model selection

## Benchmarks

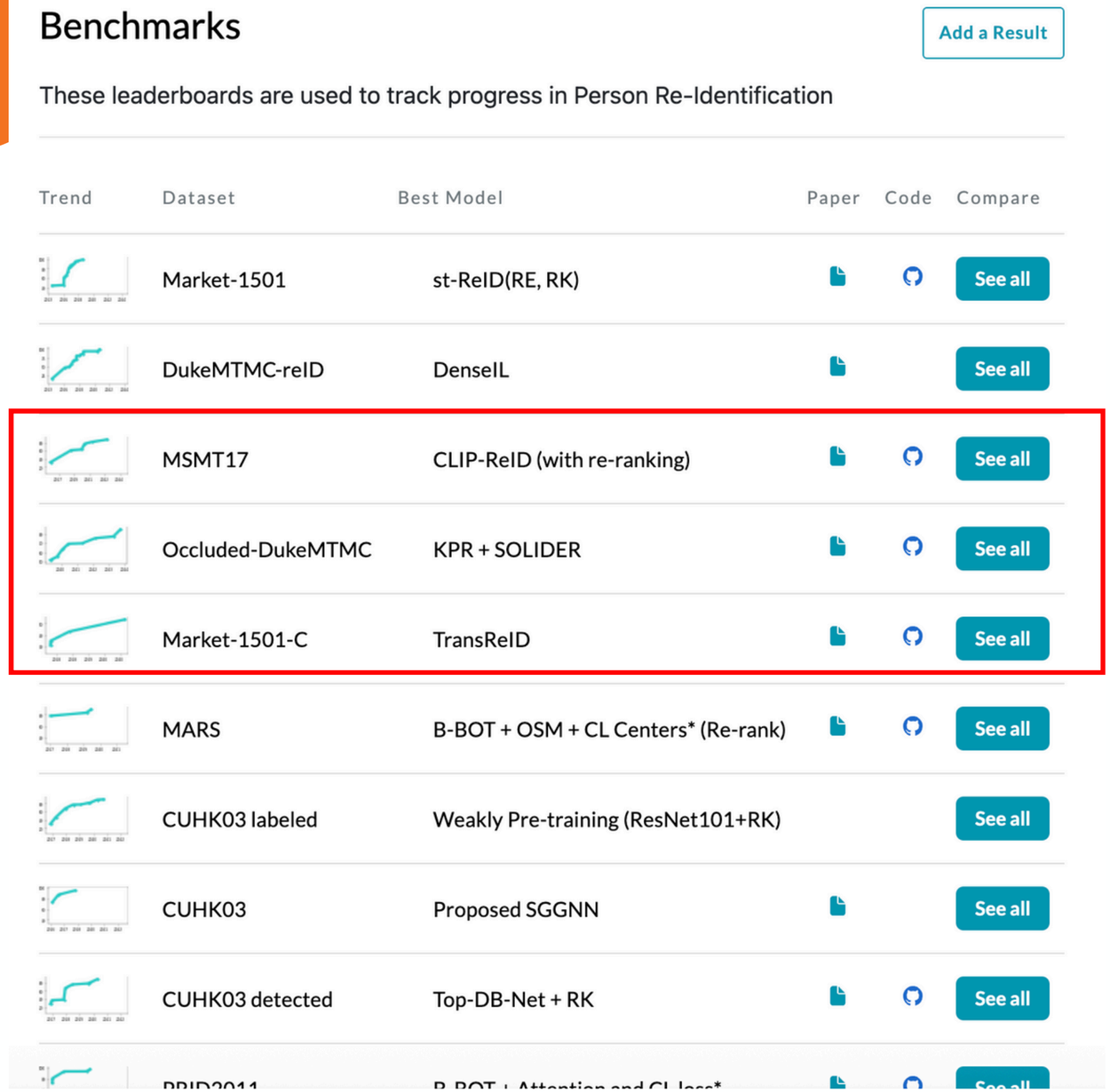
Add a Result

These leaderboards are used to track progress in Person Re-Identification

| Trend | Dataset | Best Model | Paper | Code | Compare |
|---|---|---|---|---|---|
|  | Market-1501 | st-ReID(RE, RK) | | | See all |
|  | DukeMTMC-reID | DenseIL | | | See all |
|  | MSMT17 | CLIP-ReID (with re-ranking) | | | See all |
|  | Occluded-DukeMTMC | KPR + SOLIDER | | | See all |
|  | Market-1501-C | TransReID | | | See all |
|  | MARS | B-BOT + OSM + CL Centers* (Re-rank) | | | See all |
|  | CUHK03 labeled | Weakly Pre-training (ResNet101+RK) | | | See all |
|  | CUHK03 | Proposed SGGNN | | | See all |
|  | CUHK03 detected | Top-DB-Net + RK | | | See all |
|  | PRID2011 | B-BOT + Attention and CL loss* | | | See all |

# Find a benchmarks

https://paperswithcode.com/task/person-re-identification

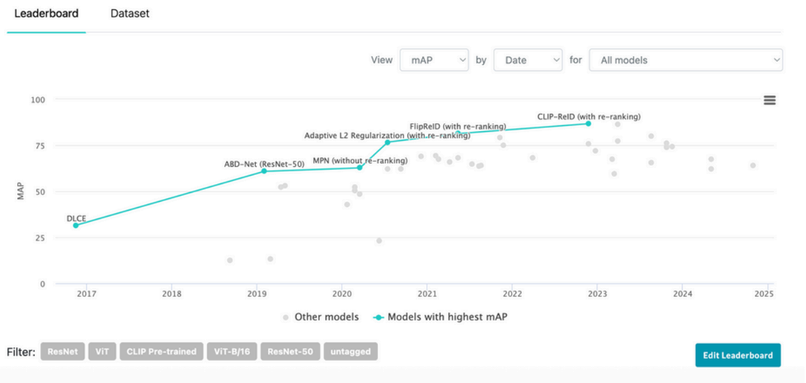



# Filter similar dataset

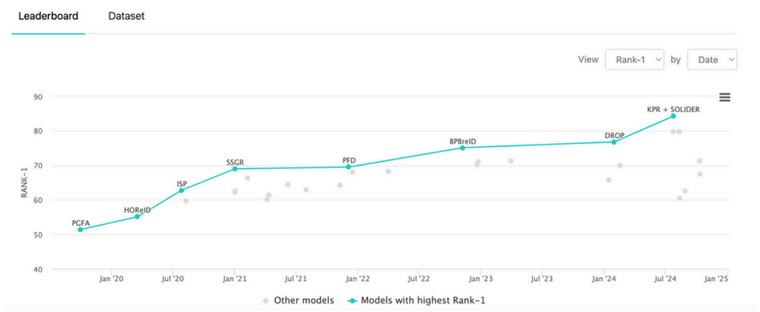
All of them are similar btw.

# Research on interesting models

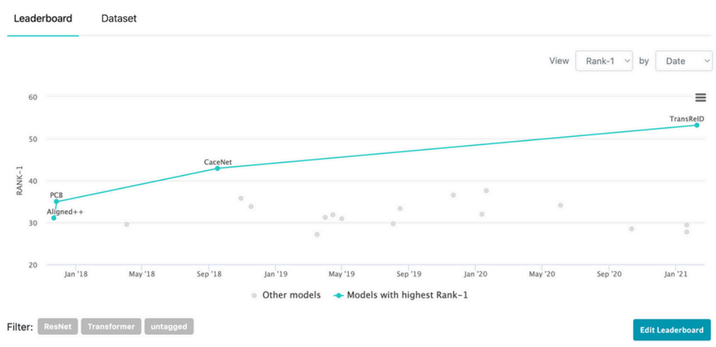
## Person Re-Identification on MSMT17




## Person Re-Identification on Occluded-DukeMTMC




## Person Re-Identification on Market-1501-C



# Compare the models

- General Accuracy: **CLIP-ReID**
- High Speed/Low Latency: **OSNet**
- Occlusion: **KPR + SOLIDER**

- General Accuracy: **CLIP-ReID**
- High Speed/Low Latency: **OSNet**
- Occlusion: **KPR + SOLIDER**
  - **Too much complex**
  - **Too much resource consuming**

# Pick the models

# Model experimental

Comparison Results (Threshold: 0.7)
OSNet: 0.5357 (Different Person)
CLIP: 0.3113 (Different Person)

c1_000352.jpg
Person 3


c3_000000.jpg
Person 2

Comparison Results (Threshold: 0.7)
OSNet: 0.7089 (Same Person)
CLIP: 0.6850 (Different Person)

c1_000352.jpg
Person 3


c3_000000.jpg
Person 1

Comparison Results (Threshold: 0.7)
OSNet: 0.8078 (Same Person)
CLIP: 0.8346 (Same Person)

c1_000217.jpg
Person 1


c1_000352.jpg
Person 1

Comparison Results (Threshold: 0.7)
OSNet: 0.6874 (Different Person)
CLIP: 0.5773 (Different Person)

c1_000217.jpg
Person 1


c1_000352.jpg
Person 2

Comparison Results (Threshold: 0.7)
OSNet: 0.7854 (Same Person)
CLIP: 0.7958 (Same Person)

c1_000217.jpg
Person 2

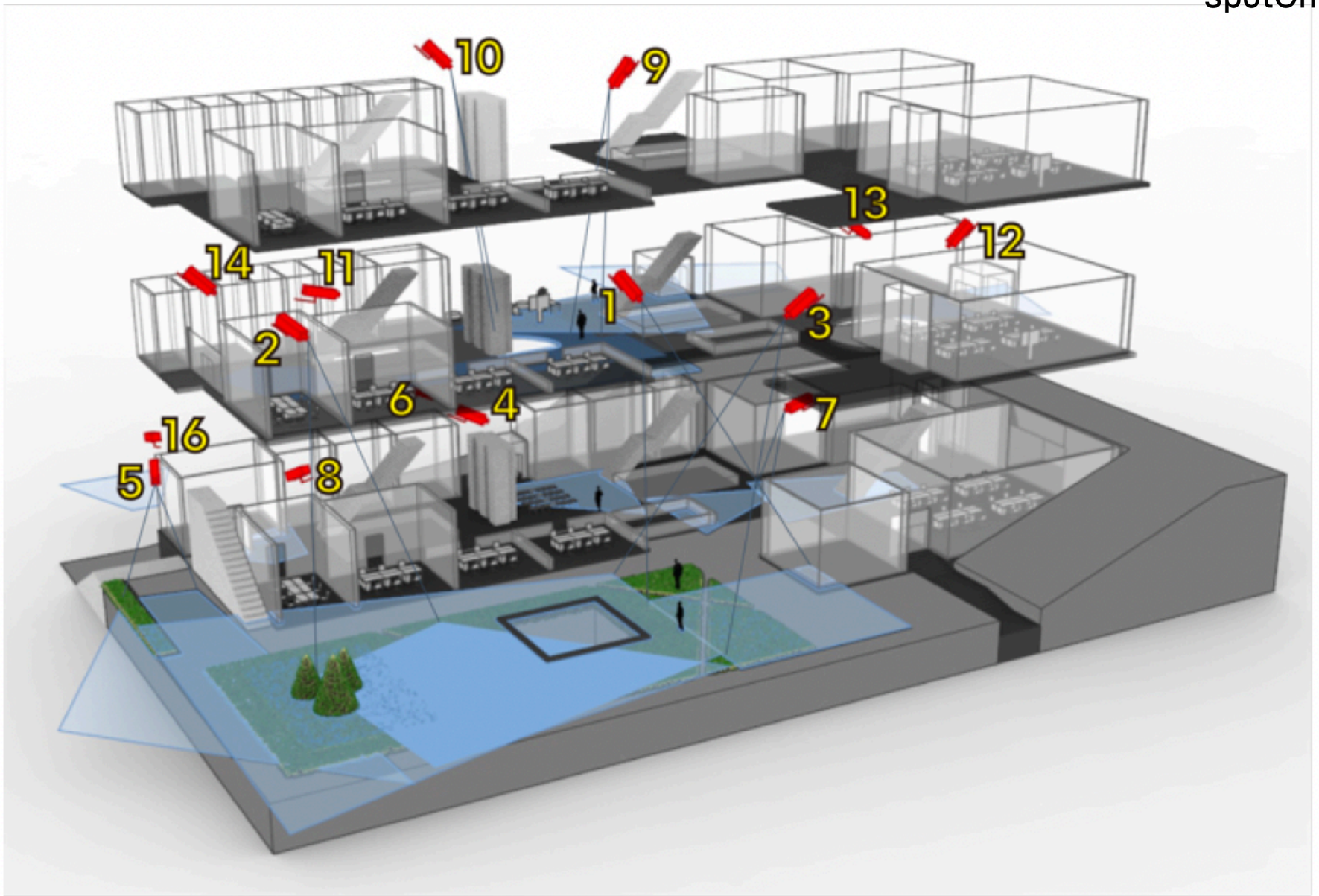

c1_000352.jpg
Person 3

# Camera view selection

# Campus

Selected Cams:

- 1
- 2
- 3
- 5




Cam 1
Cam 2
Cam 3
Cam 5
SpotOn

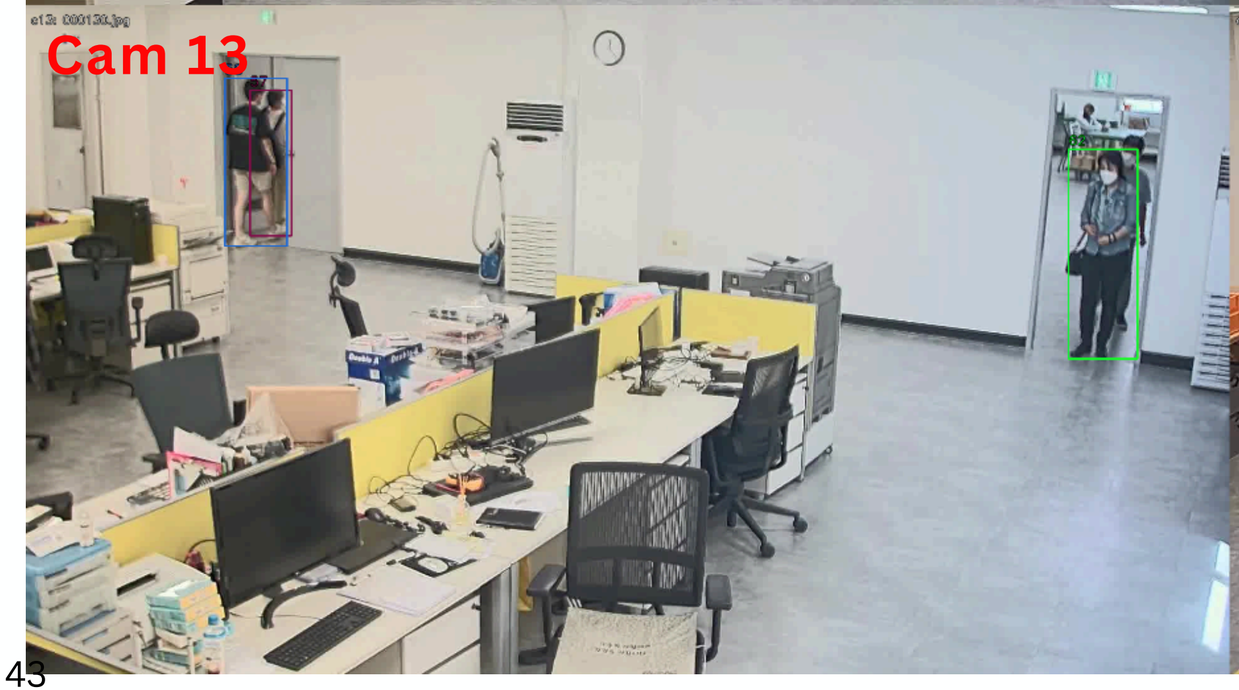
# Factory

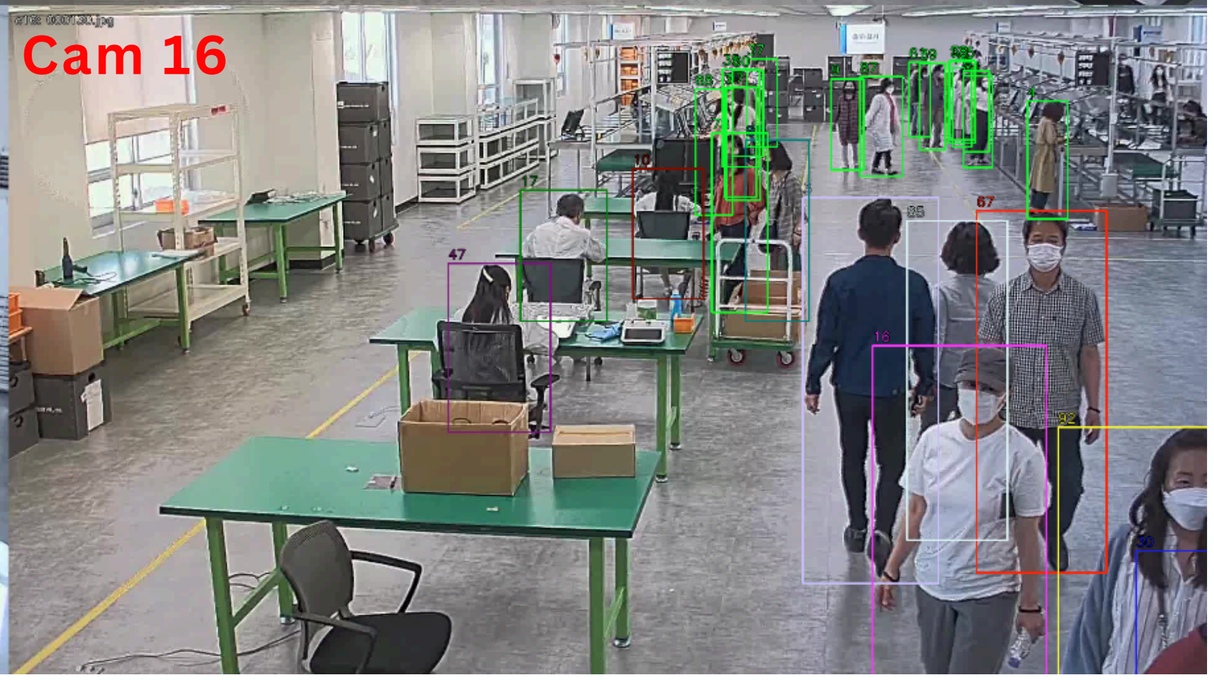
Selected Cams:

- 9
- 12
- 13
- 16



SpotOn



# What's next

- **MLOPs designing**