TP3

DNN for object detection in visible and IR images

Objective

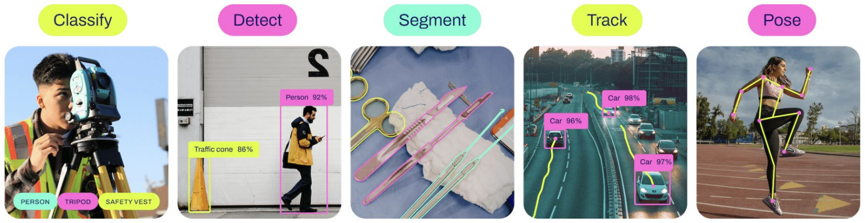
1. Realize an **application for object detection in a video stream using a deep learning network** chosen from the state of the art, with existing methodology.
2. Set up **all stages of the application**
   a. create and prepare the dataset
   b. train the network and manage learning parameters to improve learning performance
   c. deploy the network in an application that processes the video stream from a webcam and calculates the representation of the detected object, which can be used for object tracking.
3. Test two image modalities :
   a. visual data for pedestrian detection
   b. infrared camera for pedestrian detection

To do this, we're going to explore an existing framework based on two resources:

1. Roboflow - dataset creation and annotation tools
2. Ultralytics - a recent implementation of YoloV8, requiring very few lines of code to write and capable of running on the GPU as well as the CPU

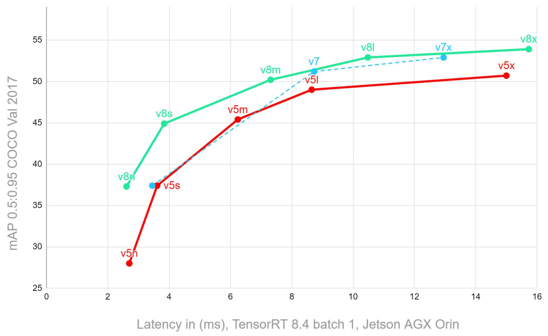
Networks / framework to use

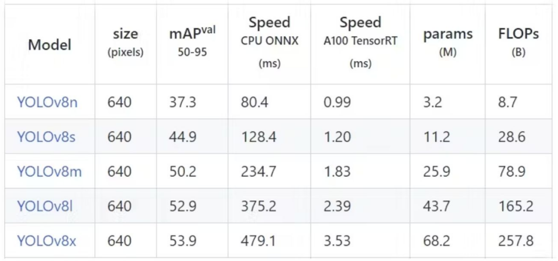
- We're going to use YoloV8 to address all the AI tasks currently being processed.



- For our purposes, this framework requires very little coding and can be run without a GPU (https://docs.ultralytics.com/guides/raspberry-pi/#perform-yolov8-inference).

- For classification, the model is based on a variant of EfficientNet
  - https://encord.com/blog/yolo-object-detection-guide/





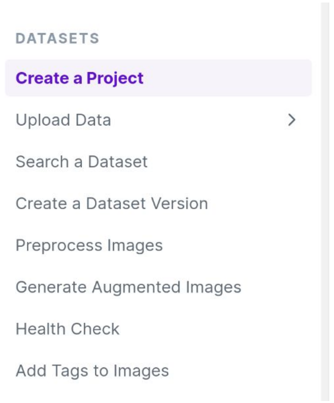
Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

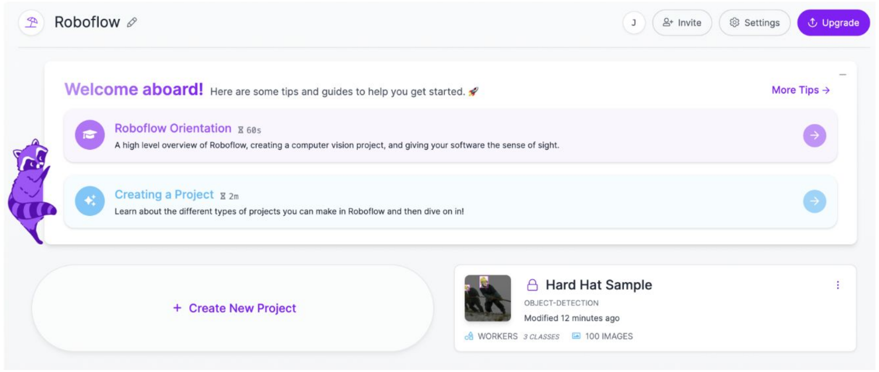
1. Dataset creation and preparation

Create an account at https://roboflow.com/ (free)

and create your own tennis ball detection dataset - just take a dozen photos!

Follow the demo in class!





Welcome aboard! Here are some tips and guides to help you get started.

Roboflow Orientation 60s
A high level overview of Roboflow, creating a computer vision project, and giving your software the sense of sight.

Creating a Project 2m
Learn about the different types of projects you can make in Roboflow and then dive on in!

Hard Hat Sample
OBJECT-DETECTION
Modified 12 minutes ago

Once you've created your dataset **(preprocessing + data augmentation),** download the database or use it from google colab.

2. Train your chosen network

- To go faster, I suggest you follow Google Colab Notebook Train YOLOv8 available on Blackboard
    - https://colab.research.google.com/drive/1CZKRto_si_3IrdOZ5nai-fVQjIvigeLP?usp=sharing
- All the parameters available for optimizing training can be found on the following web page
    - https://docs.ultralytics.com/modes/train/

    Please note that the metric used to evaluate learning performance is **mAP** -> it is based on the measure of coverage of the surface of the detected object.

    https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173

    https://www.v7labs.com/blog/mean-average-precision

Object detection

Follow the steps in the Google Colab script

We recommend that you first test a model on a pre-recorded sequence for visual images, for IR frames - take some supplementary images from the dataset

Notebook example for detection

https://colab.research.google.com/drive/1tfZQvLF3otc8yTUSIjx6M17EUqr3vo95?usp=sharing

Elements to be included in the report :

1. Application specification
2. Describe the DNN you've chosen, giving reasons for your choice (don't hesitate to do research on the Internet, and cite the advantages and disadvantages of your choice).
3. Describe and illustrate your realization/implementation (dataset, learning - the parameters selected, why)
   a. quote the metrics used for the evaluation
   b. make a note of the parameters of your experiments
4. Describe and illustrate the results: learning curves, results on the validation part, some illustrations of the results on the images obtained.
5. Test execution on PC without and with GPU (can be done in the Google Colab environment), what can we deduce?
6. Don't forget the conclusion ;-)