Lab4: Object detection (YOLO)

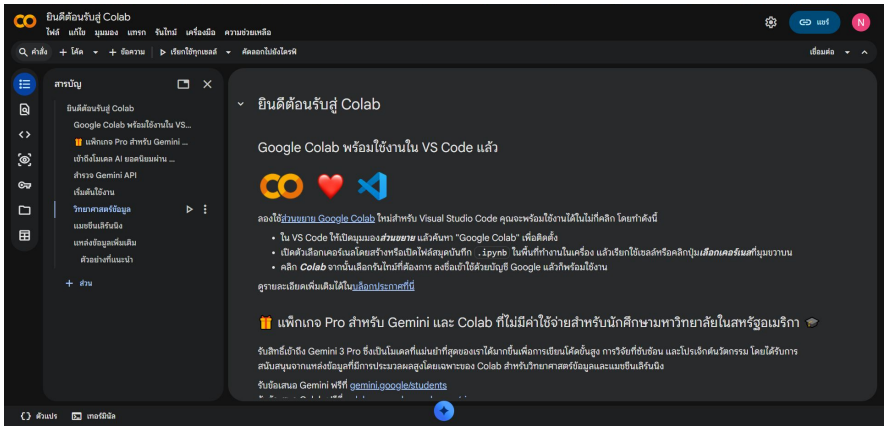
3099704 AI for Digital Health (2025/2)

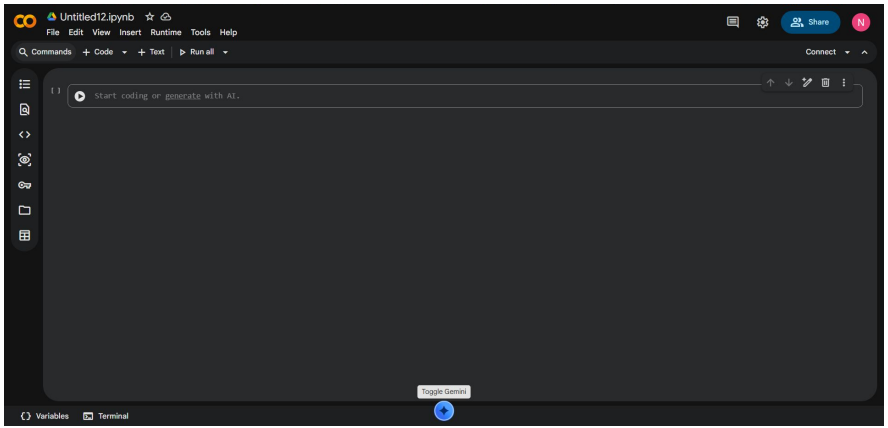
Objective

- **Create & train** object detection model from ultralytics hub and ultralytics library
- **Inference & evaluate** performent each model

Material

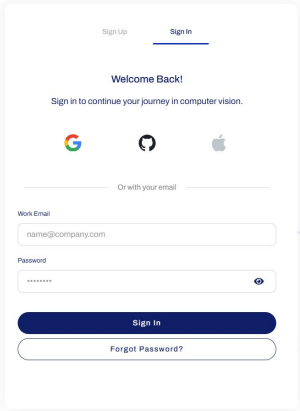
- With **Google Colab**, you don't need to install any software. All you need is a Google account, and you can start using it right away. Simply visit: https://colab.research.google.com/ or select NEW NOTEBOOK to start a new file.

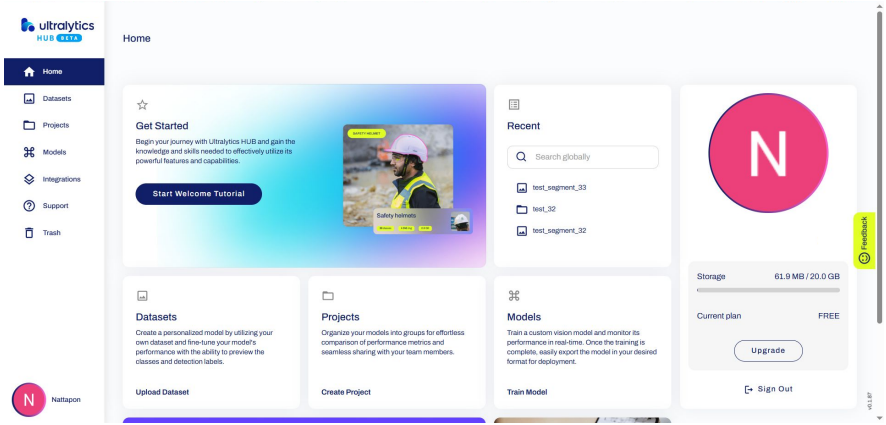




Material

- **Ultralytics HUB** is a web-based platform that can be accessed via a browser at https://hub.ultralytics.com/, enabling users to upload datasets, train YOLO models, and monitor performance without any local software installation.



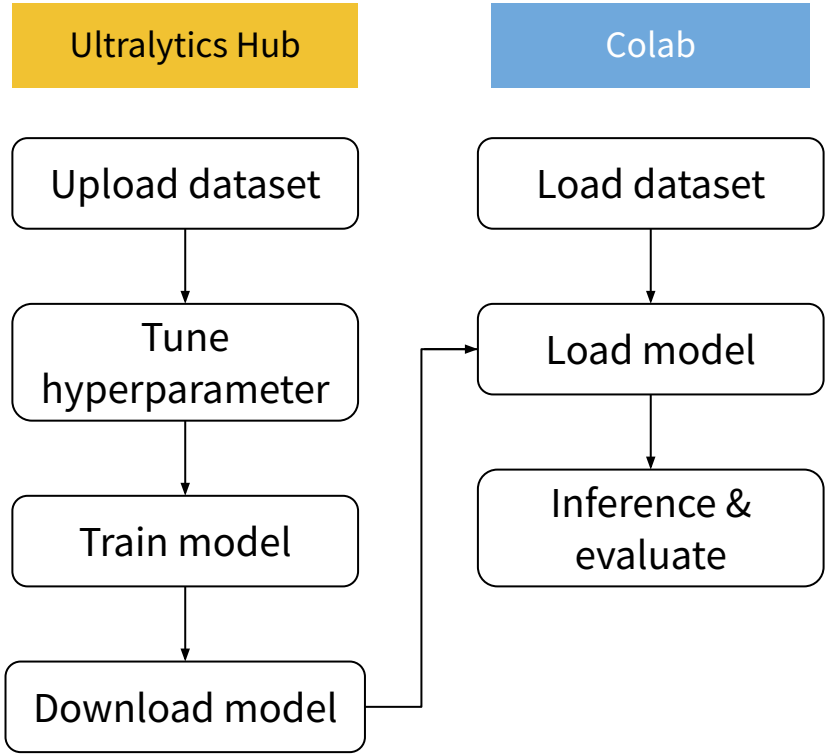


Material

- **Ultralytics is a Python library for computer vision tasks**, best known for developing and training **YOLO (You Only Look Once)** models.
- It supports common tasks such as **image classification, object detection, and image segmentation**.
- Ultralytics models can be trained and used **on a local computer or on Google Colab**.
- Install the library using **pip**

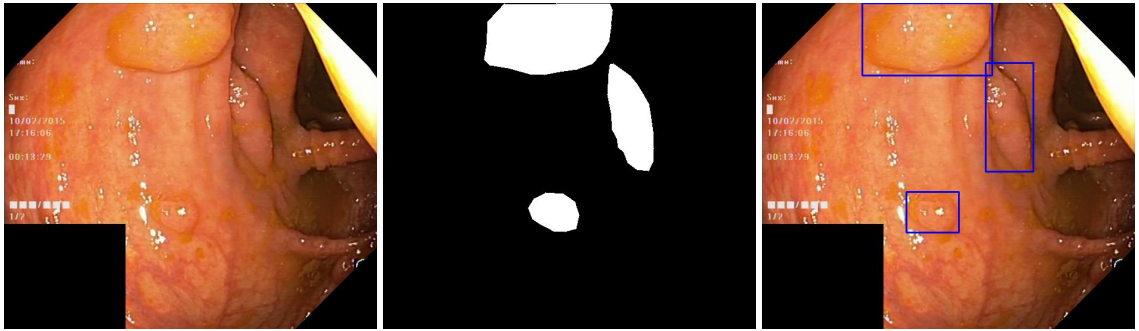
Lab4.1: YOLOv8n (Ultralytics Hub)

In this lab, you will create an image detection model using **Ultralytics HUB** and then evaluate its performance using a **Google Colab notebook.**



Dataset: kvasir dataset

- **Kvasir dataset** consists of images, annotated and verified by medical doctors (experienced endoscopists), including several classes showing anatomical landmarks, phatological findings or endoscopic procedures in the GI tract.
- The dataset consist of the images with different resolution from 720x576 up to 1920x1072 pixels in JPG format and documents in JSON format
- To simplify the experiment, we selected only 500 images containing polyps and prepared the dataset in a format compatible with YOLO training.

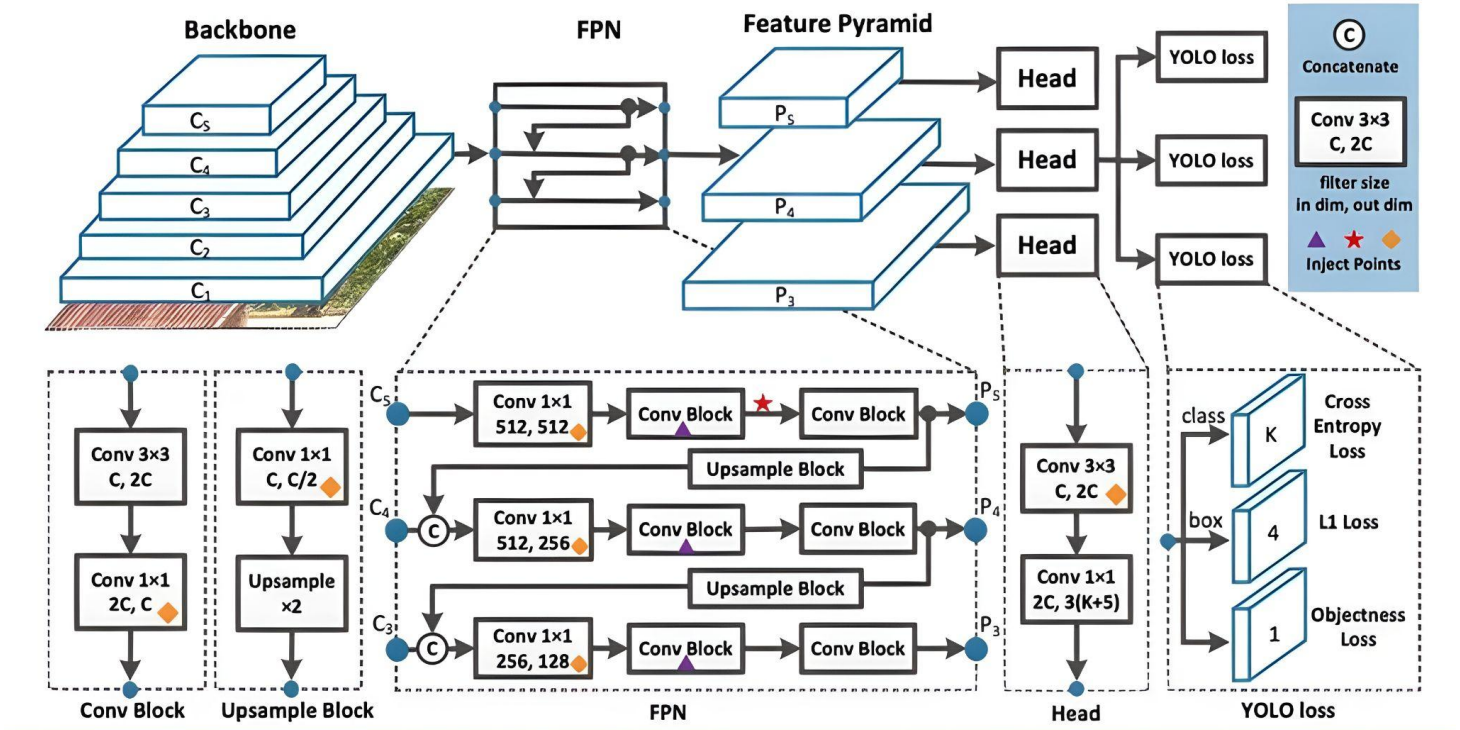


YOLOv8

In this lab, we chose to use YOLOv8, which has the following key architectural innovations:

- **Backbone**: Uses CSPNet with improved **C2f modules** for efficient feature extraction and richer representations.
- **Neck**: Combines **FPN + PAN** to fuse multi-scale features, enabling detection of both small and large objects.
- **Head**: Anchor-free and **decoupled** design, separating classification and localization for higher accuracy.
- **Loss Functions**: Employs task-specific losses (e.g., BCE for classification, DFL for bounding box regression).

YOLOv8


Backbone
FPN
Feature Pyramid
Head
Head
Head
YOLO loss
YOLO loss
YOLO loss
C
Concatenate
Conv 3×3
C, 2C
filter size
in dim, out dim
Inject Points
Conv 3×3
C, 2C
Conv 1×1
C, C/2
Conv 1×1
512, 512
Conv Block
Conv Block
Upsample Block
Conv 1×1
512, 256
Conv Block
Conv Block
Upsample Block
Conv 1×1
256, 128
Conv Block
Conv Block
Conv 1×1
2C, C
Upsample
×2
Conv 3×3
C, 2C
Conv 1×1
2C, 3(K+5)
class
K
Cross
Entropy
Loss
box
4
L1 Loss
1
Objectness
Loss
Conv Block
Upsample Block
FPN
Head
YOLO loss

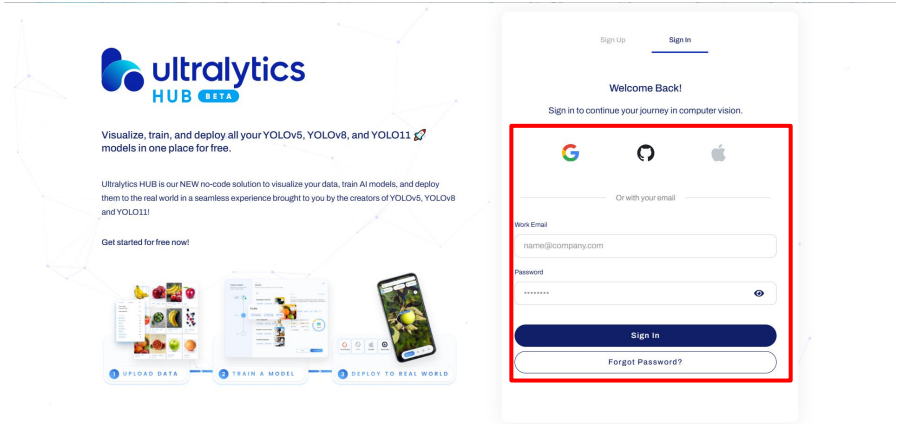
YOLOv8

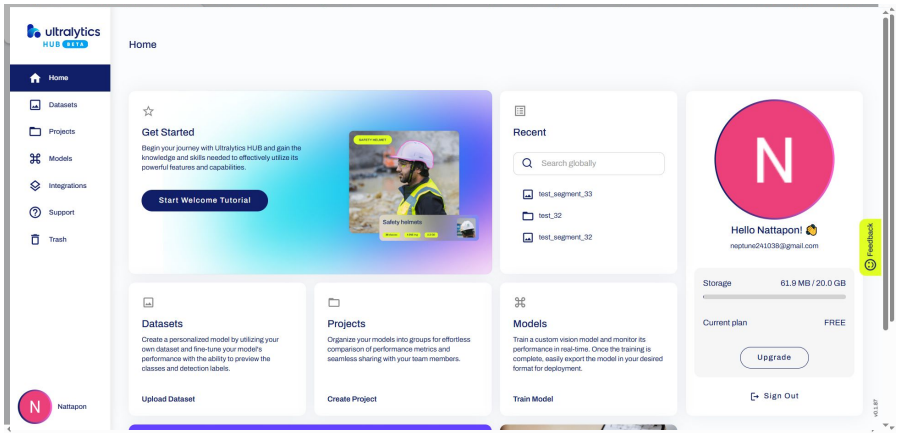
YOLOv8 is available in multiple model sizes: n (nano), s (small), m (medium), l (large), and x (extra-large). These models share the same overall architecture but differ in the number of channels (width) and repeated blocks (depth). Due to limitations, we use the nano version, which has the lowest computational cost.

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

Lab4.1: YOLOv8n (Ultralytics Hub)

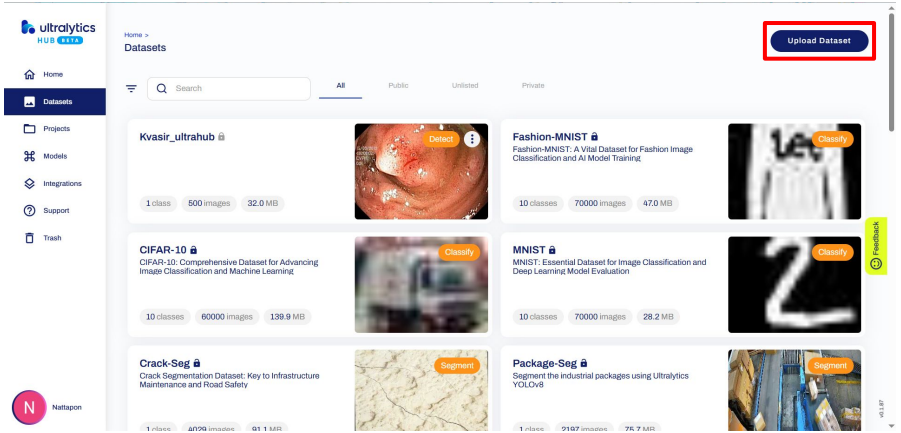
1) Sign in [Ultralytic hub]

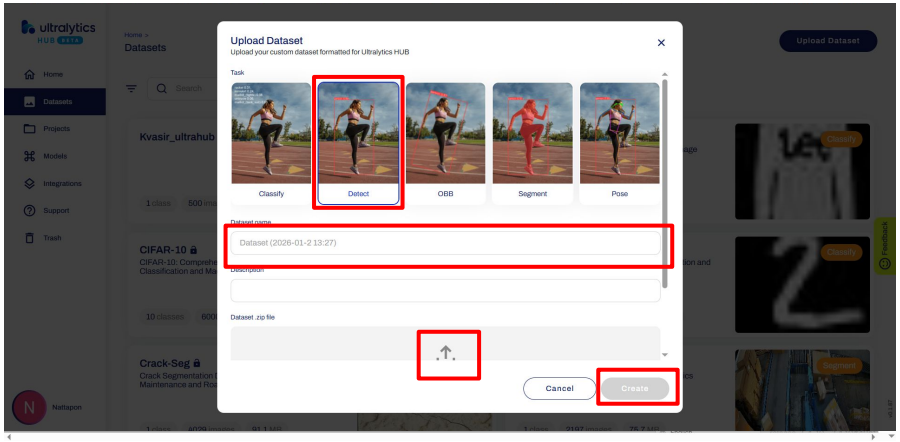




Lab4.1: YOLOv8n (Ultralytics Hub)

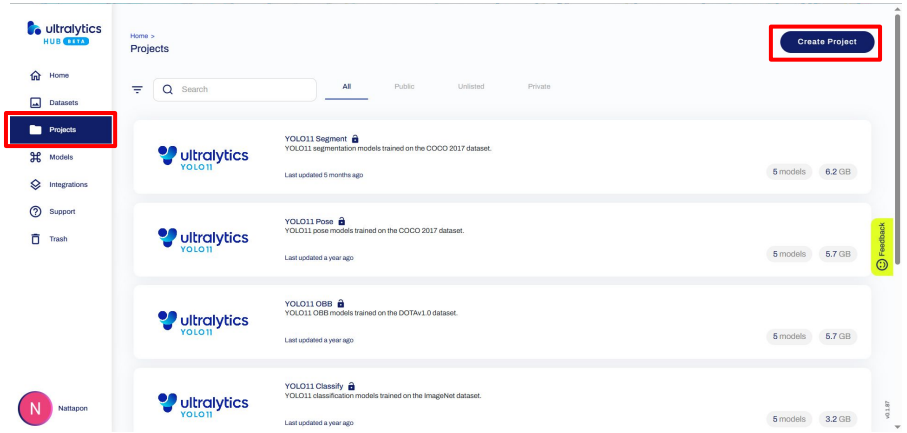
2) Download the dataset from GitHub, then upload in **Ultralytic hub**.

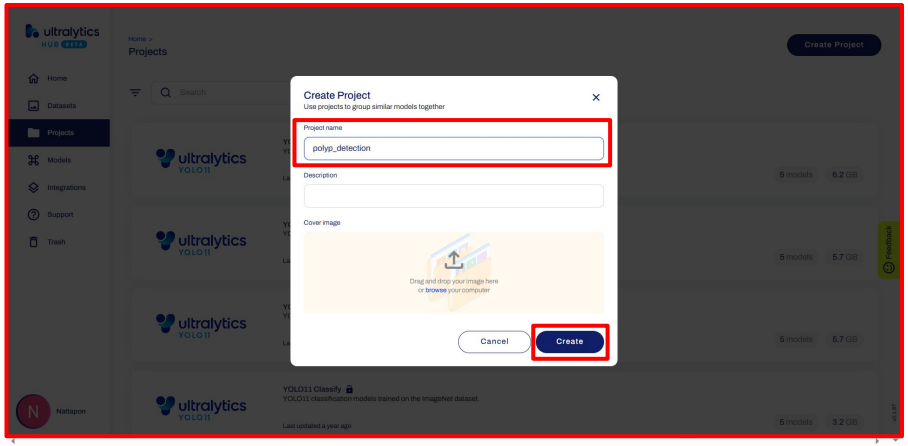




Lab4.1: YOLOv8n (Ultralytics Hub)

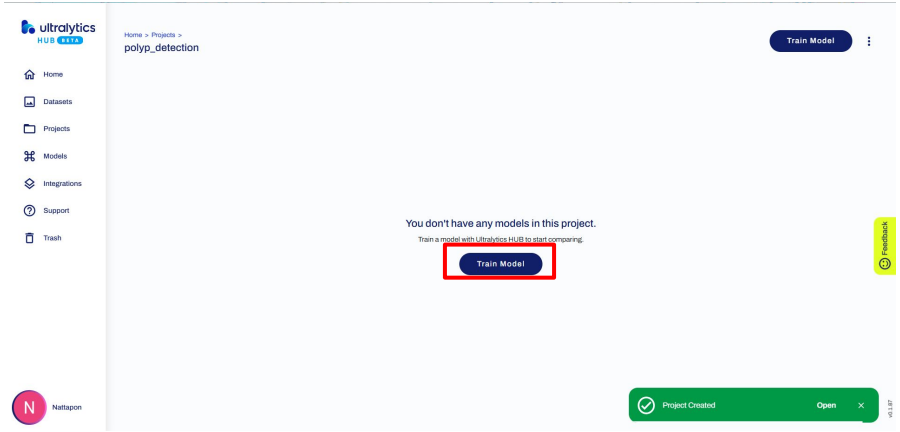
3) Create new project

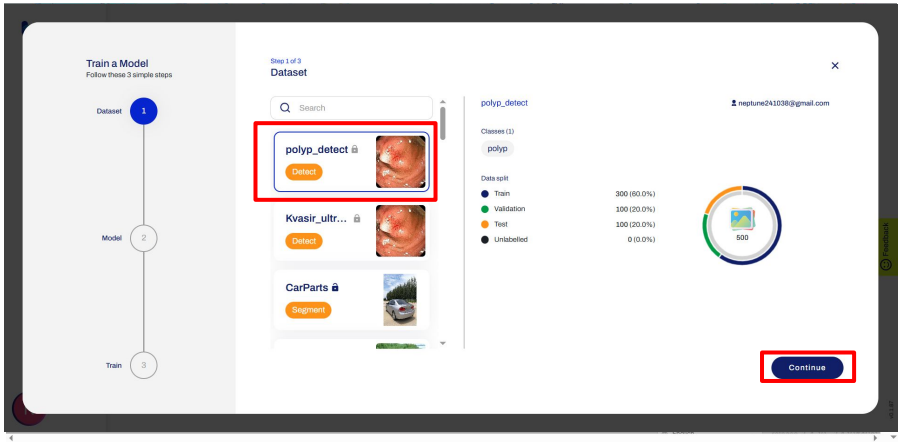




Lab4.1: YOLOv8n (Ultralytics Hub)

4) Train Model (choose dataset)

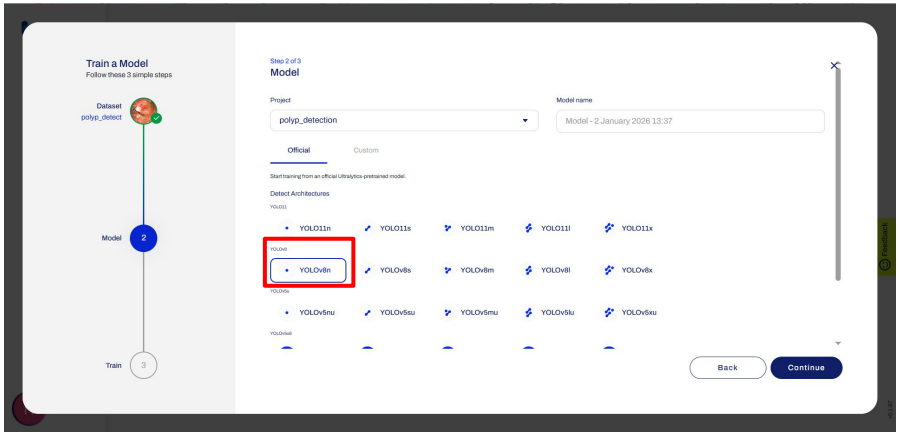


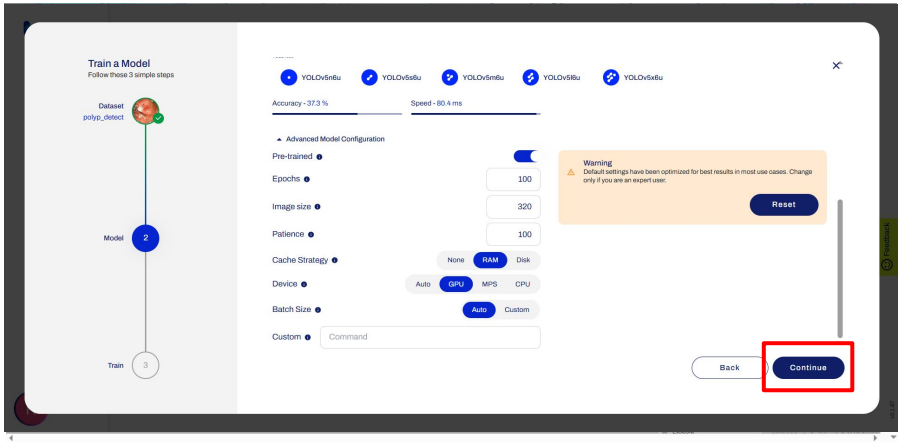


Lab4.1: YOLOv8n (Ultralytics Hub)

4) Train Model (set configuation)

- model: YOLOv8n, Epochs:100, Image size: 320, Cache: RAM, Device: GPU

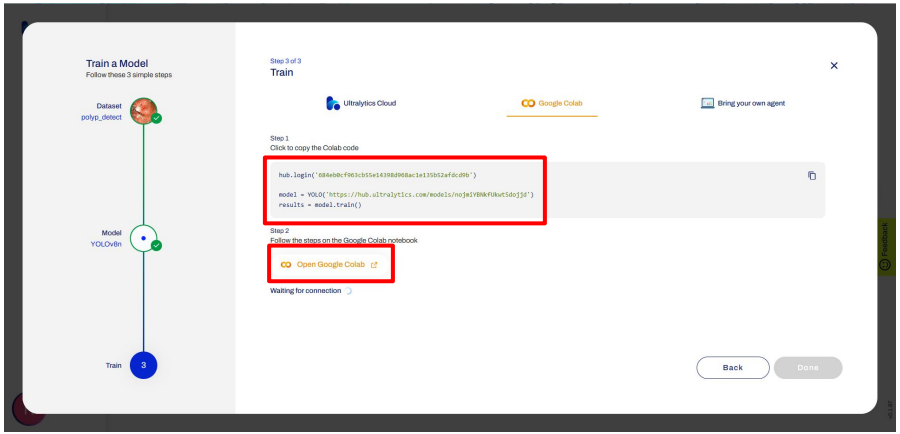


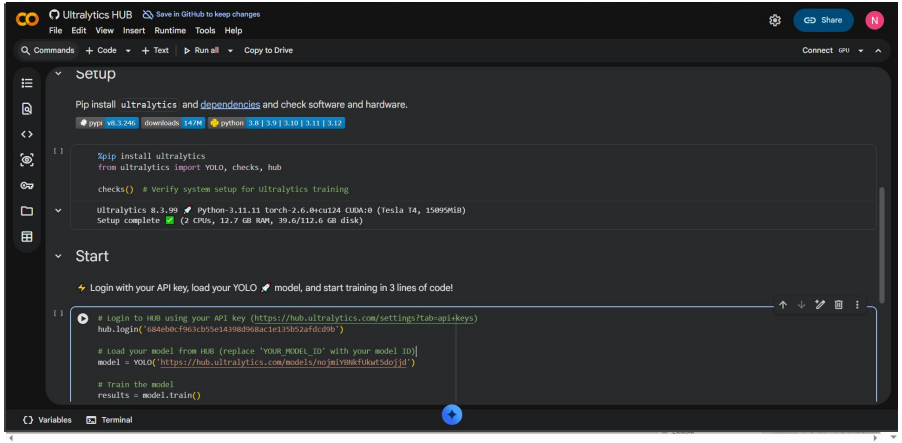


Lab4.1: YOLOv8n (Ultralytics Hub)

4) Train Model (training)

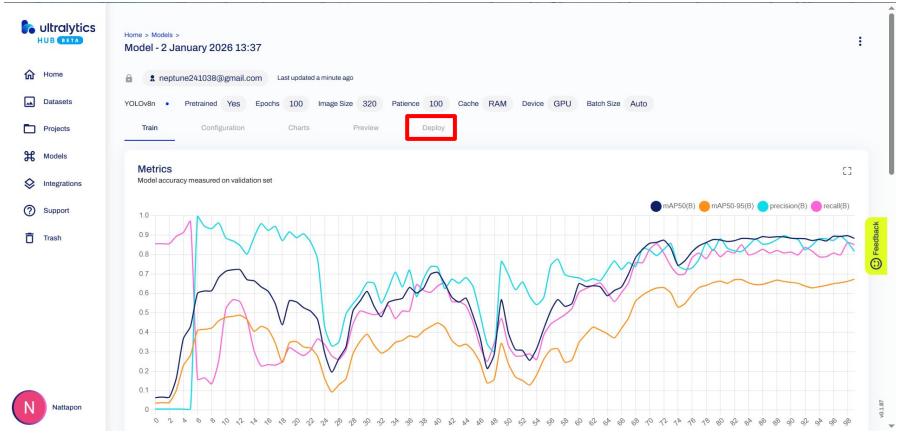
- Copy code and run in colab

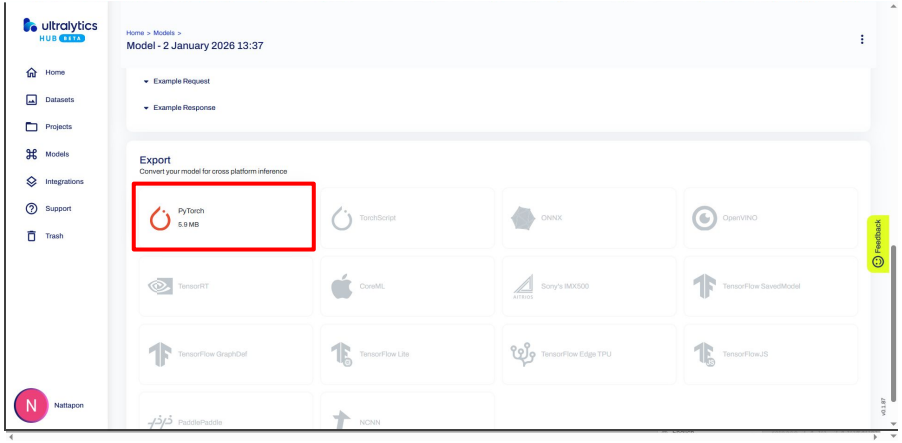




Lab4.1: YOLOv8n (Ultralytics Hub)

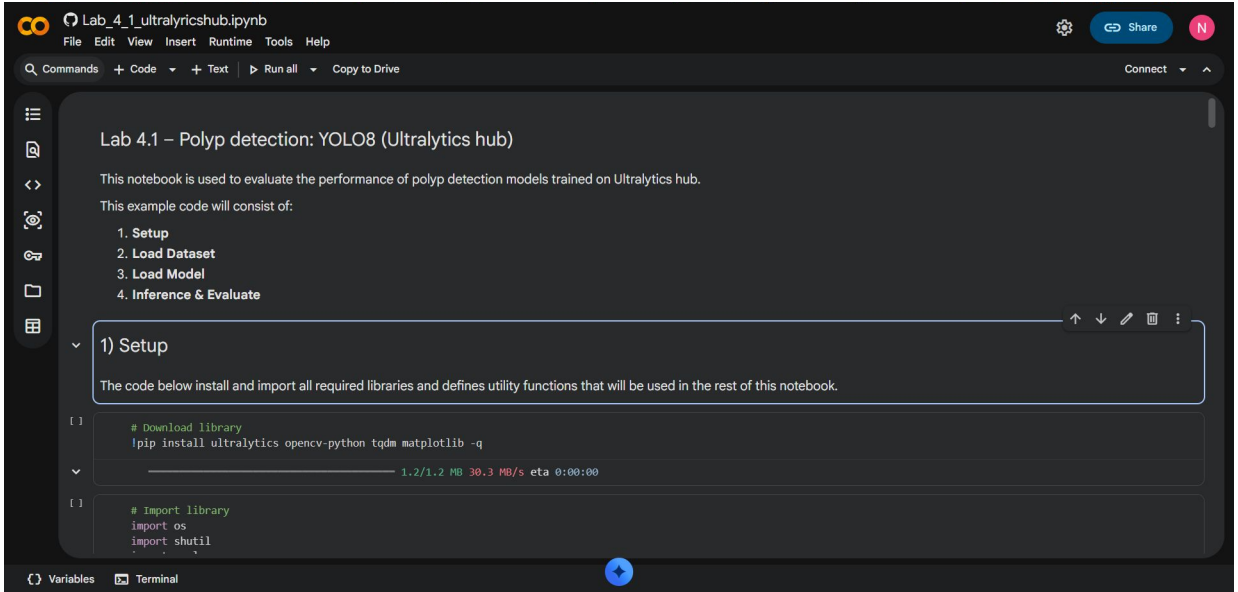
5) Export model (Pytorch version)





Lab4.1: YOLOv8n (Ultralytics Hub)

5) Open Lab_4_1_ultralyricshub in Colab, upload the pt. file exported from Ultralytics Hub, and then run the code in the notebook to evaluate the model's performance.



Lab4.1: YOLOv8n (Ultralytics Hub)

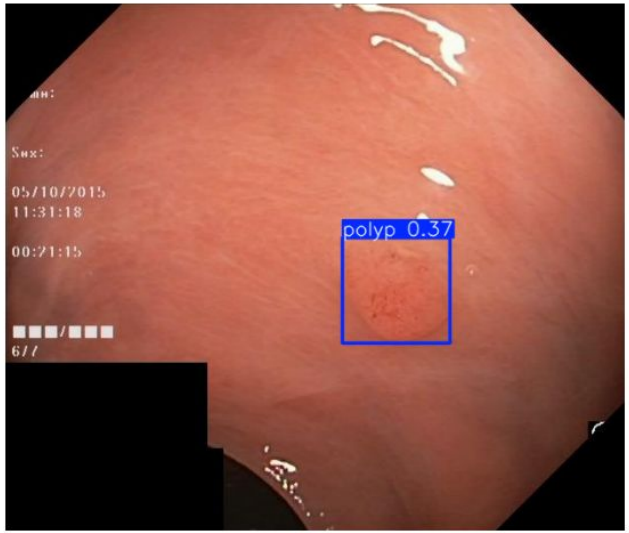
Results may vary between runs due to random seed initialization and hyperparameter tuning; however, the overall performance should be similar to the results shown on this page.

```
Testing 100 test images...
Evaluating YOLOv8 Detection: 100%|██████████| 100/100 [00:06<00:00, 14.81it/s]
 YOLOv8 Detection Performance:
• Mean IoU: 0.8022
• Avg inference time: 64.15 ms/img
```

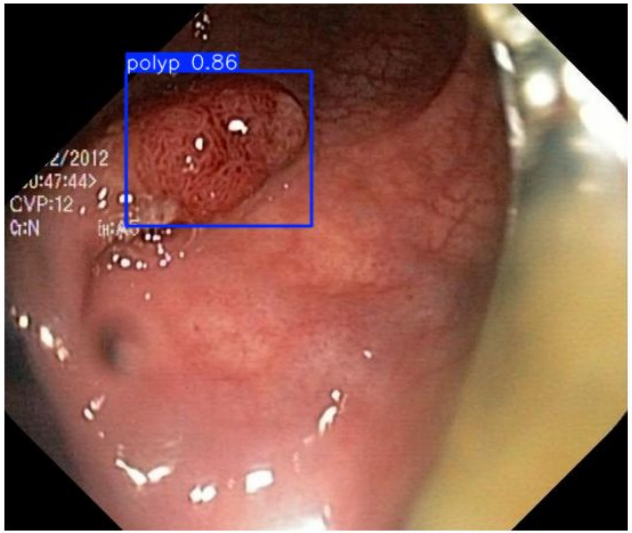
Lab4.1: YOLOv8n (Ultralytics Hub)

Results may vary between runs due to random seed initialization and hyperparameter tuning; however, the overall performance should be similar to the results shown on this page.

Prediction 1

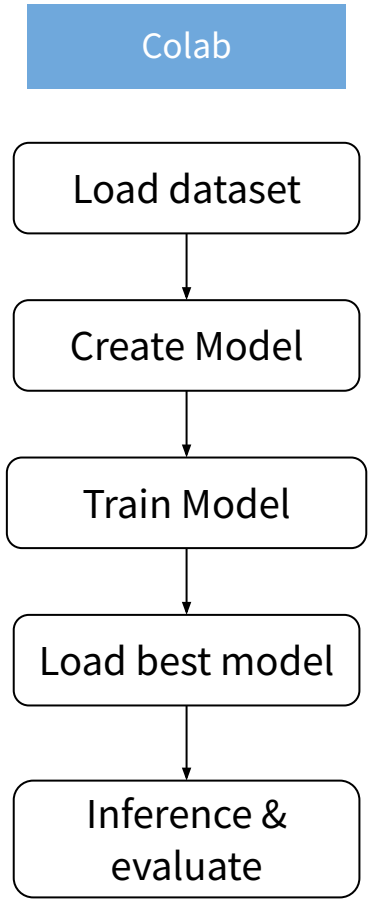


Prediction 2



Lab4.2: YOLOv8n (Ultralytics)

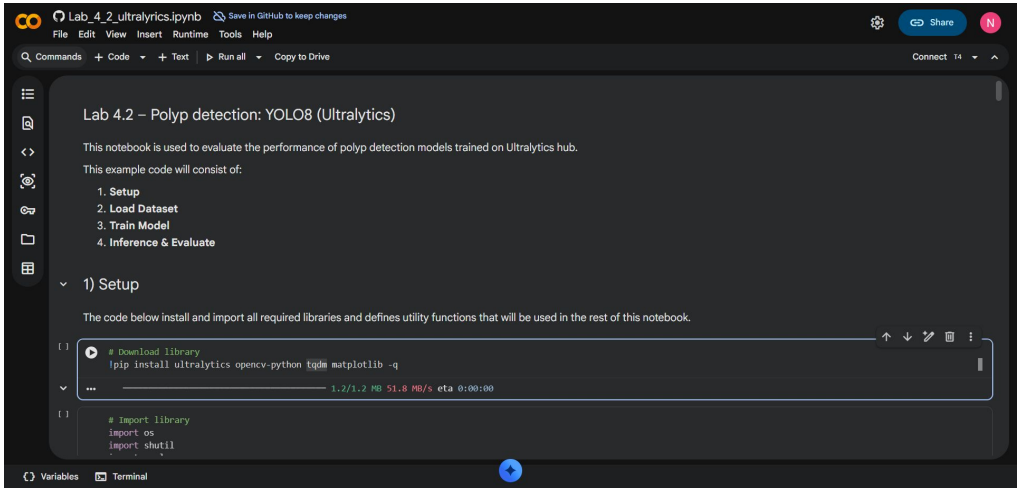
In this lab, you will create an image detection model and evaluate its performance using **Ultralytics library** in colab



Lab4.2: YOLOv8n (Ultralytics)

Run Lab_4_2_ultralyrics.ipynb (in colab)

1) Setup
2) Load Data
3) Train model
4) Inference & Evaluate



Lab4.2: YOLOv8n (Ultralytics)

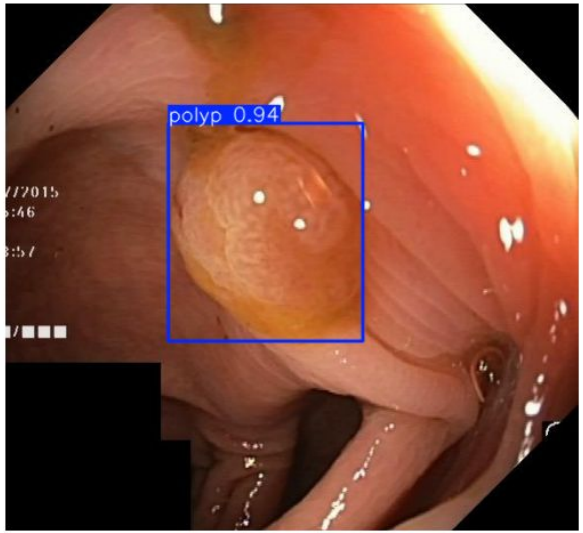
Results may vary between runs due to random seed initialization and hyperparameter tuning; however, the overall performance should be similar to the results shown on this page.

```
Testing 100 test images...
Evaluating YOLOv8 Detection: 100%|██████████| 100/100 [00:01<00:00, 84.49it/s]
 YOLOv8 Detection Performance:
• Mean IoU: 0.8439
• Avg inference time: 9.78 ms/img
```

Lab4.2: YOLOv8n (Ultralytics)

Results may vary between runs due to random seed initialization and hyperparameter tuning; however, the overall performance should be similar to the results shown on this page.

Prediction 1



Prediction 2