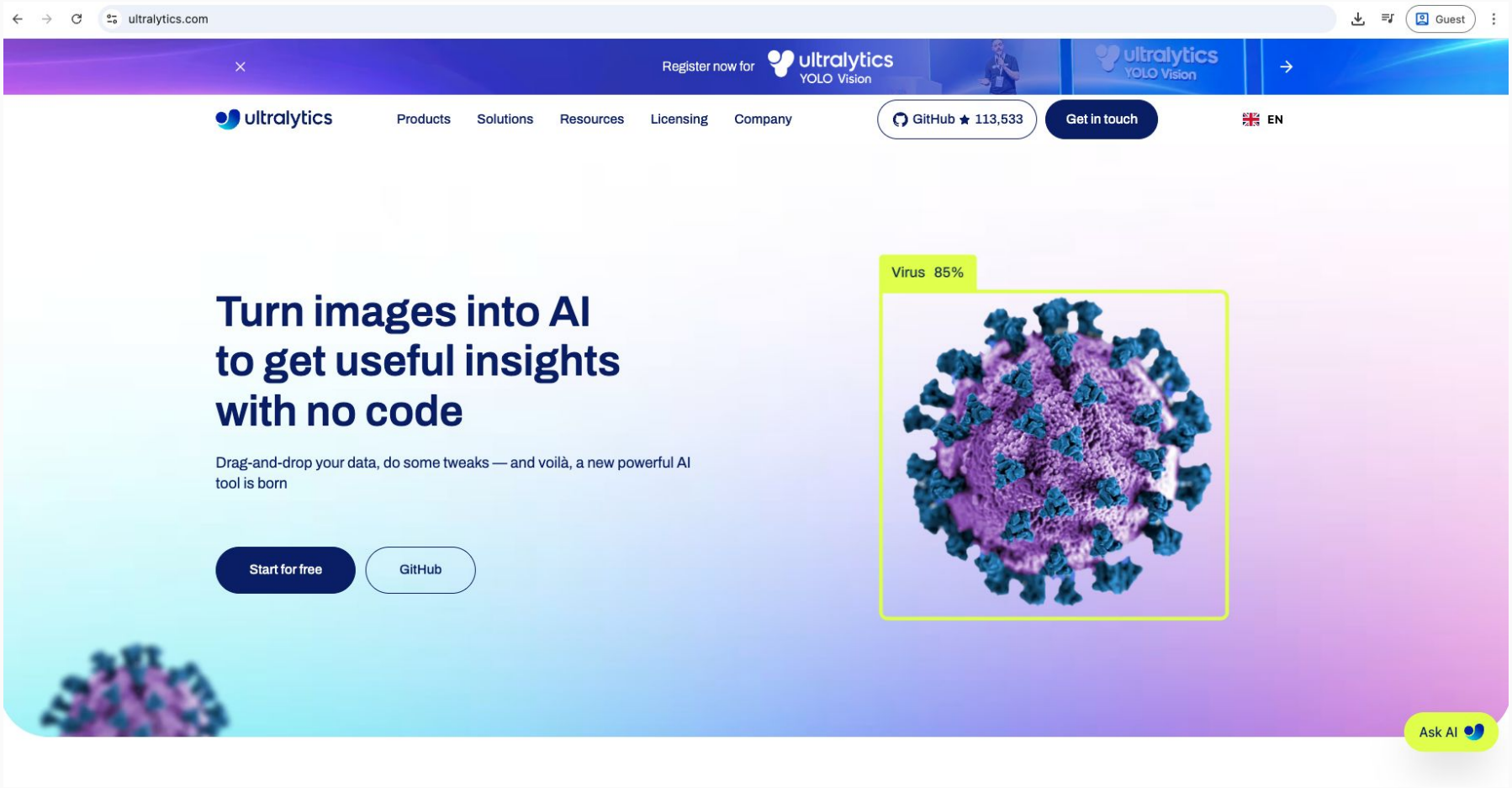
Lecture 10: Ultralytics Vision AI

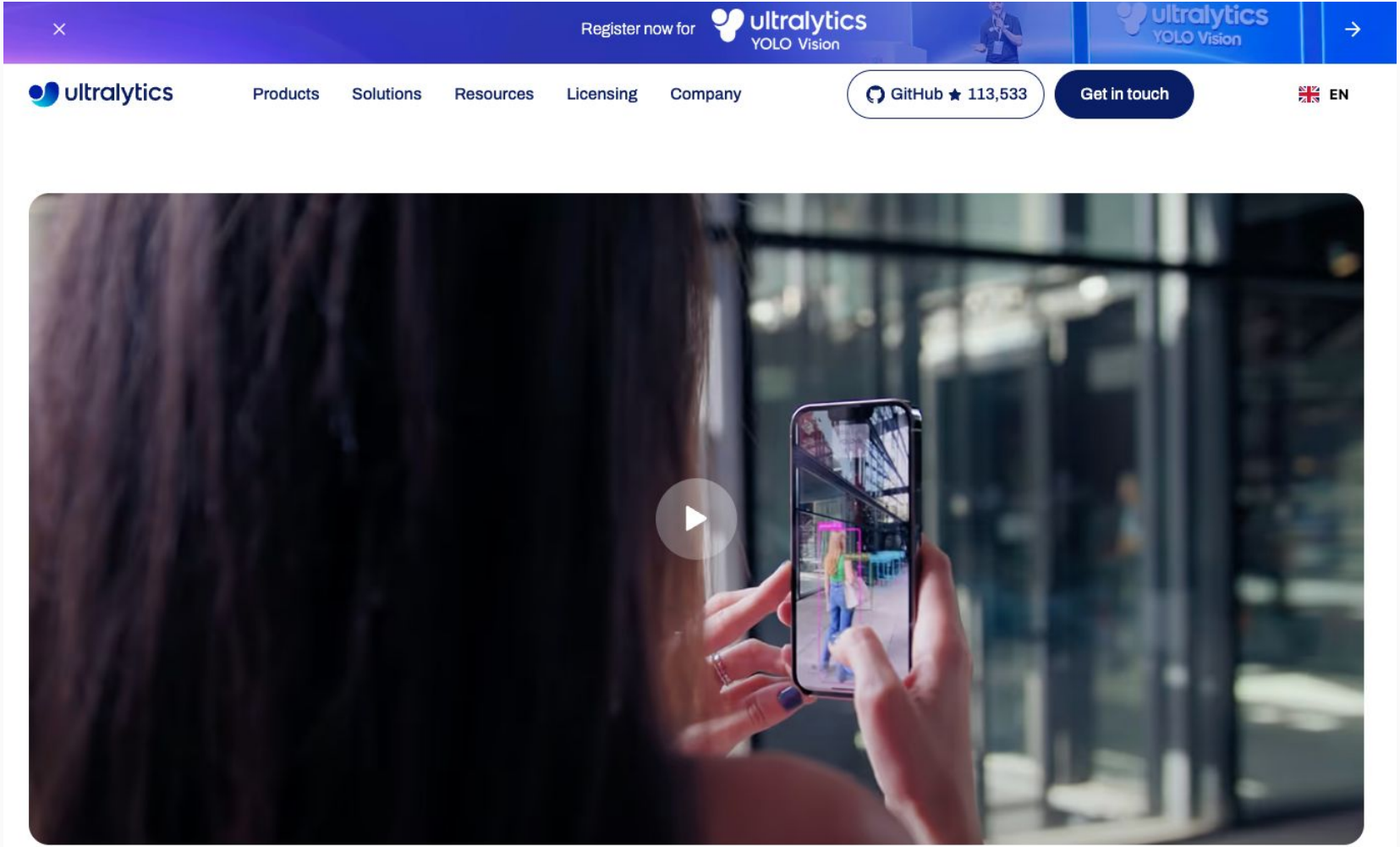
https://github.com/kaopanboonyuen/SC310005_ArtificialIntelligence_2025s1

Teerapong Panboonyuen
https://kaopanboonyuen.github.io

References

- https://www.ultralytics.com/
- https://labelstud.io/
- https://roboflow.com/
- https://cs231n.stanford.edu/index.html
- https://encord.com/blog/object-detection/
- https://neptune.ai/blog/object-detection-algorithms-and-libraries
- https://learnopencv.com/non-maximum-suppression-theory-and-implementation-in-pytorch/


ultralytics.com
Guest
Register now for ultralytics YOLO Vision
ultralytics
Products
Solutions
Resources
Licensing
Company
GitHub ★ 113,533
Get in touch
EN
Turn images into AI to get useful insights with no code
Drag-and-drop your data, do some tweaks — and voilà, a new powerful AI tool is born
Start for free
GitHub
Virus 85%
Ask AI


Register now for ultralytics YOLO Vision
ultralytics
Products
Solutions
Resources
Licensing
Company
GitHub ★ 113,533
Get in touch
EN

Register now for ultralytics YOLO Vision

ultralytics

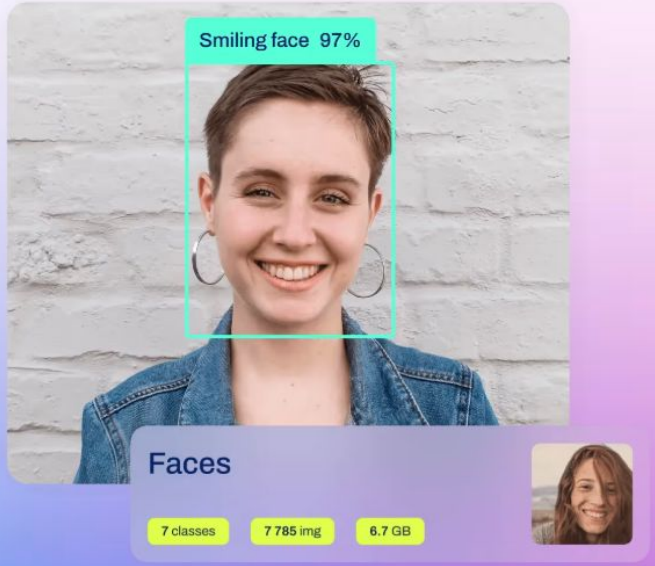
Products Solutions Resources Licensing Company

GitHub ★ 113,533 Get in touch EN

Train computer vision models in seconds with Ultralytics YOLO

Explore our state-of-the-art AI architecture to train and deploy your highly-accurate AI YOLO models like a pro

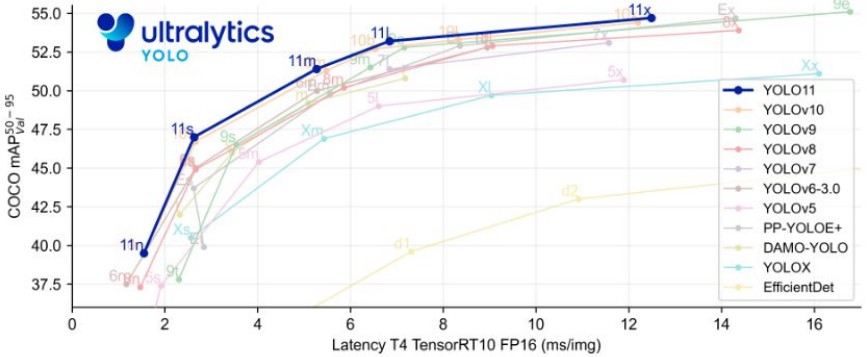
Get started GitHub



Ultralytics YOLO11

Overview

YOLO11 is the latest iteration in the Ultralytics YOLO series of real-time object detectors, redefining what's possible with cutting-edge accuracy, speed, and efficiency. Building upon the impressive advancements of previous YOLO versions, YOLO11 introduces significant improvements in architecture and training methods, making it a versatile choice for a wide range of computer vision tasks.



Introduction to Computer Vision

- **Definition**: Computer Vision enables machines to “see” and understand the visual world from images and videos.
- **Goal**: Extract meaningful information for analysis, decision-making, and automation.
- **Applications**: Healthcare, autonomous vehicles, wildlife monitoring, manufacturing, security, AR/VR.

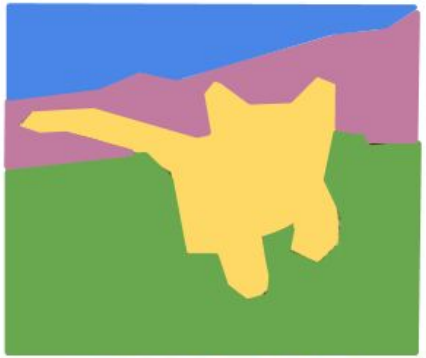
Types of Computer Vision Tasks

- **Image Classification**: Assign a label to an entire image (e.g., cat vs dog).
- **Object Detection**: Locate and classify objects within an image with bounding boxes.
- **Segmentation**:
  - **Semantic Segmentation**: Classify each pixel into a class.
  - **Instance Segmentation**: Detect objects and separate each instance.
- **Pose Estimation**: Detect keypoints on humans, animals, or objects to understand posture or movement.
- **Modern/Advanced AI Vision**: Generative AI (e.g., DALL·E, MidJourney, Text-to-Image) – will cover in advanced lectures.

https://cs231n.stanford.edu/index.html

Other Computer Vision Tasks

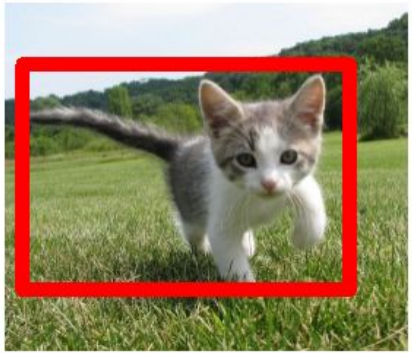
Semantic Segmentation



GRASS, CAT, TREE, SKY

No objects, just pixels

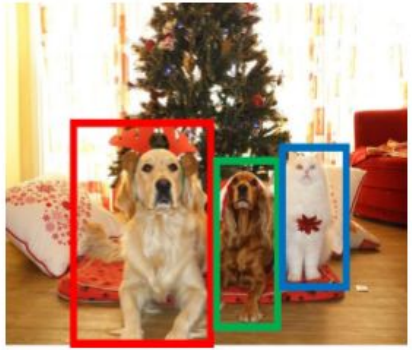
Classification + Localization



CAT

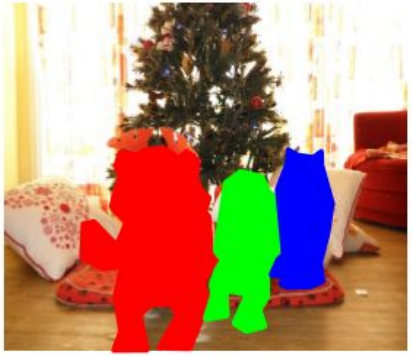
Single Object

Object Detection



DOG, DOG, CAT

Instance Segmentation

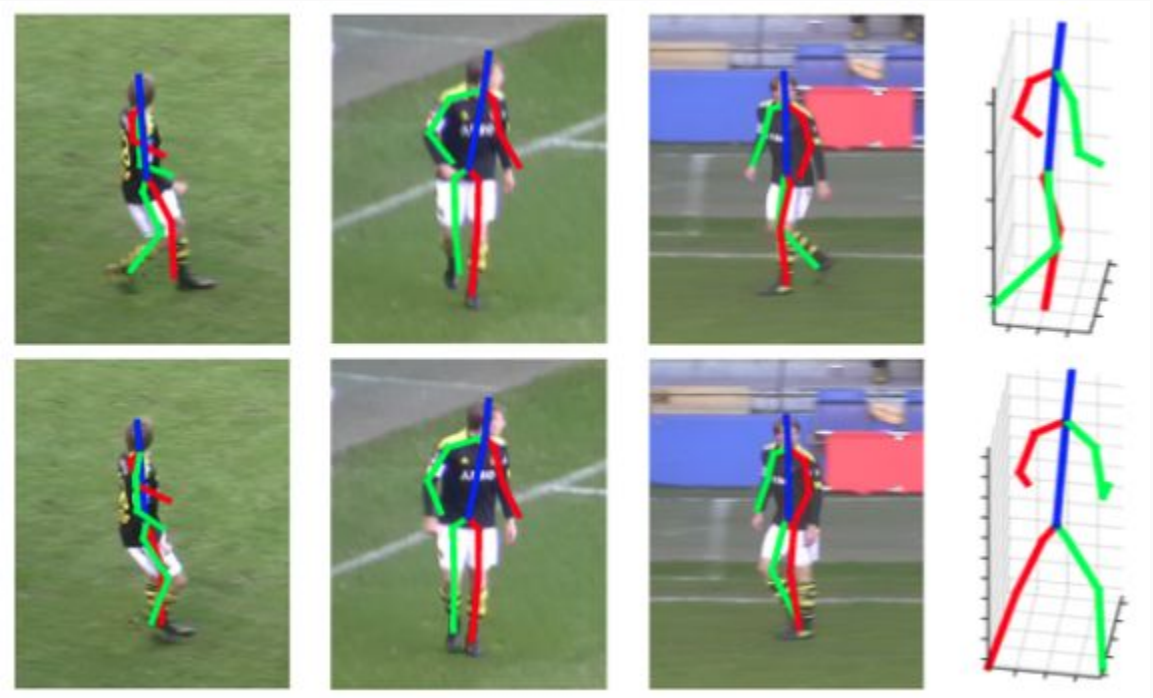


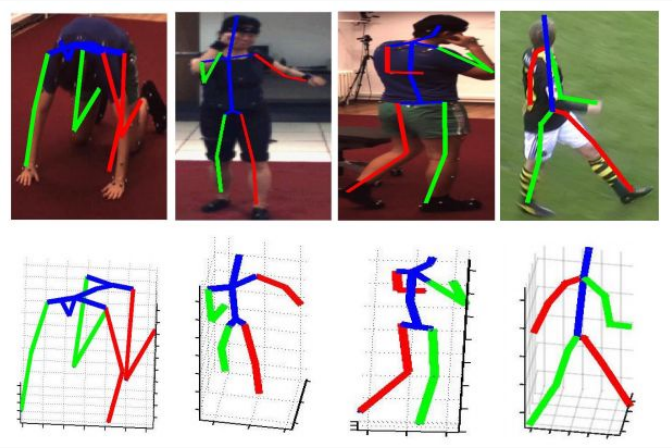
DOG, DOG, CAT

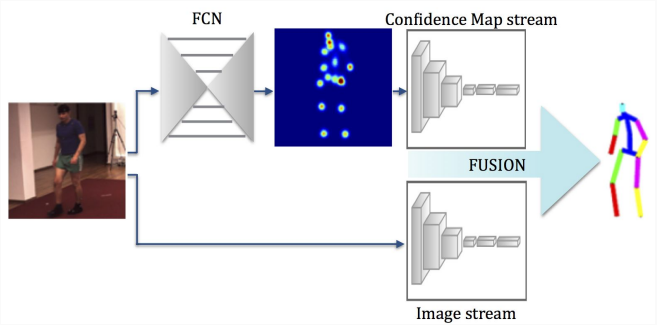
Multiple Object

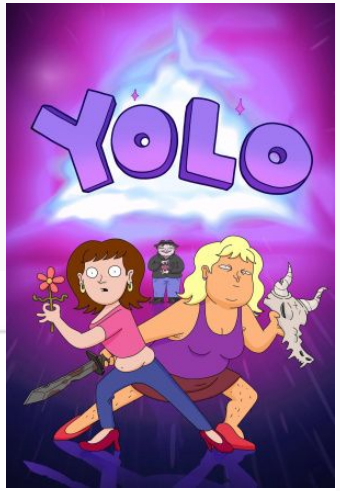
https://sigmoidal.ai/en/real-time-human-pose-estimation-using-mediapipe/

https://www.epfl.ch/labs/cvlab/research/research-surv/research-surv-human-pose-estimation/





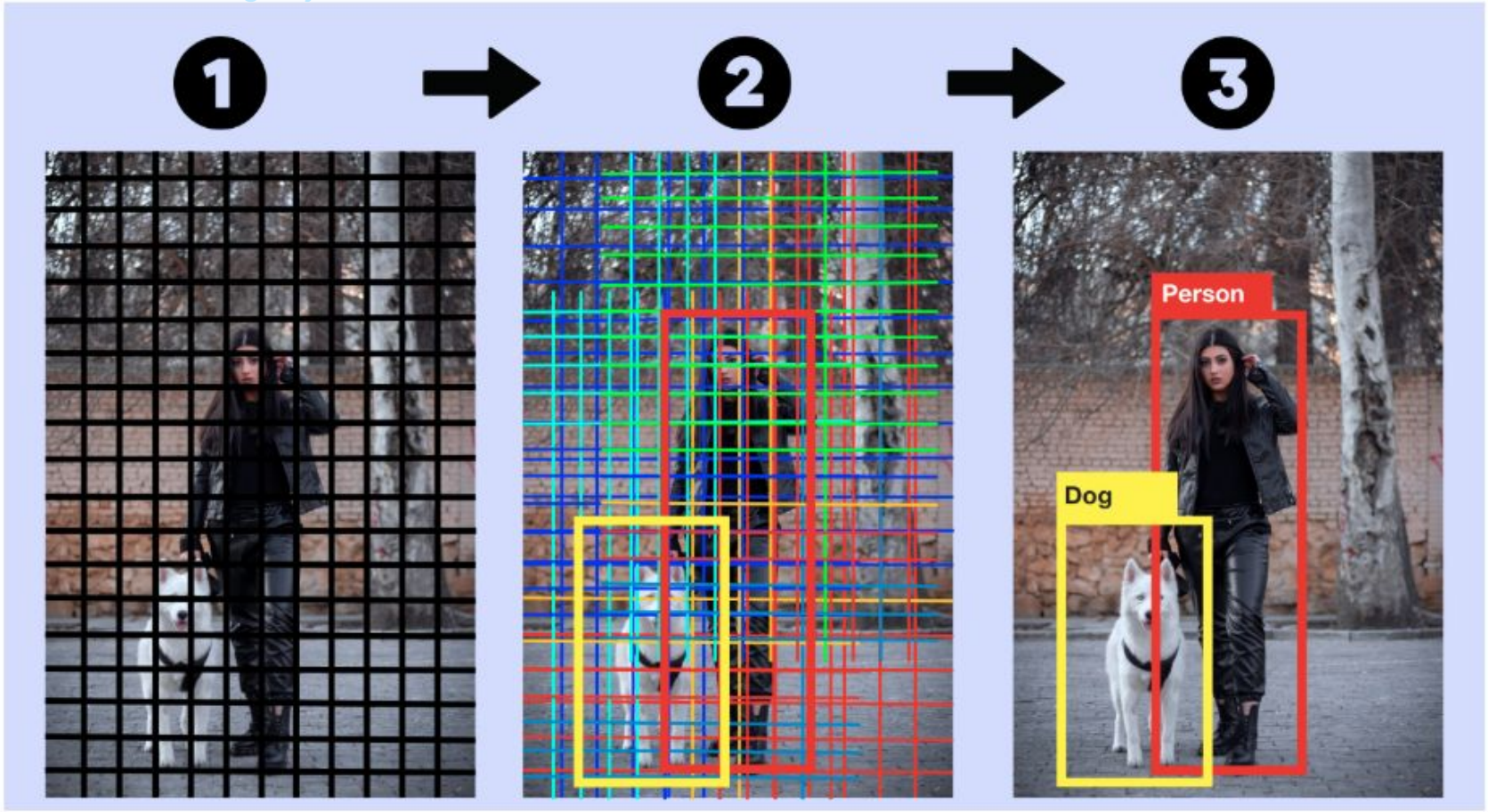




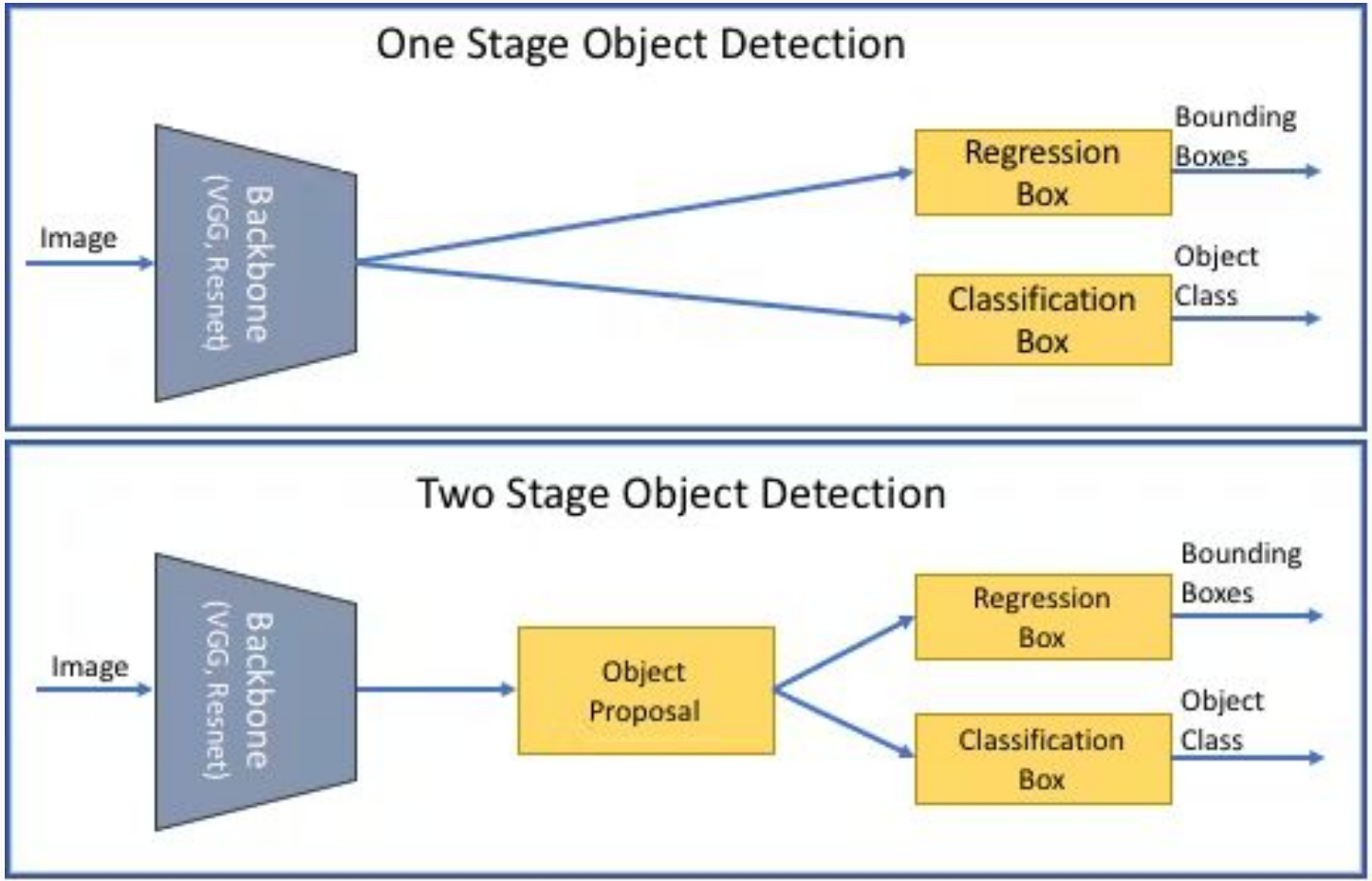
What is YOLO?

- **YOLO (You Only Look Once)**: Real-time object detection architecture.
- **Philosophy**: Single neural network predicts bounding boxes and class probabilities in one pass → extremely fast.
- **Applications**: Detection, segmentation, pose estimation.

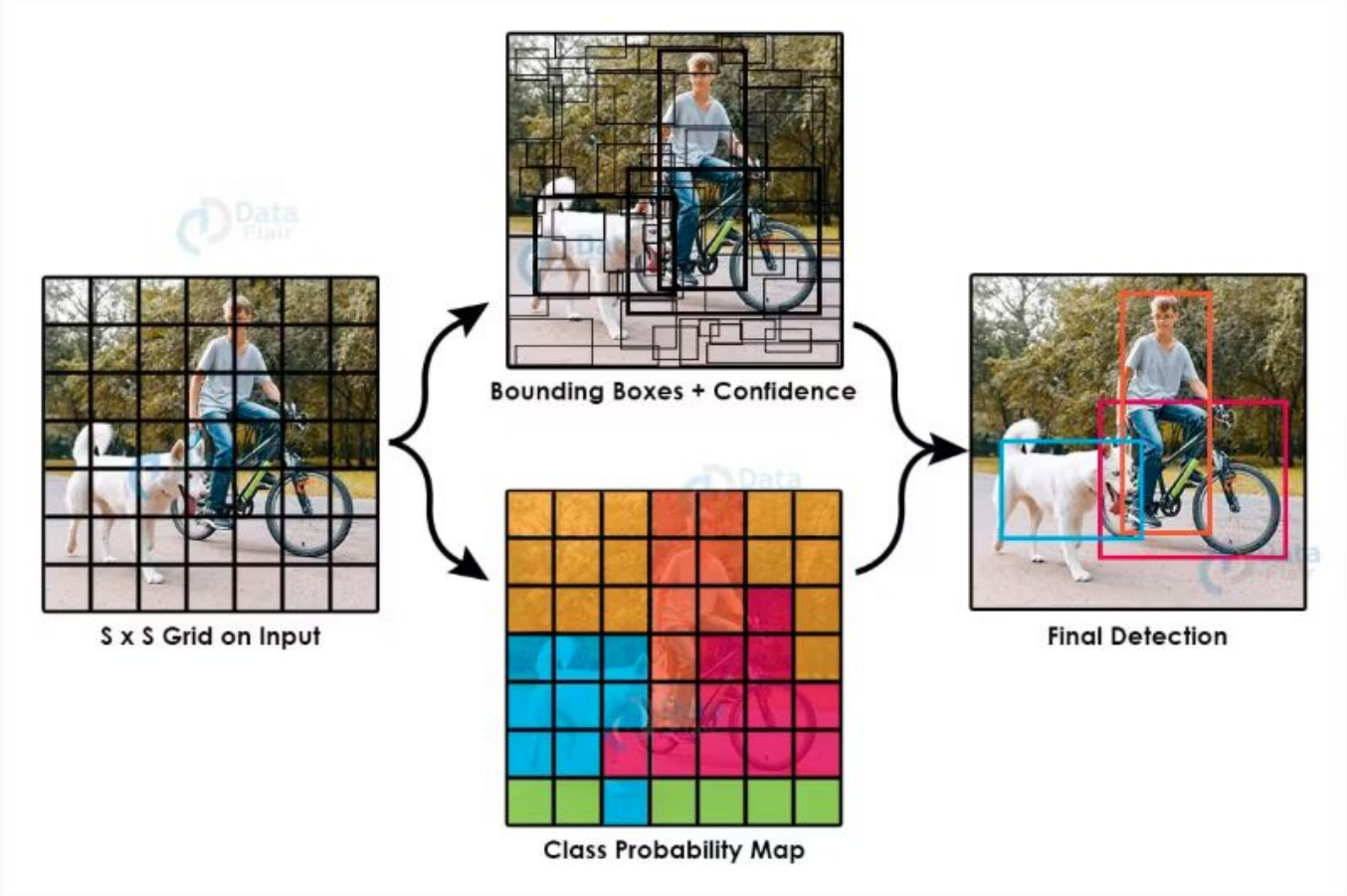
https://encord.com/blog/object-detection/



https://encord.com/blog/object-detection/

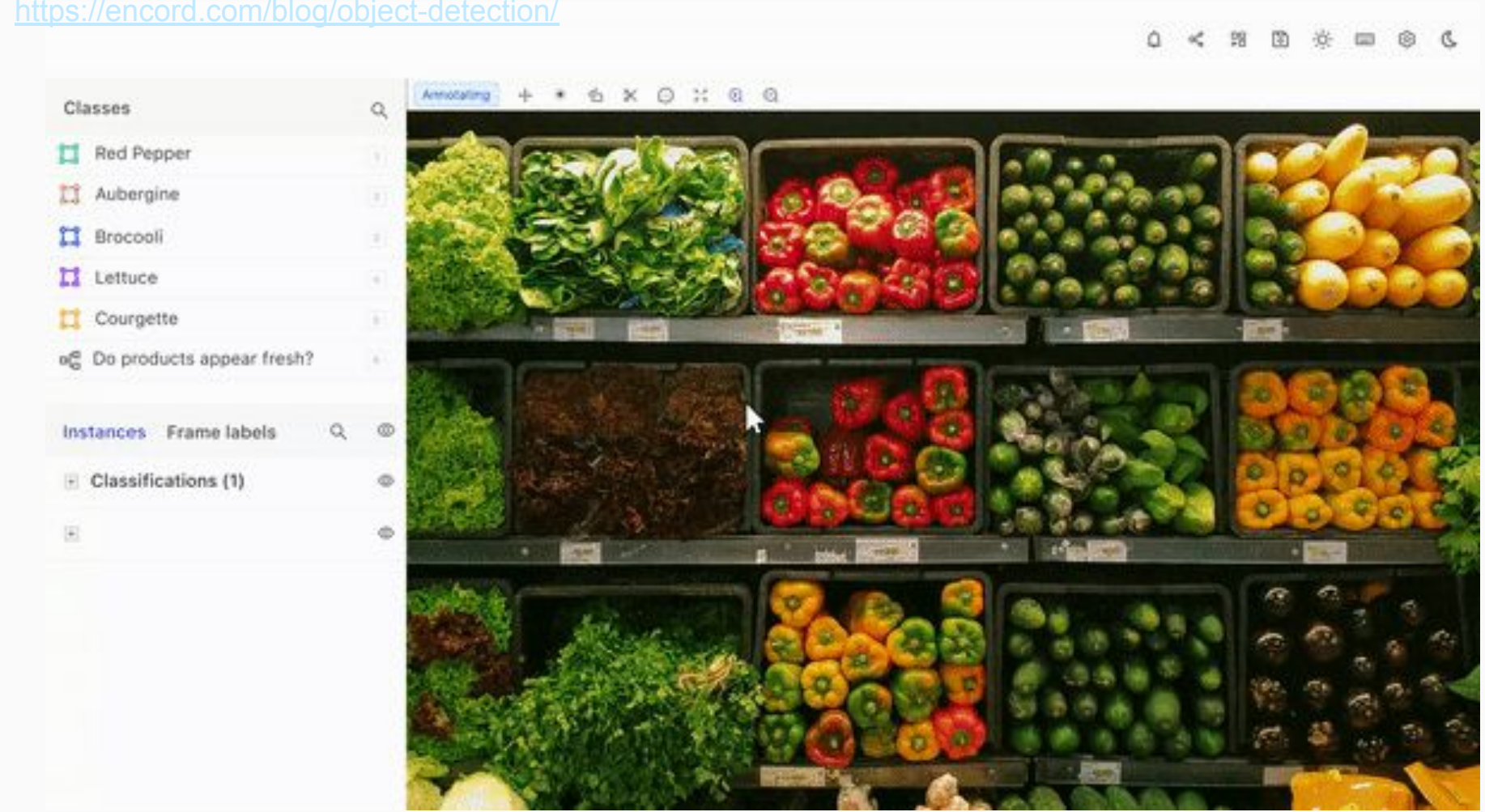


https://encord.com/blog/object-detection/

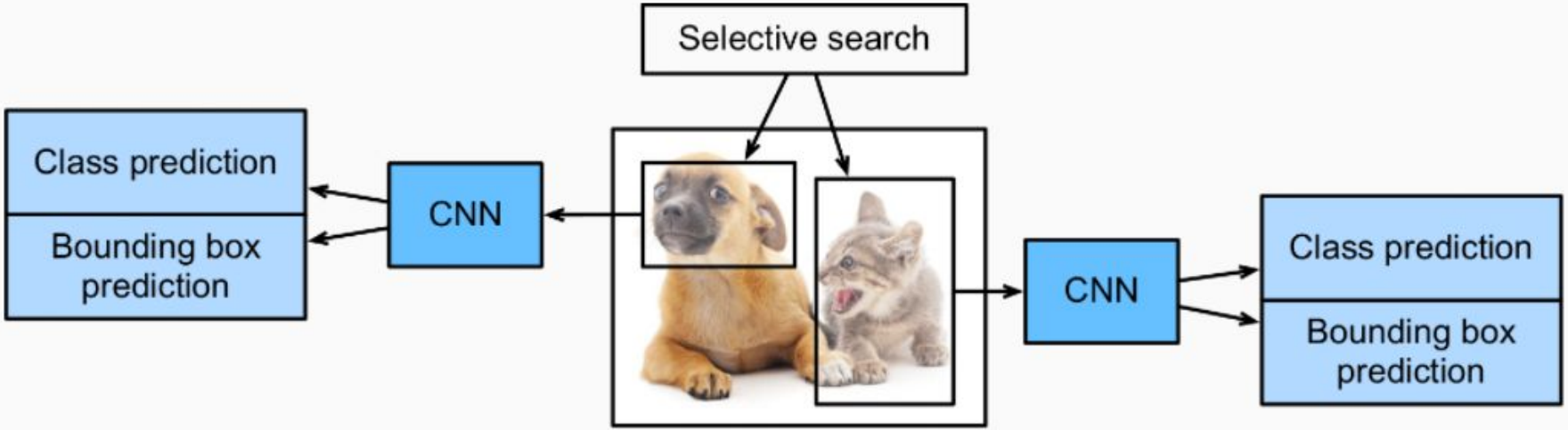


Custom Object Detection with YOLO V5

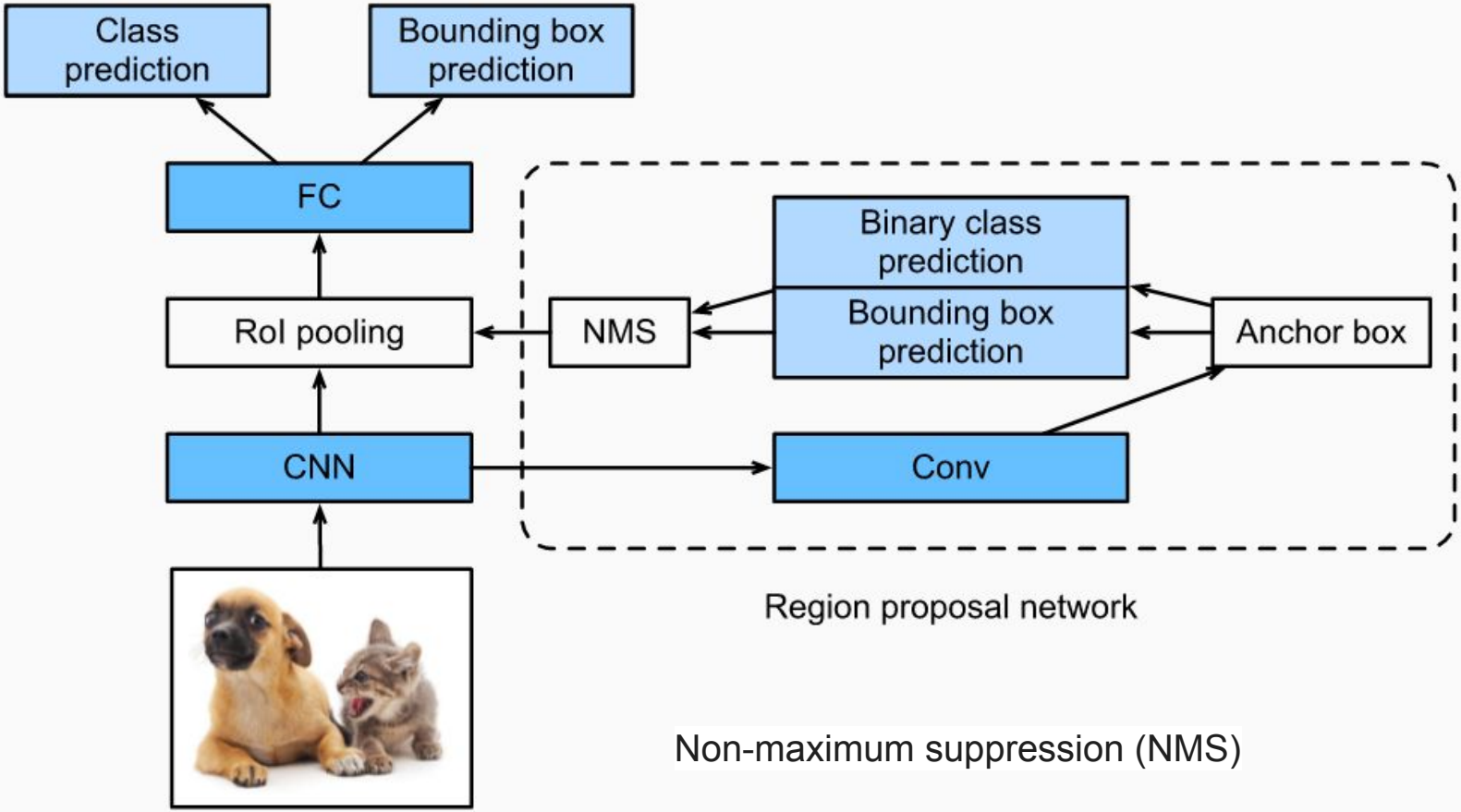
https://encord.com/blog/object-detection/



https://neptune.ai/blog/object-detection-algorithms-and-libraries



https://neptune.ai/blog/object-detection-algorithms-and-libraries



Non-maximum suppression (NMS)

https://learnopencv.com/non-maximum-suppression-theory-and-implementation-in-pytorch/

Non-maximum suppression (NMS)

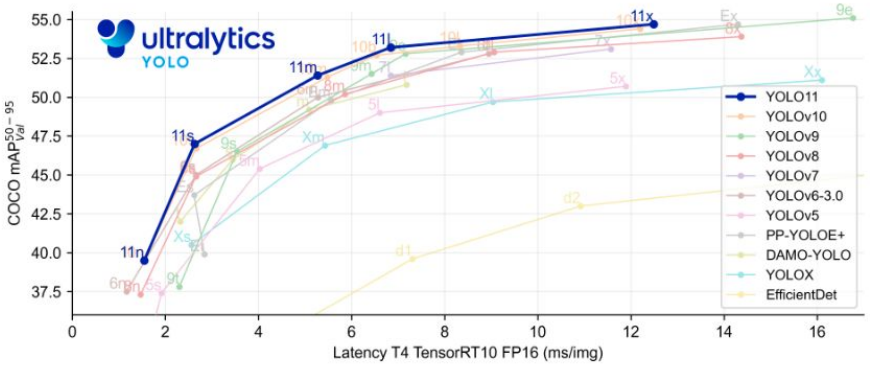
YOLO Family Overview

- **YOLOv8 Nano (n)**: Smallest, fastest, low computation, suitable for edge devices.
- **YOLOv8 Small (s)**: Lightweight, faster, good for smaller datasets or quick experiments.
- **YOLOv8 Medium (m)**: Balance between speed and accuracy, ideal for most use-cases.
- **YOLOv8 Large (l)**: High accuracy, slower inference, suitable for powerful GPUs.
- **YOLOv8 Extra Large (x)**: Maximum accuracy, large datasets, heavy computation.

Ultralytics YOLO11

Overview

YOLO11 is the latest iteration in the Ultralytics YOLO series of real-time object detectors, redefining what's possible with cutting-edge accuracy, speed, and efficiency. Building upon the impressive advancements of previous YOLO versions, YOLO11 introduces significant improvements in architecture and training methods, making it a versatile choice for a wide range of computer vision tasks.



Supported Tasks and Modes

YOLO11 builds upon the versatile model range introduced in YOLOv8, offering enhanced support across various computer vision tasks:

Model	Filenames	Task	Inference	Validation	Training	Export
YOLO11	`yolo11n.pt` `yolo11s.pt` `yolo11m.pt` `yolo11l.pt` `yolo11x.pt`	Detection	✅	✅	✅	✅
YOLO11-seg	`yolo11n-seg.pt` `yolo11s-seg.pt` `yolo11m-seg.pt` `yolo11l-seg.pt` `yolo11x-seg.pt`	Instance Segmentation	✅	✅	✅	✅
YOLO11-pose	`yolo11n-pose.pt` `yolo11s-pose.pt` `yolo11m-pose.pt` `yolo11l-pose.pt` `yolo11x-pose.pt`	Pose/Keypoints	✅	✅	✅	✅
YOLO11-obb	`yolo11n-obb.pt` `yolo11s-obb.pt` `yolo11m-obb.pt` `yolo11l-obb.pt` `yolo11x-obb.pt`	Oriented Detection	✅	✅	✅	✅
YOLO11-cls	`yolo11n-cls.pt` `yolo11s-cls.pt` `yolo11m-cls.pt` `yolo11l-cls.pt` `yolo11x-cls.pt`	Classification	✅	✅	✅	✅

Performance

Detection (COCO) | Segmentation (COCO) | Classification (ImageNet) | Pose (COCO) | OBB (DOTAv1)

See Detection Docs for usage examples with these models trained on COCO, which include 80 pre-trained classes.

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed T4 TensorRT10 (ms)	params (M)	FLOPs (B)
YOLO11n	640	39.5	56.1 ± 0.8	1.5 ± 0.0	2.6	6.5
YOLO11s	640	47.0	90.0 ± 1.2	2.5 ± 0.0	9.4	21.5
YOLO11m	640	51.5	183.2 ± 2.0	4.7 ± 0.1	20.1	68.0
YOLO11l	640	53.4	238.6 ± 1.4	6.2 ± 0.1	25.3	86.9
YOLO11x	640	54.7	462.8 ± 6.7	11.3 ± 0.2	56.9	194.9

How to Use YOLO

1. Install **Ultralytics YOLO** (`pip install ultralytics`).
2. Load a pre-trained model or define your own.

```python
from ultralytics import YOLO
model = YOLO("yolov8n.pt")  # detection
```

3. Train on your dataset with `.train()` method.
4. Evaluate and visualize results with `.val()` and inference methods.

Example

Python CLI

PyTorch pretrained `*.pt` models as well as configuration `*.yaml` files can be passed to the `YOLO()` class to create a model instance in Python:

```python
from ultralytics import YOLO

# Load a COCO-pretrained YOLO11n model
model = YOLO("yolo11n.pt")

# Train the model on the COCO8 example dataset for 100 epochs
results = model.train(data="coco8.yaml", epochs=100, imgsz=640)

# Run inference with the YOLO11n model on the 'bus.jpg' image
results = model("path/to/bus.jpg")
```

Object Detection

Back to top

Bounding box object detection is a computer vision technique that involves detecting and localizing objects in an image by drawing a bounding box around each object.

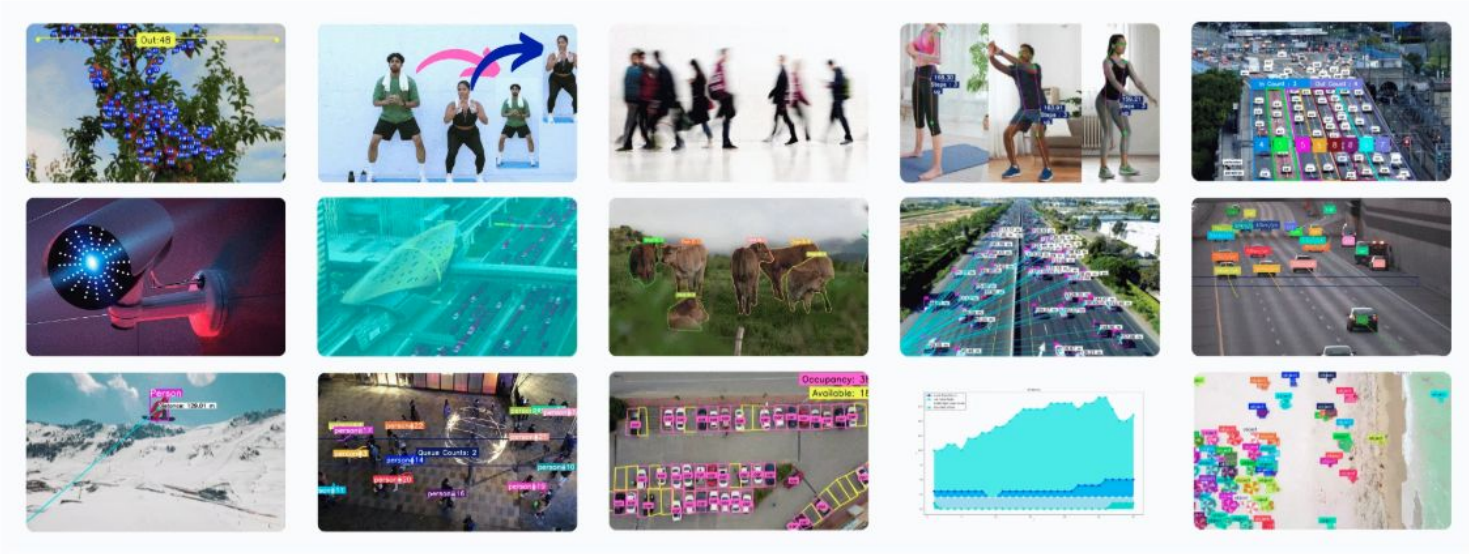
- Argoverse: A dataset containing 3D tracking and motion forecasting data from urban environments with rich annotations.
- COCO: Common Objects in Context (COCO) is a large-scale object detection, segmentation, and captioning dataset with 80 object categories.
- LVIS: A large-scale object detection, segmentation, and captioning dataset with 1203 object categories.
- COCO8: A smaller subset of the first 4 images from COCO train and COCO val, suitable for quick tests.
- COCO8-Grayscale: A grayscale version of COCO8 created by converting RGB to grayscale, useful for single-channel model evaluation.
- COCO8-Multispectral: A 10-channel multispectral version of COCO8 created by interpolating RGB wavelengths, useful for spectral-aware model evaluation.
- COCO128: A smaller subset of the first 128 images from COCO train and COCO val, suitable for tests.
- Global Wheat 2020: A dataset containing images of wheat heads for the Global Wheat Challenge 2020.
- Objects365: A high-quality, large-scale dataset for object detection with 365 object categories and over 600K annotated images.
- OpenImagesV7: A comprehensive dataset by Google with 1.7M train images and 42k validation images.
- SKU-110K: A dataset featuring dense object detection in retail environments with over 11K images and 1.7 million bounding boxes.
- HomeObjects-3K New 🚀: A dataset of annotated indoor scenes featuring 12 common household items, ideal for developing and testing computer vision models in smart home systems, robotics, and augmented reality.

COCO Format (Common Objects in Context)

- **Definition**: A widely used standard for storing annotations for object detection, segmentation, and keypoint tasks.
- **Structure**: JSON file with fields like:
  - `images` : list of images with `id` , `file_name` , `height` , `width`
  - `annotations` : list of objects with `image_id` , `bbox` (x, y, w, h), `category_id` , `segmentation` , `keypoints`
  - `categories` : list of class names with `id`
- **Supports**:
  - Bounding boxes → object detection
  - Segmentation masks → instance segmentation
  - Keypoints → pose estimation

Ultralytics Solutions: Harness YOLO11 to Solve Real-World Problems

Ultralytics Solutions provide cutting-edge applications of YOLO models, offering real-world solutions like object counting, blurring, and security systems, enhancing efficiency and accuracy in diverse industries. Discover the power of YOLO11 for practical, impactful implementations.



Ultralytics YOLO format

The Ultralytics YOLO format is a dataset configuration format that allows you to define the dataset root directory, the relative paths to training/validation/testing image directories or `*.txt` files containing image paths, and a dictionary of class names. Here is an example:

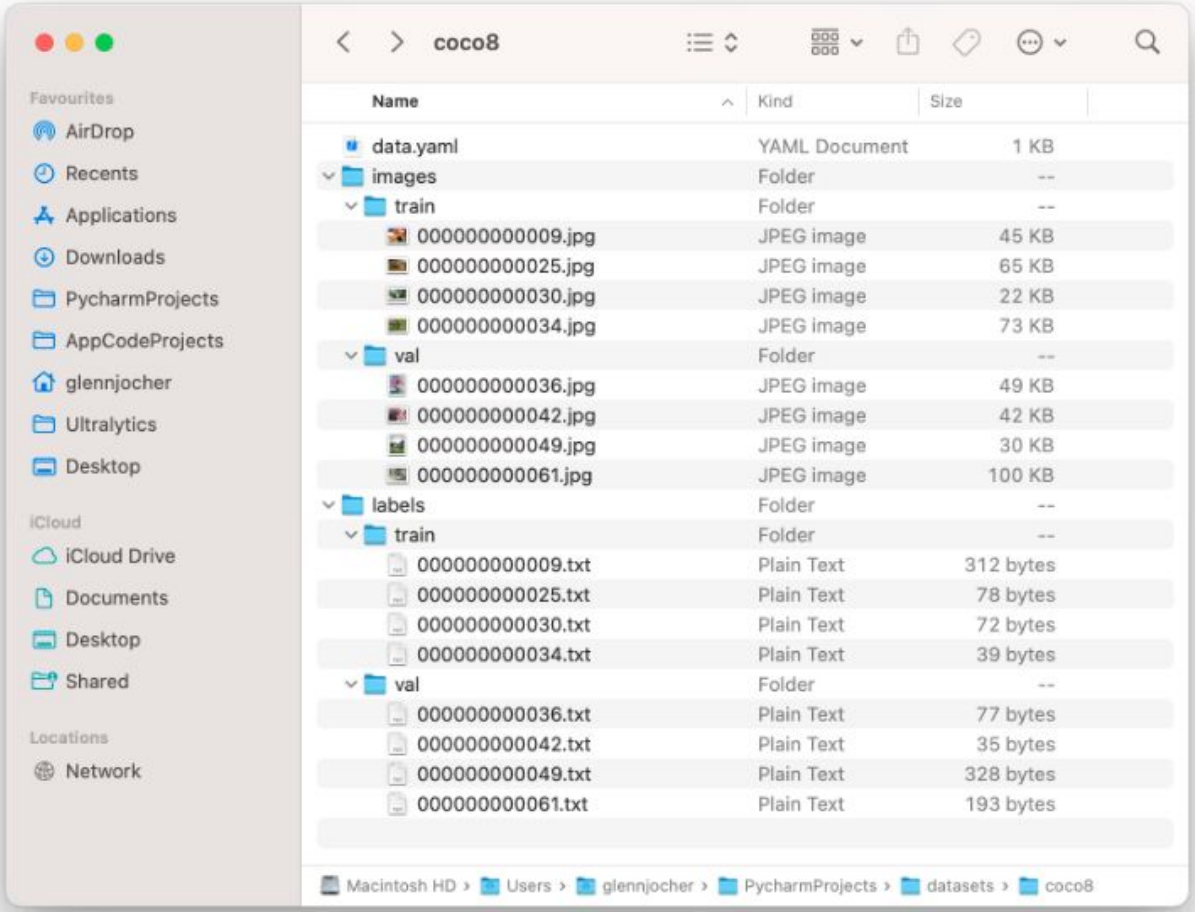
```yaml
# Train/val/test sets as 1) dir: path/to/imgs, 2) file: path/to/imgs.txt, or 3) list: [path/t
path: coco8 # dataset root dir
train: images/train # train images (relative to 'path') 4 images
val: images/val # val images (relative to 'path') 4 images
test: # test images (optional)

# Classes
names:
  0: person
  1: bicycle
  2: car
  3: motorcycle
  4: airplane
  5: bus
  6: train
  7: truck
  8: boat
  9: traffic light
  10: fire hydrant
  11: stop sign
```

https://docs.ultralytics.com/datasets/detect/#ultralytics-yolo-format

The label file corresponding to the above image contains 2 persons (class `0`) and a tie (class `27`):

zidane.txt

```
0  0.481719 0.634028 0.690625 0.713278
0  0.741094 0.524306 0.314750 0.933389
27 0.364844 0.795833 0.078125 0.400000
```


coco8
Favourites
AirDrop
Recents
Applications
Downloads
PycharmProjects
AppCodeProjects
glennjocher
Ultralytics
Desktop
iCloud
iCloud Drive
Documents
Desktop
Shared
Locations
Network
Name
Kind
Size
data.yaml YAML Document 1 KB
images Folder --
train Folder --
000000000009.jpg JPEG image 45 KB
000000000025.jpg JPEG image 65 KB
000000000030.jpg JPEG image 22 KB
000000000034.jpg JPEG image 73 KB
val Folder --
000000000036.jpg JPEG image 49 KB
000000000042.jpg JPEG image 42 KB
000000000049.jpg JPEG image 30 KB
000000000061.jpg JPEG image 100 KB
labels Folder --
train Folder --
000000000009.txt Plain Text 312 bytes
000000000025.txt Plain Text 78 bytes
000000000030.txt Plain Text 72 bytes
000000000034.txt Plain Text 39 bytes
val Folder --
000000000036.txt Plain Text 77 bytes
000000000042.txt Plain Text 35 bytes
000000000049.txt Plain Text 328 bytes
000000000061.txt Plain Text 193 bytes
Macintosh HD › Users › glennjocher › PycharmProjects › datasets › coco8

roboflow

Preparing Custom Datasets

- **Label Your Own Dataset**:
  - Recommended tool: **Label Studio** → annotate images for detection, segmentation, or keypoints.
- **Play with Public Datasets**:
  - Recommended tool: **Roboflow** → download, augment, split datasets easily.

https://labelstud.io/

Preparing Custom Datasets (Label Your Own)

- **Why label your own dataset**:
  - Real-world applications often have unique objects not found in public datasets.
  - Proper annotation ensures model learns the right features.
- **Recommended Tool: Label Studio**
  - Open-source, versatile labeling platform.
  - Supports: bounding boxes, polygons, keypoints, masks, text.
  - Workflow:
    1. Upload images.
    2. Define labeling schema (classes, keypoints).
    3. Annotate images.
    4. Export in YOLO format (`.txt` for detection, masks for segmentation, keypoints for pose).
  - Tips:
    - Start with small batches, check annotation quality.
    - Use keyboard shortcuts to speed up labeling.
    - Consistent labeling improves model performance.
- **Extra Tip for Students**:
  - Annotate edge cases (small objects, partially occluded objects) → model generalizes better.
  - Review annotations twice; mislabeled images reduce model accuracy.

labelstud.io

GUIDE HOW TO BUILD AI BENCHMARKS THAT EVOLVE WITH YOUR MODEL

Label Studio | Use Cases | Learn | Docs | Integrations | Enterprise | Search | 24,251 | Quick Start

Open Source Data Labeling Platform

The most flexible data labeling platform to fine-tune LLMs, prepare training data, or evaluate AI models.

Download OSS | Compare Versions

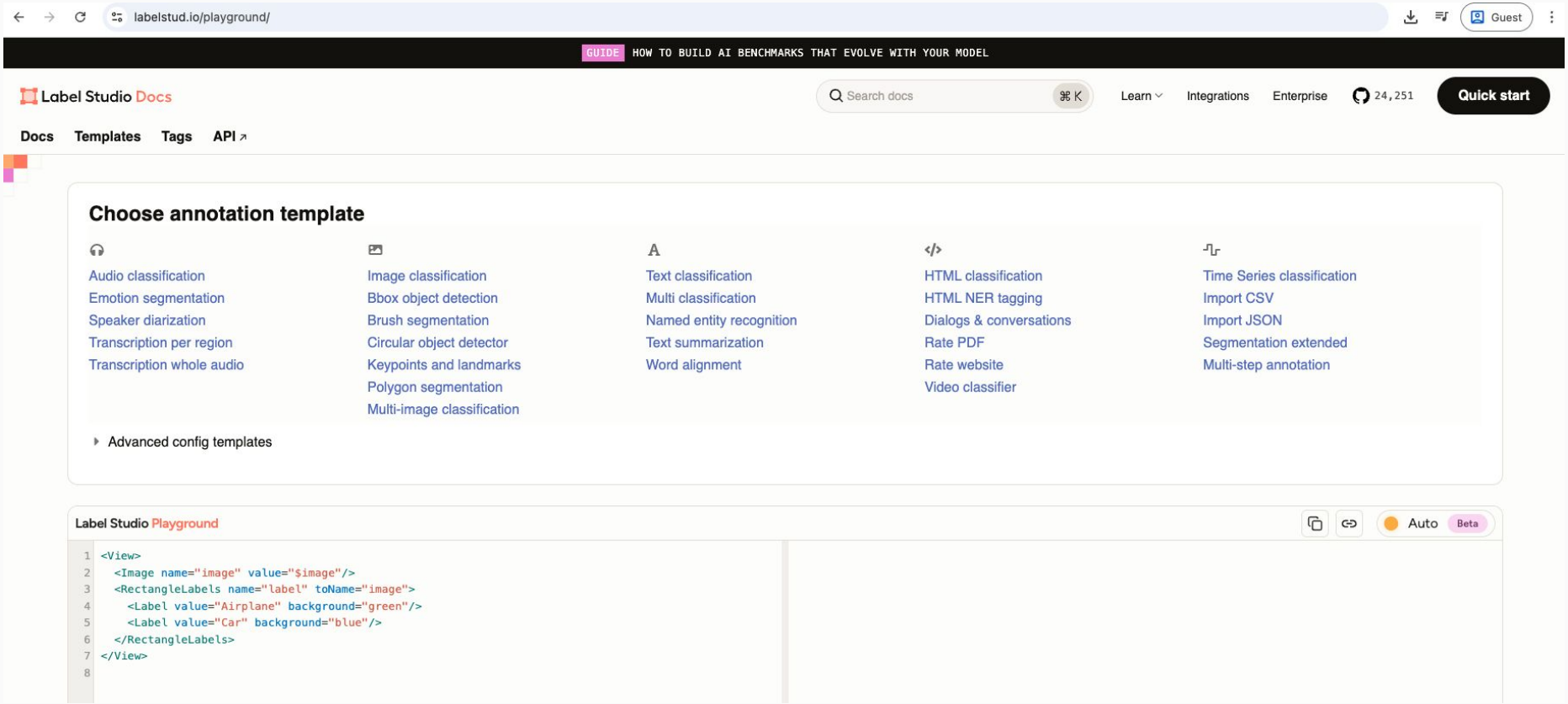
LAST COMMIT: AUGUST 19, 2025 | LATEST VERSION: NIGHTLY

Quick Start

PIP BREW GIT DOCKER

```
# Install the package
# into python virtual environment
pip install -U label-studio

# Launch it!
label-studio
```

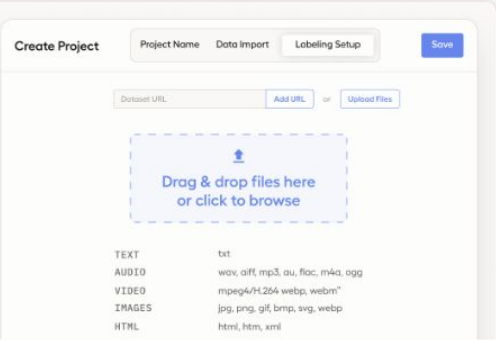
https://labelstud.io/guide



Label Studio Documentation

Discover our Quick Start installation guide, instructions on building custom UIs, pre-built labeling templates to get you labeling more quickly, and much more.

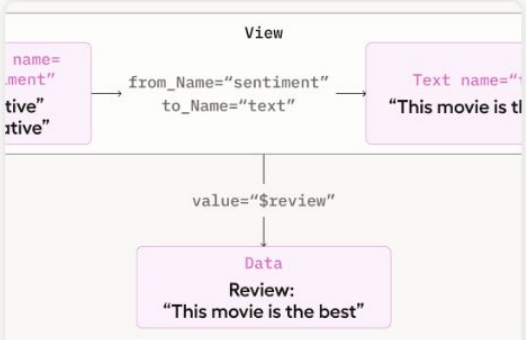
Quick Start Guide



Label Studio 101

Brand new to Label Studio? We've created a jam-packed new tutorial with the most important information to get you up and running.

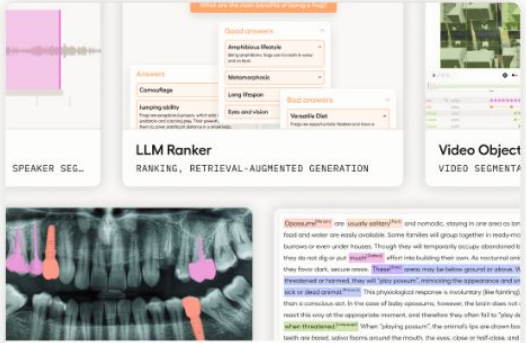
Get started →



Machine Learning with Label Studio

Learn the background knowledge and steps required to integrate Machine Learning models into your Label Studio workflow.

Learn more →



Use templates

Label Studio provides multiple out-of-the-box labeling configurations to help get you started.

Explore templates →

https://labelstud.io/templates/

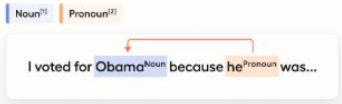
Template Gallery

Computer Vision

Object Detection, Semantic Segmentation, Image Classification

Dynamic Labels

Object Detection, Semantic Segmentation, Image Classification



Natural Language Processing

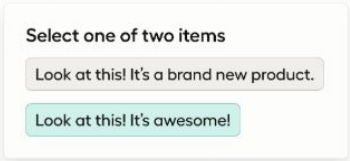
Named Entity Recognition, Text Classification, Relation Extraction

Audio/Speech Processing

Automatic Speech Recognition, Speaker Segmentation, Intent Classification

Conversational AI

Natural Language Understanding, Chatbot Response Generation, Slot Filling

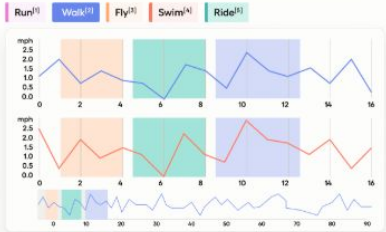


Ranking & Scoring

Pairwise Classification, Document Retrieval

Structured Data Parsing

Freeform Metadata, Tabular Data, Pdf Classification



Time Series Analysis

Activity Recognition, Forecasting, Outliers And Anomaly Detection

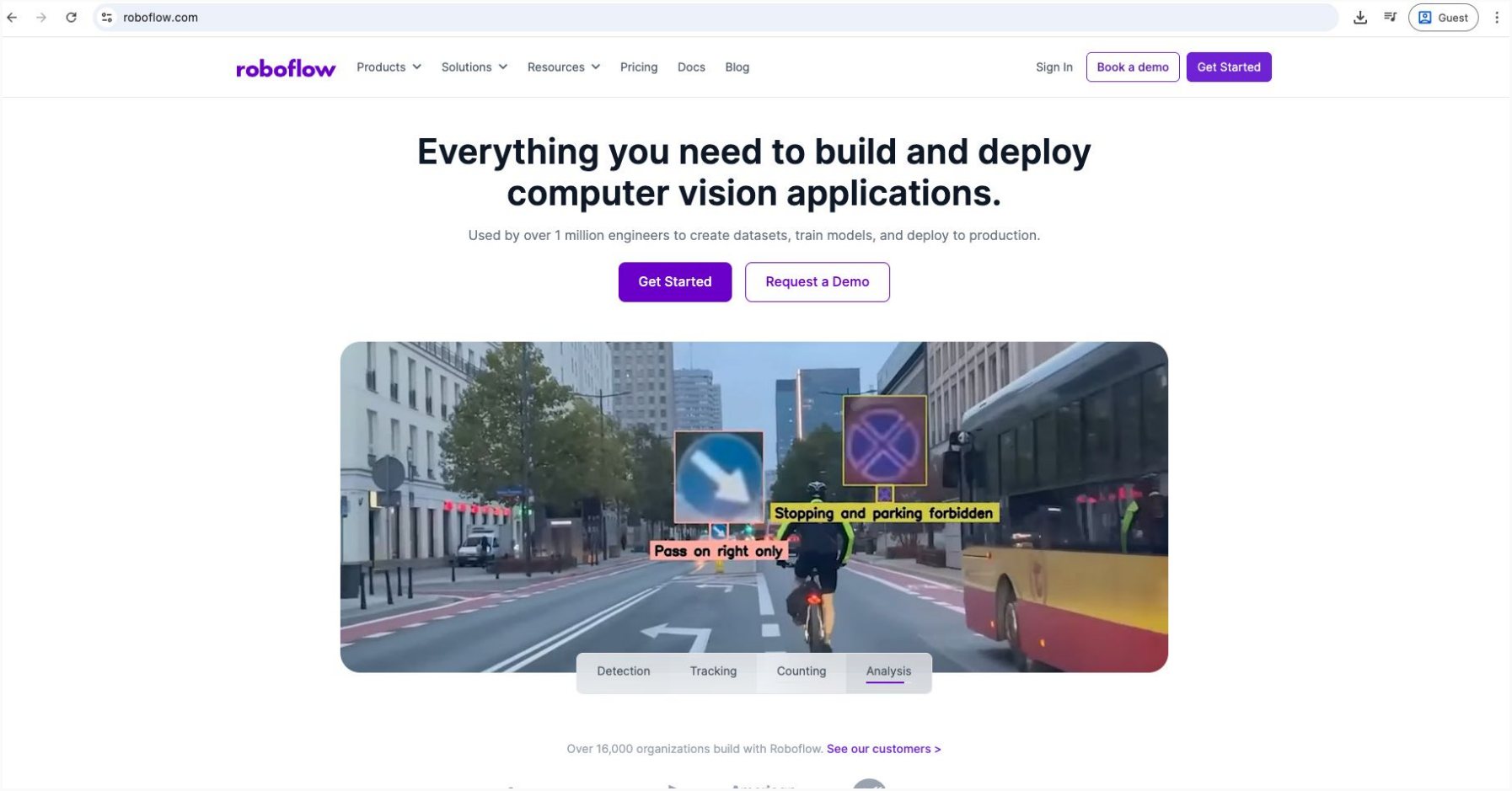
Inventory Tracking

Inventory Tracking system allows you to label exact products by given brand names illustrated by relevant sample photo. Every task with shelf photo also has a list of assiated brands to label.

Using Public Datasets (Roboflow)

- **Why use public datasets**:
  - Save time & effort for training and experimentation.
  - Standard datasets allow benchmarking and comparison.
- **Recommended Tool: Roboflow**
  - Upload or import public datasets (COCO, Pascal VOC, Kaggle, GitHub).
  - Provides:
    - **Auto-splitting** into train/val/test sets.
    - **Data augmentation**: flips, rotation, brightness, noise.
    - **Conversion** into YOLO format automatically**.
  - Steps:
    1. Import dataset or upload your images.
    2. Apply augmentation if needed.
    3. Export dataset in YOLOv8-ready format.
  - Tips:
    - Explore multiple public datasets → combine to increase variety.
    - Use preview function to verify images & annotations before training.
- **Extra Tip for Students**:
  - Use Roboflow for **quick prototyping** before investing time in labeling your own data.
  - Augment dataset to handle class imbalance or improve robustness.

https://roboflow.com/



roboflow.com

Guest

roboflow Products Solutions Resources Pricing Docs Blog Sign In Book a demo Get Started

Everything you need to build and deploy computer vision applications.

Used by over 1 million engineers to create datasets, train models, and deploy to production.

Get Started Request a Demo

roboflow Products Solutions Resources Pricing Docs Blog

PLATFORM

Universe
Open source computer vision datasets and pre-trained models

Annotate
Label images fast with AI-assisted data annotation

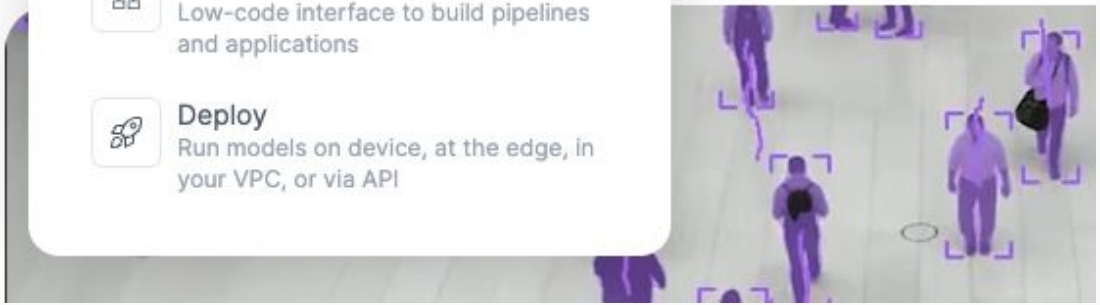
Train
Hosted model training infrastructure and GPU access

Workflows
Low-code interface to build pipelines and applications

Deploy
Run models on device, at the edge, in your VPC, or via API

vou need to
ter vision a
gineers to create datasets, t

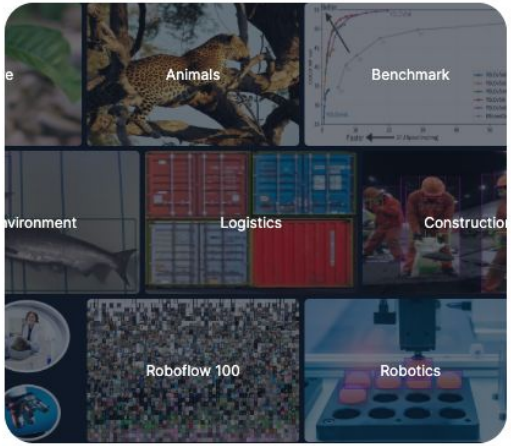
Get Started Reque



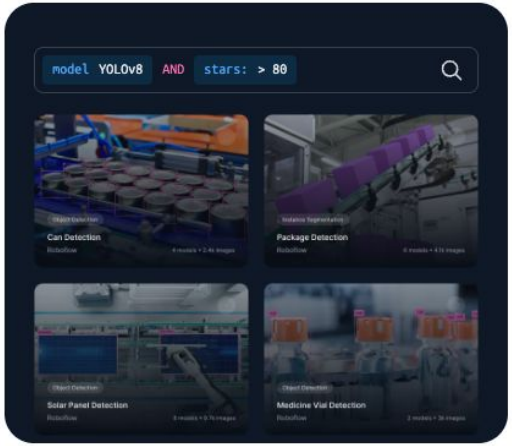
roboflow Products Solutions Resources Pricing Docs Blog Sign In Book a demo Get Started

Search 575,000+ Datasets for Any Use Case

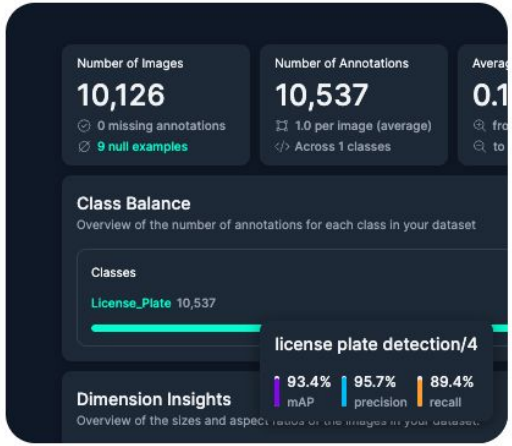
Explore and analyze datasets containing 750 million+ images



Collections of datasets and pre-trained models for specific use cases



Search and filter for relevant projects to find the best resources



Evaluate the health of datasets and accuracy of models before using them

universe.roboflow.com/roboflow-100

Guest

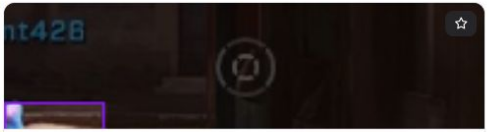
PROJECTS (100)

Object Detection Model yolov5

solar panels
by Roboflow 100
161 images • 1 model
Updated 2 years ago

Object Detection Model yolov5

soccer players
by Roboflow 100
163 images • 1 model
Updated 2 years ago



Object Detection Model yolov5

halo infinite angel videogame
by Roboflow 100
669 images • 1 model
Updated a year ago

Object Detection Model yolov5

aquarium
by Roboflow 100
638 images • 1 model
Updated 2 years ago

Object Detection Model yolov5

furniture
by Roboflow 100
689 images • 1 model
Updated 2 years ago

Object Detection Model yolov5

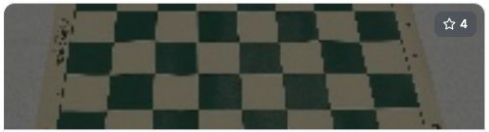
street work
by Roboflow 100
873 images • 1 model
Updated 2 years ago

Object Detection Model yolov5

leaf disease
by Roboflow 100
2501 images • 1 model
Updated 2 years ago

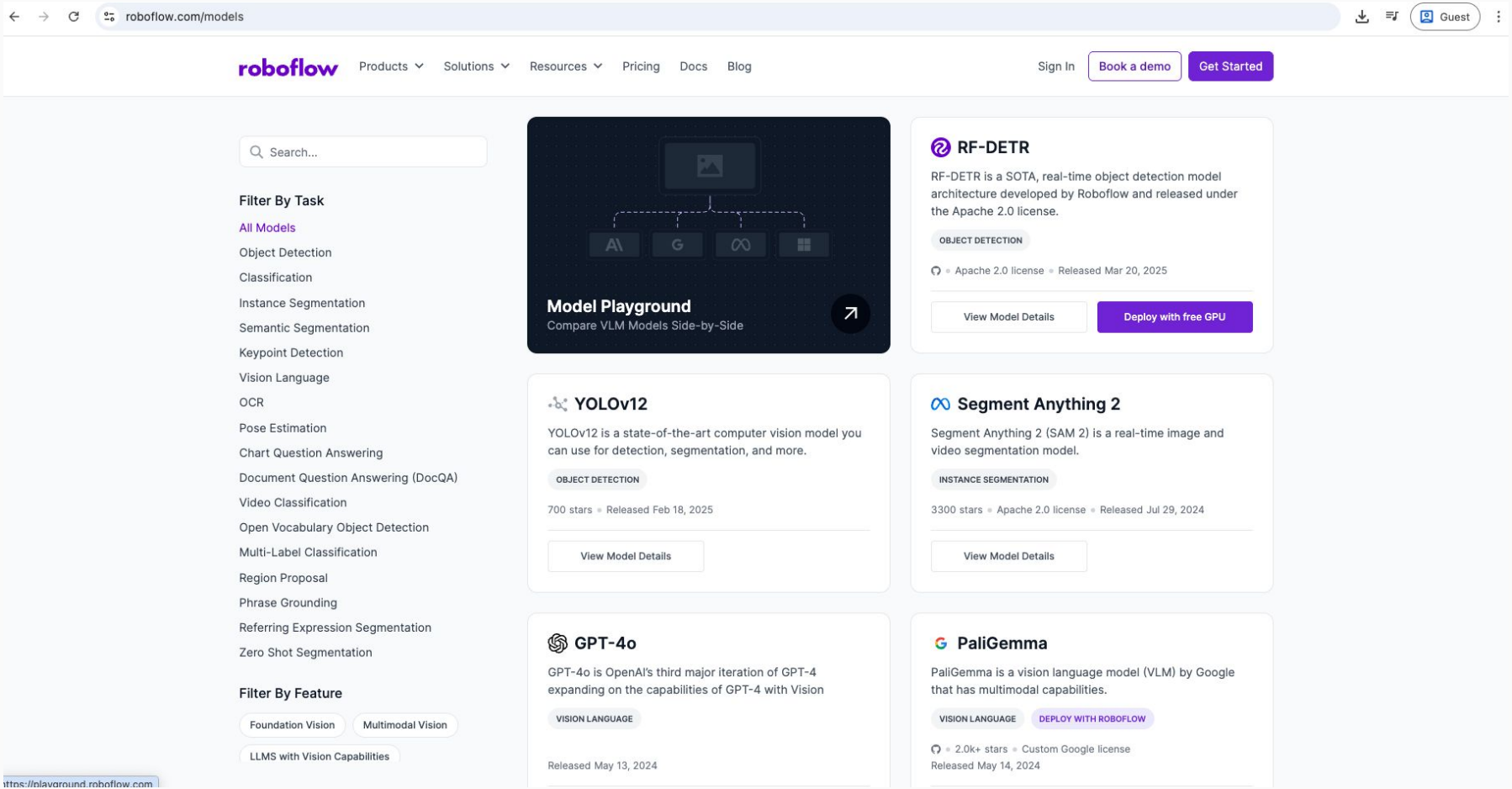
Object Detection Model yolov5

cotton
by Roboflow 100
406 images • 1 model
Updated 2 years ago



Object Detection Model yolov5

chess pieces
by Roboflow 100
289 images • 1 model
Updated 2 years ago


roboflow.com/models
Guest
roboflow
Products
Solutions
Resources
Pricing
Docs
Blog
Sign In
Book a demo
Get Started
Search...
Filter By Task
All Models
Object Detection
Classification
Instance Segmentation
Semantic Segmentation
Keypoint Detection
Vision Language
OCR
Pose Estimation
Chart Question Answering
Document Question Answering (DocQA)
Video Classification
Open Vocabulary Object Detection
Multi-Label Classification
Region Proposal
Phrase Grounding
Referring Expression Segmentation
Zero Shot Segmentation
Filter By Feature
Foundation Vision
Multimodal Vision
LLMS with Vision Capabilities
Model Playground
Compare VLM Models Side-by-Side
RF-DETR
RF-DETR is a SOTA, real-time object detection model architecture developed by Roboflow and released under the Apache 2.0 license.
OBJECT DETECTION
Apache 2.0 license • Released Mar 20, 2025
View Model Details
Deploy with free GPU
YOLOv12
YOLOv12 is a state-of-the-art computer vision model you can use for detection, segmentation, and more.
OBJECT DETECTION
700 stars • Released Feb 18, 2025
View Model Details
Segment Anything 2
Segment Anything 2 (SAM 2) is a real-time image and video segmentation model.
INSTANCE SEGMENTATION
3300 stars • Apache 2.0 license • Released Jul 29, 2024
View Model Details
GPT-4o
GPT-4o is OpenAI's third major iteration of GPT-4 expanding on the capabilities of GPT-4 with Vision
VISION LANGUAGE
Released May 13, 2024
PaliGemma
PaliGemma is a vision language model (VLM) by Google that has multimodal capabilities.
VISION LANGUAGE
DEPLOY WITH ROBOFLOW
2.0k+ stars • Custom Google license
Released May 14, 2024
https://playground.roboflow.com

YOLOv12

YOLOv12 is a state-of-the-art computer vision model you can use for detection, segmentation, and more.

OBJECT DETECTION

700 stars • Released Feb 18, 2025

View Model Details

Segment Anything 2

Segment Anything 2 (SAM 2) is a real-time image and video segmentation model.

INSTANCE SEGMENTATION

3300 stars • Apache 2.0 license • Released Jul 29, 2024

View Model Details

GPT-4o

GPT-4o is OpenAI's third major iteration of GPT-4 expanding on the capabilities of GPT-4 with Vision

VISION LANGUAGE

Released May 13, 2024

View Model Details

PaliGemma

PaliGemma is a vision language model (VLM) by Google that has multimodal capabilities.

VISION LANGUAGE DEPLOY WITH ROBOFLOW

2.0k+ stars • Custom Google license
Released May 14, 2024

View Model Details

YOLOv9

YOLOv9 is an object detection model architecture released on February 21st, 2024.

OBJECT DETECTION DEPLOY WITH ROBOFLOW

8.5k+ stars • GPL-3.0 license • Released Feb 21, 2024

YOLO-World

YOLO-World is a zero-shot object detection model.

OBJECT DETECTION DEPLOY WITH ROBOFLOW

2.9k+ stars • GPL-3.0 license • Released Jan 31, 2024

YOLOv8

YOLOv8 is a state-of-the-art object detection and image segmentation model created by Ultralytics, the developers of YOLOv5.

OBJECT DETECTION DEPLOY WITH ROBOFLOW

21.1k+ stars • AGPL-3.0 license • Released Jan 10, 2023

View Model Details Deploy with free GPU

GPT-5

GPT-5 is a multimodal language model developed by OpenAI.

VISION LANGUAGE

Released Aug 7, 2025

View Model Details

Depth Anything V2

Depth-Anything-V2 is a depth estimation model developed by researchers from HKU and TikTok.

6100 stars • Apache 2.0 license

View Model Details

GPT-4.1

GPT-4.1 is a multimodal model developed by OpenAI that comes in three sizes: GPT-4.1, mini, and nano.

Released Apr 14, 2025

View Model Details

Claude 3.7 Sonnet

Claude 3.7 is a multimodal "hybrid reasoning" model developed by Anthropic.

Released Feb 24, 2025

Phi-4 Multimodal

Phi-4 Multimodal is a multimodal language model developed by Microsoft.

VISION LANGUAGE

MIT License license

SmolVLM2

SmolVLM2 is a multimodal image and video understanding model developed by engineers on the Hugging Face TB (Textbook) Research team.

2000 stars • Apache 2.0 license • Released Feb 20, 2025

View Model Details

Moondream 2

Moondream 2 is the latest model in the Moondream series of "tiny vision language models".

7600 stars • Apache 2.0 license

View Model Details

Gemma 3

Gemma 3 is a multimodal language model developed by Google.

Gemma License license • Released Mar 11, 2025

View Model Details

OpenAI o3-mini

OpenAI o3-mini is a multimodal reasoning model developed by OpenAI.

VISION LANGUAGE

Released Jan 31, 2025

View Model Details

Qwen2.5-VL

Qwen2.5-VL is a multimodal vision-language model developed by the Qwen team at Alibaba Cloud.

VISION LANGUAGE

8600 stars • Apache 2.0 license • Released Jan 28, 2025

View Model Details

PaliGemma-2

PaliGemma-2 is a multimodal model developed by Google.

VISION LANGUAGE DEPLOY WITH ROBOFLOW

Gemma License license • Released Dec 5, 2024

View Model Details

Combining Both Approaches

- **Workflow**:
  1. Start with public dataset (Roboflow) to experiment with YOLO families (n, s, m, l, x).
  2. Label your own custom dataset (Label Studio) for your project-specific objects.
  3. Merge datasets if necessary → improve generalization.
  4. Train YOLO models using both datasets → compare performance.
- **Student Advice**:
  - Keep dataset organized: `/images/train`, `/images/val`, `/labels/train`, `/labels/val`.
  - Ensure `.yaml` config paths match dataset folder structure.
  - Always validate dataset visually before training → catch annotation mistakes early.

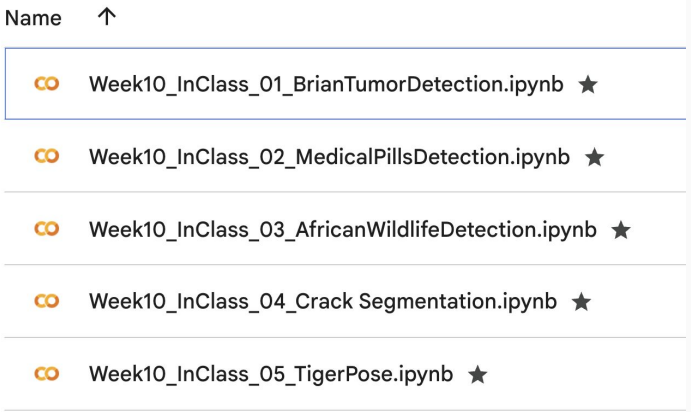
Tricks to Revise and Improve YOLO

- **Training Options**: epochs, batch size, image size, optimizer.
- **Augmentation**: Mosaic, flips, color jitter, rotation.
- **Fine-Tuning**: Start from pre-trained weights → adapt to your dataset.
- **Handling Imbalanced Classes**: Focal loss, class weights, oversampling.
- **Custom Loss Functions**: If default loss isn't sufficient for edge cases.
- **Experimentation Tricks**:
  - Mix different YOLO families (n, s, m, l, x).
  - Advanced augmentation strategies.
  - Add dropout, label smoothing, or additional heads for segmentation/pose.

Lab Today (Hands-On)

Students will run 5 in-class notebooks:

1. **Brain Tumor Detection** – binary detection
2. **Medical Pills Detection** – binary detection
3. **African Wildlife Detection** – multi-class detection
4. **Crack Segmentation** – semantic segmentation
5. **Tiger Pose Estimation** – keypoint detection



Name ↑

Week10_InClass_01_BrianTumorDetection.ipynb ★

ultralytics

Homework

- **Dataset**: Use **Brain Tumor Detection dataset** (Week10_InClass_01).
- **Task**: Run YOLOv8 models from all families: n, s, m, l, x.
- **Goal**: Find which model performs best.
- **Bonus**: Apply tricks like focal loss, advanced augmentation, or fine-tuning to beat larger models → earn extra points.