Workshop3 Image Detection

Yolo: You Only Look Once!

1. Run the Jupyter notebook YoloV8.ipynb using different YoloV8 models and parameters over the provided images and then choose your own images of the Internet that have cars, motorcycles, etc and show the results.

2. Use the script Yolo_Potholes.ipynb in Kaggle/Collab and train a Yolo V8 model, pick different versions of Yolo V8, nano to Xlarge and compare the results, also compare one trained model with the best model and create and download the video of the inference. Download this dataset from: https://www.kaggle.com/datasets/farzadnekouei/pothole-image-segmentation-dataset

3. Bonus: Pick and use a Yolo dataset from the Internet re run the code and show the results and draw your own conclusions.

Answer the following questions:

1. Which model worked better?
2. Why do you think that specific model did a better job?
3. What are the advantages of Yolo V8 over other Yolo Versions?
4. Why Yolo is better than other image detection algorithms?
5. Can you segment images using Yolo? Which Version should I use?
6. Compare versions 1 to version 8 in a table.

Want to know more? Read this paper:

Yolo V1

https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Redmon_You_Only_Look_CVPR_2016_paper.pdf

Which model worked better?

Based on the training logs, here's how the models performed:

YOLOv8n (Nano):

Parameters: 3M

mAP50: ~0.60

Training Time: ~5 min/epoch (CPU)

YOLOv8s (Small):

Parameters: 11.1M

mAP50: ~0.52

Training Time: ~12 min/epoch (CPU)

YOLOv8m (Medium):

Parameters: 25.9M

mAP50: ~0.55 (estimated, full results not visible)

Training Time: ~15 min/epoch (CPU)

Best Model: YOLOv8n (Nano) achieved the highest mAP50 of ~0.60, making it the best performer in this comparison.

2. Why did YOLOv8n perform better?

Dataset Suitability: The pothole detection task might not require the complexity of larger models.

Overfitting: Larger models (especially YOLOv8s) might be overfitting the training data.

Training Dynamics: The default hyperparameters might be better suited for YOLOv8n on this specific task.

Feature Learning: The simpler architecture might be learning more generalizable features for potholes.

3. Advantages of YOLOv8 over other YOLO Versions

Ease of Use:

Simplified API and CLI

Built-in support for tasks beyond detection (segmentation, classification)

Better documentation and community support

Performance:

Faster inference speed

Higher accuracy (mAP) compared to YOLOv5 and earlier versions

Better handling of small objects

Features:

Advanced augmentation techniques

Built-in model export to multiple formats (ONNX, TensorRT, etc.)

Advanced training features like mosaic augmentation and auto-anchors

Efficiency:

Multiple model sizes (nano to extra-large)

Better parameter efficiency

Lower computational requirements for similar performance

4. Why YOLO is better than other object detection algorithms?

Speed: Real-time performance with good accuracy.

Single-Stage Detection: Processes the entire image in one pass (unlike R-CNN).

Good Accuracy-Speed Trade-off: Better balance than many alternatives.

Versatility: Works well on various object sizes and aspect ratios.

Strong Feature Extraction: Uses advanced CNN architectures.

Active Development: Continuous improvements and updates.

5. Can you segment images using YOLO? Which version should I use?

Yes, YOLOv8 supports instance segmentation. For segmentation:

Recommended Version: YOLOv8x-seg (extra large) for best accuracy

Balanced Choice: YOLOv8m-seg (medium) for good balance of speed and accuracy

Lightweight: YOLOv8n-seg (nano) for edge devices

6. YOLO Version Comparison (v1 to v8)

Version	Year	Key Features	mAP (COCO)	Speed (FPS)
YOLOv1	2016	First version, single-stage	63.4	45
YOLOv2	2017	Batch normalization, anchor boxes	78.6	67
YOLOv3	2018	Feature pyramid, better backbone	57.9	30
YOLOv4	2020	CSPDarknet, PAN, SPP	65.7	62
YOLOv5	2020	PyTorch implementation	64.1	140
YOLOv6	2022	RepVGG, anchor-free	52.5	520
YOLOv7	2022	E-ELAN, model scaling	56.8	161
YOLOv8	2023	Anchor-free, SOTA	53.9	123

Note: Performance metrics can vary based on model size and hardware.

For your pothole detection task, I recommend:

Best Performance: YOLOv8n (as shown in your results)

For Segmentation: YOLOv8n-seg or YOLOv8m-seg

For Better Accuracy: Try YOLOv8s with more training data and data augmentation