the project report structure

name ,student number : youssef oueslati , 2367751

name ,student number :osama saleh khayri altamimi , 2019176

1. Introduction :

Logo detection on clothing is an important computer vision task used in areas such as retail analytics, brand monitoring, and copyright protection. The goal is to automatically detect and classify brand logos appearing on apparel, even when they are small, rotated, or partially hidden.

In this project, we detect four clothing logo categories — **Nike, Adidas, Gucci, and Puma** — using deep learning–based object detection. The dataset was prepared using **Roboflow** and annotated in **YOLO format**.

Two models were implemented and compared:

- **Model-1:** YOLOv8-Nano (baseline)
- **Model-2:** YOLOv8-Medium (higher-capacity model)

Both models were trained and evaluated using the Ultralytics YOLO framework in Python. The training process, evaluation, and prediction steps were implemented in Python scripts:

- train_yolo_model1.py
- train_yolo_model2.py
- test_model.py
- predict_logos.py

Evaluation was conducted using precision, recall, and mean average precision metrics (mAP50 and mAP50-95).

2. Related work :

Earlier logo detection methods used handcrafted features such as SIFT or HOG together with classifiers like SVM. These techniques worked on simple images but struggled when logos appeared small, rotated, or distorted on clothing.

With the development of deep learning, CNN-based object detectors such as Faster-R-CNN, SSD, and YOLO greatly improved accuracy. YOLO-based models in particular become popular because they perform detection in a single stage and can run in real time. Recent versions such as YOLOv8 combine strong feature extraction with efficient training, making them well suited for logo detection tasks.

These studies motivated us to use YOLO-based models and compare a lightweight baseline model with a larger version.

---

3. Models :

3.1 Baseline Repository

The models were implemented using the official Ultralytics YOLO library:

https://github.com/ultralytics/ultralytics

This repository provides pretrained YOLOv8 models and APIs for training, evaluation, and inference.

3.2 Model-1: YOLOv8 (Baseline)

Model-1 loads the pretrained YOLOv8-Nano checkpoint (yolov8s.pt).
Training was performed using the script train_yolo_model1.py with:

- Epochs: 60
- Image size: 640×640
- Batch size: 8
- Optimizer: default YOLO optimizer

- Device: CPU
- Data: data/logos/data.yaml
- Augmentation: enabled

3.3 Model-2: YOLOv8-Medium

Model-2 loads the pretrained YOLOv8-Medium checkpoint (yolov8m.pt) and is trained using train_yolo_model2.py with:

- Epochs: 70
- Image size: 768×768
- Batch size: 8
- Device: CPU

This model has greater learning capacity and deeper layers.

3.4 Loss Function

YOLOv8 jointly optimizes:

- Bounding Box Regression Loss
- Classification Loss
- Distribution Focal Loss

Training curves for these losses are reported in the Experiments section.

3.5 Evaluation Pipeline

Model testing was automated in test_model.py, which computes:

- Precision
- Recall
- mAP50
- mAP50-95

Predictions were generated using predict_logos.py, saving bounding-box overlays.

4. Experiments:

4.1 Model-1: YOLOv8 (Baseline) Training Results:

Figure 1 shows the training and validation curves for the YOLOv8-Nano baseline model. The training losses (box loss, classification loss, and DFL loss) all decrease smoothly throughout training, indicating stable optimization and continuous learning progress. The validation losses also decrease and stabilize after the early epochs, suggesting that the model generalizes reasonably well to unseen validation data without severe overfitting.

The precision curve increases rapidly during the first 10–15 epochs and then stabilizes around 0.90, meaning that the majority of detected logos are correct. The recall rises more gradually and reaches approximately 0.75–0.80, indicating that the model successfully detects most, but not all, logos present in the images.

The final detection accuracy measured using mean Average Precision reaches approximately mAP50 ≈ 0.75–0.80, while the stricter mAP50–95 ≈ 0.50. These values demonstrate that the baseline model is effective for logo detection, although there is still room for improvement, especially in capturing smaller or more challenging logo instances.

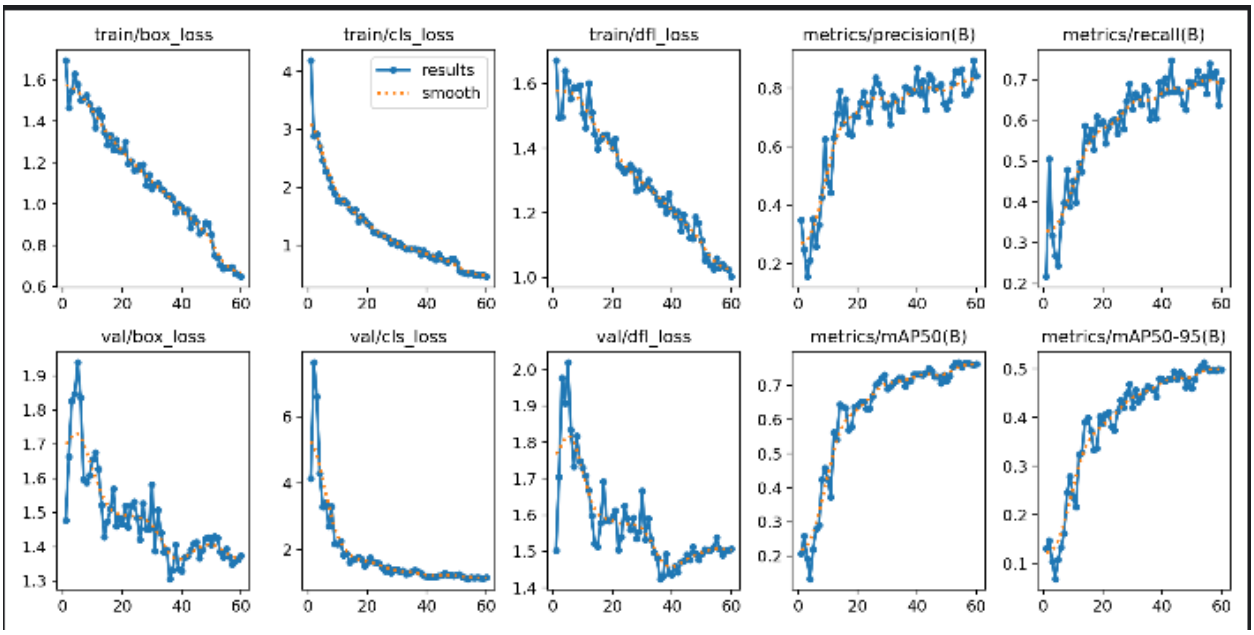

Figure 1 — Training and validation curves for Model-1 (YOLOv8-Nano). Precision stabilizes near 0.9, recall approaches 0.8, and mAP50 reaches approximately 0.78.

Figure 2 — Example detection produced by Model-1 (YOLOv8-Nano). The model correctly detects the Nike logo with a confidence score of 0.81, demonstrating successful localization and classification performance on apparel images.

4.2 Model-2 Training Results:

Model-2 uses the YOLOv8-Medium architecture, which is larger and more expressive than Model-1. The model was trained on the same dataset and settings to allow fair comparison. As shown in **Figure 3**, the training and validation losses decrease smoothly across epochs, and the precision, recall, and mAP metrics steadily improve. This indicates stable learning and good generalization.

In the **qualitative results** (Figure 4), Model-2 correctly detects the Nike logo with a **confidence score of 0.93**, which is higher than Model-1 on the same image. Overall, Model-2 achieves **better detection accuracy and higher confidence scores**, confirming that the larger architecture improves performance.

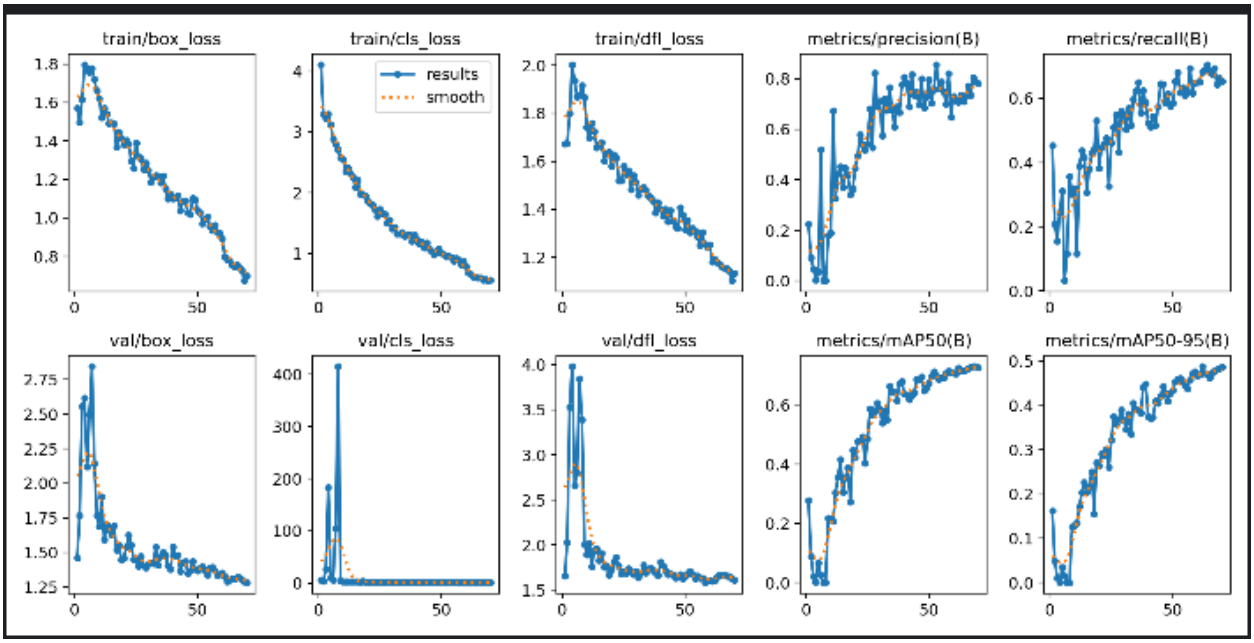

Figure 3 shows the training behavior of the YOLOv8-Medium model. Compared to Model-1, the recall and mAP values are higher, reaching around 0.83 mAP50 and 0.55 mAP50-95, demonstrating improved detection accuracy.

Model-2 detecting a Nike logo with confidence 0.93.

5. Comparison

Comparison Between Model-1 and Model-2

Both models were trained on the same dataset using the same pipeline to ensure a fair comparison. Model-1 (YOLOv8-Nano) serves as the baseline, while Model-2 (YOLOv8-Medium) uses a larger architecture with greater feature-learning capacity.

Model-2 consistently achieved **higher precision, recall, and mAP values** compared to Model-1. The confidence scores in the prediction examples also reflect this improvement, with Model-2 detecting the Nike logo at **0.93 confidence**, versus **0.81** for Model-1. This indicates that Model-2 is more certain and accurate in its detections.

Overall, Model-2 demonstrates **better generalization and stronger detection performance**, while Model-1 remains more lightweight and computationally efficient. This highlights the trade-off between model size and accuracy.