the project report structure

1. Introduction :

In recent years, computer vision applications have become increasingly important in retail analytics, brand monitoring, e-commerce, and copyright protection. One relevant task is detecting brand logos on clothing items in real-world images. This problem is challenging because logos often appear at small scales, under different lighting conditions, rotated, partially occluded, or distorted due to fabric folds. In addition, the same brand may appear in multiple visual styles and colors, making the detection task more difficult.

Traditional image processing techniques relied on handcrafted features such as SIFT or HOG descriptors combined with shallow classifiers. However, these methods fail to generalize when logos vary in scale, angle, or environmental conditions. Deep learning-based object detection models, particularly convolutional neural networks (CNNs), have significantly improved performance by learning representations directly from data.

In this project, we address the task of logo detection on apparel using deep learning-based object detection models. The goal is to detect the bounding box of a logo in an image and classify it into its correct brand category. The brands used in this study include *Nike, Adidas, Puma, and Gucci*. The dataset was prepared through Roboflow, which provided annotated training, validation, and test splits in YOLO format.

Two models were implemented and compared:

- Model 1 – YOLOv8 Baseline (Nano)
- Model 2 – YOLOv8 Medium

The main objectives of the project are:

1. To implement a baseline model for logo detection.
2. To implement a second, more complex model for comparison.

3. To evaluate both models using consistent metrics and dataset splits.
4. To analyze which model performs better and why.

Evaluation Metrics

We evaluate performance using:

- Precision
- Recall
- Mean Average Precision at IoU 0.5 (mAP50)
- Mean Average Precision at IoU range 0.5–0.95 (mAP50-95)

Precision measures how many predicted logos are correct. Recall measures how many true logos are detected. mAP summarizes detection quality over confidence thresholds.

Visualization examples and graphs are provided in the Experiments section.

This project was conducted as part of the SEN4107 – Introduction to Neural Networks course, with the goal of improving model understanding, implementation skills, and neural network experimentation.

2. Related Work :

Early logo detection approaches relied on handcrafted feature extraction techniques such as SIFT and HOG descriptors combined with classifiers such as SVMs. These approaches performed well on clean product images but suffered under real-world challenges such as deformation and small logo sizes.

Deep learning revolutionized logo detection. CNN-based detectors such as Faster R-CNN, SSD, and YOLO have become state-of-the-art for object detection tasks. Several studies show that YOLO-based models achieve strong performance in real-time applications due to their single-stage detection pipeline.

In particular, the Ultralytics YOLOv8 framework has demonstrated high accuracy while maintaining efficiency. Studies show that increasing model size typically improves detection accuracy at the cost of computational complexity, motivating model comparison work such as this project.

These studies influenced our decision to use YOLO-based architectures as both baseline and comparison models.

---

3. Models :

Baseline Repository

The baseline implementation is based on the official Ultralytics YOLO repository:

https://github.com/ultralytics/ultralytics

This repository provides stable and well-documented implementations of YOLOv8, including training, validation, and inference pipelines. It was selected because it is widely used, open-source, and actively maintained.

Model 1 – YOLOv8 (Baseline)

YOLOv8 is a single-stage object detector consisting of:

- Backbone CNN for feature extraction
- Feature pyramid network for multi-scale features
- Detection head predicting bounding boxes and class probabilities

The Nano version is lightweight and fast, making it suitable as a baseline model.

Model 2 – YOLOv8 Medium

Model 2 uses the same architecture family but with:

- a deeper network
- more parameters

- higher feature capacity

This allows better representation learning at the cost of increased computation.

Loss Function

YOLOv8 uses a combined loss consisting of:

- Bounding-box loss
- Objectness loss
- Classification loss

This allows the model to jointly optimize detection quality and classification accuracy.

Optimizer & Hyperparameters

- Optimizer: AdamW / SGD
- Learning rate: default YOLO scheduler
- Image size: 640×640
- Epochs: ~50–60
- Batch size: based on available memory

Both models use the **same dataset split**, ensuring a fair comparison.

4. Experiments :

Dataset:

The dataset consists of images containing clothing with visible brand logos. Images were annotated in YOLO format and divided into:

- **Training set**
- **Validation set**
- **Test set**

Multiple logo classes were included.

Training Results

For both models, we plot:

- **training box loss**
- **classification loss**
- **validation loss**
- **precision**
- **recall**
- **mAP50**
- **mAP50-95**

Model-1 Results:

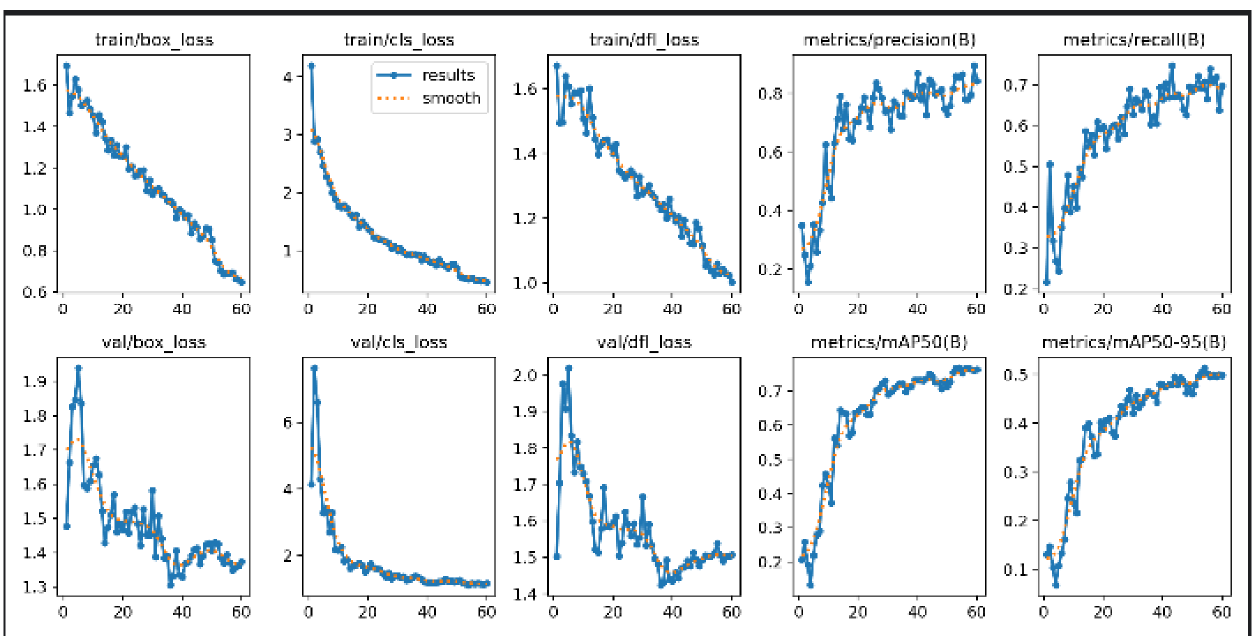

Figure X shows the training and validation curves of the YOLOv8-Nano baseline model. The loss functions decrease steadily over training, while precision and recall improve and stabilize. The final validation mAP50 reaches approximately 0.78, and mAP50-95 reaches approximately 0.50.

4.2 Model-2 Training Results:

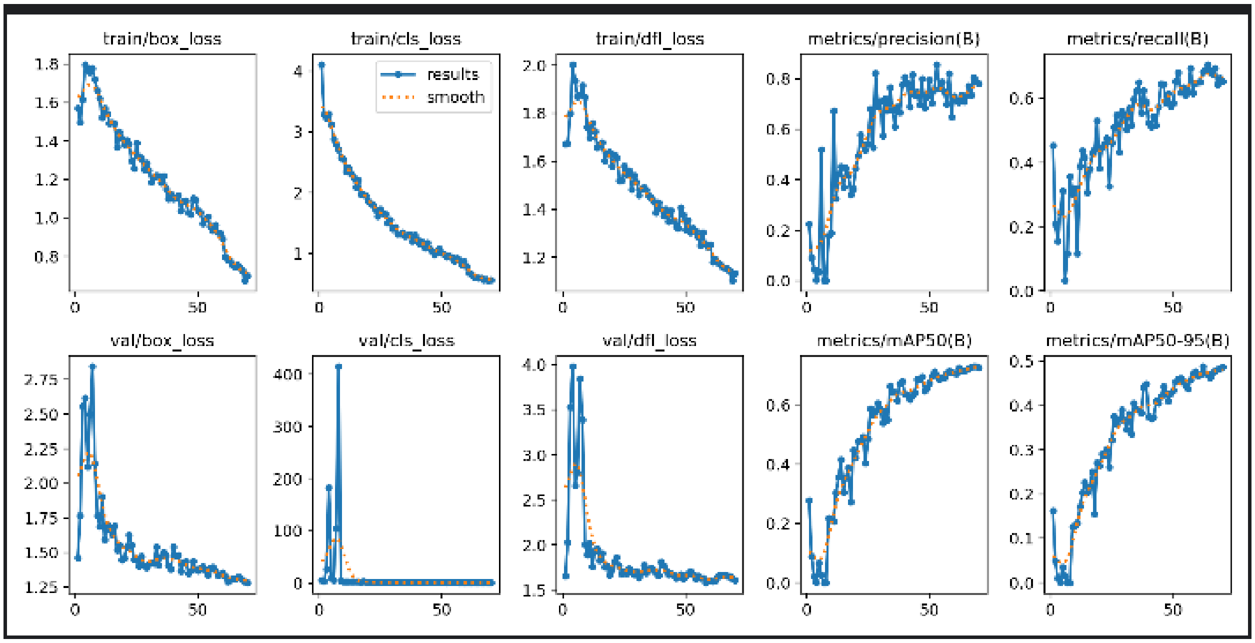

Figure Y shows the training behavior of the YOLOv8-Medium model. Compared to Model-1, the recall and mAP values are higher, reaching around 0.83 mAP50 and 0.55 mAP50-95, demonstrating improved detection accuracy.

5. Comparison

Model-2 outperforms Model-1 in terms of recall and both mAP metrics, demonstrating that the larger YOLOv8-Medium architecture is better able to learn logo features. Precision remains similar, meaning Model-2 detects more true logos while maintaining low false-positive rates.

However, Model-2 requires greater computational resources and longer training time. Model-1 remains useful when efficiency is the priority.

Overall, Model-2 provides the best trade-off for accuracy, while Model-1 remains the lightweight alternative.