LAB 2 Report: Object Detection, Semantic Segmentation and Transfer Learning

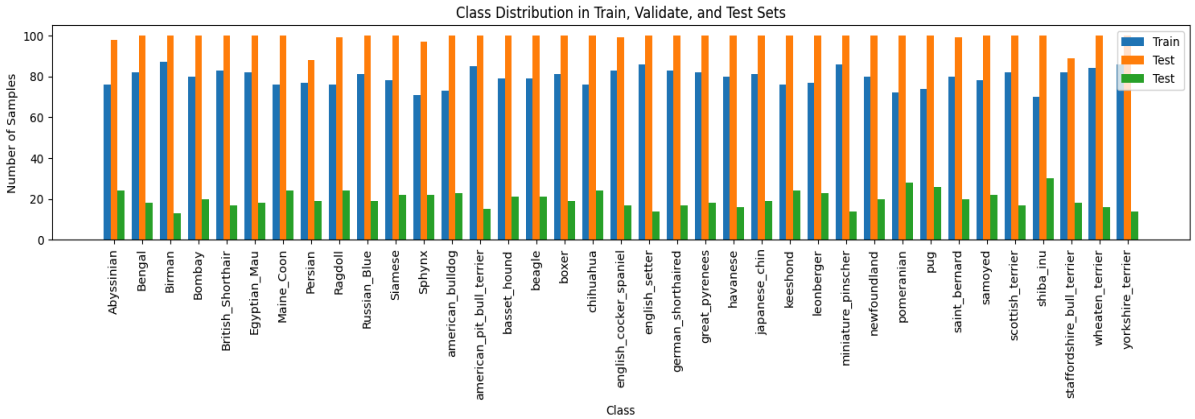
Nguyễn Triều Vương
Student ID: SE182185
FPT University - DAT301m

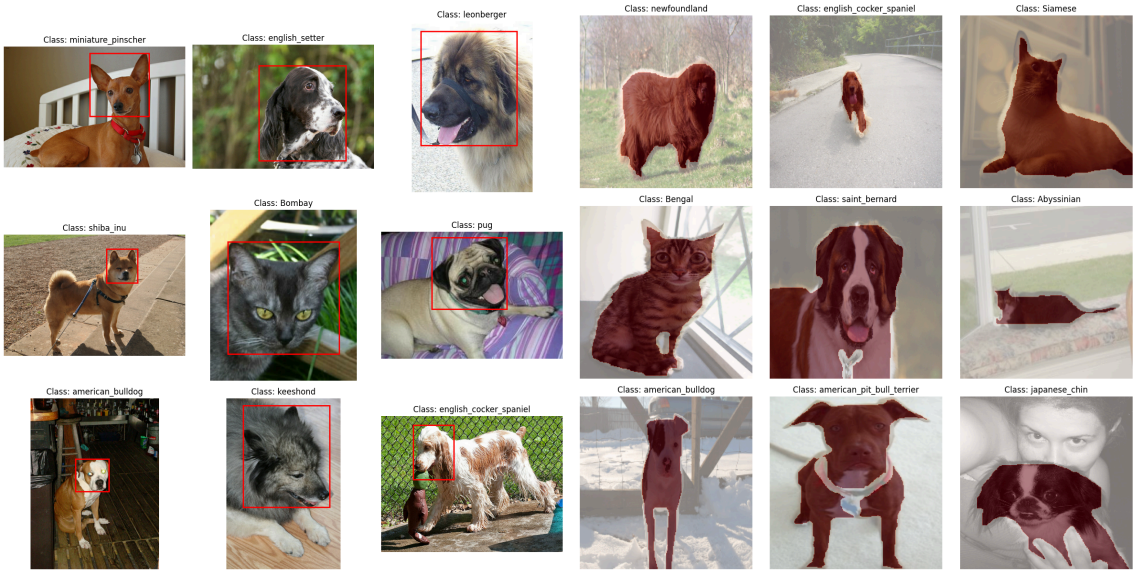
1. Introduction

To begin with, the development of CNN based architectures lead to the rise of the computer vision field in some specific tasks such as Classification, Object Detection and Semantic Segmentation. Therefore this report aims to perform a comparison between these tasks by using several models that come from CNN based architectures in this case are EfficientNet and MobilleNet.

2. Dataset

The dataset will be using the Oxford-IIIT Dataset which has 37 classes, about 200 images for each class. The images have large variations in scale, pose and lighting. Every data point contains label, bounding box coordinate and mask for each task respectively.





3. Preprocessing

Generally, in this lab images are being resized and ensure each pixel is float32. However, due to the models variant after casting images, another preprocess input will be applied to each kind of CNN based model provided by EfficientNet and MobileNet correspondingly.

The preprocessing for Mask is resize, convert into binary format and ensure each pixel is float32.

Then, float32 pixels ensured the Label and Bounding Box.

Lastly, map the preprocessing method to the final dataset with batch size of 32 for each set.

4. Object Detection

4.1 Backbone

EfficientNetB3 is the backbone for detection. Following the global average pooling of backbone output are two blocks fully connected with RELU activation function and batch normalization layers. After that split into two label branch and bounding box branch each of them has a fully connected and batch normalization as well before the final output related to each task:

+ Label output contains 37 units following is softmax activation function

+ Bounding Box output contains 4 units following sigmoid activation function

4.2 Training Loop

Model Compile:

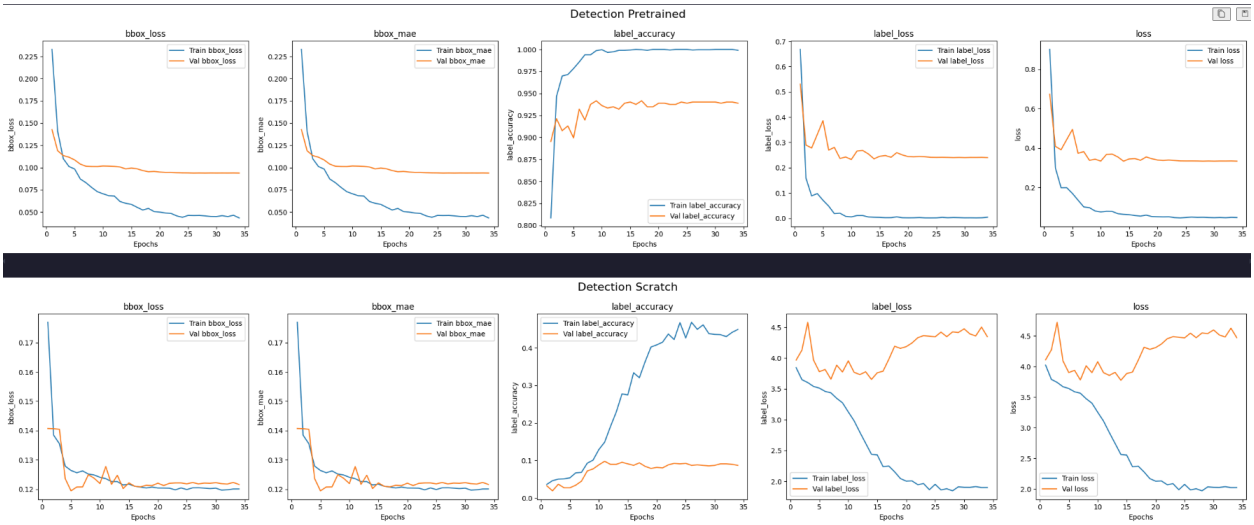
- Optimizer : Adam (learning_rate: 1e-3)
- Loss:
    + Classification: Sparse Categorical Cross Entropy
    + Detection: Mean Absolute Error
- Metrics:
    + Classification: Sparse Categorical Accuracy
    + Detection: Mean Absolute Error

Call backs:

- ReduceLROnPlateau:
    + Monitor: val_loss
    + Factor: 0.5 (new_lr = factor * old_lr)
    + Min Learning rate: 1e-6
    + Limit patience before reduce: 2 (trigger if val_loss is not improved after 2 epochs)

- Early Stopping:
    + Monitor: val_loss
    + Es patience: 20 (trigger if val_loss is not improved after 20 epochs)

4.3 Training results



Bounding Box Loss:

+ Pretrained: Smooth, low loss with slight train-val gap
+ Scratch: Noisy, higher validation loss
  → Pretrained performs better

Bounding Box MAE:

+ Pretrained: Steady, low error
+ Scratch: Noisy, higher errors
  → Pretrained is more stable and accurate

Label Accuracy:

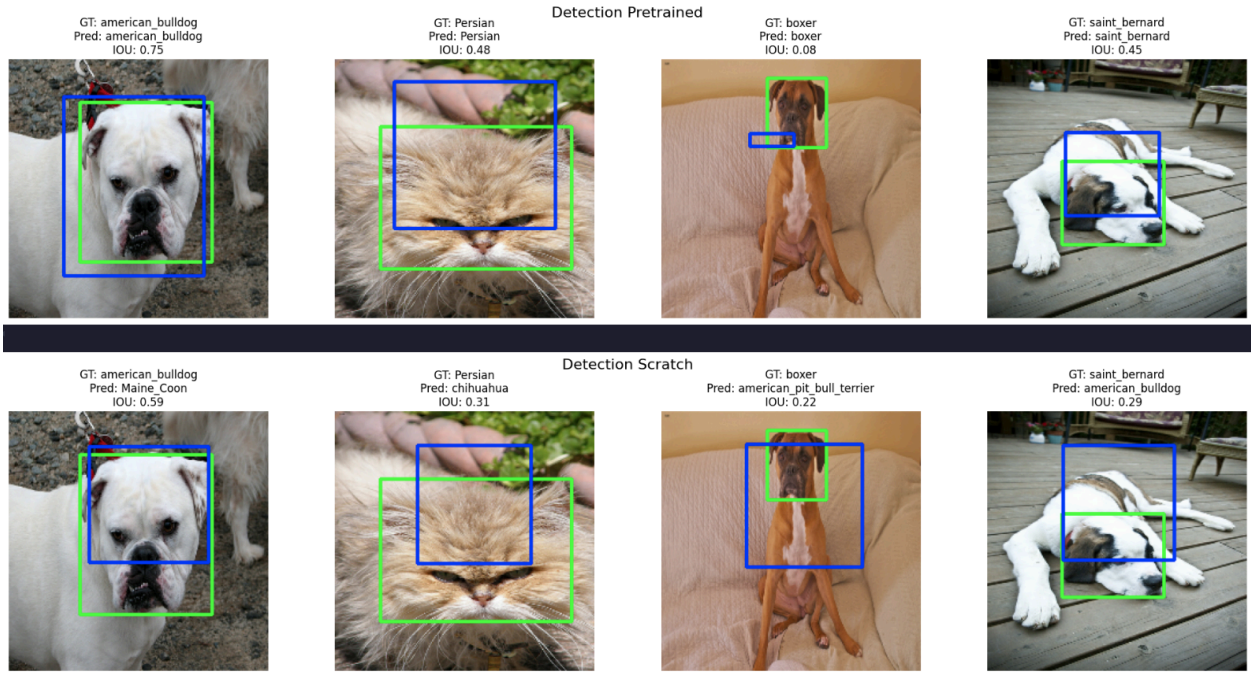
+ Pretrained: High and consistent (~1.0).
+ Scratch: Low val accuracy (~0.1), severe overfitting
  → Pretrained generalization well

Label Loss:

+ Pretrained: Low and converging
+ Scratch: Validation loss increases—overfitting
  → Pretrained is clearly better

Total Loss:

+ Pretrained: Converges with small gaps
+ Scratch: High val loss, poor generalization
  → Pretrained is superior overall



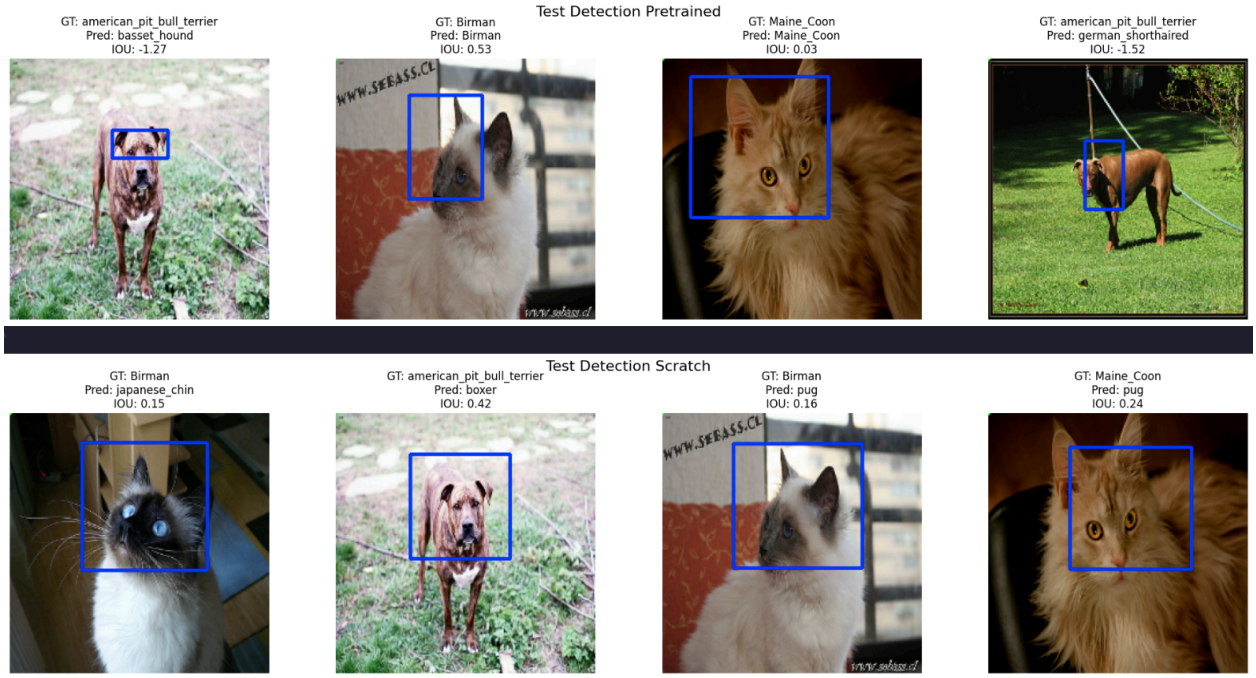
4.4 Evaluation

- Validation Label Accuracy:
  - + Pretrained: 0.9389
  - + Scratch: 0.0951

  → Pretrained model classifies labels far more accurately

- Mean Average Error:
  - + Pretrained: 0.09845525
  - + Scratch: 0.12023202

  → Pretrained has better localization with lower error



Answer Q1: The pretrained model significantly outperformed the scratch-trained model overall. It achieved higher classification accuracy (~93.9%) and lower bounding box mean absolute error (MAE ~0.098), compared to the scratch model's poor classification accuracy (~9.5%) and higher MAE (~0.122). This demonstrates that using a pretrained backbone like EfficientNet provided strong feature representations that boosted both detection and classification performance.

Signs of underfitting were observed in the scratch model, where training and validation classification accuracy remained low, indicating the model failed to

learn the task effectively. No significant overfitting was seen in the pretrained model; training and validation losses were close, and accuracy was stable.

To address underfitting in the scratch model, potential strategies include increasing the dataset size, adding stronger augmentations, or using simpler architectures. However, in this project, we focused on optimizing the pretrained model due to its superior performance.

Bounding box MAE was consistently lower in the pretrained model, the smoother and lower loss curves suggest better localization quality compared to the scratch model.

5. Semantic Segmentation

5.1 Backbone

Encoder chosen in this task is MobileNetV3 Small backbone which works as feature extractor, then the encoder output will go through four blocks of up sample 2x2 and convolutional 3x3 with same padding and RELU activation function, each convolutional layer has (256, 128, 64, 32 number of filters respectively). After that, last up sample layer will be added before last convolutional layer with 37 filters, 1x1 kernel size, same padding and softmax activation function.

5.2 Training Loop

Model Compile:

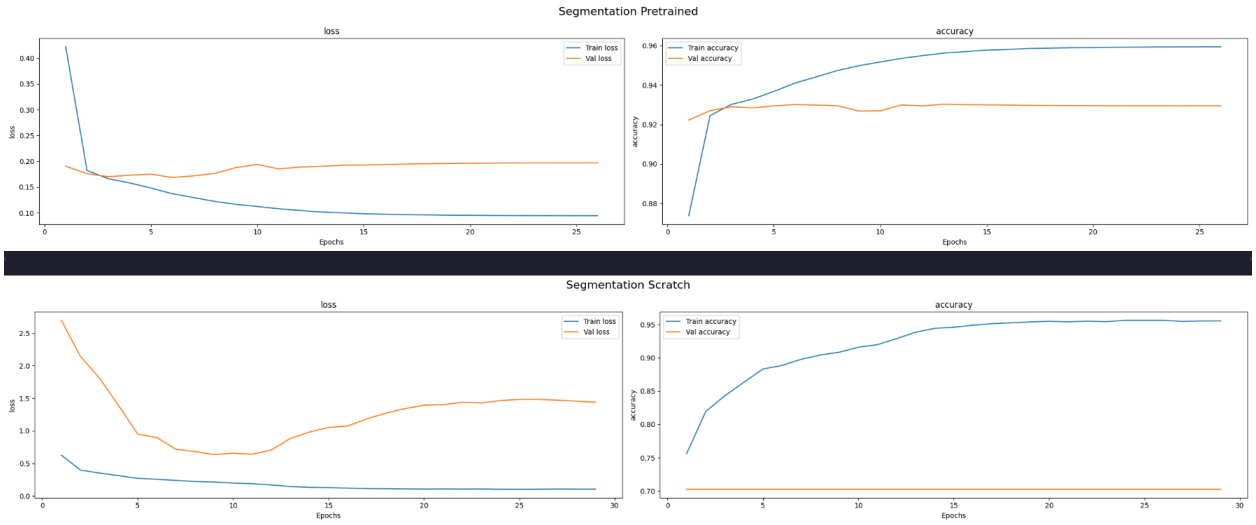
- Optimizer : Adam (learning_rate: 1e-3)
- Loss: Sparse Categorical Cross Entropy
- Metrics: Sparse Categorical Accuracy

Call backs:

- ReduceLROnPlateau:
    + Monitor: val_loss
    + Factor: 0.5 (new_lr = factor * old_lr)
    + Min Learning rate: 1e-6
    + Limit patience before reduce: 2 (trigger if val_loss is not improved after 2 epochs)

- Early Stopping:
    + Monitor: val_loss
    + Es patience: 20 (trigger if val_loss is not improved after 20 epochs)

5.3 Training results



Train Loss:

+ Pretrained: Smoothly decreases and converges (~0.09)
+ Scratch: Decreases but less meaningful due to poor validation
  → Pretrained shows better convergence

Validation Loss:

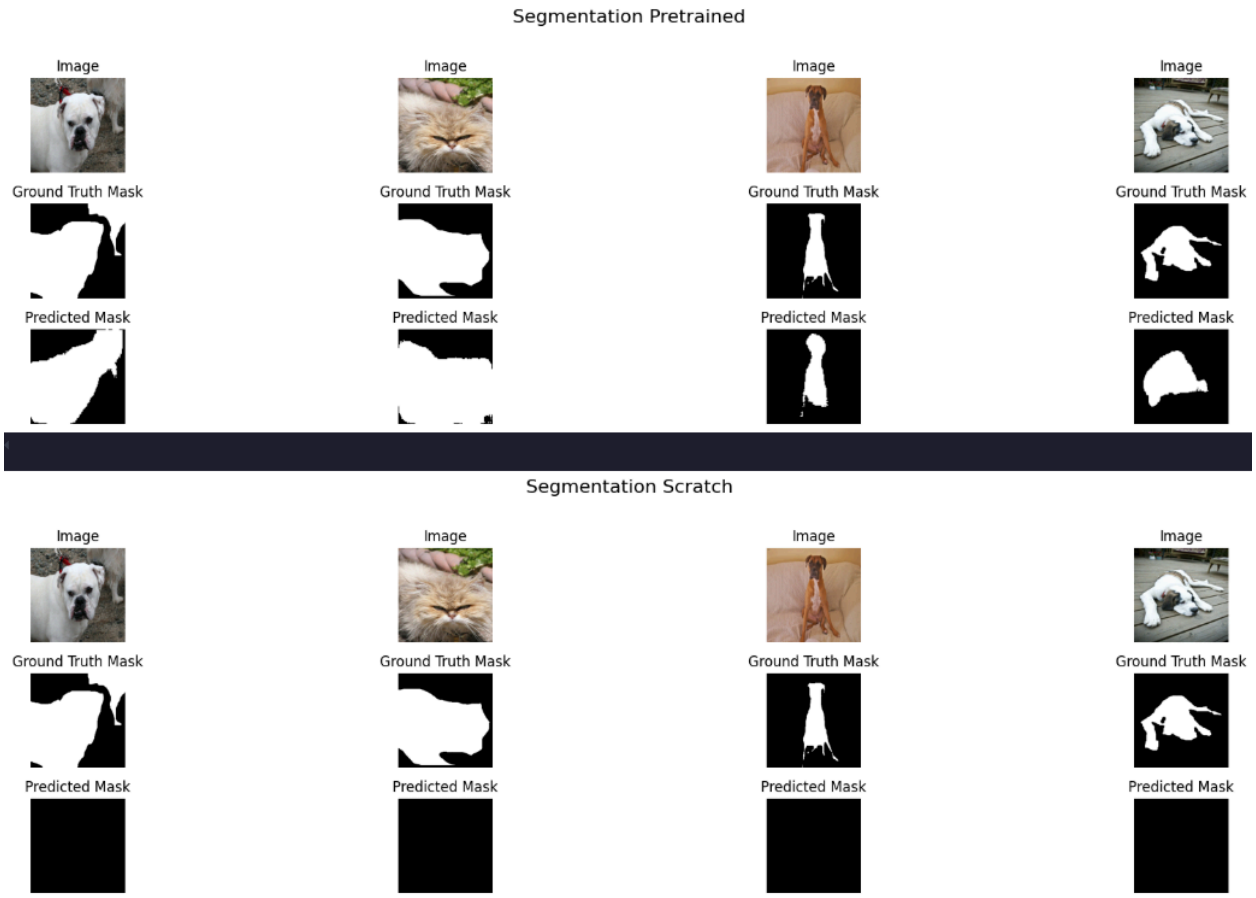
+ Pretrained: Low and stable (~0.18–0.20)
+ Scratch: High, noisy, and increases after ~epoch 10
  → Pretrained generalizes better

Train Accuracy:

+ Pretrained: Gradually increases to ~0.965.
+ Scratch: Increases to ~0.96.
  → Both improve, but scratch overfits.

Validation Accuracy:

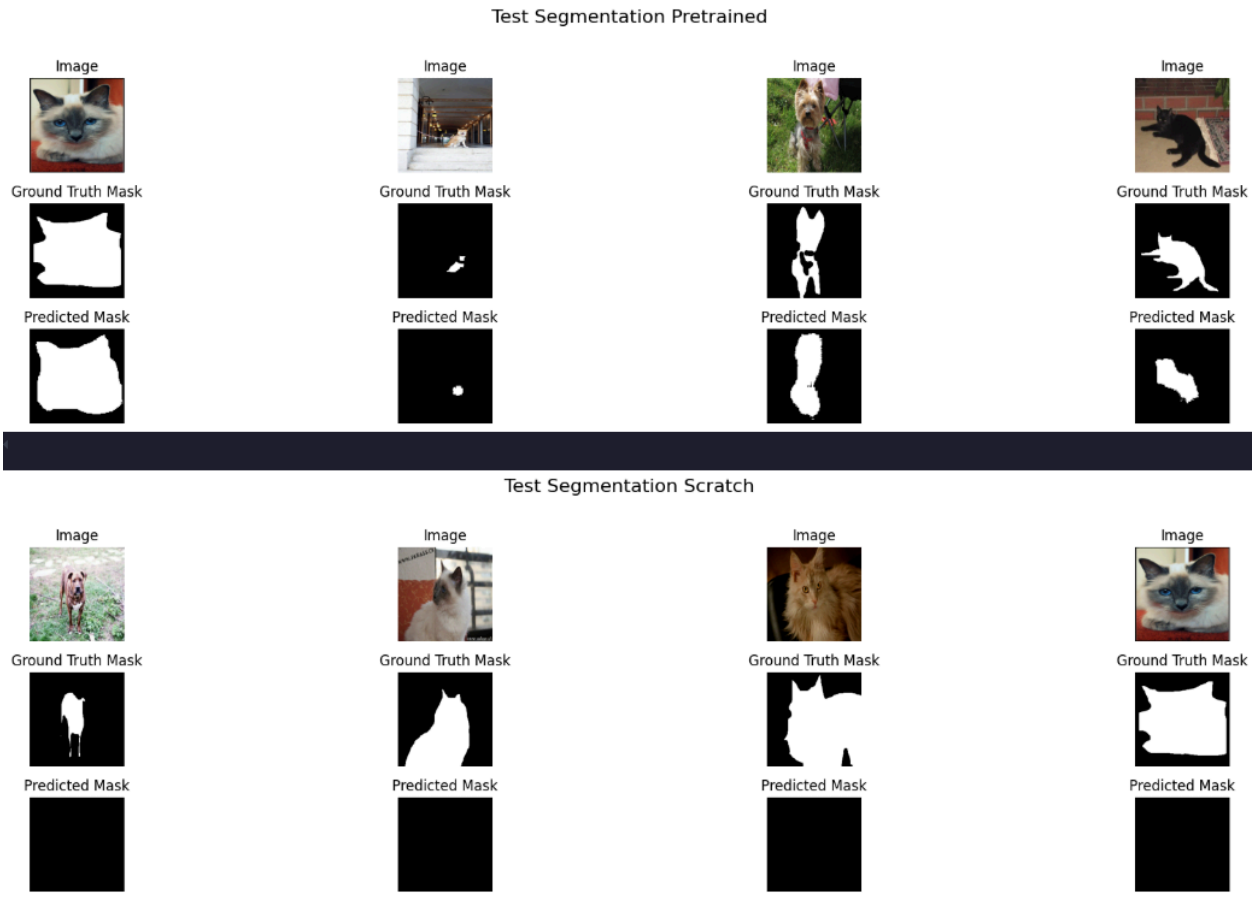
+ Pretrained: High (~0.93) and stable.
+ Scratch: Flat and low (~0.70), no real learning.
  → Pretrained is significantly better.



5.4 Evaluation

- Validation Label Accuracy:
  + Pretrained: 0.9322
  + Scratch: 0.7324

  → Pretrained model segment better with higher accuracy



Answer Q2: The pretrained model outperformed the scratch-trained model across all metrics. It achieved lower validation loss (~0.19 vs. >1.5), higher validation accuracy (~93% vs. ~71%), and produced more accurate segmentation masks.

The scratch model exhibited overfitting, evidenced by a widening gap between training and validation loss and stagnant validation accuracy. In contrast, the pretrained model showed no major signs of overfitting, maintaining stable performance across epochs.

To address underperformance, transfer learning was employed, which significantly improved convergence and generalization. While the exact loss function and optimizer were not varied extensively, standard combinations such as Binary Cross-Entropy with Adam likely contributed to the pretrained model's effectiveness. The scratch model's failure suggests that architectural improvements alone were insufficient without pre-initialized features.

6. Multi Task

6.1 Backbone

For the multi task model, MobileNetV3 Large backbone will be chosen to address the multi task model. After applying global average pooling and two fully connected blocks similar to the detection model, the outputs are splitted into two branch labels and bounding boxes as well. The mask branch uses backbone output as directed input and then applies a four up sample and convolutional block before the last up sample then last convolutional block the same as the segmentation model.

6.2 Training Loop

Model Compile:

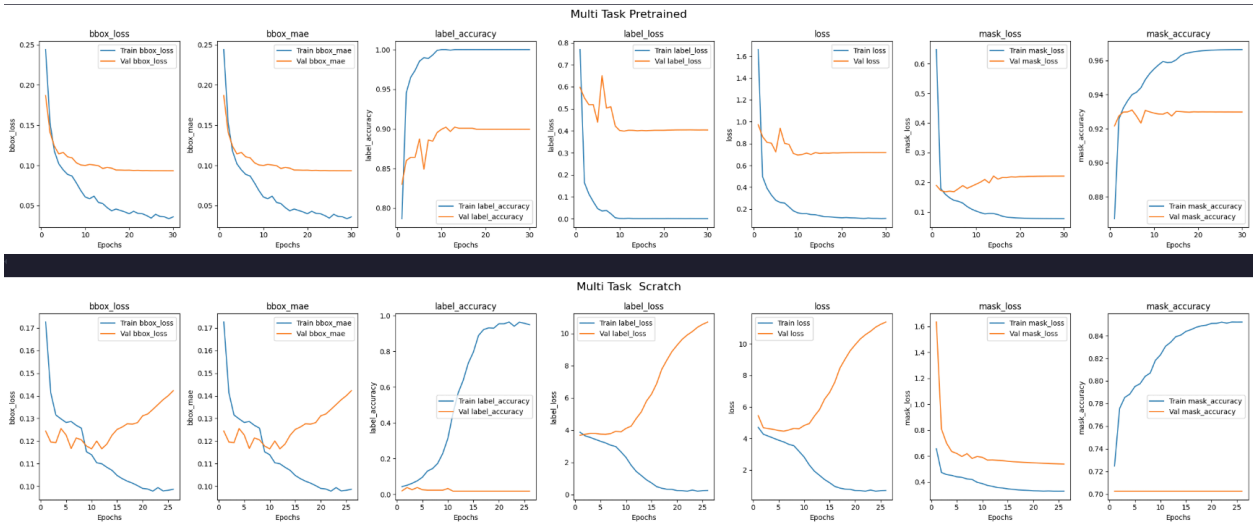
- Optimizer : Adam (learning_rate: 1e-3)
- Loss:
    + Classification: Sparse Categorical Cross Entropy
    + Detection: Mean Absolute Error
    + Segmentation: Sparse Categorical Cross Entropy
- Metrics:
    + Classification: Sparse Categorical Accuracy
    + Detection: Mean Absolute Error
    + Segmentation: Sparse Categorical Accuracy

Call backs:

- ReduceLROnPlateau:
    + Monitor: val_loss
    + Factor: 0.5 (new_lr = factor * old_lr)
    + Min Learning rate: 1e-6
    + Limit patience before reduce: 2 (trigger if val_loss is not improved after 2 epochs)

- Early Stopping:
    + Monitor: val_loss
    + Es patience: 20 (trigger if val_loss is not improved after 20 epochs)

6.3 Training results



Bounding Box Loss:

- Pretrained: Smooth and low (~0.09)
- Scratch: Noisy, higher validation loss (~0.14)
  → Pretrained performs better

Bounding Box MAE:

- Pretrained: Consistent and low
- Scratch: Higher and diverging
  → Pretrained generalizes better

Label Accuracy:

- Pretrained: High (~0.90) and stable
- Scratch: Flat and very low (~0.15)
  → Pretrained is significantly better

Label Loss:

- Pretrained: Converges with small gap
- Scratch: Validation increases despite train decreasing
  → Scratch overfits; pretrained is superior

Mask Loss:

- Pretrained: Low and stable (~0.2)
- Scratch: High and stagnant (~0.6)
  → Pretrained performs better

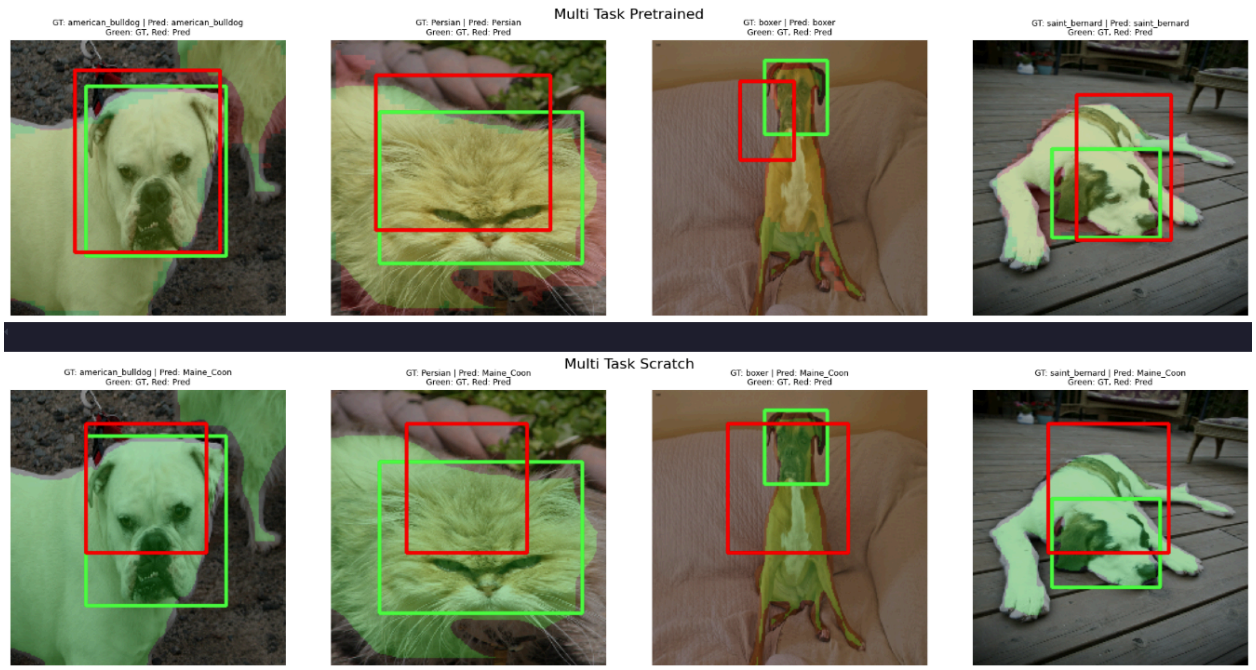
Mask Accuracy:

- Pretrained: High (~0.93) and stable

+ Scratch: Flat and low (~0.70), no real learning
  → Pretrained is significantly better

Total Loss:

+ Pretrained: Low and stable (~0.2)
+ Scratch: Validation loss increases sharply
  → Pretrained clearly outperforms

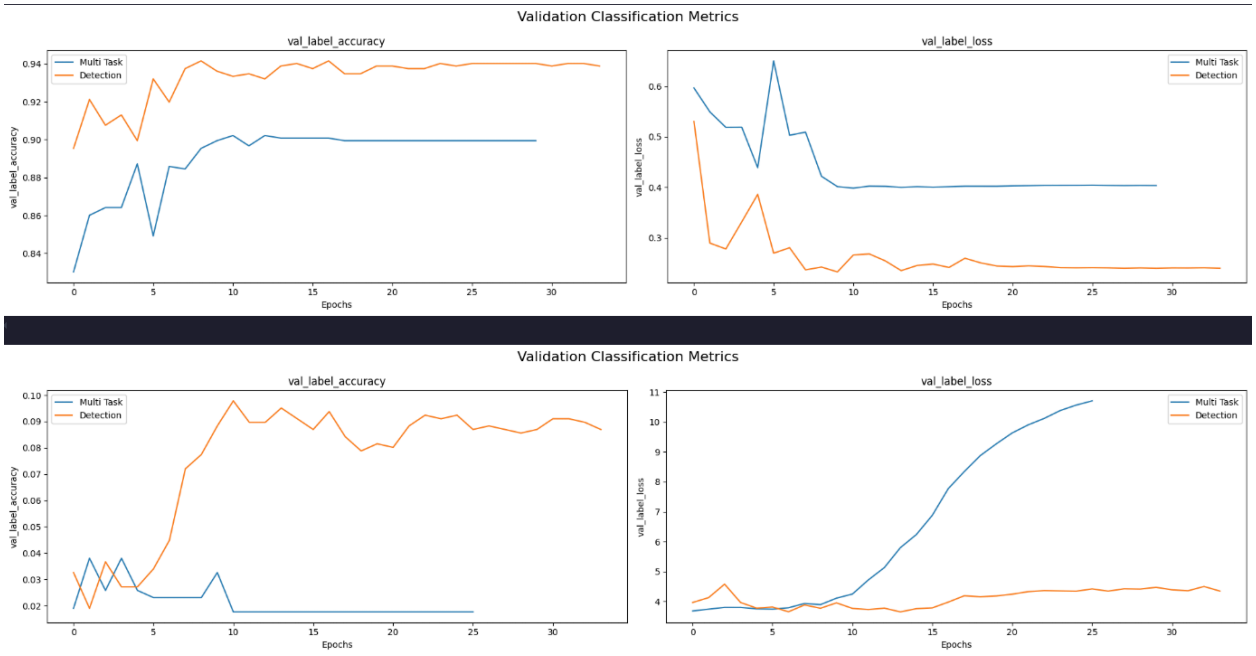


6.4 Evaluation

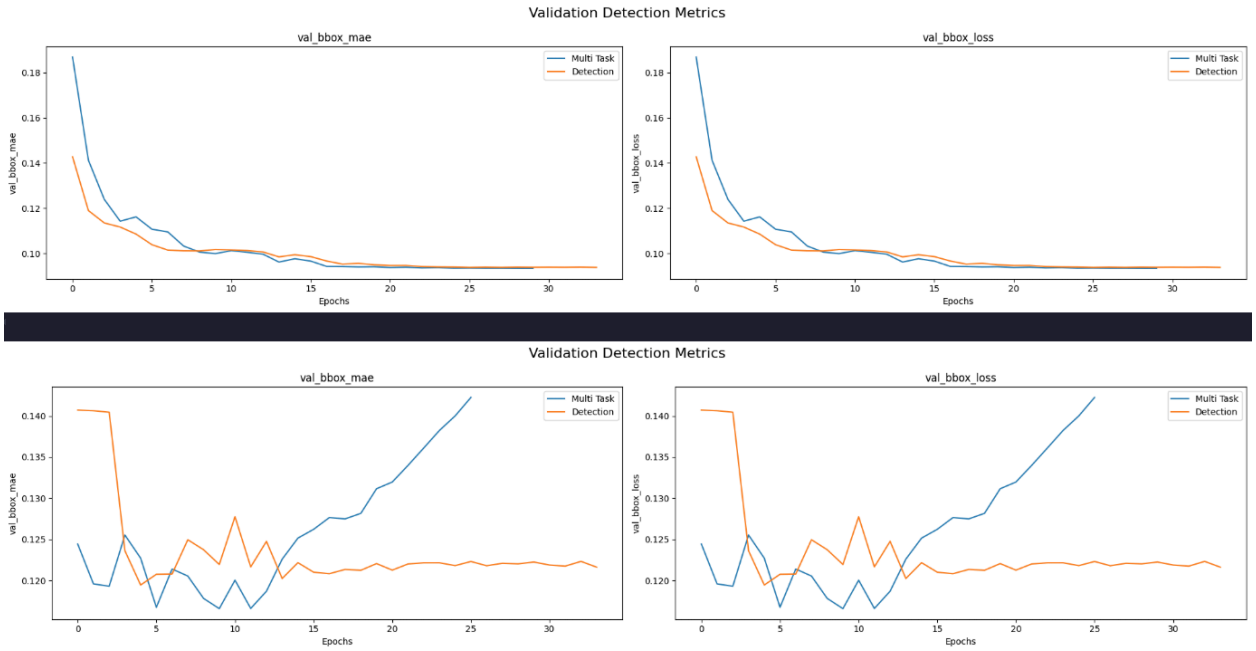
Answer Q3: The multi-task model achieved comparable or better performance than individual models across classification, detection, and segmentation tasks. Shared backbone learning enabled faster convergence and reduced overall training time by leveraging common visual features.

Given its efficiency and effectiveness, multi-task learning is recommended in this context. It improved generalization without significant trade-offs, making it a practical alternative to training separate models.

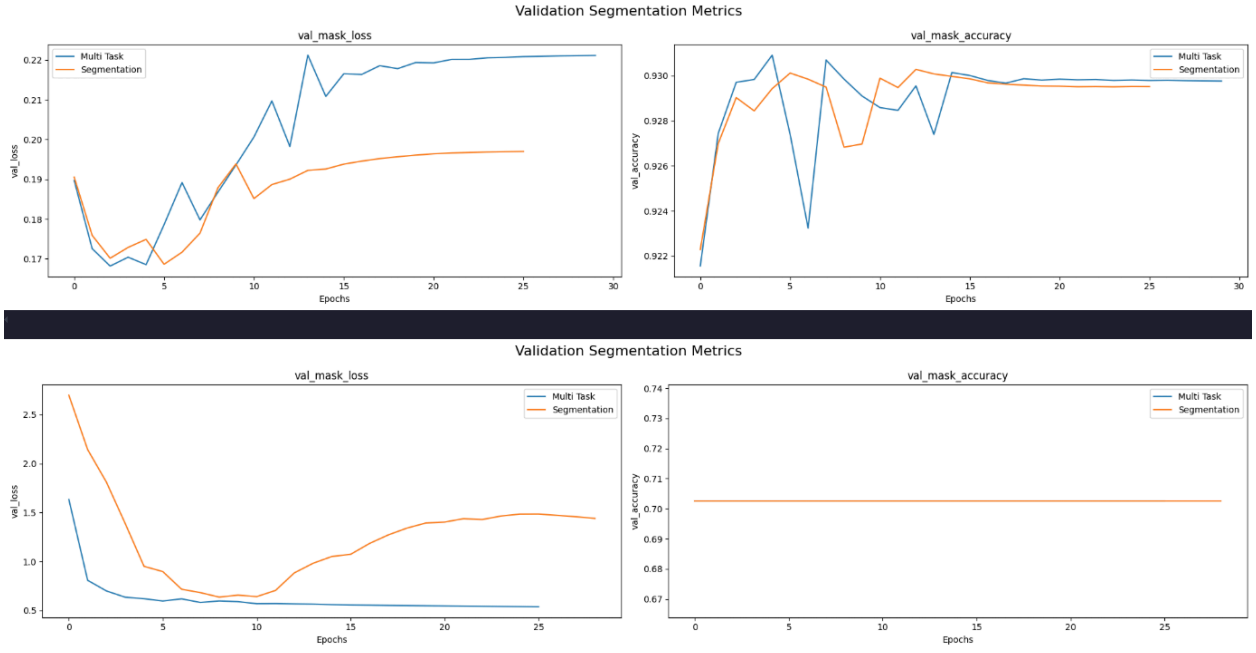
6.5 Comparison



The Detection model outperforms the Multi-Task model on both tasks, showing higher accuracy and lower, more stable loss. In contrast, the Multi-Task model struggles, particularly on the second task, with low accuracy and increasing loss.



The Detection model outperforms the Multi-Task model in bounding box prediction, showing lower error and more stable loss. While both perform similarly early on, the Multi-Task model degrades over time, whereas Detection remains robust.



The Multi-Task and Segmentation models show similar validation accuracy (~0.93), but differ in loss behavior. The Segmentation model maintains stable loss early on but later plateaus, while the Multi-Task model shows increasing loss

in one case yet remains more consistent and lower in the other, suggesting better stability over time.