CSEN342 - Program 2 (Object Detection)

Aditya Puranik
aspuranik@scu.edu
7700005906 - Rank 4, mAP Score 0.7486
Professor: Dr. David Anastasiu

I. Introduction

The program investigates the development of models for object detection, classification, and localization within images. We focus on three specific object categories: cars, medium trucks, and large trucks, labeled as 1-3 respectively.Our approach relies on fine-tuning pre-trained convolutional neural networks (CNNs) such as SSD and Yolo, which were originally developed for broader object detection tasks. We tailor these models to recognize our particular object classes.

A dataset with 14,000 photos, and 2,000 validation images — all labeled with item class and bounding box coordinates—is used in the training. The challenge is to process 2000 test images of size 960x540 to predict object class, relative bounding box coordinates, and confidence score for each object in the test set.

II. Methodology

To select the most suitable model for our transfer learning task, we evaluated various options including Faster RCNN, SSD, YOLOv5, and ultimately YOLOv8. YOLOv8 emerged as the preferred choice due to its superior performance as the current state-of-the-art model in object detection and its faster inference speed.

II-A. *Data Pre-processing*

Our training process utilized 14,000 images for training and 2,000 for validation. To accommodate the YOLOv8 model's requirements, we reorganized the dataset folder structure. While the original structure held all image IDs within a single "labels.txt"file, YOLOv8 requires separate ïmages.ªnd "labels"subdirectories within each dataset split (train, test, val). The test set, lacking labels, only utilizes the ïmages"subdirectory. Additionally, each image requires a corresponding label file with the same name for accurate image-label association. This restructuring ensured compatibility with YOLOv8's workflow.

This document corresponds to the final submission of the CSEN342 Program 2 submitted at Santa Clara Univeristy, CA, Winter'24

II-B. *0-indexing and Normalization*

Image ID:	`image_id`
Class:	c
Center Coordinates:	(cx, cy)
Width:	w
Height:	h

We had to modify the class labels in our dataset in order to comply with YOLOv8's requirements, which rely on a 0-indexed base for precise prediction. Originally designated as 1 for cars, 2 for medium trucks, and 3 for large trucks, these labels were recalibrated to align with the model's requirements, thereby ensuring compatibility and optimizing the model's predictive performance.

The initial label format employed absolute pixel values for bounding box coordinates (cx, cy) and dimensions (w, h). However, the YOLOv8 model necessitates relative coordinates for compatibility and image resizing flexibility. The YOLO model requires the relative position for these values so that it is easier to re-size the image according to the preferred size. This restructuring aligned the label format with YOLOv8's requirements and facilitated efficient image processing during training.

III. Proposed Solution

III-A. *YOLOv8 model selection hyperparameter tuning*

Given YOLOv8's availability in various sizes (nano, small, medium, large, and extra large), we investigated model selection based on our task's requirements. As our training dataset size was relatively smaller compared to some computer vision tasks, we opted to experiment with models ranging from nano to medium. Each model was trained for 75 epochs, and the following inference time and mAP50 values were obtained:

Tabla I: YOLO v8 model overview

Model	mAP (Validation Set)	Inference Time (in ms)
Nano	0.447	0.6
Small	0.583	1.4
Medium	0.663	2.0

To determine the optimal training duration, we evaluated different epoch counts within the range of 50, 75, and 100. Our analysis revealed that loss functions (box, cls, and dfl) exhibited saturation behavior around the 65-epoch mark. Based on this observation, we chose 75 epochs as the training duration for subsequent experiments.

We aimed to explore further hyperparameter tuning using the Ray Tune library from Ultralytics. This library provides tools to optimize internal loss function parameters like weight decay, loss weights, mosaic augmentation, and learning rate schedules. While we attempted to refine our best model with Ray Tune, resource limitations resulted in excessively lengthy training times even for individual 10-epoch iterations. Consequently, we opted to forgo further tuning with this method.

III-B. *Transfer learning with frozen layers*

Our objective centered on leveraging transfer learning for efficient re-training of the model on new data, avoiding the computationally expensive process of retraining the entire network from scratch. This required **freezing a portion of the initial weights, effectively preserving the learned features from the pre-trained model**. The remaining weights were then used for computing loss and subsequent updates by the optimizer. This strategy resulted in reduced resource requirements and faster training times, albeit with the potential trade-off of slightly lower final accuracy compared to full re-training.

We froze the **initial 10 layers of our model**. Employing the frozen layer strategy significantly reduced the average training time for the **YOLOv8 Medium** model, dropping it from approximately **3:50 minutes to 2:05 minutes**. This substantial improvement aligns perfectly with our objective of attaining faster training. Notably, this optimization did not result in a dramatic decline in mAP, further demonstrating the effectiveness of our approach.

IV. Simulations and tests

IV-A. *Loss Function*

The presented graphs effectively summarize the training and validation performance of the object detection model across various epochs. They highlight a clear trend of improvement, with both box and classification losses exhibiting a significant decrease. This reduction indicates an enhanced ability of the model to precisely locate and classify objects within the images.

Despite some fluctuations, the precision and recall metrics demonstrate the model's consistent detection capabilities. Furthermore, the mean Average Precision (mAP) for Intersection over Union (IoU) thresholds between 0.50 and 0.95 stabilizes after an initial period of adjustment. This stability signifies the model's increasing reliability in detecting objects with varying degrees of overlap accuracy. We can also see that after around 60 epochs, the mAP value doesn't improve much and is on a declining trend, which suggests that training should be stopped.

In conclusion, these combined metrics paint a picture of a maturing model that is progressively learning to effectively distinguish and localize the three targeted object classes within the provided dataset.

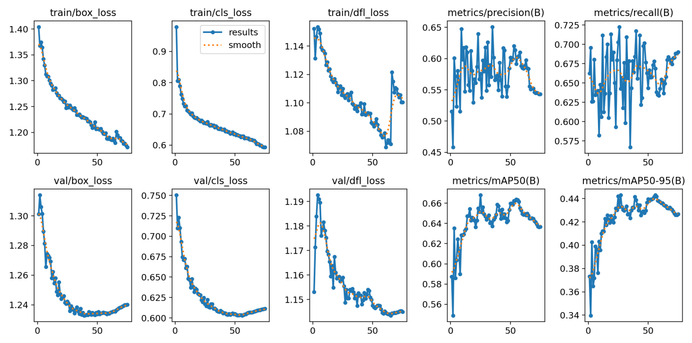

Figura 1: Loss function Plots and mAP Plots

IV-B. *Confusion Matrix*

The normalized confusion matrix provides insights into the classification efficacy of the model for three vehicle types against the background. High accuracy is observed for cars with a correct classification rate of 0.91. Medium trucks are accurately classified 55 % of the time, but a significant overlap in confusion with both cars and background suggests room for improvement.

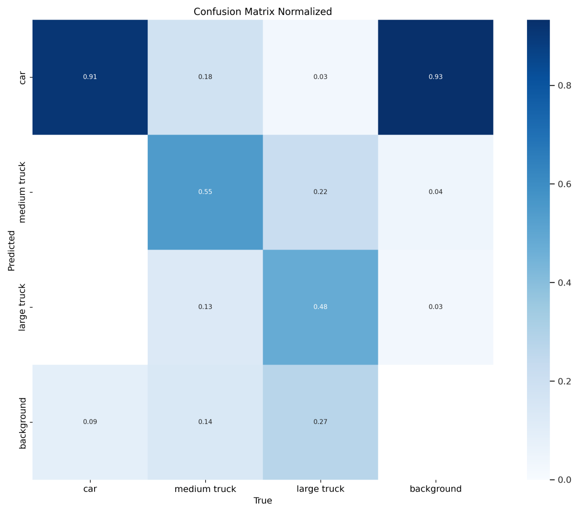

Figura 2: Confusion Matrix Normalized

V. Results & Conclusion

V-A. *Results*

The mAP50 score on the validation set come out to **0.663**, while the mAP95 score for the validation set was around **0.553**.

Tabla II: Validation Set mAP

Model	mAP50	mAP95
Cars	0.899	0.577
Medium Truck	0.551	0.379
Large Truck	0.54	0.373
All	0.663	0.443

Our transfer-learned model achieved an impressive **mAP** of *0.7486* on 50 % of test dataset, which could be improved further with lesser resource constraints and more hyperparameter tuning.

V-B. Conclusion

The **mAP score** (on 50 % data) is **0.7486 %** on the test data using our developed model. The bias-variance analysis of our object detection model's training unveils nuanced performance variations across different object classes. For cars and the background category, the model exhibits low bias, reflected in high accuracy rates. This suggests a strong understanding of the key features crucial for accurate predictions in these classes. However, for medium trucks, the model shows higher variance, evident in lower accuracy and frequent misclassifications with other classes.

While the loss curves from both training and validation phases display synchronous decreases, indicating effective learning without significant overfitting, the observed fluctuations in precision and recall across epochs hint at inconsistencies in the model's predictive power, particularly for medium trucks. This highlights the need for improved generalization capabilities. Potential solutions might involve exploring enhanced data representation techniques or refined model adjustments to mitigate the observed variance and bolster the model's robustness in classifying medium trucks.