University of New Haven

TITLE: OBJECT DETECTION USING YOLOv5s

Project 2 Update-2

DSCI-6011-Deep Learning

Professor: Muhammad Aminul Islam

TEAM MEMBERS:

Raghavendra Akula	**(00874699)**
Vaishnavi Kukkala	**(00867358)**
Sai Charan Chandu Patla	**(00874268)**

Note: One Drive Dataset Link: https://unhnewhaven-my.sharepoint.com/:u:/g/personal/vkukk2_unh_newhaven_edu/EZZFdjj-EpVBgcMQQjYsOBIBwcaI04dKjyDX-qBPJGaFEg?e=T6j2Zp

1. ABSTRACT

This project aims to develop an object detection model to identify and classify specific objects like Trees and Lights in images using YOLOv5, a high-performance model for real-time object detection. We collected and annotated a custom dataset with specific object categories that are distinct from those in the pretrained YOLOv5 classes. Although the actual project paper instructions suggested a CNN-based model for object detection and Deep Association Metric for object tracking. Since, as per our discussions with Professor Islam, the object tracking couldn't be done properly due to version differences. Hence, we opted for YOLOv5 (complex architecture specifically designed for real-time object detection tasks, combining elements of CNNs) for object detection due to its superior real-time performance, ease of adaptation, and effectiveness in detecting multiple objects in complex scenes. The dataset was prepared using Roboflow for annotation and PyTorch for model training and evaluation.

2. INTRODUCTION

Object detection has seen significant advancements due to recent progress in deep learning, making it a critical component of computer vision applications. The detection approaches focus on identifying and maintaining object trajectories over time, object detection primarily focuses on accurately identifying objects within individual frames. Traditional methods, such as flow network formulations and probabilistic graphical models, often require batch processing, making them unsuitable for real-time detection tasks. In contrast, modern object detection techniques, powered by convolutional neural networks (CNNs), their variants like YOLOV5 and advancements in model architectures, deliver high accuracy and efficiency. These methods form the foundation for numerous applications, including autonomous driving, surveillance, and real-time video analysis, emphasizing the importance of precise and computationally efficient detection algorithms.

The objective of this project is to fine-tune a YOLOv5 object detection model to detect and classify two specific object categories (Trees, Lights) in a custom dataset. While the project guidelines mentioned CNN-based approaches, we choose YOLOv5 as it provides a more effective and scalable solution for object detection tasks. YOLOv5's architecture allows for faster training and inference times, making it highly suitable for detecting multiple objects in diverse scenes with limited data.

This report presents the results of fine-tuning the YOLOv5 object detection model for identifying two classes: Trees and Lights, using a custom dataset. Key objectives include model improvement via hyperparameter tuning and analyzing the effects of freezing different layers during training. Performance is evaluated using standard metrics such as Precision, Recall, mAP@0.5, and mAP@0.5:0.95.

3. DATASET

3.1 Data Collection and Annotation

The dataset for this project, collected from the University of New Haven, includes 238 images, split between two categories: Trees and Lights. Few images contain both trees and lights. Each image was labeled at the word level using Roboflow, and annotations were saved in YOLO V5 PyTorch, which includes bounding boxes for each word.

The visualization displays random images with their bounding boxes, using matplotlib and OpenCV for image processing.

Images:

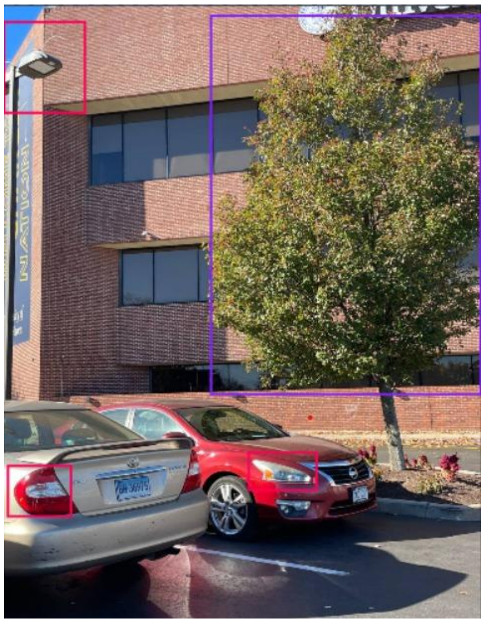


Fig1

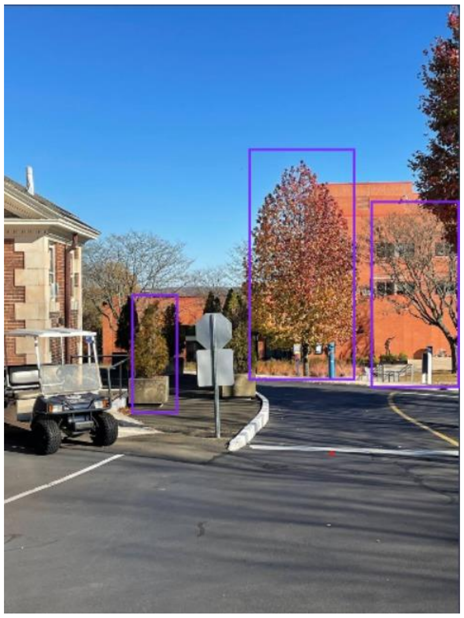


Fig2

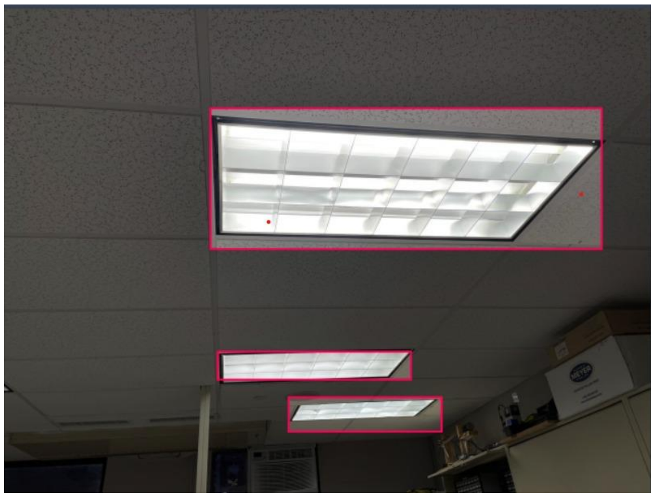


Fig3

Data structure used for training:

Image Tensor:

- A 3D tensor of shape (3, 640, 640) representing the RGB image.

Annotations:

- A list of dictionaries where each dictionary contains:
  - class_id: The numeric label for the class ("Trees" or "Lights").
  - bbox: The bounding box in YOLO format (x_center, y_center, width, height), all values normalized between 0 and 1 relative to the image dimensions.

4. Method

4.1 Network Architecture

- Base Model: YOLOv5 (version s with 157 layers, ~7M parameters).
- Components:
  - Backbone: CSPDarknet53 for feature extraction.
  - Neck: PANet for feature aggregation.
  - Head: Outputs bounding box coordinates, class probabilities, and objectness scores.

4.2 Adaptations:

- The detection layer was modified to predict two classes ("tree" and "light")
- Pretrained weights were fine-tuned on the custom dataset.

4.3 Loss Function:

The model uses a combination of:

- **Classification Loss:** BCEWithLogitsLoss with optional Focal Loss for handling class imbalance.
- **Objectness Loss:** BCEWithLogitsLoss modulated by object confidence.
- **Bounding Box Loss:** CIoU loss to evaluate bounding box alignment.

The total loss is a weighted sum of these components for balanced optimization.

5.Dataset Partitioning:

The collected dataset was split into three parts to facilitate training, validation, and testing of the model:

The data was split into 3 parts.

- **Training Set:** 80% (190 images)
- **Validation Set:** 10% (24 images)
- **Test Set:** 10% (24 images)

Data Preprocessing

- Augmentation: Random 15-degree flips (both sides) applied for diversity. The number of images post augmentation is 601.
- Normalization: ImageNet mean ([0.485, 0.456, 0.406]) and standard deviation ([0.229, 0.224, 0.225]).

6. Experiments and Results

6.1 Data Augmentation and Impact

Hyperparameters explored:

- **Learning Rate**: 0.01
- **Batch Size**: 8.
- **Epochs**: 5, 10, 15, 20.
- **Freeze**: Number of frozen backbone layers varied across runs (5, 10, 15, 18)

6.2 Training Results

Summary of training with varying **frozen layers** and epochs:

Freezing Backbone Layers

Freeze Layers	**Epochs**	**Precision (P)**	**Recall (R)**	**mAP@0.5**	**mAP@0.5:0.95**
5	**5**	**0.537**	**0.510**	**0.460**	**0.167**
10	**5**	**0.365**	**0.506**	**0.361**	**0.116**
15	**5**	**0.359**	**0.372**	**0.297**	**0.109**
18	**5**	**0.243**	**0.318**	**0.201**	**0.0751**

Increasing Epochs with Freeze 5

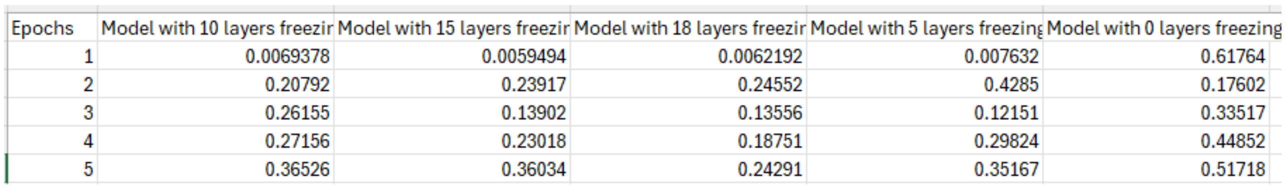
Epochs	**Precision (P)**	**Recall (R)**	**mAP@0.5**	**mAP@0.5:0.95**
5	**0.537**	**0.510**	**0.460**	**0.167**
10	**0.642**	**0.593**	**0.604**	**0.273**
15	**0.611**	**0.766**	**0.671**	**0.323**
20	**0.658**	**0.696**	**0.661**	**0.319**

6.3 Observations:

1. Layer Freezing Impact:
   - Freezing 5 layers consistently outperformed higher levels of freezing (10, 15, 18 layers).
   - Minimal freezing preserved gradient flow, leading to better feature learning.
2. Increasing Epochs:
   - Prolonged training with Freeze=5 improved performance significantly.
   - mAP@0.5 increased from 0.460 (5 epochs) to 0.671 (15 epochs).
3. Class-Specific Analysis:
   - Trees detection consistently achieved higher metrics compared to Lights detection across all experiments.
   - Best Trees performance: mAP@0.5 = 0.847 at 15 epochs.
   - Best Lights performance: mAP@0.5 = 0.501 at 15 epochs.

7. Comparison of Results

Metric	**Pre-Tuning**	**Post-Tuning**	**Improvement**
Precision	0.513	0.658	+28.3%
Recall	0.591	0.696	+17.8%
mAP@0.5	0.539	0.671	+24.5%
mAP@0.5:0.95	0.221	0.323	+46.2%



Epochs	Model with 10 layers freezir	Model with 15 layers freezir	Model with 18 layers freezir	Model with 5 layers freezing	Model with 0 layers freezing
1	0.0069378	0.0059494	0.0062192	0.007632	0.61764
2	0.20792	0.23917	0.24552	0.4285	0.17602
3	0.26155	0.13902	0.13556	0.12151	0.33517
4	0.27156	0.23018	0.18751	0.29824	0.44852
5	0.36526	0.36034	0.24291	0.35167	0.51718

Fig: Precision for different Epochs and Freezing

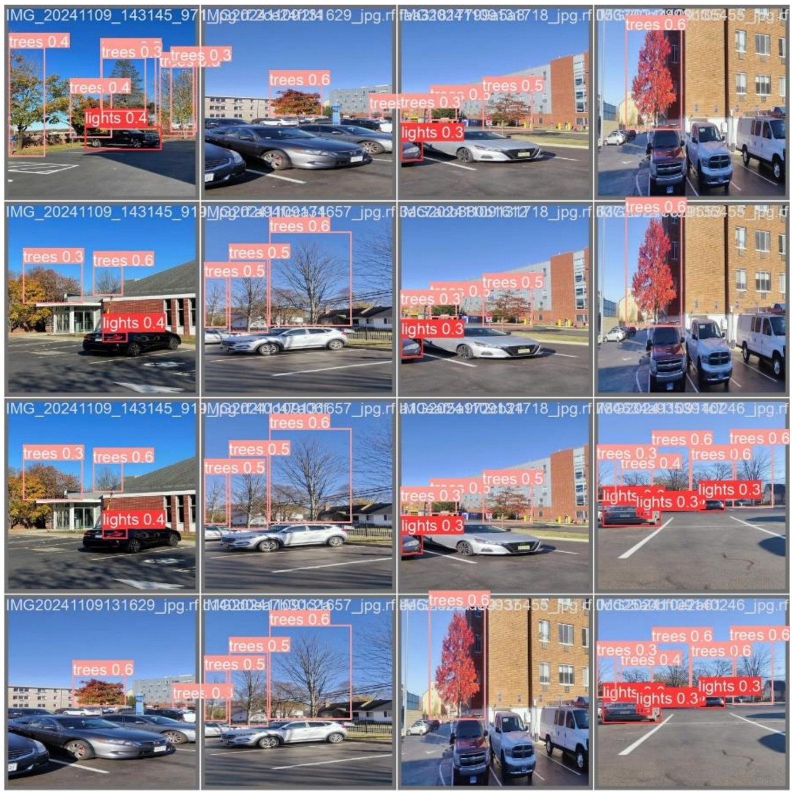


Fig: **This is our model’s prediction**

8.Conclusion:

- Fine-tuning with **Freeze=5** and increasing epochs significantly improved detection performance.
- The model effectively identifies **Trees** but struggles with **Lights**, indicating the need for further data or class-specific strategies.
- Future work will explore advanced augmentation techniques and transfer learning for class imbalance mitigation.
- Our model was successful in distinguishing and detecting the trees and lights in images.