Object Detection in Low-Light Environments Using MIRNet Image Enhancement in Conjunction with YOLO v8n Model

Ghodke Aditya Rao, Avi Mandavia

Abstract

Detection of objects in low-light images is a significant challenge due to reduced visibility and heightened noise levels. In this study, we delve into solving the issue in a way that considers all the practical constraints connected to the subject. Our strategy equips the MIRNet image enhancement model to feed enhanced images into the YOLO v8 object detection model to yield improved results. For this purpose, the study utilises NVIDIA's GPU to ensure smooth training of the heavy YOLO v8n model over the ExDark Image Dataset, the largest publicly available dataset of low-light images. The proposed methodology yields a mean average precision of 0.702 on test images by leveraging state-of-art techniques for both image enhancement as well as object detection

Keywords
Object Detection, YOLO, Image Restoration, Bounding Boxes, Precision

1 Introduction

Detecting objects in low-light images is a fundamental yet challenging task in computer vision with numerous real-world applications ranging from surveillance to autonomous driving. The ability to accurately identify objects under poor lighting conditions is important for ensuring the reliability and effectiveness of these systems. However factors such as degraded quality and minimal visibility in low-light images hamper precise object detection.

One of the primary obstacles in low-light object detection is the reduced visibility caused by insufficient illumination. Improper lighting leads to darker and noisier images, making it difficult for traditional object detection algorithms to isolate objects from the background accurately.

Moreover, low light conditions worsen the effects of noise in images, further complicating the detection process. Sensor and photon shot noises introduce unnecessary distortions, which can affect detection algorithms' ability to accurately localize objects.

Another challenge in low-light object detection is the lack of contrast between objects and their surroundings. In dimly lit environments, the contrast between objects and the background is significantly reduced, making it challenging for detection algorithms to differentiate between objects and background clutter.

Traditional approaches to tackle such challenges involve pre-processing techniques such as histogram equalization or contrast enhancement to improve image quality before applying object detection algorithms. However, these methods are not always effective, especially in extremely noisy or low contrast images.

In recent years, the development of deep learning-based approaches has improved object detection performance in low-light conditions. These approaches leverage the power of convolutional neural networks (CNNs) to learn discriminative features directly from the data to enable smooth detection of objects like in the case of well-lit images.

In this context, our study focuses on exploring the effectiveness of integrating state-of-the-art deep learning models, namely the MIRNet image enhancement model and the YOLO v8n object detection model, to address the challenges of object detection in low light images. By leveraging the capabilities of these models, we aim to enhance visibility, reduce the effects of noise, and improve object detection results in low-light environments.

2 Materials & Methods

In this section, the steps and the resources involved in object detection of low-light images are explained in detail. The steps in the procedure were - initial data preprocessing, enhancing

images using the MIRNet enhancement model, processing the data for the YOLO v8n model, training the YOLO v8n model on the enhanced images and discussing the obtained results.

2.1 Initial Data Preprocessing

2.1.1 Dataset

The study uses the ExDark Image Dataset (https://github.com/cs-chan/Exclusively-Dark-Image-Dataset) the largest publicly available image dataset comprising 7363 images with 12 object classes namely, Bicycle, Boat, Bottle, Bus, Car, Cat, Chair, Cup, Dog, Motorbike, People, and Table 1. These images are placed in their respective class folders. The individual distribution of the images can be seen in the table. The dataset has 10 varying low-light conditions like Low, Ambient, Object, Single, Weak, Strong, Window, Screen, Shadow, and Twilight. All of the images vary in size. The annotations of these images are included in a separate file known as the Groundtruth. It comprises 12 class folders, similar to the image file. Each folder contains '.txt' files of annotations corresponding to their respective images. the '.txt' files are of the format:

1. **First 16 characters:** Annotation tool data

2. **1st column:** Object class name

3. **2nd-5th column:** Bounding box coordinates [l t w h] where l is the pixel number from the left of the image, t is the pixel number from the top of the image, w is the width of the bounding box and h is the height of bounding box.

4. **6th-12th column:** occlusion and orientation annotation

5. For images with more than 1 object annotated, (2)-(4) is repeated.

Class	No. of Images
Bicycle	652
Boat	679
Bottle	547
Bus	527
Car	638
Cat	735
Chair	648
Cup	519
Dog	801
Motorbike	503
People	609
Table	505

Table 1: Class-wise Distribution of images

2.1.2 Altering Annotation Files for Compatibility with YOLO V8n Model

Several changes were made to the annotation files to guarantee that the ExDark image collection worked with the YOLO v8n object detection model. These adjustments were made to simplify the dataset's format and guarantee a smooth integration with the YOLO v8n architecture.

- **Class Label Conversion:** The first column of annotation files, which previously included the names of object classes (e.g., Bicycle, Boat), has been turned into numeric labels. Each class was assigned a unique integer identifier ranging from 0 to 11. This translation increased the model's ability to read and classify data using numerical representations.

- **Removal of Occlusion Information:** Columns 6–12 of the annotation files, which contained information about occlusions in the pictures, were removed. While occlusion information may be important in certain object detection tasks, it was unnecessary for the objectives of this study. Removing these columns streamlined the annotation files and reduced unnecessary complexity.

- **Normalisation of Bounding Box Coordinates:** The bounding box coordinates (columns 2–5) were normalized to ensure consistency and compatibility with the YOLO v8n model. Normalization involved scaling the coordinates to fall inside the range [0, 1], to ensure that they were insensitive to changes in image dimensions that may occur during further steps.

2.1.3 Preprocessing for MIRNet Image Enhancement Model

Before feeding images into the MIRNet image enhancement model, several preparation steps were undertaken to ensure compatibility and optimal performance. The following stages were involved in the process:

- **Conversion to RGB Format:** Initially, images were checked and converted to the

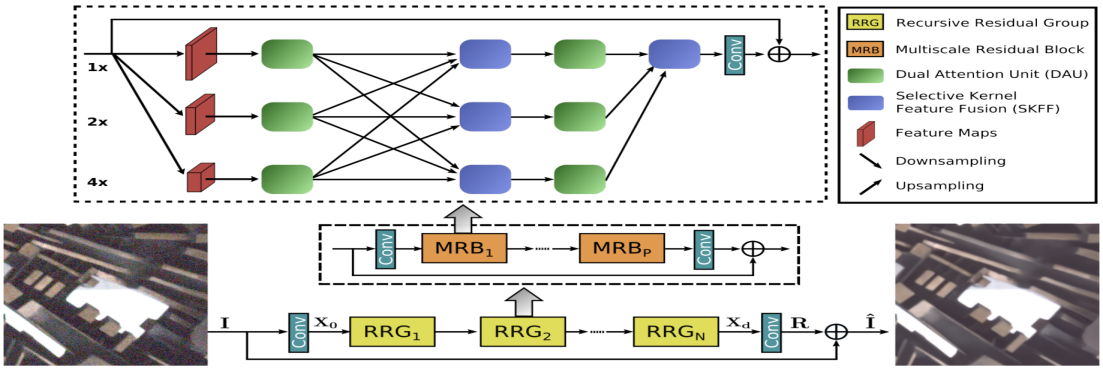

Figure 1: **MIRNet Architecture**

RGB format if they weren't already in that colour format. This step ensured that the MIRNet model handled the images consistently by standardizing the colour representation across all images.

- **Resizing Images**For the MIRNet model to work, the dimensions of the input photos must be multiples of 4. Every image was resized to 512 × 512 pixels in resolution in order to meet this requirement.

- **Conversion to Array**Following the scaling of the images, numerical arrays were used to represent the pixel values. This conversion allowed the MIRNet model to process the images as numerical data.

- **Normalisation of Pixel Values**The pixel values in the image arrays were normalised between [0, 1]. Normalization improves convergence during the optimization stage and prepares the data for model training by standardizing the pixel values.

- **Adding Batch Dimension**The final step was adding a new dimension to the image arrays to enable batching. The MIRNet model can understand several pictures simultaneously due to this dimension, which boosts throughput and computational efficiency for both training and inference.

2.2 MIRNet Image Enhancement Model

Image restoration is essential in many domains where the objective is to restore high-quality image content from the downgraded like computational photography, surveillance, and remote sensing, etc. In recent years, convolutional neural networks (CNNs) have grown to be incredibly powerful tools, outperforming conventional techniques in image restoration tasks by a significant margin. In present CNN-based methods, contextual robustness and spatial precision are traded off. While working on lower-resolution images, the model produces outputs that are less accurate in space but semantically dependable. On the other hand, while working on full-resolution images, the model produces spatially intact outputs but contextually less accurate.

But MIRNet (Multi-scale Image Restoration Network) achieves a trade-off balance by maintaining spatially precise high-resolution representations throughout the network while leveraging significant contextual information from low-resolution representations. The multi-scale residual block, a fundamental component of MIRNet, consists of several significant parts:

- Convolution streams working in parallel at many resolutions to extract features at various scales.

- Information-sharing mechanisms that facilitate communication between the various resolution streams.

- Contextual information is captured by both spatial and channel attention processes.

- Attention-based multi-scale feature aggregation.

In essence, MIRNet acquires an augmented feature set by combining contextual data from many scales and preserving high-resolution spatial details. The architecture of MIRNet can be understood better with the image shown in Fig 1.

In our study, the images produced by MIRNet turned out to be significantly better than those produced by simply auto-contrasting the image as MIRNet takes into account all the spatial context rather than simply contrasting the

Figure 2: Original Image

Figure 3: Auto Contrasted Image

Figure 4: **Enhanced Image using MIRNet Model**

lowly illuminated parts. Comparison of the original, autocontrasted and the MIRNet produced images is shown in Fig 2, 3 and 4 respectively.

2.3 YOLO V8N Model

2.3.1 Preprocessing and Setup for YOLO

Several post-processing steps were taken to prepare the enhanced images for integration into the YOLO v8n model after implementing the MIRNet model on the image arrays.

Denormalization and Clipping

- The output image arrays from MIRNet were denormalized by multiplying their pixel values by 255 to restore them to their original scale.
- Later, pixel values were reduced to the permitted range of 0 to 255. Values greater than 255 were restricted to 255 and values less than 0 were set to 0 to preserve image integrity.

Conversion to Image and Reshaping

- The clipped pixel values in the image array were then translated back into image format to allow further processing.
- To comply with the input dimension criteria of the YOLO v8n model, the image size was converted into 640 × 640.

Dataset Partitioning

- The images were then partitioned into training, validation and testing subsets with 80%, 15% and 5% ratio.
- Paths to the training and validation images were meticulously recorded in a YAML file, along with corresponding class labels and their assigned numerical identifiers (0, 1, 2, ..., 11).
- The paths of training and validation folders were included in the YAML file along with the numbers mapped to each classes (0,1,2....11), so that the YOLO could be trained using the YAML file.

Thereafter, NVIDIA GPUs were configured for the YOLO v8n model for faster and smoother execution of the model.

2.3.2 Model and Execution

The You Only Look Once (YOLO) model is notable for its efficiency, accuracy, and real-time

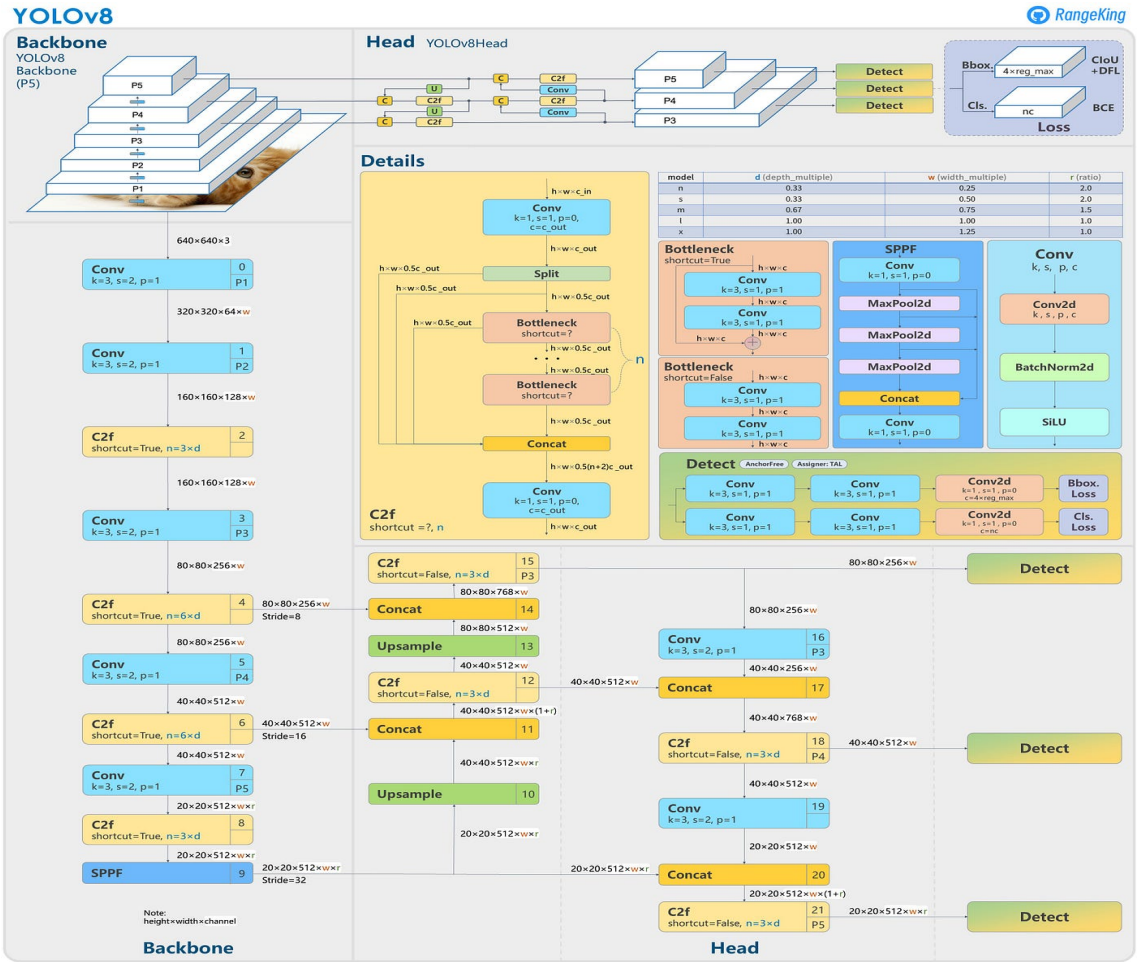

Figure 5: **YOLOv8 Architecture**

capabilities, making it one of the most innovative approaches of object detection. Bounding boxes and class probabilities for items in an image are predicted by a single neural network in a single pass with YOLO's single-stage approach.

The backbone network of the YOLO design is a Convolutional Neural Network (CNN), like Darknet or ResNet. Its task is to analyze the input image and derive important high-level features for object detection. These features are then input into the prediction head, which is composed of convolutional layers and uses these properties to forecast bounding boxes and class probabilities for objects in the image. The input image is represented by a grid of cells that YOLO utilizes to forecast bounding boxes for each cell, class probabilities for each category of objects, and confidence scores that indicate the likelihood of an object being present.

1. **Input Processing**:

    - To meet the input requirements of the model, the input image is scaled and normalized as needed.

2. **Feature Extraction**:

    - The backbone network extracts high-level features relevant to object detection from the pre-processed image. These features incorporate object characteristics and spatial information from the provided image.

3. **Grid Cell Division**:

    - YOLO divides the input image into a grid of cells. Each cell is responsible for predicting bounding boxes and class probabilities for objects localized within its spatial region.

4. **Bounding Box Prediction**:

    - The model makes multiple bounding box predictions based on the center coordinates, width, height, confidence score, and class probabilities for each grid cell. Class probabilities indicate the likelihood that an object will fall into a particular class, while the confidence score indicates the chance that

an object will be inside a bounding box.

5. **Non-Maximum Suppression**:
   - To cut down on redundant detections, overlapping bounding boxes with lower confidence scores are eliminated using non-maximum suppression. This ensures that only the most certain and non-overlapping bounding boxes are retained.
6. **Output Generation**:
   - The output of the YOLO model is a set of bounding boxes, each coupled with a confidence score and class probabilities. These are the detected objects in the input image.

In January 2023, after numerous updates to the YOLO model, YOLO v8 was released to produce improved object detection results.

- **CSPDarknet-AA Backbone Architecture**:
  - Yolo v8n uses the CSPDarknet-AA backbone architecture, a sophisticated variation of the CSPDarknet series. This architecture improves detection accuracy by enhancing feature extraction capabilities.
- **Multi-Scale Object Detection**:
  - Yolo v8n's built-in multi-scale object recognition feature enables the model to recognize objects of various sizes in an image. This technique enhances the model's robustness and adaptability, leading to more accurate detections over a range of object scales.
- **ELU Activation Function**:
  - YOLO v8n uses the Exponential Linear Unit (ELU) activation function to speed up learning by resolving the vanishing gradient problem.
- **GIoU Loss Function**:
  - Yolo v8n employs an enhanced version of the Intersection over Union (IoU) metric, known as the Generalized Intersection over Union (GIoU) loss function. GIoU improves prediction and object localization accuracy by considering bounding box shape and size.
- **Performance Improvements**:
  - Yolo v8n shows a remarkable 1.2% increase in average precision (AP) over Yolo v7. Additionally, YOLO v8n reduces the model's weight file size by 80.6 M, increasing deployment efficiency and feasibility in environments with limited resources.

The architecture of the YOLO model can be better understood with the image in Fig 5. Our study feeds the YOLO v8n model with a batch size of 32 and runs 50 epochs of the model.

3 Results and Discussion

This section covers both general object detection assessment metrics and metrics specific to assessing the performance of YOLO v8.

Fundamental Evaluation Metrics

- **Intersection over Union (IoU):** The idea of Intersection over Union, or IoU, quantifies the amount of overlap between a ground truth bounding box and a predicted bounding box.

$$IoU = \frac{Area_of_Intersection}{Area_of_Union} \tag{1}$$

- **Average Precision (AP):** By calculating the area under the precision-recall curve, AP provides an overview of the model's precision and recall performance across a range of confidence criteria. It provides a single number that represents the model's total detection accuracy.

$$AP = \int_0^1 \text{precision}(r)\, dr \tag{2}$$

- **Mean Average Precision (mAP):** mAP builds upon AP by averaging the AP values across multiple object classes. It is particularly useful when dealing with many object identification classes because it offers a comprehensive evaluation of the model's performance across various categories.
- **Precision and Recall:** By determining the proportion of true positives among all positive predictions, a model's precision can be used to assess its ability to avoid false positives. Recall measures how well the model can identify each and every instance of a given class.

$$Precision = \frac{True_Positives}{True_Positives + False_Positives} \tag{3}$$

Class	Instances	P	R	mAP50	mAP50-95
all	3416	0.77	0.607	0.702	0.442
bicycle	168	0.849	0.702	0.807	0.546
boat	204	0.778	0.636	0.721	0.373
bottle	188	0.797	0.596	0.667	0.418
bus	102	0.917	0.804	0.904	0.697
car	467	0.807	0.615	0.723	0.46
cat	145	0.718	0.6	0.698	0.412
chair	363	0.741	0.501	0.607	0.371
cup	206	0.77	0.537	0.636	0.399
dog	119	0.536	0.65	0.666	0.432
motorbike	170	0.762	0.585	0.696	0.416
people	1043	0.807	0.577	0.694	0.394
table	241	0.756	0.481	0.599	0.389

Table 2: Final Results of YOLO v8n on Validation Set

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

- **F1 Score:** The F1 Score is the harmonic mean of precision and recall, providing a balanced assessment of the model's performance while considering both false positives and false negatives.

$$F1_Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{5}$$

Metrics Specific to YOLO v8

The following are the metrics specific to the YOLO v8n model:

- **Class:** Denotes the name of the object class, such as "bottle", "bus", or "car".
- **Images:** Indicates the number of images in the validation set that contain the object class.
- **Instances:** Provides the count that the class appears across all images in the validation set.
- **Box (P, R, mAP50, mAP50-95):**
  - **P (Precision):** Reflects the precision of the particular class.
  - **R (Recall):** Reflects the recall of the particular cass.
  - **mAP50:** Mean average precision calculated at an IoU threshold of 0.50. (0.50 is a lesser strict threshold)
  - **mAP50-95:** Average of mean average precision calculated at IoU thresholds ranging from 0.50 to 0.95, providing insights into the model's performance across varying levels of detection difficulty.

In our case,we have trained the model with a batch size of 32, running for a total of 50 epochs on the NVIDIA GPU. The training process took approximately 6.5 hours to complete. We have a validation set that comprises a total of 1104 images. The values of above-described metrics specific to YOLO v8n of our experiment are shown in Table 2.

From the Table, we can see that the **Bus** class has the maximum mAP50 of **0.904**. This could be due to only 102 instances of bus present in the dataset, the least of them all. Also, the model has yielded decent mAP50 values indicating that the model has been able to produce a greater ratio of correct positive predictions for the IOU threshold of 0.5. The mAP50 of all classes is **0.702**.

The precision-recall curve for threshold 0.5 shown in Fig 6 can be referred to for comparing the metric curves for all object classses. It is also evident from the curve that the Bus object encapsulates the largest area under its curve, thus yielding maximum mAP50 value.

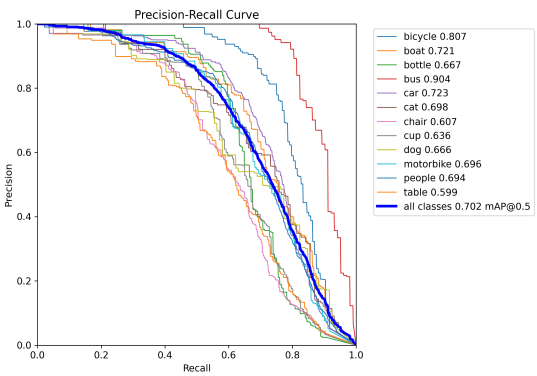

Figure 6: **Precision-Recall Curve**

Precision and Recalls individually, against the range of all threshold values from 0 to 1 can be

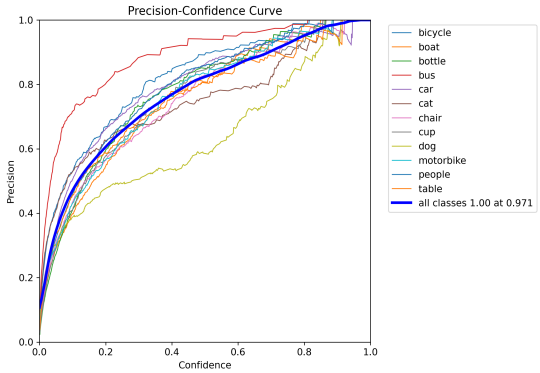

Figure 7: **Precision-Confidence Curve**

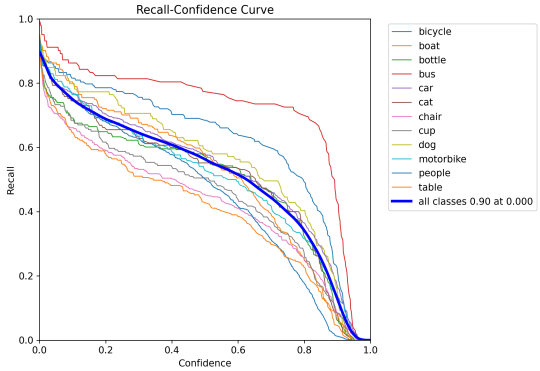

Figure 8: **Recall-Confidence Curve**

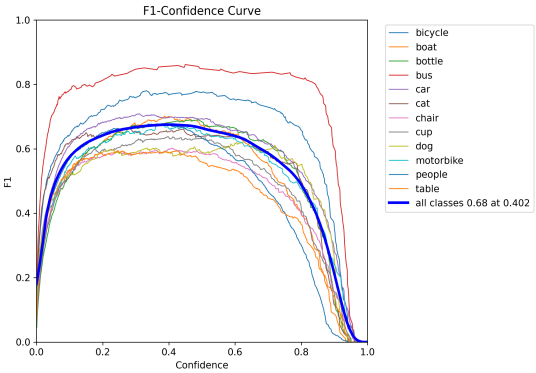

Figure 9: **F1-Confidence Curve**

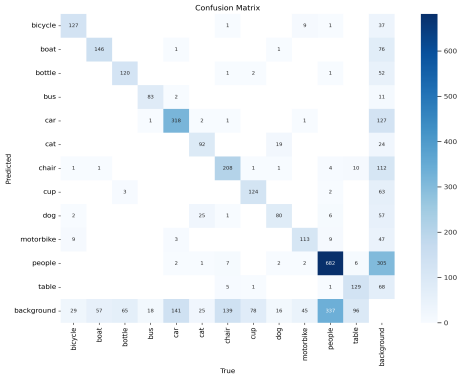

Figure 10: **Confusion Matrix**

viewed in Fig 7 and Fig 8 respectively. For better understanding of the number of positives and negatives, the confusion matrix in Fig 10 can be read. From the confusion matrix, we can observe how in the "Bus" row, there are only 11 objects detected as "background" as opposed to the large number of objects detected as "background" for other classes. The F1 curve vs confidence threshold graph is shown in Fig 9.

The variation of the mAP50 and mAP50-95 metrics with epochs is shown in Fig 11.

Conclusions

Our findings infer that YOLO v8n model has the greatest mean Average Precision (mAP50) of 0.904 when it comes to classifying "Bus" objects due to the minimum number of instances of that class in the validation set. The model has a mAP50 of 0.702 overall and performs well across a range of object classes. A deeper understanding of the model's functionality is also possible through the examination of precision-recall curves and confusion matrices, which demonstrate the model's capacity to detect objects properly while reducing false positives.

Our results show how effective the undertaken methodology is in detecting objects in poorly lit areas. The model's flexibility to various confidence criteria and the significant improvements in precision rates have verified our approach as workable. These findings advance the field of computer vision research and have applications ranging from surveillance to autonomous driving systems.

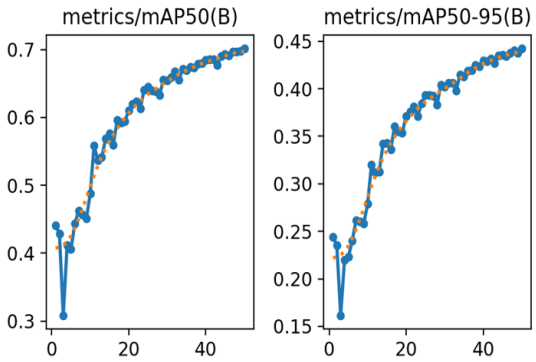

Figure 11: **mAP50 and mAP50-95 metrices**