NAME: HARSIMRAT SINGH

SID:23104086

BRANCH: ELECTRICAL ENGINEERING

SUBJECT: TRAFFIC AND PEDESTRIANS DISTINCTION USING A YOLO V8 SEGMETATION MODEL

PROJECT REPORT

Data Collection

The dataset for this project was created by sourcing traffic scene videos from Pexels, an open-access multimedia website. A selection of diverse traffic environments was chosen to ensure coverage of various scenarios involving vehicles, bikes, and pedestrians. More details of sources can be found in the source markdown file in the repository

Dataset Preparation

To transform the video sources into usable image data, a multi-step process was employed. First, frames were systematically extracted from videos sourced from platforms like Pexels and YouTube using Python scripts and the OpenCV library. A frame similarity algorithm was then applied to identify and remove duplicate or highly similar frames, which helped to improve the dataset's diversity and reduce redundancy.

Following the extraction and curation process, manual annotation was performed using the Labeller platform. During this step, each image was meticulously labelled with regions of interest, and the objects within them were classified into three distinct categories: vehicles, bikes, and pedestrians.

Finally, the annotated images were split into training and validation sets. This split was carefully executed to ensure a representative distribution of objects and scenarios, which is essential for robust and accurate model evaluation.

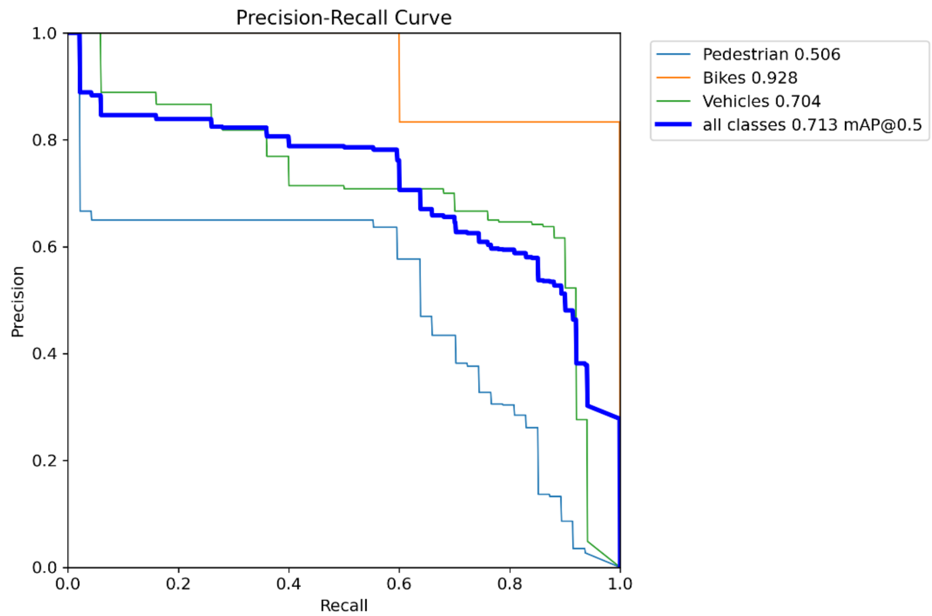
Model Training

The processed dataset was used to fine-tune the YOLOv8n segmentation model. Training was performed on Google Colab utilizing an NVIDIA T4 GPU to enable efficient, large-scale learning.
Data was organized according to the YOLO annotation format, and hyperparameters such as batch size and image resolution were adjusted for optimal performance. The model learned to detect and segment target objects, with progress monitored via performance metrics such as precision, recall, and mean average precision (mAP) for both bounding box and mask predictions.

Model Evaluation

After training, the model's effectiveness was rigorously validated on the held-out validation set. The model's performance was measured using standard object detection metrics, providing a comprehensive view of its strengths and weaknesses across different object classes: Pedestrians, Bikes, and Vehicles.

Precision-Recall Curve Analysis

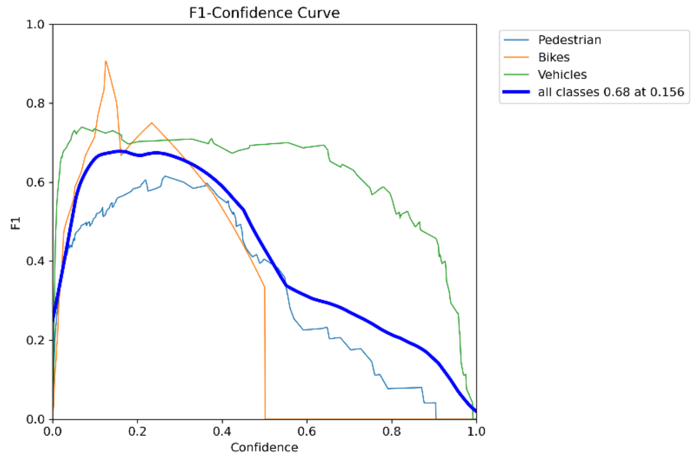


The Precision-Recall (PR) Curve provides a detailed look at the trade-off between the model's precision and recall for each class.

- Vehicles show the highest performance with a mAP of 0.704, indicating the model is highly effective at detecting vehicles.
- Bikes also perform exceptionally well, achieving a mAP of 0.928. This suggests the model is very precise and has high recall when identifying bikes.
- Pedestrians have the lowest mAP at 0.506, highlighting that this is the most challenging class for the model to detect accurately.

The all classes mAP@0.5 score is 0.713, demonstrating a strong overall performance.

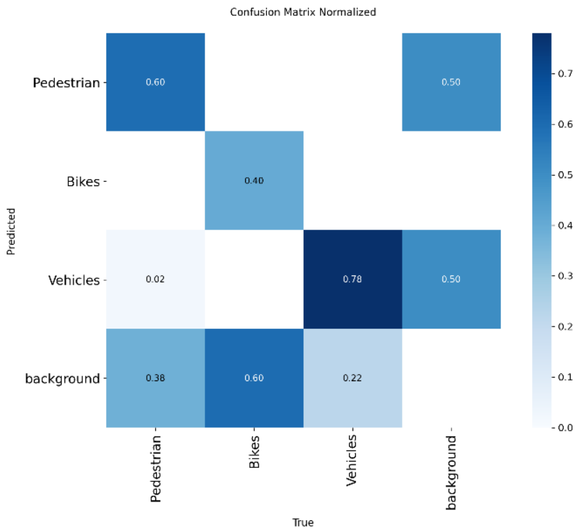
F1 and Confidence Curve Analysis



The F1-Confidence Curve helps determine the optimal confidence threshold for the model. The F1 score, which is the harmonic mean of precision and recall, is a key indicator of a model's balance.

- The highest F1 score for all classes is 0.68, which occurs at a confidence threshold of 0.156. This is the ideal threshold to use for balancing precision and recall in real-world applications of this model.

Confusion Matrix Analysis



The normalized confusion matrix provides a crucial insight into common classification errors.

- The model correctly predicted 60% of Pedestrians, but 38% were misclassified as 'background.' This explains why the recall for the pedestrian class is lower than the others.

- The model successfully identified 78% of Vehicles. However, it confused a significant number of Vehicles with 'background' and Pedestrians with 'background' as well, with 22% and 38% respectively.
- The model is highly accurate at identifying bikes, with 40% of the true bike instances correctly classified. The confusion matrix also shows that a significant number of bikes were classified as 'background'

Visual inspection of the images confirmed the accurate segmentation and classification of vehicles, bikes, and pedestrians across the dataset, even in challenging scenarios. The evaluation data provides a solid foundation for understanding the model's capabilities and pinpointing areas for future improvement, particularly in addressing the confusion between objects and the background.

Application Deployment

To demonstrate the trained model's practical utility, a web-based application was developed for real-time video processing and object tracking. Users upload a video file, which is processed frame by frame through the YOLOv8n segmentation model. The app overlays detected and segmented objects on the video output, and generates a structured JSON file summarizing the object class, position, and frame index for each detection.
The resulting tool provides an accessible interface for traffic analysis, surveillance, or other related tasks requiring automated object tracking and statistics.

Summary and Future Work

This project demonstrates the development of a complete, end-to-end pipeline for traffic scene segmentation and tracking, starting from data acquisition through to deployment of a web application for real-time analysis. The pipeline successfully integrates video data collection, frame extraction, data curation with labelling, deep learning model training, and application deployment.
Future enhancements could significantly expand the scope and improve the performance of the system:

- Expanding Object Classes:
  While the current model segments vehicles, bikes, and pedestrians, future work could expand to include additional important classes such as traffic signs, road markings, or different vehicle subcategories (e.g., trucks, buses). This would allow for richer scene understanding and improved situational awareness.
- Model-Assisted Labelling for Dataset Efficiency:
  Integrating an active learning or model-assisted labelling loop can reduce manual annotation effort. The trained model could pre-label new images for human reviewers to verify and correct, speeding up dataset expansion and quality improvement.
- Advanced Data Augmentation & Fine-Tuning:
  Applying sophisticated augmentation techniques such as geometric transformations, photometric variations, and synthetic data generation can enhance model robustness to various real-world conditions such as weather, lighting, and occlusions. Continued hyperparameter tuning and

experimentations with novel loss functions and architectures may further improve detection accuracy and mask segmentation quality.

- Optimizing Web App for Scalability and Real-Time Performance:
  Enhancing the deployed web application by leveraging optimized inference runtimes (e.g., Tensors, ONNX Runtime), model quantization, or edge deployment can enable real-time performance on a wider range of hardware. Improvements to the UI/UX and inclusion of analytics dashboards could also enhance usability.