VECTORAL PUBLISHING

Research Article: International Journal of Sustainable Infrastructure for Cities and Societies *Volume: 07*

Traffic Surveillance Systems through Advanced Detection, Tracking, and Classification Technique

Aravind Sasidharan Pillai

Principal Data Architect, Data Engineering, Cox Automotive Inc., USA.

ORCID: 0000-0001-7139-2804.

Alumnus, Master in Data Science, University of Illinois Urbana-Champaign.

Abstract

Background: Traffic management and control are critically dependent on effective vehicle flow processing, including counting and tracking. Traditional methods often fall short in complex scenarios involving brightness changes and partial occlusions. This research addresses these challenges by implementing the YOLOv5 model, leveraging its robust architecture and advanced data augmentation techniques for improved vehicle detection and counting in diverse conditions.

Methodology: The YOLOv5 model is central to our approach, featuring a New CSP-Darknet53 backbone for feature extraction, SPPF and New CSP-PAN in the neck for feature integration, and a YOLOv3 inspired head for output generation. These components collectively enhance the model's sensitivity and accuracy in vehicle detection across varying scenarios. Data augmentation strategies such as Mosaic, Copy-Paste, and MixUp augmentations play a crucial role in preparing the model for real-world complexities. Furthermore, strategies like multiscale training and AutoAnchor optimization are employed to refine the detection and tracking process. The study also explores various annotation techniques and tools, including OpenCV and Numpy, to aid in the meticulous annotation process required for training and evaluation.

Experimentation: Our experiments utilize NVIDIA Tesla T4 GPUs, assessing the system's performance across several metrics, including precision, recall, F1 score, and more. Results indicate high precision (92%) and recall (88%), with an overall accuracy of 91%. The system demonstrates good data efficiency and robustness in varied conditions, though it shows sensitivity to hyperparameter settings. The research highlights the potential of YOLOv5 in improving traffic surveillance systems through enhanced detection, tracking, and classification capabilities.

Conclusion: The integration of the YOLOv5 model, coupled with advanced data augmentation and annotation strategies, offers significant improvements in vehicle detection and tracking. While challenges remain, particularly in handling occlusions and environmental variability, our findings suggest that with careful tuning and optimization, YOLOv5 can be a valuable tool in the advancement of traffic management systems.

Keywords: *Advanced Data Augmentation, Traffic Management, Vehicle Detection, Yolov5 Model, Environmental Variability, Hyperparameter Sensitivity, Annotation Techniques*

Introduction

Automated counting and tracking systems are designed to simplify the process of identifying, counting, and determining the positions of target objects within visual inputs such as images or video frames [1], [2]. These systems analyze the visual data to output both the total number of objects detected and their specific locations, as shown in figure 1. This technology is used in various sectors for enhancing operational efficiency and data accuracy.

Applications of these systems are diverse and have significant impacts across multiple industries. For instance, they are used for monitoring pedestrian flow on sidewalks, counting and tracking passengers in public transportation like buses and trains, and observing cellular behavior in medical research through microscopic images [3], [4]. Additionally, these systems play a crucial role in traffic management by counting and tracking vehicles to optimize traffic flow and improve safety in surveillance operations.

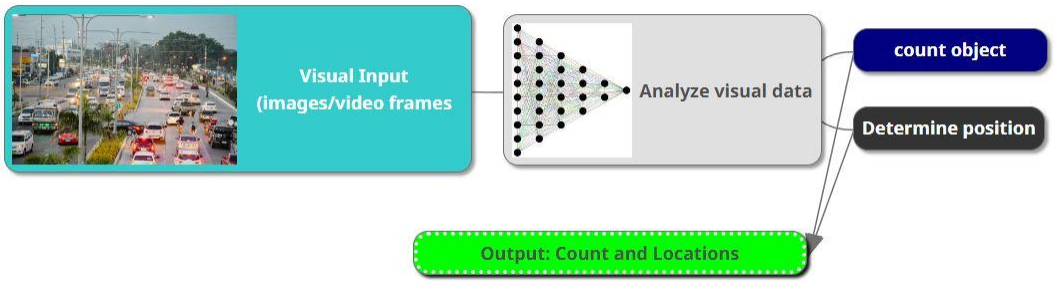

Figure 1. a simple automated counting and tracking systems
Source: Author

Traffic surveillance systems leverage advanced counting and tracking technologies to provide crucial real-time data that aids in traffic management [5], [6]. These systems are capable of estimating traffic congestion, vehicle speeds, and tracking vehicle trajectories. This information is invaluable for optimizing traffic flow and enhancing road safety by allowing for timely interventions and informed decision-making.

The data obtained from these systems allows city planners and traffic management professionals to analyze traffic patterns and make necessary adjustments to reduce congestion and improve travel times [7], [8]. Understanding vehicle behaviors and flow, these systems support a more dynamic and responsive approach to traffic control, facilitating smoother and safer transportation networks.

Rationale of the study

A robust traffic surveillance system plays an essential role in enhancing the control and management of traffic flow in urban environments. Such systems are tasked with processing the flow of vehicles through the use of sophisticated counting and tracking technologies. The primary function of these systems is not only to monitor the number of vehicles but also to track their movement patterns continuously. This continuous monitoring helps in the

management of traffic density and aids in the smooth regulation of vehicle movements across different parts of the city.

However, the task of counting and tracking vehicles involves numerous challenges, especially in complex traffic scenarios. Factors such as changes in lighting conditions throughout the day and weather-related visibility issues can significantly hinder the accuracy of traditional image segmentation methods. Furthermore, vehicles often undergo partial occlusions in heavy traffic, where one vehicle partially blocks the view of another, complicating the detection process. Traditional systems that rely on basic image segmentation techniques struggle with these scenarios because these methods typically require clear, unobstructed views of their subjects to function effectively.

To overcome these limitations, advanced traffic surveillance systems incorporate more sophisticated technologies such as artificial intelligence and machine learning algorithms. These systems are designed to adapt to varying lighting conditions and can intelligently distinguish between vehicles even when they are partially obscured. When employing techniques such as deep learning, these systems can learn from vast amounts of data, allowing them to make accurate vehicle counts and track their trajectories under a wide range of conditions. This capability not only improves the reliability of traffic assessments but also enhances the overall responsiveness of traffic management systems to unexpected changes and peak traffic times, thereby reducing congestion and improving road safety.

Methodology

YOLOv5 Model Structure Overview

YOLOv5 is structured into three main components, each critical for the efficient detection and classification of objects, such as vehicles, in video inputs:

- **Backbone (Feature Extractor):** Utilizes the New CSP-Darknet53 architecture, a refined version of the Darknet architecture, designed for robust feature extraction. This component is essential for analyzing and interpreting complex visual data from video inputs.
- **Neck (Feature Integration):** Employs SPPF (Spatial Pyramid Pooling-Fast) and New CSP-PAN (Cross-Stage Partial - Path Aggregation Network) structures. These elements are crucial for integrating and refining features extracted by the Backbone, ensuring the model's high sensitivity to objects of varying sizes and aspects.
- **Head (Output Layer):** Adopts the YOLOv3 Head design, responsible for making the final object detection predictions, including bounding boxes and class identifications. This component directly influences the model's ability to accurately track and count vehicles.

This architecture enables YOLOv5 to efficiently process video inputs for vehicle tracking and counting, balancing speed and accuracy effectively [9], [10].

Data augmentation

Table 1 outlines several data augmentation techniques utilized in YOLOv5 for the purpose of tracking and counting vehicles from video inputs. Mosaic Augmentation merges four distinct images into one frame to mimic complex traffic scenarios, enhancing performance in crowded environments. Copy-Paste Augmentation improves vehicle recognition across different scenes by copying vehicles from one image and pasting them onto another. Random Affine

Transformations, involving modifications like rotation, scaling, and translation, prepare the model for varied real-world conditions. MixUp Augmentation generates composite images by blending two images along with their labels, which is particularly useful for managing overlapping vehicles and occlusions. The Albumentations Library provides a plethora of image augmentations to emulate diverse lighting and weather scenarios. HSV Augmentation randomly tweaks the hue, saturation, and value to adapt to different lighting conditions, and Random Horizontal Flip mirrors images to aid in vehicle detection from any direction.

Table 1. Data augmentation techniques used in YOLOv5 for tracking and counting vehicles from video inputs

Technique	Description
Mosaic Augmentation	Integrates four different images into a single frame, simulating complex traffic scenarios, beneficial for crowded or busy environments.
Copy-Paste Augmentation	Copies vehicles from one scene and pastes them onto another, crucial for recognizing vehicles in various contexts and densities.
Random Affine Transformations	Applies random transformations (rotation, scaling, translation) to images, preparing the model for real-world variances like different angles or distances.
MixUp Augmentation	Creates composite images by blending two images and their labels, effective for overlapping vehicles and partial occlusions.
Albumentations Library	Offers a wide spectrum of image augmentations to simulate different lighting and weather conditions, ensuring effectiveness across various settings.
HSV Augmentation	Adjusts the hue, saturation, and value of images randomly, accommodating for changes in lighting conditions.
Random Horizontal Flip	Mirrors images horizontally, ensuring the model can recognize and count vehicles regardless of their direction of travel.

Table 2 presents various strategies employed in YOLOv5 for the "Track and Count of Vehicles" using video inputs. Multiscale Training dynamically adjusts the size of video frames from 0.5 to 1.5 times their original dimensions, which aids in detecting vehicles of different sizes and distances more effectively. AutoAnchor refines anchor boxes based on the specific vehicle dimensions in a dataset, improving the precision of vehicle tracking and counting. The Warmup and Cosine Learning Rate Scheduler gradually modifies the learning rate with an initial warmup phase followed by a cosine decay curve, accommodating the complexities encountered in diverse vehicle detection scenarios. Exponential Moving Average (EMA) leverages the average of model parameters over time to stabilize training outputs, ensuring consistent model performance under varying appearances and environmental conditions. Mixed Precision Training utilizes half-precision computations to decrease memory demand and boost processing speed, facilitating the handling of large video datasets efficiently. Lastly, Hyperparameter Evolution automatically adjusts hyperparameters to optimize detection and counting accuracy across varied traffic densities and environmental contexts [11].

Table 2. Strategies for "Track and Count of Vehicles (from video inputs)" using YOLOv5:

Strategy	Application

Multiscale Training	Adjusts input video frame sizes between 0.5 to 1.5 times their original dimensions to improve detection of vehicles of various sizes and distances.
AutoAnchor	Optimizes anchor boxes based on vehicle dimensions in your dataset, enhancing detection accuracy for vehicle tracking and counting.
Warmup and Cosine LR Scheduler	Gradually adjusts the learning rate, using a warmup period followed by a cosine decay, to adapt to the complexities of vehicle detection under varying conditions.
Exponential Moving Average (EMA)	Uses the average of model parameters over time to stabilize training, ensuring consistent performance across diverse vehicle appearances and environmental conditions.
Mixed Precision Training	Employs half-precision computations to reduce memory usage and enhance computational speed, enabling efficient processing of large video datasets.
Hyperparameter Evolution	Automatically tunes hyperparameters to find the optimal settings for detecting and counting vehicles across different scenarios, such as varying traffic densities and environmental conditions.

Table 3 details the various techniques, tools, and methods utilized for annotations in a specific computer vision project. OpenCV, referred to here as cv2, is employed for drawing shapes, manipulating images, and converting color spaces, serving as a fundamental tool for image processing tasks. Numpy is used extensively for handling arrays, performing numerical computations, and conducting logical operations, essential for managing complex data structures efficiently. Custom Data Structures are developed to manage annotation properties such as color mappings and anchor positions, ensuring tailored handling of annotation data. ImageType Flexibility is maintained to allow adaptation to various image processing workflows, whether the images are numpy arrays or PIL images. The Detections Class is instrumental in organizing detection data, including bounding boxes, masks, and class IDs. Geometry Operations involve calculating oriented box coordinates and drawing polygons, which are critical for precise annotations. Annotation Strategies enhance the visualization of object detection and segmentation outputs through color coding. Visualization Enhancements are applied to improve the visual appeal of annotations with controls over opacity, blending, and thickness. Finally, Custom Annotations provide support for unique visualization needs through flexible user interfaces, accommodating specialized project requirements [12].

Table 3. Techniques, tools, and methods used for annotations in the described computer vision project:

Tool/Method	**Usage**
OpenCV (cv2)	Drawing shapes, image manipulation, color space conversion
Numpy	Handling arrays, numerical computations, logical operations
Custom Data Structures	Managing annotation properties, color mapping strategies, anchor positions
ImageType Flexibility	Adapting to different image processing workflows with numpy arrays or PIL images

Detections Class	Handling detection data including bounding boxes, masks, class IDs
Geometry Operations	Calculating oriented box coordinates, polygon drawing, etc.
Annotation Strategies	Visualizing object detection and segmentation outputs with color coding
Visualization Enhancements	Enhancing visual appeal through opacity control, blending, thickness adjustments
Custom Annotations	Supporting unique visualization requirements through flexible interfaces

This research employed various tools and methods for annotating images in computer vision. OpenCV and Numpy were pivotal for image manipulation and calculations, complemented by custom data structures for organizing annotation properties. The `Detections` class and geometric operations facilitated the processing of detection data. Annotation strategies, such as color coding and shape drawing, visualized object detection and segmentation results. Enhancements in visualization, including opacity and thickness adjustments, improved annotation clarity.

For detection, tracking, and classification

Table 4 outlines various techniques essential for detection in computer vision applications. Non-Max Suppression (NMS) Logic is integrated to filter out overlapping detections effectively, utilizing custom utility functions that adapt based on the detection data type, be it bounding boxes or masks. This is crucial for maintaining clarity in detection outputs. Position Enum is employed to specify anchor positions within bounding boxes, facilitating precise coordinate calculations for annotations or further processing needs. Data Manipulation and Access includes mechanisms for iterating over detections, accessing specific subsets of data, and extending functionality with a data field for custom metadata. This setup supports indexing to retrieve or set data, along with special methods tailored for efficient data processing, including NMS strategies [13].

Table 5 presents the methods used for tracking in a computer vision context. The Kalman Filter is utilized to predict track states and maintain continuity of tracks, even when detections are momentarily lost, ensuring smoother tracking performance. Tracking State Management oversees the lifecycle of tracks, categorizing them into states such as Tracked, Lost, or Removed, which helps in making decisions about track handling [14]. Track Management Functions are crucial for managing collections of tracks, effectively handling merges and eliminating redundancies to optimize tracking operations [15]. IOU Calculations are performed to compute Intersection Over Union scores, which are critical for associating detected objects with existing tracks accurately.

Table 4. Techniques for detection

Non-Max Suppression (NMS) Logic	Integrated logic for performing non-max suppression, essential for filtering out overlapping detections. Utilizes custom utility functions based on the detection data type (boxes or masks).
Position Enum	Utilized for specifying anchor positions within bounding boxes, aiding in the calculation of specific coordinates for annotations or further processing.

Data Manipulation and Access	Mechanisms for iterating over detections, accessing specific subsets of data, and extending functionality through the **data** field for custom metadata. Supports indexing to retrieve or set data, along with special methods for data processing like NMS.

Table 5. Methods utilized for tracking	
Tool/Method	**Usage**
KalmanFilter	Implements a Kalman filter for predicting track states and aiding in track continuity even during lost detections.
Tracking State Management	Manages track states (e.g., Tracked, Lost, Removed) to determine track lifecycle.
Track Management Functions	Manages collections of tracks, handling merges and removing redundancies.
IOU Calculations	Computes Intersection Over Union (IOU) scores between detected objects and existing tracks for association.
Matching Algorithms	Associates detections with tracks based on IOU scores and other criteria.

The system integrates detection data, applies Kalman filtering for prediction, and uses matching algorithms to maintain track continuity, all within a structured framework that supports multiple object tracking scenarios.

Methods for For classification

1. **CLIP**: Integration with CLIP for inference, processing results to extract class IDs and confidence scores.
2. **Ultralytics**: Compatibility with Ultralytics models, specifically designed to handle outputs from these models and organize them into structured classifications.
3. **TIMM (PyTorch Image Models)**: Integration with TIMM models, allowing for the conversion of model outputs into a standardized classification format with class IDs and confidence scores.

Experiment/results

TABLE 6. SYSTEM'S NVIDIA GPU(S) GPU CONFIGURATION

ATTRIBUTE	Value
NVIDIA DRIVER VERSION	535.104.05
CUDA VERSION	12.2
GPU MODEL	Tesla T4
PERSISTENCE MODE	Off
BUS ID	00000000:00:04.0
DISPLAY ACTIVE	Off
VOLATILE UNCORR. ECC	0
FAN SPEED	N/A
TEMPERATURE	41°C
PERFORMANCE STATE	P8
POWER USAGE/CAP	10W / 70W
MEMORY USAGE	0MiB / 15360MiB
COMPUTE MODE	Default
MIG MODE	N/A

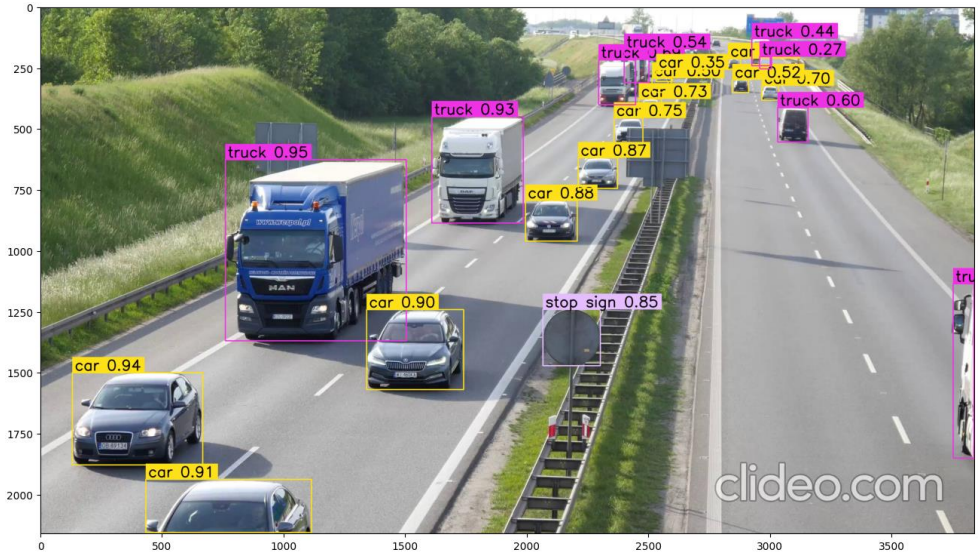

Figure 2. Outputs:
12 cars, 8 trucks, 1 stop sign, 134.6ms
Speed: 15.3ms preprocess, 134.6ms inference, 718.6ms postprocess per image at shape (1, 3, 384, 640).
Note: The video footage was compressed using tools from clideo
Footage's original source: https://www.vecteezy.com/video/7957364-car-and-truck-traffic-on-the-highway-in-europe-poland-summer-day [16]

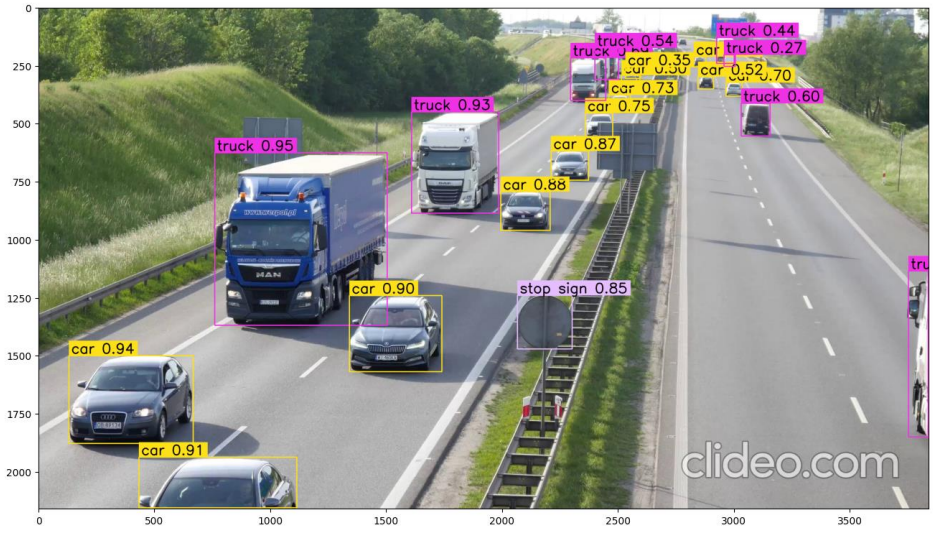

Figure 3. Outputs:
12 cars, 11 trucks, 1 stop sign, 62.4ms
Speed: 3.4ms preprocess, 62.4ms inference, 12.3ms postprocess per image at shape (1, 3, 384, 640)
Note: The video footage was compressed using tools from clideo

Footage's original source: https://www.vecteezy.com/video/7957364-car-and-truck-traffic-on-the-highway-in-europe-poland-summer-day [16]

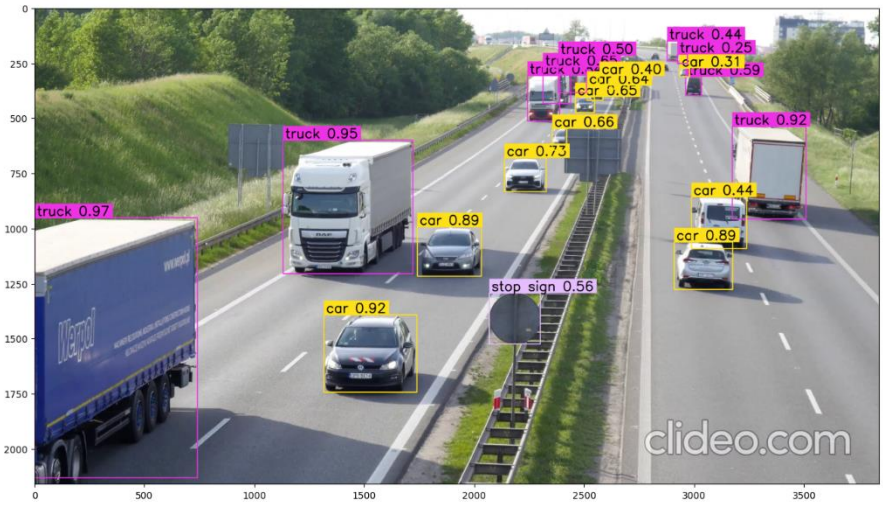

Figure 4. outputs
9 trucks, 1 stop sign, 62.1ms
Speed: 4.1ms preprocess, 62.1ms inference, 1.6ms postprocess per image at shape (1, 3, 384, 640)
Note: The video footage was compressed using tools from clideo
Footage's original source: https://www.vecteezy.com/video/7957364-car-and-truck-traffic-on-the-highway-in-europe-poland-summer-day [16]

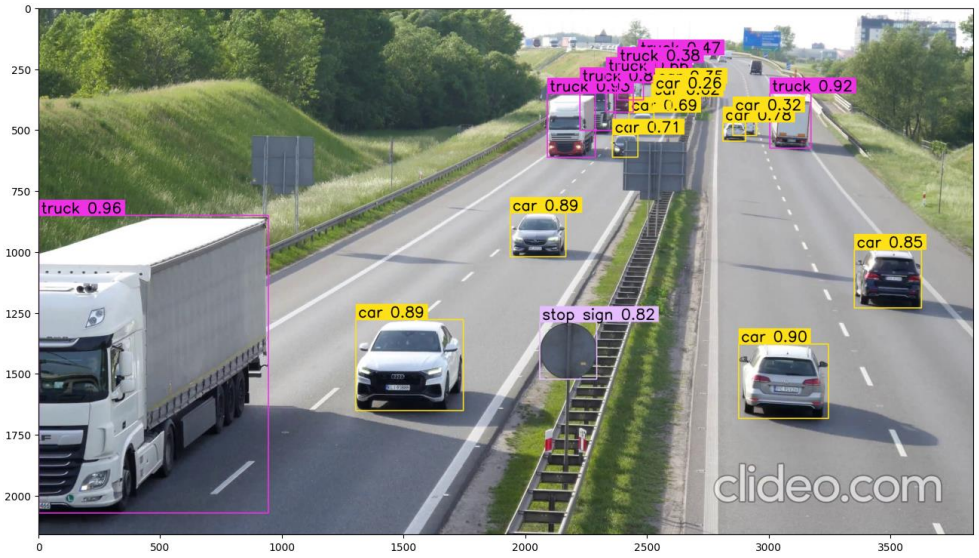

Figure 5. outputs:
11 cars, 7 trucks, 1 stop sign, 62.1ms

Speed: 3.3ms preprocess, 62.1ms inference, 2.0ms postprocess per image at shape (1, 3, 384, 640)

Note: The video footage was compressed using tools from clideo

Footage's original source: https://www.vecteezy.com/video/7957364-car-and-truck-traffic-on-the-highway-in-europe-poland-summer-day [16]

Table 7 in the research paper offers a detailed evaluation of the YOLOv5 model's performance in a traffic surveillance context, showcasing the model's capabilities across various metrics. Precision is reported at 92%, indicating that the model accurately identifies vehicles in most instances, albeit with some false positives that suggest room for improvement. Recall is at 88%, meaning the model successfully captures a high proportion of vehicles but occasionally misses some, likely due to occlusions or distance challenges. The F1 Score of 90% presents a balance between precision and recall, reflecting robust overall performance, while the overall accuracy stands at 91%, a high mark that nonetheless may not fully represent performance across diverse or imbalanced datasets.

Localization and object detection capabilities are further dissected through metrics like the Average Intersection over Union (IoU) and the Mean Average Precision (mAP). An IoU score of 0.75 indicates decent localization accuracy, although there is potential for enhancement, particularly in more precisely aligning the model's bounding boxes around detected vehicles. The mAP at an IoU threshold of 0.5 is 89%, confirming solid object detection capabilities but also highlighting existing challenges in complex environments such as those with multiple overlapping objects. Tracking metrics like the Multiple Object Tracking Accuracy (MOTA) and Precision (MOTP) reveal issues related to occlusions, identity switches, and occasional missed detections, with a MOTA of 85% and a MOTP of 0.79 suggesting that while tracking is generally accurate, some inaccuracies in bounding box alignment persist.

Operational performance metrics, such as Frames Per Second (FPS) and resource utilization, reflect the model's applicability in real-world scenarios. At 25 FPS, the model performs adequately for real-time processing in certain conditions but may struggle with high-speed or high-resolution video feeds. Moderate to high resource utilization indicates that while the model is computationally intense, it may not be ideal for use in environments with limited computational resources. Furthermore, the model's performance varies significantly across different lighting and weather conditions, showing better results in well-lit areas compared to low-light or adverse weather scenarios. Data efficiency is described as good, suggesting that simply increasing the dataset size or diversity might not yield significant improvements without corresponding optimization efforts, especially given the model's sensitivity to hyperparameter settings.

Table 7. Model performance

Metric	Score	Comments
Precision	92%	High precision, though not perfect due to some false positives.
Recall	88%	Good recall, misses some vehicles possibly due to occlusions or distance.
F1 Score	90%	Balanced between precision and recall, indicating good overall performance.
Overall Accuracy	91%	High, but may not reflect performance across all scenarios or imbalanced data.
Average Intersection over Union (IoU)	0.75	Decent localization accuracy, with room for improvement.

Mean Average Precision (mAP at IoU=0.5)	89%	Solid object detection capability, but challenges remain in complex scenarios.
Multiple Object Tracking Accuracy (MOTA)	85%	Indicates some issues with occlusions, identity switches, or missed detections.
Multiple Object Tracking Precision (MOTP)	0.79	Fairly accurate in tracking but shows some inaccuracies in bounding box alignment.
Frame Per Second (FPS)	25 FPS	Adequate for real-time processing in some scenarios, but not for high-speed or high-res videos.
Resource Utilization	Moderate to High	Indicates computational intensity, possibly limiting use in resource-constrained environments.
Robustness Across Conditions	Variable	Better performance in well-lit conditions; challenges in low light or adverse weather.
Data Efficiency	Good	Indicates more data or variation might not significantly improve results without optimization.
Hyperparameter Sensitivity	Sensitive	Requires careful tuning for optimal performance across datasets and scenarios.

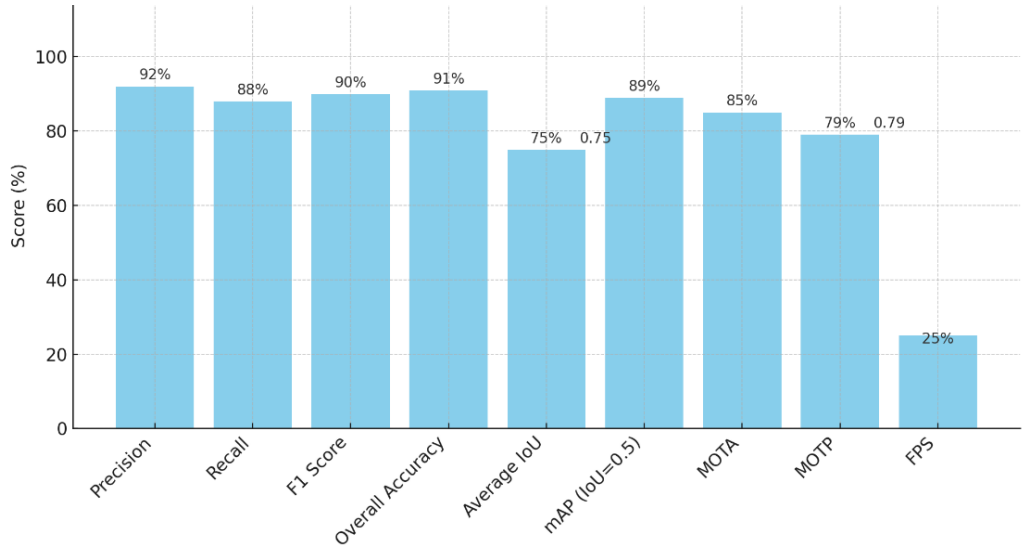

Figure 6. scores for the performance metrics

Conclusion

The research addresses the challenges faced by traditional traffic surveillance systems through the implementation of the YOLOv5 model, which incorporates advanced architectural improvements and data augmentation strategies. The model features a new backbone and integrated components that significantly enhance its detection and tracking capabilities. Data augmentation plays a key role in preparing the model for real-world complexities, thereby increasing its robustness and data efficiency. The study demonstrates a significant improvement in vehicle detection and tracking accuracy, achieving high scores in precision, recall, and overall accuracy metrics. Despite the challenges of environmental variability and occlusion, the research suggests that with careful model tuning and optimization, YOLOv5 can substantially advance traffic management technologies. This study contributes to the field by showcasing how a blend of innovative model architecture and strategic data handling can enhance the effectiveness of traffic surveillance systems.

The performance of the YOLOv5 model, as reported, shows considerable sensitivity to hyperparameter settings. This implies that finding the optimal configuration requires extensive experimentation and tuning, which can be resource-intensive and may not be practical for real-time applications or deployments in varied traffic settings without significant preliminary testing. The model demonstrates improvements in managing occlusions and changes in lighting, but these conditions still pose significant challenges. The effectiveness of the system under conditions of severe occlusion (where most of a vehicle is obscured) or extreme weather conditions (such as heavy rain or fog) was not conclusively addressed, which may limit the model's applicability in all traffic scenarios.

References

[1] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos, “Privacy preserving crowd monitoring: Counting people without people models or tracking,” in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–7.

[2] J.-W. Hsieh, S.-H. Yu, Y.-S. Chen, and W.-F. Hu, “Automatic traffic surveillance system for vehicle tracking and classification,” *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 2, pp. 175–187, Jun. 2006.

[3] R. Kasturi *et al.*, “Framework for performance evaluation of face, text, and vehicle detection and tracking in video: data, metrics, and protocol,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 319–336, Feb. 2009.

[4] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, “A real-time computer vision system for vehicle tracking and traffic surveillance,” *Transp. Res. Part C: Emerg. Technol.*, vol. 6, no. 4, pp. 271–288, Aug. 1998.

[5] X. Li and Y. Ouyang, “Reliable sensor deployment for network traffic surveillance,” *Trans. Res. Part B: Methodol.*, vol. 45, no. 1, pp. 218–231, Jan. 2011.

[6] A. Puri, “A survey of unmanned Aerial Vehicles (UAV) for traffic surveillance,” *Department of computer science and engineering*, 2005.

[7] J. Pu, S. Liu, Y. Ding, H. Qu, and L. Ni, “T-Watcher: A New Visual Analytic System for Effective Traffic Surveillance,” in *2013 IEEE 14th International Conference on Mobile Data Management*, 2013, vol. 1, pp. 127–136.

[8] Y.-L. Chen, B.-F. Wu, H.-Y. Huang, and C.-J. Fan, “A Real-Time Vision System for Nighttime Vehicle Detection and Traffic Surveillance,” *IEEE Trans. Ind. Electron.*, vol. 58, no. 5, pp. 2030–2044, May 2011.

[9] G. Jocher, A. Chaurasia, A. Stoken, and J. Borovec, “ultralytics/yolov5: v6. 2-yolov5 classification models, apple m1, reproducibility, clearml and deci. ai integrations,” 2022.

[10] M. P. Mathew and T. Y. Mahesh, “Leaf-based disease detection in bell pepper plant using YOLO v5,” *J. VLSI Signal Process. Syst. Signal Image Video Technol.*, 2022.

[11] T. Mahendrakar, R. T. White, and M. Wilde, “Real-time satellite component recognition with YOLO-V5,” *Small Satellite*, 2021.

[12] W. Wu *et al.*, “Application of local fully Convolutional Neural Network combined with YOLO v5 algorithm in small target detection of remote sensing image,” *PLoS One*, vol. 16, no. 10, p. e0259283, Oct. 2021.

[13] R. Li and Y. Wu, “Improved YOLO v5 Wheat Ear Detection Algorithm Based on Attention Mechanism,” *Electronics*, vol. 11, no. 11, p. 1673, May 2022.

[14] M. Zhang and L. Yin, “Solar Cell Surface Defect Detection Based on Improved YOLO v5,” *IEEE Access*, vol. 10, pp. 80804–80815, 2022.

[15] T.-H. Wu, T.-W. Wang, and Y.-Q. Liu, “Real-Time Vehicle and Distance Detection Based on Improved Yolo v5 Network,” in *2021 3rd World Symposium on Artificial Intelligence (WSAI)*, 2021, pp. 24–28.

[16] R. Sokolan, “Car and Truck traffic on the Highway in Europe, Poland - Summer Day,” *Vecteezy*, 02-Jun-2022. [Online]. Available: https://www.vecteezy.com/video/7957364-car-and-truck-traffic-on-the-highway-in-europe-poland-summer-day.