Journal of Transportation Technologies, 2027, 17(1), 1-53
https://www.scirp.org/journal/jtts
ISSN Online: 2160-0481
ISSN Print: 2160-0473

Intelligent Vehicle Traffic Recognition Using Deep Learning Algorithms: A Class-Specific Evaluation of YOLOv8n

Ashikur Rahman Mohammad[1,2,3], Jun Li[3*], Yanfeng Han[1,2], Sheikh Sakibul Hasan[1,2]

[1]College of Mechanical and Vehicle Engineering, Chongqing University, Chongqing, China
[2]State Key Laboratory of Mechanical Transmission for Advanced Equipment, Chongqing University, Chongqing, China
[3]School of Mechatronics and Vehicle Engineering, Chongqing Jiaotong University, Chongqing, China
Email: l2400108@stu.cqu.edu.cn, *cqleejun@163.com, fyh-0220@163.com, sakibzjut@gmail.com

How to cite this paper: Mohammad, A.R., Li, J., Han, Y.F. and Hasan, S.S. (2027) Intelligent Vehicle Traffic Recognition Using Deep Learning Algorithms: A Class-Specific Evaluation of YOLOv8n. *Journal of Transportation Technologies*, **17**, 1-53.
https://doi.org/10.4236/jtts.2027.171001

Received: November 1, 2026
Accepted: November 24, 2026
Published: November 27, 2026

Copyright © 2027 by author(s) and Scientific Research Publishing Inc.
This work is licensed under the Creative Commons Attribution-NonCommercial International License (CC BY-NC 4.0).
http://creativecommons.org/licenses/by-nc/4.0/

Open Access

Abstract

Urban-road recognition for intelligent vehicles requires evaluation data whose labels, geometry, and partitions support the intended task. This study presents a reproducible class-specific audit of archived YOLOv8n experiments and an open traffic export containing 2582 images and 14,074 annotations. The analysis combines annotation-format inspection, box geometry, a fixed sample of 84 annotated regions, exact and perceptual image-identity checks, and extraction of three saved validation reports. The archive contains 72 polygon rows and 14,002 box rows. In 44 mixed-format files, a fresh import under the recorded software version reinterprets 74 box rows as two-point polygons, including 20 validation rows. After normalization, the original verifier returns all 14,074 intended boxes with identical class IDs and coordinates at float32 precision. The AI-assisted inspection, checked by one author, finds registration characters under the bike label, punctuation-like regions under the triple-riding label, and miniature vehicles within the vehicle sample. These observations restrict the semantic interpretation of the saved class scores. The 400-epoch report records mAP@0.50 of 86.3% and mAP@0.50:0.95 of 54.0%; class strict AP ranges from 26.8% to 92.4%. Exact hashing identifies four cross-partition duplicate groups, while perceptual screening identifies additional candidates requiring review. Because validation informed checkpoint selection and an independent test report is absent, these results characterize the exported labels rather than establish urban-road generalization. The contribution is an executable evidence audit and verified annotation-format normalization, supported by an improvement workflow that separates data repair from subsequent model training and independent testing.

Keywords

Deep Learning, YOLOv8n, Object Detection, Annotation Consistency, Intelligent Vehicles

1. Introduction

Intelligent-vehicle traffic recognition depends on the relationship between what a detector predicts and what its evaluation data actually represent. Object detection can provide observations for road-user tracking, counting, or subsequent safety analysis, but a numerical score is meaningful only for the specified annotation units and evaluation protocol. COCO defines detection measures across increasingly strict overlap criteria, while BDD100K demonstrates the diversity of driving imagery and perception tasks [1] [2]. Dataset documentation and intended-use reporting are therefore part of the evidence supporting a transport application [3] [4].

Modern detectors provide different ways of locating and classifying image regions. YOLO predicts boxes and classes in one network pass [5]; Faster R-CNN proposes regions and then classifies and refines them [6]; DETR uses object queries and set prediction [7]. Subsequent work addresses efficiency, matching, and training design through architectures such as YOLOv7, RT-DETR, and DEIM [8]-[10]. Their success motivates candidate algorithms for road perception, while the validity of a particular application still depends on its labels, data independence, and implementation. Benchmark research shows that annotation errors can affect model comparisons, and leakage research explains why a nominal split alone does not establish independent evaluation [11] [12].

The present study examines saved YOLOv8n experiments from the thesis study [13] and the supplied public Traffic dataset export [14]. Its seven exported names are Triple_riding, bike, motorcycle, number_plate, vehicle, with_helmet, and without_helmet. These strings suggest several transportation tasks, yet names alone cannot establish what each box encloses. Helmet, rider-group, and plate studies illustrate why the unit of annotation matters: a helmet region, a rider-helmet relation, a rider group, and a plate transcription are distinct targets [15]-[17]. The export has no codebook establishing those distinctions.

Our archive audit adds two concerns beyond unequal class support. First, a fixed inspection sample contains registration characters under the bike name and punctuation-like regions under Triple_riding, alongside the expected two-wheeler or rider-group examples. The vehicle sample contains both miniature vehicles and full-size vehicle photographs. Second, the label export mixes five-column boxes with polygon rows. The file-level import rule in the recorded software version can consequently change the geometry of otherwise conventional box rows. These observations concern task validity and implementation compatibility; they cannot be resolved by increasing the number of training epochs.

The research questions are therefore: 1) how consistently do the exported names and serialized coordinates define the evaluated targets; 2) what class support, box geometry, and cross-partition similarity are present in the archive; and 3) what can the saved model reports establish when read alongside those findings? The contributions are:

- A reproducible audit of all 2582 images and 14,074 annotation rows, combining file identity, row representation, class-level image support, box geometry, and a fixed, traceable annotation sample.
- A source-checked coordinate-import experiment and a non-destructive normalization procedure. The procedure removes representation-dependent coordinate changes while preserving class IDs and row counts; semantic relabeling remains a separate task.
- Executable extraction and reconciliation of the saved YOLOv8n validation reports, including explicit macro-averaging, class-level contrasts, and the boundaries of recorded processing time.
- An evidence-guided workflow connecting the completed audit and normalization to subsequent annotation review, controlled model comparisons, and independent evaluation.

This is an empirical audit of an archived experiment and its supplied export. All detector scores remain historical validation summaries. There is no fresh model evaluation on normalized labels, no preserved historical label cache, and no independent test report. Section 2 positions the study within detection, data quality, and transportation applications. Section 3 specifies the audit and recorded experiment; Sections 4 and 5 report and interpret the evidence; Section 6 states the supported conclusions.

2. Related Work

2.1. Visual Representations and Detection Architectures

Convolutional networks established the feature-learning basis for modern visual recognition. AlexNet demonstrated the effectiveness of deep convolutional features at large scale, while Inception and residual networks developed multi-branch processing and identity shortcuts to improve feature efficiency and optimization [18]-[20]. General accounts of deep architectures explain why learned hierarchical representations replaced many hand-designed pipelines, but they do not by themselves specify a detector or validate a transport application [21]. This distinction matters in the present study: it analyzes archived detector outputs and their evaluation context.

Object detectors combine these visual features with mechanisms for locating and classifying multiple instances. The original YOLO formulation predicts boxes and class probabilities in a single network pass [5]; SSD similarly performs dense detection at multiple feature scales, including in pedestrian-focused implementations [22]. Later traffic-oriented work has explored multi-scale YOLO features, feature optimization and fusion, and deep-learning detectors for autonomous-ve-

hicle scenes [23]-[25]. These studies motivate design choices for dense urban scenes, where object scale, occlusion, and visual clutter can vary within one frame. Their reports are tied to their own data and evaluation protocols and are not numerical baselines for the dataset analyzed here.

Region-based methods offer a different allocation of computation. Faster R-CNN first proposes candidate regions and then classifies and refines them using shared features [6]. Mohammad and Li also studied Faster R-CNN for urban-road target detection, providing a directly relevant application of the region-proposal approach [26]. Subsequent studies have adapted this family to small helmet targets and traffic imagery, illustrating the potential value of proposal coverage for small or partially occluded objects [27]. Other work uses deformable convolutions with SSD or shifted-window Transformer backbones to alter feature sampling and long-range context [28] [29]. Such modifications can be relevant to distant road users, but improvements must be established under a common split, annotation policy, and matching rule. The earlier urban-road study is treated here as related prior work; its scores are not used as a matched benchmark because the available records do not establish equivalent data, labels, or evaluation protocols.

Feature-pyramid and attention components form another strand of detector design. Feature Pyramid Networks combine semantically strong coarse features with higher-resolution features through top-down and lateral paths, supporting predictions at multiple scales [30]. Spatial pyramid pooling, squeeze-and-excitation, and convolutional block attention modules provide alternative mechanisms for aggregating context or reweighting channels and spatial locations [31]-[33]. Feature-fusion detection and hybrid CNN-Transformer work provide further design alternatives [24] [34]. These components are design options, not independent evidence that any one block improves the seven classes evaluated in this article.

Efficiency-focused backbones address the cost of deployment. MobileNet uses depthwise separable convolution, and its later versions refine the balance of latency and representation capacity [35]-[37]. ShuffleNet and ShuffleNet V2 use channel grouping and channel shuffle operations, with the latter emphasizing measured hardware efficiency; GhostNet generates additional feature maps through inexpensive operations [38]-[40]. Their relevance is that detector speed depends on backbone computation and hardware, not only on parameter count. The present work reports saved YOLOv8n timing on an RTX 4090 and does not claim embedded-device performance.

Loss design also changes the localization objective. Distance-IoU and generalized-IoU losses add geometric information beyond overlap alone, while distributional regression methods supervise box coordinates through discrete distributions [41]-[43]. These formulations aim to make box optimization more informative, especially when overlap is initially small or boxes have different shapes. The saved YOLOv8n implementation uses its recorded complete-IoU and distribution-focal-loss configuration; the alternative losses in this literature were not evaluated as ablations here. This is why a class-specific AP profile should be inter-

preted as the output of a complete model-and-training configuration rather than as an isolated architectural effect.

2.2. Transformer Detection and Set Prediction

Transformer detection reframes the output as a set of objects rather than a dense collection of independently thresholded candidates. The attention operation introduced in the Transformer architecture relates queries to keys and values [44]. DETR applies an encoder-decoder with learned object queries and bipartite matching to predict a permutation-invariant set of boxes and labels; the original paper and official implementation describe the formulation and training pipeline [7] [45]. Analyses of image-to-set prediction and deep-set methods provide context for the matching, cardinality, and loss choices involved in this formulation [46]-[48]. This provides a conceptual contrast with YOLOv8n's dense feature-map prediction and post-processing.

Traffic-specific and hybrid Transformer studies investigate whether global context can aid recognition in crowded scenes. Mohammad and Li's earlier study examines CNN- and DETR-based intelligent road recognition for autonomous vehicles, linking set-prediction methods directly to the road-recognition task considered here [49]. Recent real-time detector work examines transformer efficiency, matching, and convergence [9] [10] [34]. These research directions are potentially relevant where local appearance is ambiguous, such as in overlapping riders or vehicles. The prior CNN-DETR paper is included as related work rather than as a performance comparator because equivalence of dataset, label definitions, and evaluation protocol has not been established. The current labels do not encode rider-helmet relations, and no Transformer model was trained on the supplied export. The authors' earlier DETR and Faster R-CNN frameworks are described separately below; neither was evaluated in the current experiment.

2.3. Prior DETR and Faster R-CNN Frameworks

The authors' earlier road-recognition work considered query-based and region-proposal detectors [26] [49]. Their framework illustrations and thesis-reported dataset descriptions are retained as methodological context. The current export has no matched DETR or Faster R-CNN experiment; historical source manifests and checkpoints for those studies are unavailable.

2.3.1. CNN-Transformer Road-Object Detection

DETR combines CNN image features with a Transformer encoder-decoder and learned object queries. Prediction heads return object classes and boxes, including a no-object class for unmatched queries [7]. Attention relates image and query features [44]; bipartite matching assigns target objects to distinct queries.

Figure 1 connects image-feature encoding, query decoding, and set prediction. It describes the detector family used in the earlier work; no new DETR predictions are reported here.

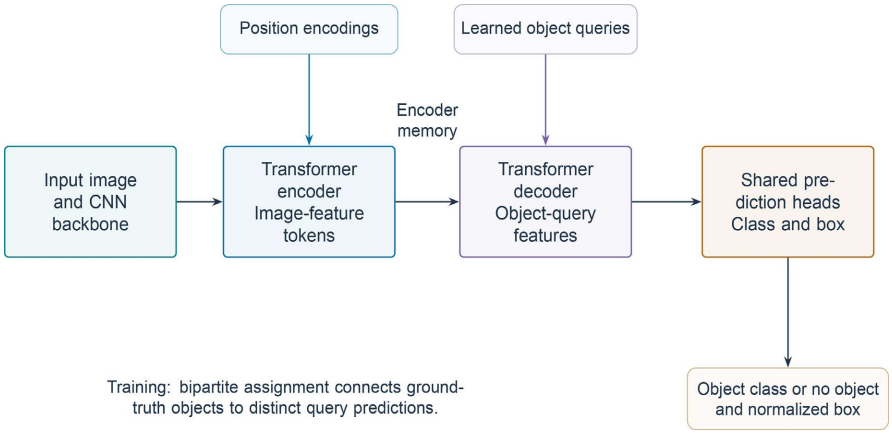


Figure 1. Vector schematic of the DETR stages underlying the prior CNN-DETR road-recognition framework [13] [49]. The encoder-decoder and set-prediction mechanism follows DETR [7].

2.3.2. Region-Proposal Road-Object Detection

Faster R-CNN shares convolutional features between a region proposal network (RPN) and a region classifier/refiner [6]. The authors' earlier urban-road study used this family [26].

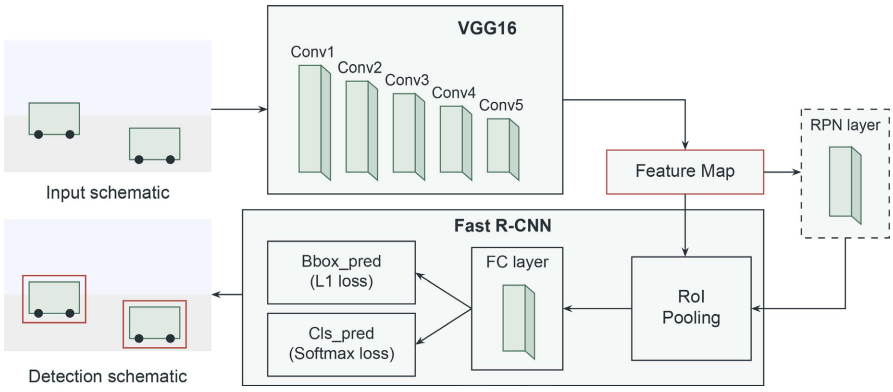


Figure 2. Vector Faster R-CNN processing flow corresponding to the thesis framework and prior urban-road study [13] [26]. Input/output icons are schematic. The standard proposal/refinement design follows Faster R-CNN [6].

Figure 2 illustrates the shared VGG16 features, proposals, RoI pooling, and classification/regression stages of the depicted framework. The icons describe processing rather than archive a detector output.

Figure 3 shows the $2k$ objectness and $4k$ coordinate outputs for k anchors at a feature-map location. Proposal coverage and filtering are additional evaluation choices in a future matched study. Standard attention, assignment, and RPN equations are retained in the Supplementary framework derivations.

2.3.3. Prior Datasets and Comparability

Table 1 separates historical thesis dataset descriptions from the supplied export. Earlier reported scores use inconsistent model assignments and metric terminol-

ogy and lack the records needed for a matched comparison. They are therefore preserved as source context rather than reproduced as a ranking. A future comparison requires verified labels, independent source groups, shared evaluation rules, and documented model-specific training.

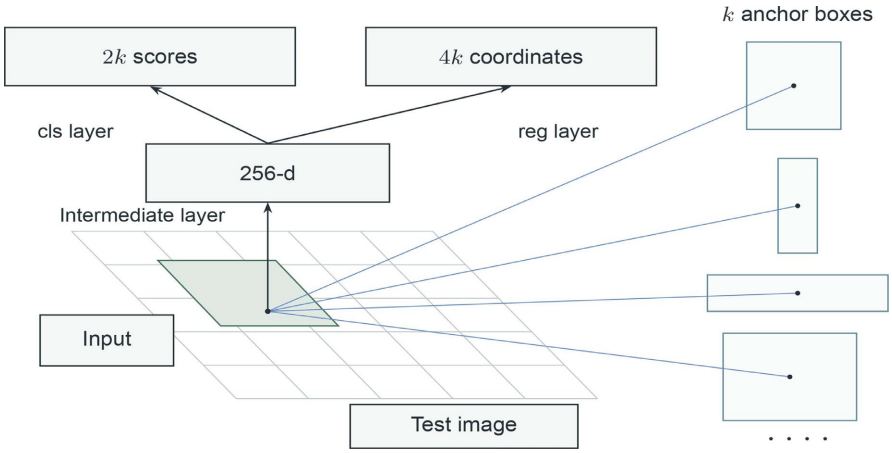


Figure 3. Anchor-based RPN schematic associated with the prior urban-road framework [13] [26]. Objectness and box-regression heads follow the standard formulation [6].

Table 1. Dataset descriptions reported for the prior author studies and the current YOLOv8n audit. Prior-study counts are thesis-reported and could not be checked against source manifests or model artifacts.

Method context	Data and partition described	Evidence boundary
Prior DETR study	The thesis reports 7986 road images and a 70/15/15% training/validation/test split.	Thesis narrative only; no source manifest, checkpoint, or per-image predictions were supplied for this audit.
Prior Faster R-CNN study	The thesis refers to BDD100K and lists 69,719 training and 10,000 test samples; their relation to the study's annotations is unclear.	Thesis description only; exact split manifest, label mapping, checkpoint, and prediction records were not supplied.
Current YOLOv8n audit	2582 images: 1864 training, 482 validation, and 236 test.	Counts were checked against the supplied export; exact-image overlap affects train/validation independence, and no test report was supplied.

2.4. Dataset Documentation, Label Validity, and Evaluation Integrity

Dataset documentation records what examples represent, how they were col-

lected, and which uses their labels support. Datasheets make those questions explicit rather than leaving users to infer meaning from class names [3]. Model cards similarly connect reported performance with intended use and evaluation context [4]. These ideas are directly relevant when an export combines region types or source domains under a small set of names.

Northcutt *et al.* show that label errors can change benchmark comparisons and support checking annotated examples before interpreting a model ranking [11]. Their analysis concerns other datasets; it does not establish an error rate for the export studied here. TIDE offers a complementary decomposition of detection errors when predictions and targets are available [50]. The present audit can inspect source contents and coordinate import, while its saved summaries lack the instance outputs needed for that decomposition.

Kapoor and Narayanan organize leakage mechanisms and advocate documenting the evidence supporting evaluation independence [12]. For traffic imagery, repeated scenes and related camera frames are plausible dependencies, so exact hashing and perceptual screening serve different purposes. A duplicate-file test establishes byte identity; a similarity screen identifies candidates for review. Neither supplies a source-video manifest or proves scene-disjoint evaluation.

2.5. Road-Scene Labels, Small Objects, and Task Definitions

Road-scene datasets vary in geography, camera position, class ontology, and task scope. COCO supplies a general object-detection benchmark with a defined evaluation protocol, whereas BDD100K emphasizes diversity in driving videos and supports several perception tasks [1] [2]. The export used in this work attaches seven names to numeric class IDs: triple riding, bike, motorcycle, number plate, vehicle, with helmet, and without helmet [14]. Similar-sounding class names across datasets do not guarantee equivalent annotation rules. As a result, reported scores from one benchmark cannot be treated as direct evidence for a different class inventory.

Earlier traffic applications have targeted specific road users and safety-related objects. The thesis bibliography includes pedestrian detection with lightweight SSD, optimized YOLO-based traffic-sign detection, and small-object helmet detection using a Faster R-CNN variant [22] [27] [51]. A separate study addresses license-plate recognition under occlusion, which combines localization and character-reading concerns that are not captured by a generic vehicle box [52]. These examples show why class definitions and downstream output requirements must be stated explicitly: detecting a plate region is not the same as reading its characters, and detecting a helmet-related region does not establish which rider wears it.

Recent work sharpens these task distinctions. Studies of electric-moped helmets and triple-riding violations consider rider-level safety categories, while plate-recognition systems join a detector with a recognition stage [15]-[17]. Urban pedestrian-and-vehicle studies also use detections as inputs to crossing-behavior analysis rather than treating box scores as the final transport outcome [53]. Traf-

fic-sign studies focus on small, often distant targets, and adverse-weather papers make condition robustness an explicit experimental factor [54]-[56]. Other recent work selects feature paths or model variants for degraded visual conditions and compares detector families under a stated benchmark [57]-[59]. These studies inform the choice of diagnostics, but their classes, datasets, and conditions differ from those of the present export.

Small and infrequent objects deserve explicit attention because aggregate means can hide weak categories. Patch-based refinement has been proposed for early detection of small, distant objects, and TIDE separates several error types that an aggregate AP score combines [50] [60]. In the supplied export, the bike and vehicle IDs dominate annotation support; the contents audit is needed before assigning those counts a road-user meaning. The review therefore treats support and class-wise results as central evidence, while avoiding causal attribution to scale, occlusion, or weather when instance-level predictions and condition annotations are unavailable.

2.6. Recent YOLOv8-Based Traffic Detection and Improvement Strategies

Ten recent primary studies published in 2024-2026 provide specific candidate changes for YOLOv8-based road perception. Their relevance lies in the mechanisms and evaluation questions they examine. Differences in model scale, camera viewpoint, label definitions, and data partitions prevent their reported scores from forming a common benchmark for the archived seven-class experiment.

2.6.1. Efficient Features and Multi-Scale Localization

Liu *et al.* introduce BGS-YOLO, a YOLOv8n-based road-target detector combining weighted bidirectional feature fusion, global attention, and SimAM-C2f processing [61]. Their sequential component comparisons motivate separating the contribution of feature fusion from attention. Its COCO-derived vehicle, bicycle, and pedestrian task has a different label inventory and evaluation protocol from the present export. The fusion and attention designs supply hypotheses for a matched comparison.

Hong *et al.*'s L-YOLO studies lightweight road detection from a YOLOv8s baseline on KITTI remapped to car, pedestrian, and cyclist [62]. The method combines an L-HGNetV2 backbone, a P2 feature level, CStar processing, FPIoU2 regression, and channel pruning. Its contribution is to consider the added computation of fine-resolution features together with later compression. This suggests evaluating accuracy and cost at each stage of a proposed modification, rather than assuming that a smaller final parameter count guarantees lower pipeline latency.

Zheng *et al.* examine roadside detection through YOLO-RC, combining an MBConv-based backbone, CBAM, a C3FB neck, and weighted feature-pyramid fusion [63]. Evaluation on infrastructure-view DAIR-V2X-I and sampled UA-DETRAC frames makes viewpoint part of the experimental setting. For the present study, this motivates recording camera source and perspective before con-

structing evaluation groups. An improvement measured from roadside cameras does not automatically establish the same behavior in forward-facing vehicle imagery.

Wang *et al.* focus on traffic signs using a YOLOv8n detector with attention-scale fusion, a P2 detection layer, lightweight multi-kernel processing, and Wise-IoU regression [64]. Their TT100K ablations distinguish feature, convolution, and loss changes. Traffic signs differ from the current export's classes, but the study provides a concrete way to test whether preserving finer spatial features helps small targets. Such a test here first requires measured box-size strata and verified annotation rules.

Liu *et al.*'s FRCP-YOLO is directly based on YOLOv8n and combines FasterNet feature extraction, residual coordinate attention, a high-resolution branch and small-object head, and PIoU v2 regression [65]. The authors report component and loss comparisons on a remapped KITTI task and an additional BDD100K subset. The study motivates separate hypotheses about feature detail, context, and box regression. Those hypotheses need matched ablations on a clean partition before any of them can explain the AP gaps in the saved reports.

2.6.2. Helmet Targets and Inference Resolution

Deng *et al.*'s Helmet Net combines SENet attention, LConv processing, and WIoU within YOLOv8n, and evaluates sliced inference with SAHI separately [66]. This separation matters: changing the detector and changing the number or resolution of inference windows affect different parts of the pipeline. Its helmet dataset does not establish equivalence to the current rider-related boxes. A corresponding experiment would require an explicit target entity, box convention, and processing-time boundary before comparing ordinary and sliced inference.

2.6.3. Adverse Conditions and Complete Pipeline Cost

Liu *et al.*'s YOLO-DC combines deformable convolution, cross-channel coordinate attention, feature fusion, and a grouped detection head, with evaluation on UA-DETRAC and DAWN [67]. Its ablations show that the grouped head alone can reduce AP even when the full combination improves the study's result. This provides a useful methodological lesson: an additional module is a hypothesis whose effect can depend on its surrounding features. Component and interaction tests should accompany any proposed combination.

Deng *et al.*'s LAGS-YOLO couples luminance-adaptive enhancement and glare suppression with a YOLOv8-based detector using P2 features, BiFPN, and PIoU [68]. Its nighttime BDD100K subset retains car annotations only. The study motivates judging enhancement by its detection outcomes, while preserving the distinction between a nighttime car task and broader road-user recognition. Neither brighter images nor selected night examples establish robustness for the current helmet, rider, or plate labels.

Jaiswal *et al.*'s YOLOv8s-WAMNet combines a hybrid convolution-Transformer representation with cross-dimensional attention, contextual fusion, and

a modified localization objective [69]. The evaluation distinguishes synthetic WEDGE imagery from natural DAWN weather data, and the reported limitations include rare small categories and severe visibility loss. The model uses the s scale, so it is not an interchangeable nano configuration. Its evaluation design supports reporting class support and environmental strata together when assessing condition-related changes.

Jayaraman examines weather classification followed by condition-specific preprocessing, ESRGAN enhancement, and a tuned YOLOv8 detector on DAWN [70]. The reported full chain takes substantially longer than raw detection, illustrating a tradeoff that detector-only timing would conceal. The study also identifies mixed or unseen weather and absent cross-dataset validation as limitations. For an urban-traffic system, a preprocessing candidate therefore needs a matched unenhanced baseline and a timing measurement that includes the enhancement stage.

Table 2 converts these studies into experimental questions for the current evidence gaps.

Table 2. Literature-motivated candidate experiments for a future YOLOv8n urban-road study. All changes require fresh training or inference and independent evaluation; no gain is claimed for the archived model.

Experimental question	Literature motivation	Required comparison in this task
Does finer spatial processing improve localization?	P2 features, small-object heads, and regression changes [64] [65].	Verify box definitions; stratify object size; change one factor and compare per-class AP across IoUs.
Can feature efficiency preserve class performance?	Attention/fusion and compression studies [61] [62].	Retain a matched baseline; record class AP, parameters, and complete pipeline time on common hardware.
Does the result depend on scene source?	Roadside-view fusion and weather/scale analysis [63] [67].	Group source scenes before splitting; report viewpoint and condition strata with adequate support.
Does condition adaptation help the intended classes?	Glare-aware enhancement and hybrid weather features [68] [69].	Compare against unenhanced inputs; distinguish real and synthetic conditions; check each class.
What is the cost of extra inference processing?	Sliced helmet inference and conditional preprocessing [66] [70].	Define the boxed entity; retain filtering settings; time all windows and preprocessing stages.

Table 2 supplies candidate tests rather than an implemented improved detector. The priorities must be selected from a documented error review after annotation and partition repair. This connects the literature to the proposed workflow in Figure 4 without attributing any published module's benefit to the archived YOLOv8n results.

2.7. Traffic-Monitoring Sensors and Platforms

Traffic monitoring predates current deep detectors and has used several sensing modalities. Inductive-loop measurements have been used to discriminate oversaturated traffic states, while earlier video systems extracted traffic parameters with conventional computer-vision software [71] [72]. Video-based incident studies extend this idea by analyzing traffic state over a road segment, showing that a camera pipeline may be designed around an operational event rather than object labels alone [73]. These approaches establish the broader monitoring context but do not provide class-wise object-detection benchmarks.

Aerial imagery has supported wide-area traffic measurement and vehicle tracking. Coifman *et al.* studied roadway monitoring from an unmanned aerial vehicle, and a later review organized the use of UAVs for traffic information collection [74] [75]. Airborne-video work has used ego-motion guidance for tracking, while other studies estimate traffic parameters from wide-area drone footage [76] [77]. HOG-based aerial vehicle detection and frame-difference methods illustrate earlier feature-engineered alternatives to learned detectors [78] [79]. These camera and platform differences affect object scale, viewpoint, and temporal continuity, so results cannot be transferred to the fixed or forward-facing image conditions in the current evaluation without a matched experiment.

2.8. Tracking and Transportation-State Estimation

Object detection is often an intermediate measurement for a larger transportation analysis. Video tracking and speed estimation require temporal association and geometric calibration beyond per-frame boxes [80]; a recent SORT-CNN study likewise combines detection and association for multi-target traffic surveillance [81]. Vehicle-image recognition has also been framed as one component of sustainable urban traffic management [82]. Queue-length estimation from low-penetration mobile sensors and origin-destination estimation from floating-car data use different measurement models again [83] [84]. Trajectory-based congestion estimation uses sequential records rather than image boxes [85]. Hidden-Markov traffic-state identification likewise concerns aggregate states [86]. Such applications require temporal consistency and calibration beyond a class label.

Another group studies how congestion forms, propagates, and dissipates. Cellular-automaton simulations have represented road-jam dynamics and sporadic congestion behavior [87] [88]. Network-based simulation examines congestion propagation mechanisms [89], while speed-based models predict expressway congestion [90]. These works address transport dynamics at temporal and network

scales; they may use observations derived from sensors or trajectories, but they do not validate the per-class precision or localization of the detector studied here.

Deep-learning traffic-state research predicts speed, volume, or congestion from sequential measurements. One line uses deep learning for traffic-speed prediction and large traffic-flow datasets [91] [92]. Other work combines traffic-flow data with Dempster-Shafer reasoning, reports short-term flow prediction, or develops hybrid deep-learning forecasts [93]-[95]. A hybrid model in Transportation Research Part C and a multi-3D-CNN citywide flow model further illustrate spatial-temporal forecasting objectives [96] [97]. These methods predict future network conditions; their predictive targets and error measures should not be conflated with object-detection AP.

Probability-based models and forecasts from online open data address congestion prediction at a different measurement level [98] [99]. PCNN work explicitly uses deep convolutional features for short-term congestion prediction [100]. Spatial-temporal studies also identify congestion-propagation patterns [101]. A rectangle-based model provides another treatment of congestion propagation in urban transportation [102]. This related literature informs the downstream relevance of traffic observations, while also clarifying a necessary boundary: the present audit examines detector outputs; estimating traffic flow, congestion, queue length, travel time, speed, or control benefit requires additional measurements.

2.9. Research Gap and Study Position

The literature connects object localization, relational or temporal interpretation, and transportation-state estimation, but each level requires its own annotation and measurement definitions. An important question for an archived detector experiment is therefore whether its input representation and exported labels support the interpretation assigned to its scores. This study addresses that question through direct archive evidence, a reproducible import-compatibility experiment, and a fixed annotation sample.

Table 3 summarizes ten recent primary studies in road-user detection and difficult visual conditions. The additional YOLOv8-based studies motivate candidate tests in Table 2. Their results retain each source's task and protocol; they do not rank models on the current export.

Table 3 places the algorithm choices in context. The completed contribution is the archive audit and verified coordinate normalization. Figure 4 separates that evidence from the subsequent model experiments required after semantic review and independent partitioning.

3. Materials and Methods

3.1. Study Design and Improvement Workflow

The study audits seven exported class IDs, their coordinate representation, and the archived detection experiment. Standard YOLOv8n provides an available compact baseline with pretrained weights and an established training and valida-

tion pipeline [103]. The logged inference model contains 3,007,013 parameters, 168 fused layers, and an estimated 8.1 GFLOPs. These properties identify the baseline and its recorded computational scale; comparative benefits require matched evaluation.

Table 3. Selected recent primary studies in traffic and road-user detection (2024-2026). Reported results belong to each source's own dataset and protocol and are not directly comparable with this study.

Study	Target and method	Reported finding relevant to this manuscript
Fu *et al.* (2025) [15]	Electric mopeds and helmets; modified YOLOv8n.	The authors report AP increases of 1.37 percentage points for electric mopeds and 3.16 points for helmets, with a 2.27-point increase in overall mAP on their test set.
Deshpande *et al.* (2025) [16]	Two-wheeler localization, rider counting, triple-riding detection, and plate extraction.	A staged surveillance pipeline reports task-specific accuracies for triple-rider, two-wheeler, and plate detection; its outputs depend on rider counting and vehicle association, not only box classification.
Tao *et al.* (2024) [17]	License-plate localization followed by character recognition.	YOLOv5-PDLPR reports 99.4% recognition accuracy and 159.8 frames/s on the CCPD evaluation described by the authors; plate recognition is a separate task from plate-box detection.
Damarasingh *et al.* (2025) [53]	YOLOv8 detection of pedestrians and vehicles on Cape Town urban roads.	The detected road-user observations support a subsequent analysis of pedestrian crossing behavior, illustrating the separation between detection and downstream transport analysis.
Chaman *et al.* (2026) [59]	Comparative YOLOv8-YOLOv12 evaluation for ADAS/ITS tasks.	The study uses a common multi-class dataset and reports accuracy and latency measures, illustrating the value of matched evaluation protocols across model versions.
Du *et al.* (2024) [54]	Small traffic signs; YOLOv8s with multi-scale features, selective-kernel attention, and WIoU.	The authors report mAP@0.50 gains of 3.2 and 5.1 percentage points over YOLOv8s on CCTSDB and TT100K, respectively; the targets and label distribution differ from the seven classes here.
Luo *et al.* (2024) [55]	Traffic signs; YOLOv8-CE with coordinate attention, EIoU, and focal loss.	The article reports a 2.8% mAP increase over its YOLOv8 baseline and a 96-ms inference time on Jetson Nano; this timing is hardware- and pipeline-specific.
Kang *et al.* (2024) [56]	Road-object detection on DAWN in rain, fog, sand, and snow.	The improved YOLOv9 study evaluates weather-specific data augmentation, attention, and loss changes, supporting condition-stratified testing rather than claims based on unlabelled examples.

Continued

Gao *et al.* (2025) [57]	Vehicle detection under fog, rain, and low illumination with dynamic enhancement routing.	YOLO-DER reports mAP@0.50 gains over YOLOv12 on BDD100K, Foggy Cityscapes, and ExDark; its results depend on explicit adverse-condition datasets.
Du *et al.* (2026) [58]	Vehicle and pedestrian detection under adverse weather with MLE-YOLO.	On a custom dataset, the YOLOv11-based method reports higher F1 and mAP alongside reduced parameter count and computation; the study evaluates a different ontology and environmental protocol.

The study combines a retrospective audit of saved runs with a new, source-checked annotation-import experiment. The archive analysis examines class names, row formats, geometry, and partition similarity. The compatibility experiment verifies a coordinate-format normalization without training or evaluating a detector. Historical class metrics and timing are extracted separately from the supplied notebook outputs. Figure 4 connects this completed audit to a proposed improvement cycle that would require fresh training and independent testing.

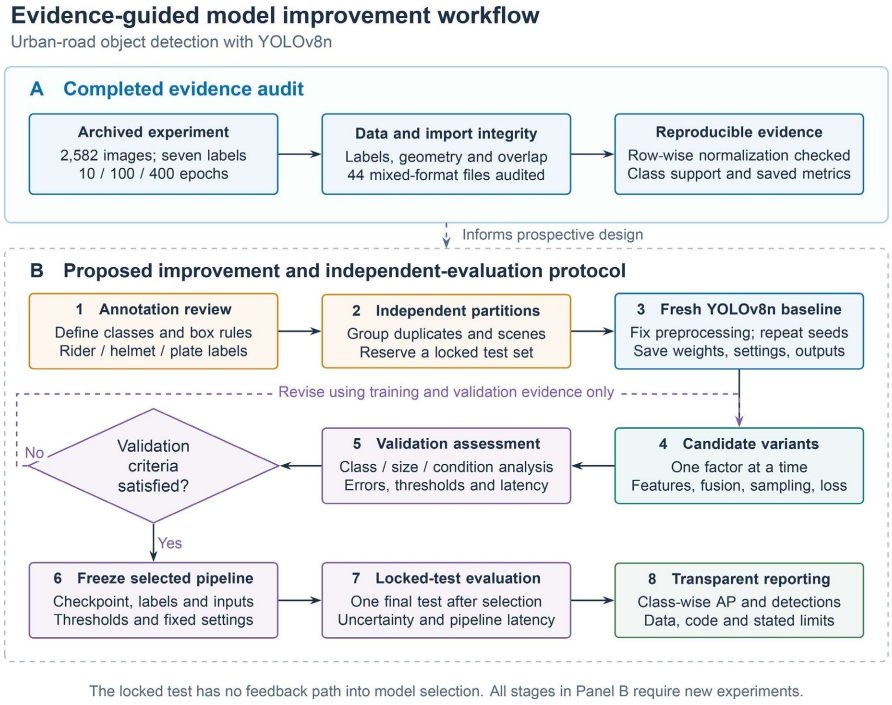


Figure 4. Evidence-guided workflow for YOLOv8n urban-road detection. Panel A summarizes the completed archive, annotation-import, and saved-output audit, including verified format normalization. Panel B presents a prospective improvement protocol with annotation review, independent partitions, controlled validation, and locked-test evaluation; these stages require new experiments.

Figure 4 makes data definition and independence prerequisites for model improvement. Its feedback loop uses training and validation evidence to select candidates, with prespecified criteria for per-class performance and computation.

Only after the checkpoint, score threshold, and image policy are frozen is the locked test evaluated. Test outputs do not feed back into tuning.

In the proposed cycle, candidate changes would be selected after instance-level error review. The literature motivates tests of finer spatial features and box regression [64] [65], efficient fusion or compression [61] [62], and separately timed condition or inference processing [66] [70]. **Table 2** specifies the associated evaluation questions. The current audit identifies evidence gaps; the lower workflow panel specifies prospective experiments.

3.2. YOLOv8n Architecture and Inference

3.2.1. Multi-Scale Feature Extraction

The description is checked against Ultralytics version 8.0.236, which corresponds to the saved run logs [104]. The nano configuration applies depth and width multipliers of 0.33 and 0.25. Its backbone combines convolutional downsampling, C2f feature-processing blocks, and fast spatial-pyramid pooling (SPPF). Its feature aggregation path uses upsampling, concatenation, and further downsampling to combine different resolutions. Detection uses feature levels P3, P4, and P5, with strides 8, 16, and 32 relative to the input. For a square 640-pixel input, their nominal grids are 80 × 80, 40 × 40, and 20 × 20; actual rectangular inference shapes can differ. **Figure 5** shows how multiple feature resolutions feed the detection head. These features provide capacity for targets of different apparent sizes, but the diagram does not demonstrate improved performance for an object-size group.

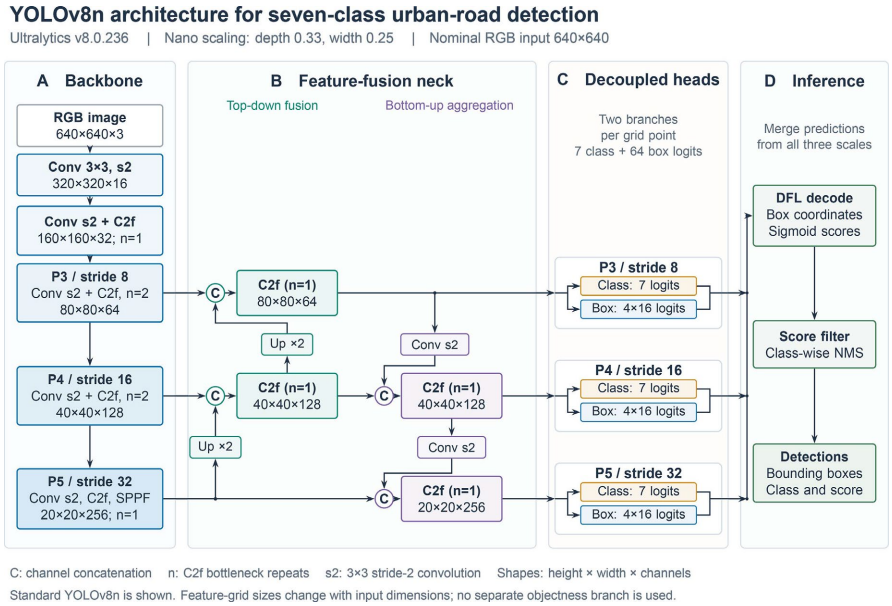


Figure 5. Standard seven-class YOLOv8n architecture based on Ultralytics version 8.0.236: backbone, top-down and bottom-up feature fusion, decoupled prediction heads, and inference processing. Tensor dimensions correspond to a nominal 640 × 640 RGB input; n denotes C2f bottleneck repeats. The diagram describes the baseline rather than an implemented architectural improvement.

3.2.2. Decoupled Prediction and Distributional Box Regression

The head separates class prediction from box regression and uses feature-grid anchor points instead of predefined anchor-box shapes. The standard head uses 16 bins for each of four box-side distances [104]. For one side,

$$\hat{d} = \sum_{k=0}^{15} k p_k, \quad p_k = \frac{\exp(z_k)}{\sum_{j=0}^{15} \exp(z_j)}. \tag{1}$$

Equation (1) converts regression logits z_k into an expected side distance $\hat{d}$ in feature-grid units. Feature stride maps these distances into image coordinates. The framework subsequently filters decoded boxes and class scores using non-maximum suppression. The model-summary flag fused describes inference-layer fusion in the standard implementation.

3.3. Dataset Provenance and Annotation

3.3.1. Dataset Source and Label Representation

The supplied metadata identifies the public Traffic dataset, version 1, from Roboflow Universe workspace rishabh-8qjb7, project traffic-cli7q, and declares a CC BY 4.0 license [14]. The archive contains 1864 training, 482 validation, and 236 test images (2582 total). Its seven exported class names and validation support of 2443 boxes match the saved validator report. The supplied materials do not preserve a dataset checksum recorded with each historical run, so run-to-archive identity cannot be verified cryptographically. One training image has an empty label file and contributes no object instance to annotation totals. The evidence does not establish that the entire export was collected by the authors.

A conventional five-column YOLO detection row contains a class index and normalized box center and size. The export also contains longer polygon rows, which are handled explicitly below. For image width W and height H,

$$(\tilde{x}, \tilde{y}, \tilde{w}, \tilde{h}) = \left(\frac{x_c}{W}, \frac{y_c}{H}, \frac{w}{W}, \frac{h}{H}\right). \tag{2}$$

Equation (2) expresses location and extent relative to image dimensions: (x_c, y_c) is the center and (w, h) the width and height. Class indices follow the exported class-name order. All seven source labels are retained verbatim. The export has no formal class-definition, boundary, or adjudication guide. Exported strings are therefore treated as names attached to numeric IDs. A separate inspection examines the actual contents of selected boxes; it does not infer the provider's original class mapping or establish validated rider-level events.

Table 4 reports annotation rows by exported class ID, not image counts or verified semantic objects. Bike and vehicle account for 2172 of 2443 validation instances (approximately 88.9%). With helmet has 26 instances; motorcycle, number plate, and without helmet have 33 - 41 each. These supports motivate class-level reporting and limit estimate stability. The archive count gives 4326 training bike annotations.

Table 4. Annotation-row counts by exported class ID in the supplied dataset; rows include boxes and polygons.

Class	Training	Validation	Test
Triple riding	541	130	64
Bike	4326	1199	605
Motorcycle	426	33	14
Number plate	529	41	21
Vehicle	3983	973	456
With helmet	282	26	14
Without helmet	338	41	32
Total	10425	2443	1206

3.3.2. Annotation Contents and Box Geometry

The audit reads the supplied ZIP directly, avoiding dependence on a potentially incomplete local extraction. Each image and label file has a SHA-256 manifest. Five-column rows are interpreted as normalized boxes, while longer coordinate sequences are represented by their enclosing rectangle for the row-wise geometry analysis. All 14,074 rows remain traceable to a file and one-based row number. Representation checks include field counts, finite values, class range, normalized coordinates, and duplicate rows; the original files are preserved.

Class support is tabulated both as annotation count and as the number of distinct images containing a label. For normalized width $\tilde{w}_i$ and height $\tilde{h}_i$, the descriptive area and small-area indicator are

$$A_i = 100\tilde{w}_i\tilde{h}_i, \quad S_i = \mathbf{1}\left\{WH\tilde{w}_i\tilde{h}_i < 32^2\right\}. \tag{3}$$

Equation (3) measures area as a percentage of the exported image and records whether its pixel area falls below 32^2. Every supplied image is 640 × 640. The threshold is a descriptive size convention in these resized images, not a claim about original-camera scale or a recomputation of small-object AP. Quartiles use linear interpolation at index $(n-1)p$.

To inspect annotation units, we ordered validation boxes by the SHA-256 digest of their relative image path and row number. Within each class, we retained the first box per image and selected the first twelve distinct images. An image could contribute a region to more than one class. The resulting 84 regions came from 57 distinct validation image files, of which 16 contributed to multiple classes. Selection used neither model confidence nor visual appearance. Each crop follows the row-wise box geometry and has an image and row identifier. An AI-assisted pass produced initial descriptions of the visible contents. The exact AI model and version, prompt text, and prompting sequence are unavailable in the retained inspection records. Ashikur Rahman Mohammad manually checked all 84 regions. The records contain the final descriptive observations but no separate log of uncertain cases or changes made during that check.

The observations describe the contents of the marked region. A plate character or punctuation-like mark occupies only part of a plate, whereas a plate region covers the plate as a unit. We distinguish boxes covering multiple riders from those covering a two-wheeler or a motorcycle and rider. Vehicle regions distinguish miniature vehicles, full-size vehicle photographs, and whole-vehicle regions where the record does so. Head-related regions describe visible head or helmet content without inferring helmet use or assigning a helmet to a particular rider. The supplement provides the full sample and the recorded descriptions. This diagnostic review establishes examples of variation in visible content. Estimating dataset-wide prevalence or inter-reviewer agreement would require a separate design [3] [4].

3.3.3. File-Format Compatibility and Normalization

The pinned v8.0.236 source selects a polygon-processing branch for an entire label file if any row has more than six fields [104]. Let L be a file's rows, r_i one row, and $\mathcal{E}$ the enclosing-rectangle operation on coordinate pairs. Its fresh-import rule can be written as

$$B_i^{\text{file}} = \begin{cases} \mathcal{E}(\mathbf{p}_i), & \max_{r_j \in L} |r_j| > 6, \\ B_i^{\text{row}}, & \text{otherwise}, \end{cases} \tag{4}$$

where B_i^{row} follows the individual row representation and $\mathbf{p}_i$ groups all coordinates after the class ID into pairs. Equation (4) exposes the ambiguity in mixed files: a five-column box becomes the two points (x_c, y_c) and (w, h) in the polygon branch. The resulting rectangle generally differs from its serialized center/size box.

The compatibility experiment reproduces this source condition and executes the unmodified archived segments2boxes and xyxy2xywh coordinate functions on NumPy arrays. The full detector and training implementation are not executed. Reconstructed coordinates are compared with those functions, and IoU measures the change from row-wise interpretation. The original historical label cache is unavailable, so this experiment establishes conditional fresh-import behavior, not the exact targets used in every archived run.

A separate canonical export writes each row in five-column form: existing box rows retain their center/size coordinates, and polygon rows become their row-wise envelopes. Automated checks require unchanged class IDs and row counts and agreement between the canonical import and row-wise geometry. This normalization resolves a representation issue; no semantic label is reassigned. The original archive and all saved model scores remain unchanged. Python 3.12.14, Pillow 12.3.0, and NumPy 2.3.5 were used for the new audit, separately from the historical training environment.

To verify the canonical export through the original image and label checks, we ran the unmodified verify_image_label routine from Ultralytics v8.0.236 in detection mode with seven classes on all 2582 image and canonical-label pairs [104]. We loaded the routine and its helper definitions from the pinned source and used

Python 3.12.14, NumPy 2.3.5, and Pillow 13.0.0 for this verification. The routine can repair JPEGs in place, so each pair was processed in a temporary directory. We independently reconstructed the intended boxes from the original archive, using the coordinates of each box row or the enclosing rectangle of each polygon row. For each image, we compared the returned row count, ordered class IDs, and normalized coordinates at float32 precision. We also recorded empty or missing labels, rejected pairs, duplicate removals, and any changes to the temporary image files.

3.3.4. Partition Integrity and Similarity Screening

We retained the training, validation, and test directories from the export. For each image, we computed SHA-256 hashes of the original file bytes and of the decoded RGB pixel bytes prefixed by the image dimensions and color mode. We grouped matching hashes separately for each representation and compared group membership across the two methods. The supplementary table summarizes group counts, and the evidence files retain filenames and hashes. The SHA-256 checksum of the supplied ZIP archive is 393e6c8b44be70a6f90c1ec01f4885f2e9902bcda7bcf98283bb21296a7cc1a6. We report the historical scores as validation diagnostics from runs that preceded the overlap audit. The effect of the overlap on AP is unknown.

The perceptual screen converts each image to grayscale, resizes it to 9 × 8 using LANCZOS interpolation, and records 64 row-wise left-versus-right intensity comparisons. Hamming-distance thresholds of 0, 2, and 4 define transparent screening settings for every cross-partition image pair. Pair counts and unique participating images are retained separately. For illustration, non-identical-file candidates are ordered by the mean absolute RGB difference between 32 × 32 thumbnails; display ranks one and three are illustrated. These heuristics identify candidates for inspection, not confirmed near-duplicate prevalence or scene labels. The experiment does not tune a detector using these outputs. Partition independence and semantic label validity remain requirements for subsequent performance evaluation [11] [12].

3.4. Image Preprocessing and Data Augmentation

The archive README states that images were auto-oriented and resized to 640 by 640 pixels using a stretch operation, and that no augmentation was applied during export. For an original image of width W and height H, the documented export mapping is

$$x' = 640x/W, \quad y' = 640y/H, \quad w' = 640w/W, \quad h' = 640h/H, \tag{5}$$

where (x, y, w, h) denote the original box center and dimensions in pixels. Equation (5) describes the non-aspect-preserving export resize documented by the dataset provider; it does not describe internal training or video-inference transforms. The saved training image-size parameter is 640, and the training records list online geometric and photometric augmentation settings (**Table 5**). Thus, export-time

transformation and train-time augmentation are distinct stages. Run-specific arguments and transformed sample tensors were not supplied, so details beyond the logged settings cannot be independently reconstructed. Inference logs show a 384-by-640 frame shape for one video, but do not preserve enough arguments to verify the exact runtime resize policy.

Table 5 lists saved training-time settings, separate from the archive's export preprocessing. Photometric changes vary color and illumination, whereas translation, scale changes, and flips vary spatial appearance. Mosaic combines multiple training images. These are configuration choices rather than measured weather or geographic robustness improvements. Augmentation does not create a new validation or test partition. A fixed closure during the final ten epochs also gives the three runs different relative mosaic schedules.

Table 5. Saved train-time augmentation parameters, distinct from the archive's stretch resize; these settings are not measured robustness gains.

Operation	Saved setting
Horizontal flip	Probability 0.5
Translation	Configured fraction 0.1
Scale variation	Configured gain 0.5
HSV variation	Hue 0.015; saturation 0.7; value 0.4
Mosaic	Probability 1.0; closed during final 10 epochs

3.5. Recorded Training and Validation Protocol

3.5.1. Software, Hardware, and Training Configurations

The saved notebook output records Ultralytics 8.0.236, Python 3.8.10, PyTorch 2.0.0 with CUDA 11.8, and an NVIDIA GeForce RTX 4090 with approximately 24 GB of memory. It shows each configuration loading pretrained yolov8n.pt and adapting the class output to seven; later runs do not resume their predecessor's checkpoint. The supplied notebook and an output-extraction script are included in the accompanying package. The script reads the final saved best-checkpoint report in each training cell without executing its code. Original checkpoint weights, historical label caches, run-specific args.yaml files, and complete per-image predictions remain unavailable, preventing a fresh reproduction of model inference from the package.

Table 6 distinguishes shared settings and optimizer choices. Automatic selection yields AdamW for the shorter runs and SGD for 400 epochs. Seed 0 and deterministic mode are configured, but repeated-seed experiments are unavailable. Training duration, optimizer, and relative mosaic exposure vary together; their comparison cannot isolate the effect of epoch count.

3.5.2. Loss Functions and Checkpoint Evaluation

The implementation combines classification, overlap-based regression, and distribution focal loss [43] [104]. Using the recorded loss coefficients.

Table 6. Training settings from the saved configurations.

Setting	10-epoch run	100-epoch run	400-epoch run
Initial weights	yolov8n.pt	yolov8n.pt	yolov8n.pt
Image-size parameter	640	640	640
Batch size	16	16	16
Optimizer (automatic)	AdamW	AdamW	SGD
Logged learning rate	0.000909	0.000909	0.01
Logged momentum parameter	0.9	0.9	0.9
Weight decay (applicable group)	0.0005	0.0005	0.0005
Seed	0	0	0
Mosaic closure (final epochs)	10	10	10

$$\mathcal{L} = 7.5\mathcal{L}_{\text{box}} + 0.5\mathcal{L}_{\text{cls}} + 1.5\mathcal{L}_{\text{DFL}}. \tag{6}$$

Equation (6) reproduces the recorded weighting. Version 8.0.236 uses binary cross-entropy classification and complete-IoU box regression, with DFL supervising side-distance distributions. For a continuous target $0 \le y < 15$ between bins $k = \lfloor y \rfloor$ and $k+1$,

$$\mathcal{L}_{\text{DFL}}(y, p) = -(k+1-y)\log p_k - (y-k)\log p_{k+1}. \tag{7}$$

Equation (7) weights adjacent bins according to the target's position. Together with Equation (1), it explains distributional regression. Neither expression represents an added module in this study.

Ultralytics v8.0.236 selects best.pt using the detection fitness score on the validation split. The score weights mAP@0.50 by 0.10 and mAP@0.50:0.95 by 0.90, with zero weight on precision and recall. The trainer saves the checkpoint when the current fitness equals the best recorded fitness [104]. The saved validator reports evaluate the selected checkpoints on the validation split.

For reporting precision and recall, the validator separately selects the common confidence-curve index that maximizes the smoothed mean F1 across classes [104]. The supplied reports do not retain the numerical confidence threshold. AP is integrated over confidence-ranked detections at the indicated IoU criteria. The archive includes a test directory, and the supplied materials contain only validation reports. The qualitative images and videos have neither matched test annotations nor independently verified checkpoint lineage, so we use them for illustration.

3.6. Class-Specific Evaluation Metrics

For class c, true positives TP_c, false positives FP_c, and false negatives FN_c define precision and recall at the chosen confidence point and IoU 0.50:

$$\begin{aligned} P_c &= \frac{TP_c}{TP_c + FP_c}, \quad R_c = \frac{TP_c}{TP_c + FN_c}, \\ P_{\text{macro}} &= \frac{1}{7}\sum_{c=1}^{7} P_c, \quad R_{\text{macro}} = \frac{1}{7}\sum_{c=1}^{7} R_c. \end{aligned} \tag{8}$$

Equation (8) distinguishes class-wise fractions from the equal-class macro summaries printed in the aggregate report. The implementation interpolates confidence curves before selecting the shared maximum-smoothed-mean-F1 index; its reported values need not equal a pooled object-level fraction. The saved values use the framework's validation-selected common confidence point described above; the threshold value is unavailable. They are not a new fixed-threshold experiment and should not be interpreted as deployment operating characteristics.

For predicted and annotated boxes B_p, B_g,

$$\mathrm{IoU}\left(B_p, B_g\right) = \frac{\left|B_p \cap B_g\right|}{\left|B_p \cup B_g\right|}. \tag{9}$$

Equation (9) measures spatial overlap; a higher matching threshold requires more closely aligned boxes. Class AP conceptually summarizes interpolated precision over recall:

$$\mathrm{AP}_{c,\tau} = \int_0^1 p_{c,\tau}\left(r\right)\mathrm{d}r, \quad \mathrm{mAP}_{\tau} = \frac{1}{7}\sum_{c=1}^{7}\mathrm{AP}_{c,\tau}. \tag{10}$$

Equation (10) distinguishes the class measure from its macro-average over seven labels. Numerical matching and integration follow the saved validator; the integral is explanatory rather than a new metric computation. Stricter AP averages ten overlap thresholds:

$$\mathrm{AP}_{c,50:95} = \frac{1}{10}\sum_{j=0}^{9}\mathrm{AP}_{c,0.50+0.05j}, \quad \mathrm{mAP}_{50:95} = \frac{1}{7}\sum_{c=1}^{7}\mathrm{AP}_{c,50:95}. \tag{11}$$

Equation (11) explains the COCO-style threshold average [1]. Class annotation counts do not directly weight mAP, although they determine the evidence supporting each AP estimate.

Two descriptive contrasts are computed from saved AP:

$$G_c = 100\left(\mathrm{AP}_{c,0.50} - \mathrm{AP}_{c,50:95}\right), \quad \Delta_c = 100\left(\mathrm{AP}^{400}_{c,50:95} - \mathrm{AP}^{100}_{c,50:95}\right). \tag{12}$$

Equation (12) expresses both contrasts in percentage points. G_c describes sensitivity to the overlap criterion, and Δ_c describes a change between recorded configurations. They are arithmetic summaries rather than new detection metrics or causal error decompositions. Per-object outputs would be required for an error analysis such as TIDE [50].

3.7. Timing Definitions and Measurement Scope

The validator separates preprocessing, model inference, and post-processing. Their arithmetic total is

$$T_{\mathrm{stages}} = T_{\mathrm{pre}} + T_{\mathrm{infer}} + T_{\mathrm{post}}. \tag{13}$$

Equation (13) excludes camera acquisition, video decoding, display, communication, and downstream tracking. Sustained throughput distributions, embedded-device power, and camera-to-alert latency are unavailable; processing cost is interpreted within the logged GPU environment.

4. Results

4.1. Image Identity and Cross-Partition Similarity

Direct audit of the supplied ZIP confirms 1864 training, 482 validation, and 236 test images, all 640 × 640 pixels. The 14,074 annotation rows comprise 10,425 training, 2443 validation, and 1206 test entries. The audit found one empty training label file and zero missing label files. It found zero rows with invalid field counts, nonnumeric or non-finite values, invalid class IDs, or invalid normalized box centers or sizes. It also found zero row-wise boxes with edges outside the image by more than 10^{-6} in normalized coordinates and zero duplicate numeric rows within a label file.

SHA-256 hashing of file bytes and decoded RGB data identified the same ten duplicate groups, with identical membership. Four groups cross partition boundaries, two between training and validation and two between training and test. Decoded RGB hashing found zero additional groups, including zero additional cross-partition groups. The known training-validation exposure is two of 482 validation images (0.415%), containing two vehicle rows. Training-test exposure is two of 236 test images (0.847%). Parsed annotation rows agree within the exact-duplicate groups. The effect of this overlap on AP is unknown. **Supplementary Table S1** gives the group counts.

Table 7 reports the additional screening results. At Hamming distance zero, 213 train-validation pairs involve 134 validation images; at distance at most four, 640 pairs involve 256 validation images. The corresponding train-test screen yields 93 and 286 pairs. These sets include exact duplicates and can contain false matches. They are candidate counts rather than measured leakage prevalence or independent-scene counts.

Table 7. Cross-partition candidate pairs from a 64-bit difference-hash screen at three Hamming-distance thresholds.

Threshold	Partitions	Candidate pairs	Images, first	Images, second
0	train-valid	213	190	134
0	train-test	93	87	66
0	valid-test	36	34	32
2	train-valid	422	359	220
2	train-test	180	164	105
2	valid-test	62	56	52
4	train-valid	640	467	256
4	train-test	286	253	131
4	valid-test	84	73	63

Image counts are unique within each partition pair and threshold. Candidate sets include exact duplicates. These counts describe a screen; they are not confirmed near-duplicate or shared-scene prevalence.



Figure 6. Two deterministically selected cross-partition comparisons with different file hashes and similar decoded appearance. Each pair shows its two partition members side by side. Selection and filenames are recorded in the audit manifest.

Figure 6 shows why byte identity alone is an incomplete screen: the displayed pairs have closely matching image content despite different file hashes. The full candidate list remains available for review. Neither these examples nor the difference-hash threshold establishes the source-video identity of every candidate.

4.2. Observed Annotation Contents and Box Geometry

The fixed inspection sample contains twelve selected images for each label. Ten Triple_riding examples enclose punctuation-like plate marks and two enclose rider groups. Eleven bike examples enclose plate characters or marks and one a two-wheeler region. The vehicle sample contains ten miniature-vehicle examples and two full-size vehicle photographs. Eleven number_plate examples enclose plate regions, while one encloses a whole vehicle. Helmet-related examples generally enclose head regions, but their class names and crops do not establish an independently verified rider-helmet relation. One without-helmet crop covers a scene/background region in the encoded image.

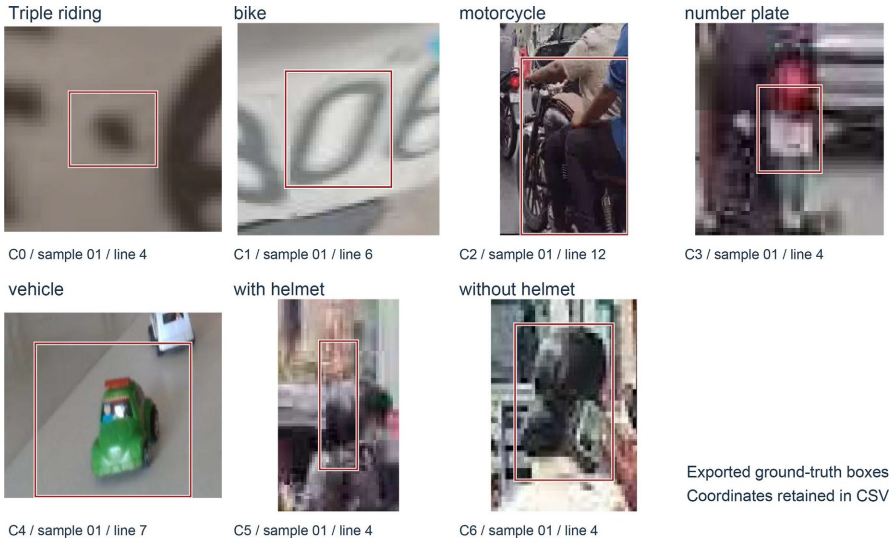


Figure 7. The first hash-selected annotation example for each exported label, using row-wise coordinates. Red rectangles show source annotations; crops are enlarged for inspection. These are observed label contents, not new detector predictions or corrected class definitions.

Figure 7 makes the different units visible, including the character and punctuation regions attached to road-object names. The full sample and descriptive observations appear in **Supplementary** Table S2 and its accompanying figures. Because the sample is a fixed diagnostic selection with possible related images, its counts are not population mislabeling rates. Original provider intent, the cause of any class-map inconsistency, and a complete corrected ontology remain unknown.

Table 8 distinguishes annotation support from image support. For example, 1199 bike rows occur in 254 validation images; 26 with-helmet rows occur in seventeen images. Median relative area is 0.062% for Triple_riding, 0.130% for number_plate, and 2.463% for vehicle. The small-area shares are 86.9%, 78.0%, and 0.8%, respectively. These are geometric descriptions of the exported rows. In light of the observed punctuation and miniature content, they cannot be interpreted directly as rider-group scale or road-vehicle representativeness.

Table 8. Validation image support and row-wise bounding-box geometry in the supplied export.

Label	Images	Boxes	Area median (IQR), %	Small, %
Triple riding	130	130	0.062 (0.046 - 0.087)	86.9
bike	254	1199	0.577 (0.410 - 1.066)	3.9
motorcycle	16	33	4.960 (1.483 - 14.984)	3.0
number plate	13	41	0.130 (0.075 - 0.197)	78.0
vehicle	214	973	2.463 (1.729 - 4.662)	0.8
with helmet	17	26	0.507 (0.185 - 1.023)	30.8
without helmet	24	41	1.266 (0.416 - 2.061)	17.1

Area is a percentage of the exported 640 × 640 image. IQR gives the 25th-75th percentiles. Small denotes area below 32^2 pixels in this export. These are annotation descriptors, not size-stratified AP. Each image can contain several labels.

4.3. Conditional Import Behavior and Verified Normalization

The export contains 14,002 five-column box rows and 72 polygon rows. Polygon rows occur in 58 files; 44 of those files also contain conventional boxes.

Table 9 shows the consequence of Equation (4). The pinned file-level rule reinterprets 74 box rows as two-point polygons: 52 in training, twenty in validation, and two in test. Every affected row has class ID 1 (bike). All 74 reconstructed boxes have IoU below 0.50 with their row-wise interpretation; the median is 0.029 for the twenty validation rows. These twenty rows constitute approximately 1.7% of the 1199 validation entries with that ID. The result concerns coordinate interpretation, not a human-verified localization error rate.

Figure 8 displays the spatial change introduced by treating center/size values as coordinate pairs. The reconstruction agrees with the archived coordinate functions to a maximum absolute normalized-coordinate difference of 6.82×10^{-8}. The

historical cache is absent, so the figures do not prove which cached targets every saved training or validation call used.

Table 9. Annotation representation and conditional fresh-import changes under the pinned v8.0.236 parser.

Split	Polygon files	Mixed files	Polygon rows	Box rows changed	Median IoU
Training	41	33	50	52	0.087
Validation	13	9	16	20	0.029
Test	4	2	6	2	0.143

A mixed file contains both five-column box rows and longer polygon rows. Changed rows are five-column entries dispatched as two-point polygons by the file-level rule; all 74 have class ID 1 (bike). IoU compares row-wise box interpretation with the reconstructed parser result. Historical label caches are unavailable.

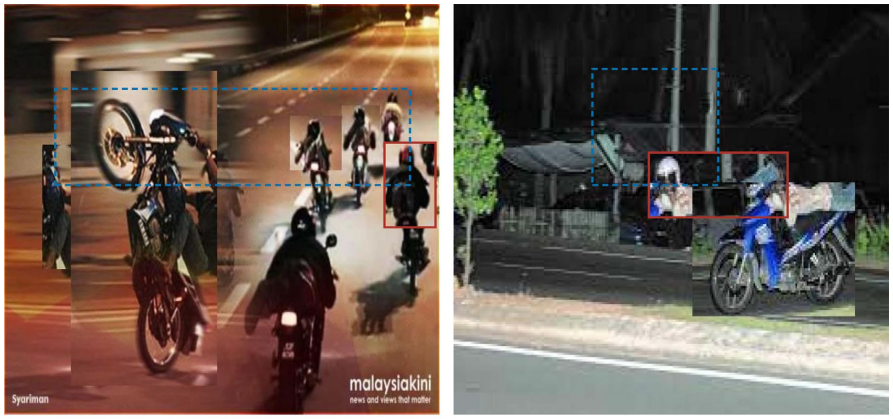


Figure 8. Two hash-selected validation examples illustrating the file-format ambiguity. Solid red boxes use each row's conventional center/size interpretation; dashed blue boxes use the reconstructed fresh-import rule from v8.0.236.

The pinned verify_image_label routine accepted all 2582 image and canonical-label pairs and returned all 14,074 intended rows. The returned counts were 10,425 for training, 2443 for validation, and 1206 for test. Ordered class IDs matched for every image. The maximum absolute normalized-coordinate difference from the intended row-wise boxes was zero at float32 precision. The verifier flagged one empty training label file, zero missing label files, and zero rejected pairs. It removed zero duplicate rows and repaired zero images. These results verify the coordinate representation of the canonical export. The descriptive content observations and the historical model scores address separate questions.

4.4. Historical Class-Specific Validation Results

The 400-epoch configuration has the highest reported aggregate AP values among the three saved validation reports. Table 10 presents its class-specific results with validation support.

Table 10 shows support ranging from 26 with_helmet boxes to 1199 bike boxes, a 46.1:1 ratio. bike and vehicle account for 2172 of 2443 validation instances (88.9%). AP@0.50:0.95 ranges from 26.8% for without_helmet to 92.4% for vehicle. These values describe the exported class IDs in the supplied validator output. The annotation-content findings above preclude interpreting the corresponding strings as seven validated road-object categories.

Figure 9 visualizes the gap between the two overlap criteria. The largest gaps are for Triple_riding (47.8 percentage points), with_helmet (43.8 points), and number_plate (43.6 points); the vehicle gap is 7.0 points. These differences show sensitivity to the chosen localization criterion, but do not identify whether scale, occlusion, annotation ambiguity, or another error type caused it.

Table 10. Results by exported label from the saved 400-epoch validation report, class support n and share of validation objects (metrics in %). G_c is the AP@0.50–AP@0.50:0.95 gap in percentage points. Validation is not independent of training.

Class	Support n (share, %)	Precision	Recall	AP@0.50	AP@0.50:0.95	G_c
Triple riding	130 (5.3)	94.4	90.7	96.8	49.0	47.8
Bike	1199 (49.1)	97.4	96.7	98.8	87.9	10.9
Motorcycle	33 (1.4)	91.4	96.1	92.0	60.7	31.3
Number plate	41 (1.7)	81.5	70.7	73.3	29.7	43.6
Vehicle	973 (39.8)	98.5	98.7	99.4	92.4	7.0
With helmet	26 (1.1)	71.1	75.6	75.2	31.4	43.8
Without helmet	41 (1.7)	83.3	53.7	68.3	26.8	41.5

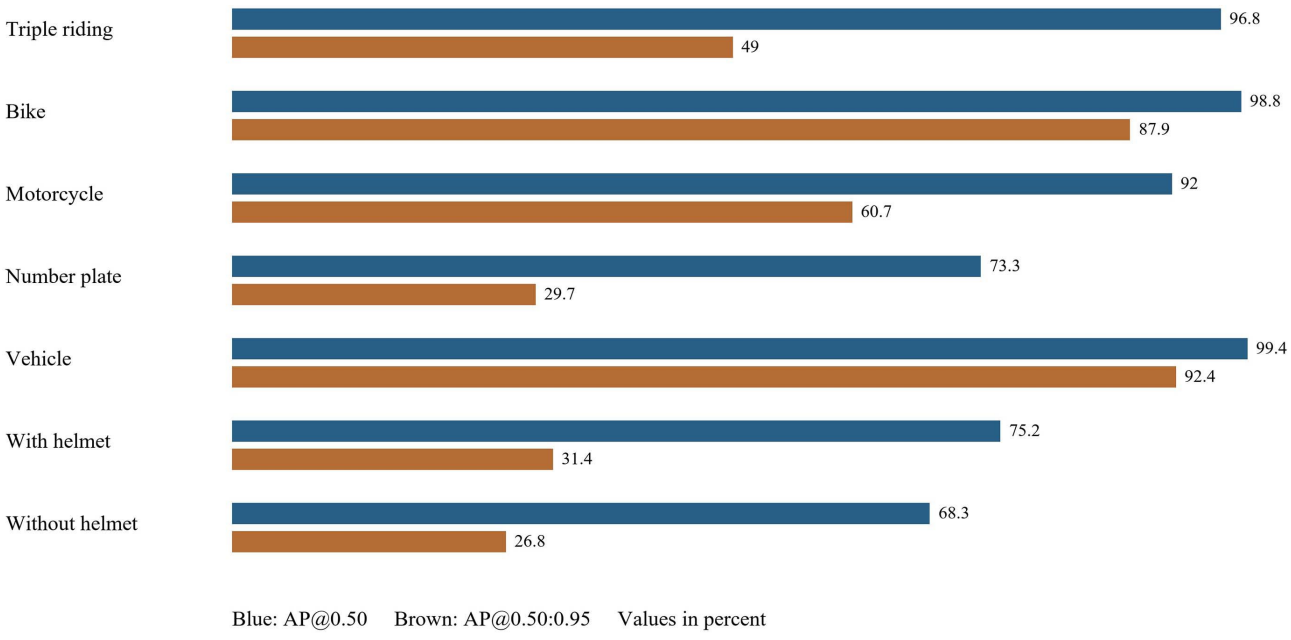


Figure 9. Saved validation AP at IoU 0.50 and 0.50:0.95 by exported class name. Annotation contents are heterogeneous; the bars do not establish seven validated road-object categories.

For the exported label without_helmet, saved class recall is 53.7%, compared with a class-macro recall of 83.2%; recall is 75.6% for with_helmet and 70.7% for number_plate. As described in Section 3, the reported P/R values use a common confidence point selected by maximum smoothed mean F1 on this same validation split. Its numeric threshold is unavailable. These figures therefore do not establish performance at a preselected enforcement or deployment threshold.

The class profile also distinguishes recognition at a permissive overlap criterion from performance across stricter criteria. For example, Triple_riding has AP@0.50 of 96.8% while its threshold-averaged AP is 49.0%. Equation (11) explains why the latter score places additional demands on box alignment. Neither percentage is an image-classification accuracy, and their difference is not a count of incorrectly localized instances. Identifying the underlying errors would require the matched predictions and annotations [50].

4.5. Comparison of Recorded Training Configurations

4.5.1. Aggregate Validation Results

Table 11 compares the aggregate validator outputs for the three saved configurations.

Table 11. Saved best-checkpoint validation summaries in percent; the validation split contains exact duplicates across partitions.

Epochs	Optimizer	P_{macro}	R_{macro}	mAP@0.50	mAP@0.50:0.95
10	AdamW	82.8	75.0	80.5	46.8
100	AdamW	84.4	83.2	83.3	52.5
400	SGD	88.2	83.2	86.3	54.0

Table 11 reports mAP@0.50:0.95 of 46.8%, 52.5%, and 54.0% for the 10-, 100-, and 400-epoch configurations, respectively. The 100-to-400 difference is 1.5 percentage points. Because optimizer, learning rate, training duration, and relative mosaic exposure differ, these numbers compare complete recipes and cannot isolate an epoch effect. Validation also informed checkpoint selection, limiting comparative inference.

The two intervals have different descriptive patterns. From 10 to 100 epochs, the saved reports differ by 8.2 points in recall and 5.7 points in strict mAP. From 100 to 400 epochs, precision increases by 3.8 points and AP@0.50 by 3.0 points, while aggregate recall remains 83.2% after rounding to one decimal place. Equal rounded recall does not imply that the same objects were recovered, particularly because the operating point is selected separately for each report. Each run starts from pretrained weights rather than resuming the preceding run, so the rows describe separate optimization histories.

4.5.2. Class-Specific Changes

Table 12 shows how class AP@0.50:0.95 varies across the same saved configurations.

Table 12. Class AP@0.50:0.95 (%) across configurations; Δ_c is the 400-minus-100 difference in percentage points.

Class	Validation objects	10 epochs	100 epochs	400 epochs	Δ_c
Triple riding	130	39.5	44.6	49.0	+4.4
Bike	1199	77.3	84.3	87.9	+3.6
Motorcycle	33	55.3	59.7	60.7	+1.0
Number plate	41	23.9	28.1	29.7	+1.6
Vehicle	973	81.7	89.3	92.4	+3.1
With helmet	26	26.2	29.1	31.4	+2.3
Without helmet	41	23.9	32.1	26.8	−5.3

Table 12 shows that the class profiles are not uniform. For without_helmet, strict AP increases from 23.9% to 32.1% and then decreases to 26.8%; its 100-to-400 difference is −5.3 points. The remaining six classes have positive arithmetic differences over this interval. These are descriptive contrasts only: there is one saved run per configuration, settings differ, and no per-image predictions or repeated-seed results are available.

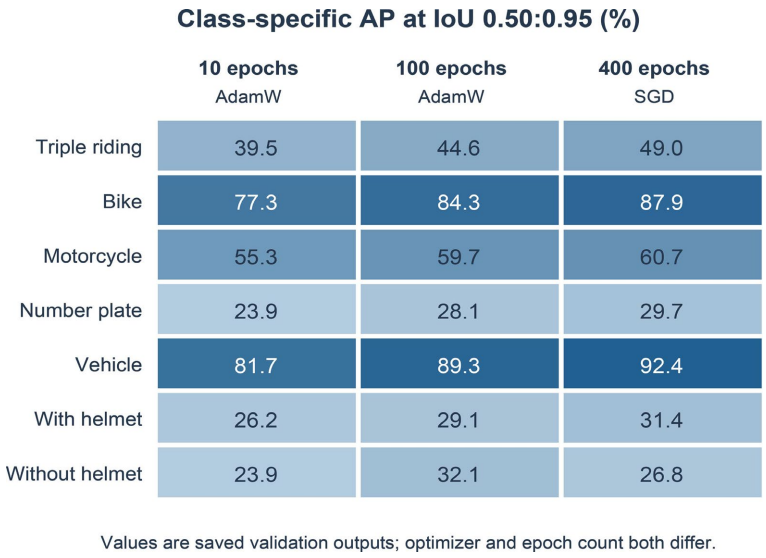


Figure 10. Class-specific AP@0.50:0.95 for the three saved configurations; the configurations are not controlled repetitions.

Figure 10 makes the minority-class variation and the without_helmet reversal visible. It supports reporting per-class results in future experiments but does not establish that one training schedule is statistically or operationally superior.

4.6. Training and Validation Histories

Figures 11-13 retain the supplied training and validation plots. Their smoothed traces summarize the same runs and are not uncertainty bands. Losses describe optimization, whereas AP summarizes ranked detections under spatial matching.

Exact reported scores come from the saved best-checkpoint tables, not visual estimates of a final plotted epoch.

4.6.1. 10-Epoch Configuration

Figure 11 shows declining training losses and generally increasing validation metrics, with non-monotonic early trajectories. Several quantities are still changing at the end. This is a record of pretrained-model adaptation; it neither establishes a sufficient training budget nor isolates the benefit of pretraining.

4.6.2. 100-Epoch Configuration

Figure 12 shows steep initial loss changes followed by more gradual evolution. Recall and AP@0.50 become comparatively stable later, while precision fluctuates. The aggregate histories do not identify which image sources or annotation units account for the changes; Table 12 supplies the complementary saved class profile.

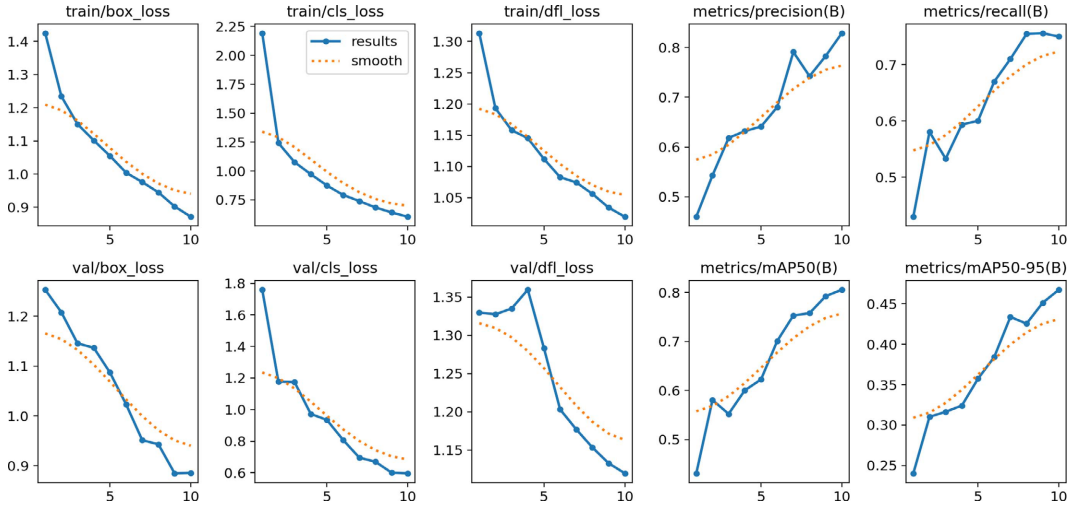


Figure 11. Saved training and validation curves for the 10-epoch configuration.

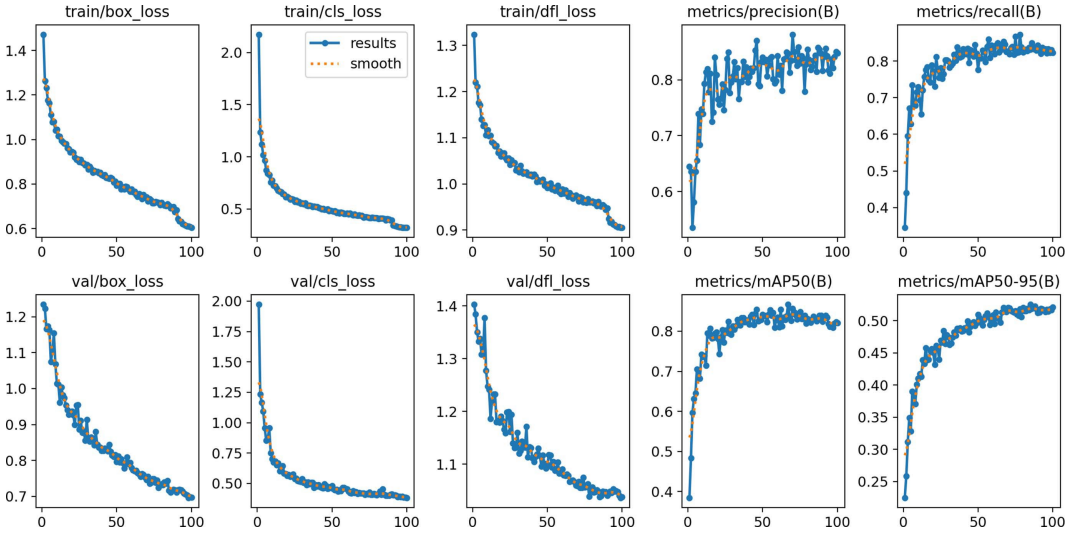


Figure 12. Saved training and validation curves for the 100-epoch configuration.

4.6.3. 400-Epoch Configuration

Figure 13 shows continuing loss reductions and broad plateaus in aggregate recall and AP. A sustained late increase in validation loss is not visible. Nevertheless, the plot does not resolve semantic target consistency, reconstruct historical import targets, or assess independent scenes. Differences in optimizer and augmentation exposure remain part of the comparison in Table 6.

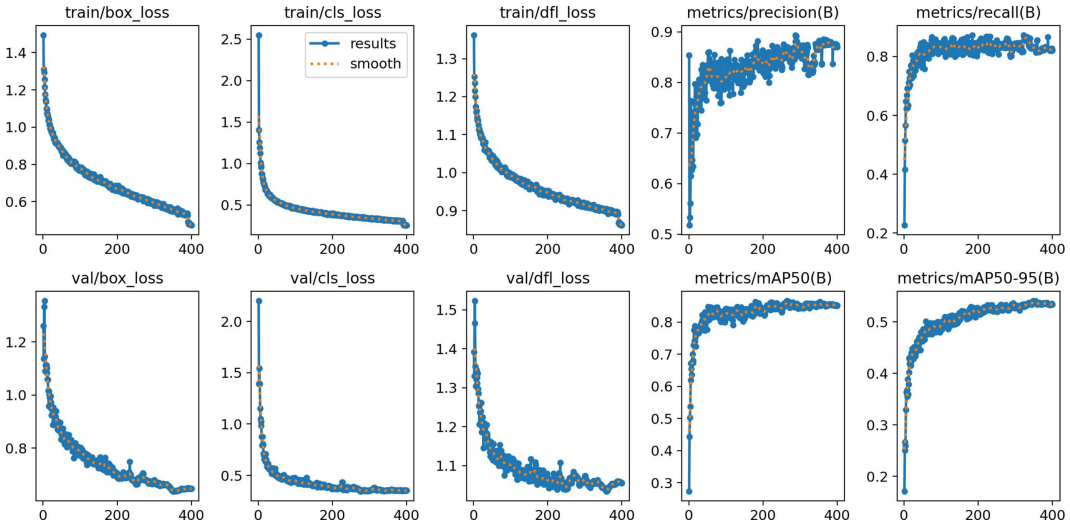


Figure 13. Saved training and validation curves for the 400-epoch configuration.

4.7. Recorded Processing Time

The saved validator reports separate preprocessing, inference, and post-processing times. Table 13 reproduces those entries.

Table 13. Per-Image timing reported by the 400-epoch validator on an NVIDIA RTX 4090.

Processing stage	Time (ms/image)
Preprocessing	0.6
Model inference	5.3
Post-processing	2.3
Arithmetic total	8.2

Table 13 gives 8.2 ms per image as the arithmetic sum of the three reported stages. The 5.3-ms inference component is the model-stage timing on an RTX 4090; neither figure includes camera acquisition, video decoding, communications, display, or downstream processing. The evidence therefore does not establish end-to-end or embedded-platform latency.

Within this logged total, inference accounts for approximately 64.6%, post-processing for 28.0%, and preprocessing for 7.3%; rounding explains the small departure from 100%. Preprocessing and post-processing together therefore account for about 35.4% of the included stages. Equation (13) makes this measurement

boundary explicit and shows why model inference alone incompletely describes the recorded processing cost. These entries are stage summaries without repeated timing trials or latency distributions. Their reciprocal is not a measured streaming frame rate, and they do not establish response-time guarantees under changing scene density, input resolution, or system load.

4.8. Qualitative Traffic-Scene Predictions

Figure 14 presents six supplied traffic-scene prediction images. They complement the numerical summaries by showing the appearance of the supplied output overlays. Their acquisition conditions, sampling procedure, and checkpoint lineage are not documented sufficiently for a separate quantitative evaluation.



Figure 14. Supplied YOLOv8n vehicle predictions from six traffic-scene frames. The images are illustrative; no matched ground truth or sampling protocol was supplied.

Figure 14 contains forward-facing views with vehicles at different apparent scales and positions, including cross traffic, nearby vehicles truncated by image boundaries, and a visibly darker night scene. The montage predominantly displays the broad vehicle label, so it does not visually substantiate the full seven-class profile. Visible scene content without a displayed box cannot be counted as a false negative because exhaustive annotations and the original filtering settings are absent. Similarly, a confidence label does not verify a box's category or spatial agreement with ground truth. These examples identify situations for a future annotated error review; condition-specific AP, tracking continuity, and minority-class performance require additional records.

5. Discussion

5.1. Semantic Validity of a Class-Specific Score

The principal finding is that the interpretation of a class score depends on the regions carrying that numeric ID. A detector can match its supplied targets while the exported class name suggests a different application. The registration-character examples under bike and punctuation-like examples under Triple_riding demonstrate this distinction directly. The vehicle sample also mixes miniature objects and full-size vehicle photographs. Consequently, strict AP of 87.9% for the bike name and 92.4% for the vehicle name cannot establish equivalent performance for real road users. This conclusion concerns the supplied evidence; it does not establish when or why the provider's class mapping acquired these associations.

The geometric summaries reinforce the need to inspect the unit of annotation. The median Triple_riding box occupies only 0.062% of an exported image, but the sample shows that many selected regions are punctuation-like marks. Calling this a small-rider-group detection problem would assign an unsupported meaning to the measurement. Similarly, the 88.9% combined support of the bike and vehicle IDs is an annotation-frequency statement, not a verified share of road vehicles. Dataset documentation and label-error research motivate this separation between a numerical benchmark target and its intended application [3] [11].

The single-author check covers all selected regions. The diagnostic sample supports descriptions of those regions and cannot establish dataset-wide prevalence or inter-reviewer agreement. For the helmet-related names, the observations concern visible head or helmet content. Establishing helmet use, rider association, or traffic violations would require separate annotations and criteria. A corrected ontology would require provider information and a documented review of the relevant image collection.

5.2. Coordinate Import and Representation Repair

The second finding is an implementation-specific representation hazard. The supplied files contain 72 polygon rows, and the pinned loader decides whether to process coordinates as polygons at the file level. In mixed files, this rule interprets conventional center and size values as two points. The compatibility experiment identifies the resulting change for 74 box rows. The separate verifier run confirms that the canonical five-column export returns the intended class IDs and box coordinates for every image.

This intervention is narrow and testable. It preserves every class ID and annotation row, converts each polygon into its enclosing rectangle, and removes the measured disagreement between row-wise geometry and subsequent box import. It does not decide whether a region called bike actually contains a two-wheeler. Format normalization and semantic relabeling are separate operations, and neither retroactively changes the metrics printed by an earlier experiment.

The historical label cache is particularly important. Matching software version, class names, and support counts provides useful provenance, but cannot prove which parsed targets were cached during each saved run. The reconstructed import result is therefore conditional on a fresh import of this exact archive. No performance degradation, gain after normalization, or explanation of helmet AP is inferred from it. All affected representation rows have the bike ID, so the finding alone does not explain the weak helmet scores.

5.3. Partition Independence and Statistical Support

Exact hashing establishes a small, precisely identified overlap: two validation images and two test images have training counterparts. It does not estimate their effect on AP. Perceptual screening exposes a much larger set of candidates and supplies examples of closely matching content with different file hashes. Candidate counts depend on the declared hash threshold and can contain false matches; they are not confirmed shared-scene counts. A future split should use source provenance and reviewed similarity groups before training, following the general principle that evaluation independence is a property to establish rather than assume [12].

The distinction between image and object support also matters. Twenty-six with-helmet rows occur in seventeen validation images, while many bike rows can occur within a single plate image. Objects in one frame and near-repeated scenes are not independent repetitions. Rounded class summaries and annotation counts cannot provide valid AP uncertainty intervals. Preserved predictions and source groups would permit an image- or scene-aware analysis; no binomial interval or significance test is manufactured from the available counts.

5.4. Training Recipes and Transport Interpretation

The saved class profile remains useful as an auditable record. The macro-average strict AP increases by 1.5 points between the 100- and 400-epoch reports, while the without-helmet label decreases by 5.3 points. Equal weighting of classes explains how these changes can coexist, but the comparison does not isolate a training factor: optimizer, learning rate, duration, and relative mosaic exposure differ. Nor do the AP@0.50-AP@0.50:0.95 gaps identify a particular error cause. A framework such as TIDE would require matched predictions and targets for its decomposition [50].

The prior DETR and Faster R-CNN frameworks remain relevant alternatives for a subsequent, well-defined road-object task. Their architecture descriptions and historical dataset reports are retained as context, with no unmatched numerical ranking. Any comparison should share validated class definitions and independent data, while allowing documented model-appropriate optimization. The standard YOLO summary flag fused denotes inference-layer fusion; it does not establish a DETR-YOLO hybrid.

For transportation use, object localization is one measurement stage. Rider as-

sociation, plate reading, tracking, counting, queue estimation, and control outcomes require their own labels and validation. The recorded RTX 4090 timing describes three validator stages and does not establish camera-to-alert latency or an embedded-system guarantee. Intended use and evaluation context should accompany any later released model [4].

5.5. Scope, Limitations, and Subsequent Evaluation

The study examines how visible label contents, coordinate import, and partition structure affect the interpretation of class-specific traffic-detection scores in one supplied export and archived notebook. One author checked every region in the AI-assisted visual inspection. Provider intent and independent annotation adjudication remain unavailable. The audit does not establish population mislabeling rates, historical cached targets, score bias, or performance on independent scenes.

The completed normalization makes one processing step inspectable but leaves the larger model study prospective. A subsequent experiment should first verify and repair class definitions, retain source-group information, create independent partitions, and establish a fresh baseline. Candidate architecture or training changes should follow measured error types, with controlled settings, repeated seeds, retained predictions, and a frozen validation-selected operating point. A final test should assess the selected pipeline once. These steps define the model-improvement portion of **Figure 4**; the present results do not claim it has been executed.

6. Conclusions

This study provides a reproducible class-specific audit of saved YOLOv8n experiments and their supplied traffic-data export. The archive contains 2582 images and 14,074 annotation rows. Of these, 72 rows contain polygons, and 44 files mix polygons with conventional boxes. The recorded software version's fresh-import rule would reinterpret 74 five-column rows as two-point polygons. After normalization, the original verifier returned all 14,074 intended rows with identical class IDs and coordinates at float32 precision. The result concerns label import. Detector performance on the normalized export has not been measured.

The annotation sample also restricts the meaning of the exported class names. Registration characters occur under bike, punctuation-like marks occur under Triple_riding, and the vehicle sample includes miniature objects. The saved class AP values therefore cannot be treated as validated performance for seven consistently defined urban-road categories. Box counts, image support, and apparent size must be interpreted together with the observed annotation contents.

The archived 400-epoch report records mAP@0.50 of 86.3% and mAP@0.50:0.95 of 54.0%. Its strict class AP ranges from 26.8% to 92.4%; the 100-to-400 comparison combines a 1.5-point aggregate increase with a 5.3-point decrease for the without-helmet label. These remain descriptive differences between complete saved recipes. Exact cross-partition duplicates, perceptual similarity candidates,

validation-based selection, and unavailable historical targets prevent independent generalization or causal interpretation. The logged 8.2-ms stage sum is limited to the recorded RTX 4090 validation environment.

The practical outcome is an inspectable audit package: source and annotation manifests, fixed inspection examples, import-compatibility checks, normalized labels, and a notebook-output extractor. For intelligent-vehicle research, its central implication is that semantic consistency and coordinate compatibility must be established before a detector score is interpreted as road-recognition evidence. A subsequent model study should validate the class definitions, repair and group the data before training, retain its predictions and settings, and evaluate a frozen pipeline on independent scenes.

Data and Code Availability

The supplied Traffic dataset export was identified in its metadata as Roboflow Universe workspace rishabh-8qjb7, project traffic-cli7q, version 1. Its export metadata declares CC BY 4.0 [14]. The authors identify this export as the data used for the archived YOLOv8n experiment. Current public access to the original download page could not be verified during proof checking.

Its export metadata declares CC BY 4.0 [14]. The authors identify this export as the data used for the archived YOLOv8n experiment.

The accompanying source archive includes the supplied notebook with saved outputs; an extractor for its best-checkpoint reports; the complete image/annotation manifests; coordinate-format and import checks; a canonical label export; the fixed annotation-inspection manifest; and code reproducing the audit tables and figures. The canonical labels retain class IDs and annotation counts and do not constitute semantic relabeling. The pinned evaluator and coordinate-source files are supplied with their source URLs, checksums, and upstream license. Reproduction instructions and dependency versions are included with the code. These materials permit the archive analyses and saved-output extraction to be repeated without a detector checkpoint. Original weights, complete prediction records, run-specific argument files, and an independent test report remain unavailable, so fresh model inference and the historical cached targets cannot be reproduced from this package.

Funding

This work was supported by the National Natural Science Foundation of China under Grants U25A20296 and 52475052.

Author Contributions

Ashikur Rahman Mohammad: Data curation, Software, Formal analysis, Investigation, and Writing—original draft. Li Jun: Supervision and Funding acquisition. Yanfeng Han: Supervision and Funding acquisition. Sheikh Sakibul Hasan: Formal analysis and Investigation (supporting; assistance with experiments and sim-

ulations).

Acknowledgements

The authors accept responsibility for verification of the literature review, manuscript text, audit code, figures, and observations from the fixed annotation sample. Numerical evidence is traceable to the supplied archive, notebook outputs, and documented audit computations. The authors are accountable for the accuracy and interpretation of the reported findings and for the final content of the manuscript.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Lin, T., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., *et al.* (2014) Microsoft COCO: Common Objects in Context. In: Fleet, D., *et al.*, Eds., *Computer Vision—ECCV* 2014, Springer International Publishing, 740-755. https://doi.org/10.1007/978-3-319-10602-1_48

[2] Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., *et al.* (2020) BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning. 2020 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Seattle, 13-19 June 2020, 2633-2642. https://doi.org/10.1109/cvpr42600.2020.00271

[3] Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J.W., Wallach, H., III, H.D., *et al.* (2021) Datasheets for Datasets. *Communications of the ACM*, **64**, 86-92. https://doi.org/10.1145/3458723

[4] Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., *et al.* (2019) Model Cards for Model Reporting. *Proceedings of the Conference on Fairness, Accountability, and Transparency*, Atlanta, 29-31 January 2019, 220-229. https://doi.org/10.1145/3287560.3287596

[5] Redmon, J., Divvala, S., Girshick, R. and Farhadi, A. (2016) You Only Look Once: Unified, Real-Time Object Detection. 2016 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, 27-30 June 2016, 779-788. https://doi.org/10.1109/cvpr.2016.91

[6] Ren, S., He, K., Girshick, R. and Sun, J. (2017) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **39**, 1137-1149. https://doi.org/10.1109/tpami.2016.2577031

[7] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A. and Zagoruyko, S. (2020) End-to-End Object Detection with Transformers. In: Vedaldi, A., *et al.*, Eds., *Computer Vision—ECCV* 2020, Springer International Publishing, 213-229. https://doi.org/10.1007/978-3-030-58452-8_13

[8] Wang, C., Bochkovskiy, A. and Liao, H.M. (2023) YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. 2023 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Vancouver, 17-24 June 2023, 7464-7475. https://doi.org/10.1109/cvpr52729.2023.00721

[9] Zhao, Y., Lv, W., Xu, S., Wei, J., Wang, G., Dang, Q., *et al.* (2024) DETRs Beat YOLOs

on Real-Time Object Detection. 2024 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Seattle, 16-22 June 2024, 16965-16974. https://doi.org/10.1109/cvpr52733.2024.01605

[10] Huang, S., Lu, Z., Cun, X., Yu, Y., Zhou, X. and Shen, X. (2025) DEIM: DETR with Improved Matching for Fast Convergence. 2025 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Nashville, 11-15 June 2025, 15162-15171. https://doi.org/10.1109/cvpr52734.2025.01412

[11] Northcutt, C.G., Athalye, A. and Mueller, J. (2021) Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks. *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 7-10 December 2021, 1-13. https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/f2217062e9a397a1dca429e7d70bc6ca-Abstract-round1.html

[12] Kapoor, S. and Narayanan, A. (2023) Leakage and the Reproducibility Crisis in Machine-Learning-Based Science. *Patterns*, **4**, Article ID: 100804. https://doi.org/10.1016/j.patter.2023.100804

[13] Mohammad, A.R. (2024) Intelligent Vehicle for Urban Road Traffic Recognition System Using Deep Learning Algorithm. Master's Thesis, Chongqing Jiaotong University.

[14] Roboflow Contributor Rishabh-8qjb7 (n.d.) Traffic Dataset, Version 1. Dataset Identifier Traffic-cli7q. Seven-Class Object-Detection Dataset; CC BY 4.0 License as Stated in the Supplied Export Metadata.

[15] Fu, S., Wei, D. and Zhou, L. (2025) Improved YOLOv8-Based Algorithm for Detecting Helmets of Electric Moped Drivers and Passengers. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, **29**, 349-357. https://doi.org/10.20965/jaciii.2025.p0349

[16] Deshpande, U.U., Shanbhag, S., Patil, R., Chate, R.A.A., Das Chagas Silva Araujo, S., Pinto, K., *et al.* (2025) Automatic Two-Wheeler Rider Identification and Triple-Riding Detection in Surveillance Systems Using Deep-Learning Models. *Discover Artificial Intelligence*, **5**, Article No. 104. https://doi.org/10.1007/s44163-025-00263-3

[17] Tao, L., Hong, S., Lin, Y., Chen, Y., He, P. and Tie, Z. (2024) A Real-Time License Plate Detection and Recognition Model in Unconstrained Scenarios. *Sensors*, **24**, Article No. 2791. https://doi.org/10.3390/s24092791

[18] Krizhevsky, A., Sutskever, I. and Hinton, G.E. (2017) ImageNet Classification with Deep Convolutional Neural Networks. *Communications of the ACM*, **60**, 84-90. https://doi.org/10.1145/3065386

[19] Szegedy, C., Ioffe, S., Vanhoucke, V. and Alemi, A. (2017) Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, **31**, 4278-4284. https://doi.org/10.1609/aaai.v31i1.11231

[20] He, K., Zhang, X., Ren, S. and Sun, J. (2016) Deep Residual Learning for Image Recognition. 2016 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, 27-30 June 2016, 770-778. https://doi.org/10.1109/cvpr.2016.90

[21] Bengio, Y. (2009) Learning Deep Architectures for AI. *Foundations and Trends® in Machine Learning*, **2**, 1-127.

[22] Qian, W. and Wang, J. (2022) Pedestrian Target Detection Based on Lightweight SSD Algorithm. *Computer Simulation*, **39**, 487-491.

[23] Ouyang, J., Wang, Z. and Liu, S. (2022) Improved YOLO v4 Target Detection Method with Multi-Scale Features. *Journal of Jilin University* (*Science Edition*), **60**, 1349-

1355. (In Chinese) https://doi.org/10.13413/j.cnki.jdxblxb.2021501

[24] Xie, Y., Bao, Z., Zhang, N., *et al.* (2022) Target Detection Algorithm Based on Feature Optimization and Deep Fusion. *Journal of Zhejiang University* (*Engineering Edition*), **56**, 2403-2415. (In Chinese) https://doi.org/10.3785/j.issn.1008-973X.2022.12.009

[25] Shi, Q. (2022) Research on the Application of Target Detection Technology Based on Deep Learning Methods in Autonomous Vehicles. *Journal of Jingdezhen University*, **37**, 58-61. (In Chinese)

[26] Mohammad, A.R. and Li, J. (2023) Urban Road Target Detection Based on Faster R-CNN. *Electronic Technology and Software Engineering*, No. 1, 149-152. (In Chinese)

[27] Zhu, Y., Du, J., Liu, Y., *et al.* (2022) Research on Small Target Helmet Detection Algorithm Based on Improved Faster R-CNN. *Electronic Production*, **30**, 64-66+83. (In Chinese) https://doi.org/10.16589/j.cnki.cn11-3571/tn.2022.19.004

[28] Jiang, C., Qian, Y., Yao, X., *et al.* (2022) Target Detection Method Based on Deformable Convolution Improved SSD Algorithm. *Electronic Measurement Technology*, **45**, 116-122. (In Chinese)

[29] Cheng, J. and Deng, H. (2023) Research on Target Detection Algorithm Based on Swin-Transformer. *Automation Application*, **64**, 157-160+164. (In Chinese)

[30] Lin, T.Y., Dollar, P., Girshick, R., He, K., Hariharan, B. and Belongie, S. (2017) Feature Pyramid Networks for Object Detection. 2017 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Honolulu, 21-26 July 2017, 936-944. https://doi.org/10.1109/cvpr.2017.106

[31] He, K., Zhang, X., Ren, S. and Sun, J. (2015) Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **37**, 1904-1916. https://doi.org/10.1109/tpami.2015.2389824

[32] Hu, J., Shen, L. and Sun, G. (2018) Squeeze-and-Excitation Networks. 2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, 18-23 June 2018, 7132-7141. https://doi.org/10.1109/cvpr.2018.00745

[33] Woo, S., Park, J., Lee, J.Y. and Kweon, I.S. (2018) CBAM: Convolutional Block Attention Module. In: Ferrari, V., *et al.*, Eds., *Computer Vision—ECCV* 2018, Springer International Publishing, 3-19. https://doi.org/10.1007/978-3-030-01234-2_1

[34] Jin, Z. (2022) Research on Improved DETR Target Detection Method Based on Hybrid Fusion Backbone of CNN and Transformer. *Wireless Internet Technology*, **19**, 109-112. (In Chinese)

[35] Howard, A.G., Zhu, M.L., Chen, B., Kalenichenko, D., Wang, W.J., Weyand, T., *et al.* (2017) MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. https://doi.org/10.48550/arXiv.1704.04861

[36] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. and Chen, L. (2018) MobileNetV2: Inverted Residuals and Linear Bottlenecks. 2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, 18-23 June 2018, 4510-4520. https://doi.org/10.1109/cvpr.2018.00474

[37] Howard, A., Sandler, M., Chen, B., Wang, W., Chen, L., Tan, M., *et al.* (2019) Searching for MobileNetV3. 2019 *IEEE/CVF International Conference on Computer Vision* (*ICCV*), Seoul, 27 October-2 November 2019, 1314-1324. https://doi.org/10.1109/iccv.2019.00140

[38] Zhang, X., Zhou, X., Lin, M. and Sun, J. (2018) ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. 2018 *IEEE/CVF Conference on*

Computer Vision and Pattern Recognition, Salt Lake City, 18-23 June 2018, 6848-6856. https://doi.org/10.1109/cvpr.2018.00716

[39] Ma, N., Zhang, X., Zheng, H. and Sun, J. (2018) ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In: Ferrari, V., *et al.*, Eds., *Computer Vision—ECCV* 2018, Springer International Publishing, 122-138. https://doi.org/10.1007/978-3-030-01264-9_8

[40] Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C. and Xu, C. (2020) GhostNet: More Features from Cheap Operations. 2020 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Seattle, 13-19 June 2020, 1577-1586. https://doi.org/10.1109/cvpr42600.2020.00165

[41] Zheng, Z., Wang, P., Liu, W., Li, J., Ye, R. and Ren, D. (2020) Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression. *Proceedings of the AAAI Conference on Artificial Intelligence*, **34**, 12993-13000. https://doi.org/10.1609/aaai.v34i07.6999

[42] Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I. and Savarese, S. (2019) Generalized Intersection over Union: A Metric and a Loss for Bounding Box Regression. 2019 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Long Beach, 15-20 June 2019, 658-666. https://doi.org/10.1109/cvpr.2019.00075

[43] Li, X., Wang, W., Wu, L., Chen, S., Hu, X., Li, J., Tang, J. and Yang, J. (2020) Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection. *Advances in Neural Information Processing Systems* 33, 6-12 December 2020, 21002-21012. https://doi.org/10.48550/arXiv.2006.04388

[44] Vaswani, A., *et al.* (2017) Attention Is All You Need. *Proceedings of the* 31*st International Conference on Neural Information Processing Systems*, Long Beach, 4-9 December 2017, 6000-6010. https://doi.org/10.48550/arXiv.1706.03762

[45] Facebook AI Research (2020) DETR: Official Implementation. Model and Training Configuration Source. https://github.com/facebookresearch/detr

[46] Pineda, L., Salvador, A., Drozdzal, M., Romero, A. (2019) Elucidating Image-to-Set Prediction: An Analysis of Models, Losses, and Datasets. https://doi.org/10.48550/arXiv.1904.05709

[47] Rezatofighi, S.H., Kaskman, R., Motlagh, F.T., Shi, Q., Cremers, D., Leal-Taixé, L. and Reid, I. (2018) Deep Perm-Set Net: Learn to Predict Sets with Unknown Permutation and Cardinality Using Deep Neural Networks. https://doi.org/10.48550/arXiv.1805.00613

[48] Rezatofighi, S.H., B. G., V.K., Milan, A., Abbasnejad, E., Dick, A. and Reid, I. (2017) DeepSetNet: Predicting Sets with Deep Neural Networks. 2017 *IEEE International Conference on Computer Vision* (*ICCV*), Venice, 22-29 October 2017, 5257-5266. https://doi.org/10.1109/iccv.2017.561

[49] Mohammad, A.R. and Li, J. (2023) Intelligent Vehicle Autonomous Driving Path Based on CNN and DETR: Research on Road Intelligent Recognition. *Changjiang Information and Communication*, No. 11, 32-34. (In Chinese)

[50] Bolya, D., Foley, S., Hays, J. and Hoffman, J. (2020) TIDE: A General Toolbox for Identifying Object Detection Errors. In: Vedaldi, A., *et al.*, Eds., *Computer Vision—ECCV* 2020, Springer International Publishing, 558-573. https://doi.org/10.1007/978-3-030-58580-8_33

[51] Pan, H., Wang, M. and Zhang, F. (2022) Traffic Sign Detection and Recognition Method Based on Optimized YOLO-V4. *Computer Science*, **49**, 179-184. (In Chi-

nese) https://doi.org/10.11896/jsjkx.220300251

[52] Guan, X. and Li, Y. (2022) Occluded License Plate Recognition Based on Improved ResNet Network. *Agricultural Equipment and Vehicle Engineering*, **60**, 58-63. (In Chinese)

[53] Damarasingh, S., Nnene, O.A. and P. Zuidgeest, M.H. (2025) Detecting Pedestrians and Vehicles on Urban Roads Using Computer Vision. *Transportation Research Procedia*, **89**, 541-549. https://doi.org/10.1016/j.trpro.2025.05.081

[54] Du, S., Pan, W., Li, N., Dai, S., Xu, B., Liu, H., *et al.* (2024) TSD-YOLO: Small Traffic Sign Detection Based on Improved YOLO V8. *IET Image Processing*, **18**, 2884-2898. https://doi.org/10.1049/ipr2.13141

[55] Luo, Y., Ci, Y., Zhang, H. and Wu, L. (2024) A YOLOv8-CE-based Real-Time Traffic Sign Detection and Identification Method for Autonomous Vehicles. *Digital Transportation and Safety*, **3**, 82-91. https://doi.org/10.48130/dts-0024-0009

[56] Kang, N., Ma, F., Wan, W., Wang, D., Yao, H. and Sheng, K. (2024) Improved YOLOv9-Based Objects Detection in Adverse Weather Conditions for Autonomous Driving. *IFAC-PapersOnLine*, **58**, 267-271. https://doi.org/10.1016/j.ifacol.2024.11.155

[57] Gao, R., Omar, M.H. and Mahmuddin, M. (2025) YOLO-DER: A Dynamic Enhancement Routing Framework for Adverse Weather Vehicle Detection. *Electronics*, **14**, Article No. 4851. https://doi.org/10.3390/electronics14244851

[58] Du, D., Bi, M., Xie, Y., Liu, Y., Qi, G. and Guo, Y. (2026) MLE-YOLO: A Lightweight and Robust Vehicle and Pedestrian Detector for Adverse Weather in Autonomous Driving. *Digital Signal Processing*, **168**, Article ID: 105628. https://doi.org/10.1016/j.dsp.2025.105628

[59] Chaman, M., Maliki, A.E., Dahou, H. and Hadjoudja, A. (2026) Benchmarking YOLO-Based Deep Learning Models for Real-Time Object Detection in Hybrid ADAS and Intelligent Transportation Systems. *Results in Engineering*, **29**, Article ID: 108942. https://doi.org/10.1016/j.rineng.2025.108942

[60] Zhang, T., Kasichainula, K., Zhuo, Y., Li, B., Seo, J. and Cao, Y. (2024) Patch-Based Selection and Refinement for Early Object Detection. 2024 *IEEE/CVF Winter Conference on Applications of Computer Vision* (*WACV*), Waikoloa, 3-8 January 2024, 718-727. https://doi.org/10.1109/wacv57701.2024.00078

[61] Liu, X., Chu, Y., Hu, Y. and Zhao, N. (2024) Enhancing Intelligent Road Target Monitoring: A Novel BGS-YOLO Approach Based on the YOLOv8 Algorithm. *IEEE Open Journal of Intelligent Transportation Systems*, **5**, 509-519. https://doi.org/10.1109/ojits.2024.3449698

[62] Hong, J., Ye, K. and Qiu, S. (2025) Study on Lightweight Strategies for L-YOLO Algorithm in Road Object Detection. *Scientific Reports*, **15**, Article No. 7649. https://doi.org/10.1038/s41598-025-92148-9

[63] Zheng, Z., Zhao, J., Fan, J., Bai, R., Zhao, J. and Liu, J. (2025) A Complex Roadside Object Detection Model Based on Multi-Scale Feature Pyramid Network. *Scientific Reports*, **15**, Article No. 15992. https://doi.org/10.1038/s41598-025-99544-1

[64] Wang, G., Jin, P., Qi, Z. and Li, X. (2025) Traffic Sign Detection Method Based on Improved YOLOv8. *Scientific Reports*, **15**, Article No. 19385. https://doi.org/10.1038/s41598-025-03792-0

[65] Liu, D., Wang, C., Li, X., Zhao, X., Yin, C., Liu, Y., *et al.* (2026) FRCP-YOLO: Road Object Detection Algorithm Based on Improved YOLOv8n. *PLOS ONE*, **21**, e0342084. https://doi.org/10.1371/journal.pone.0342084

[66] Deng, L., Zhou, J. and Liu, Q. (2024) Helmet Net: An Improved YOLOv8 Algorithm for Helmet Wearing Detection. *International Journal of Networked and Distributed Computing*, **12**, 329-343. https://doi.org/10.1007/s44227-024-00040-1

[67] Liu, Z., Zhu, M., Gao, B. and Zhao, K. (2026) YOLO-DC for Vehicle Detection Using Deformable Convolutional Networks and Cross-Channel Coordinate Attention. *Scientific Reports*, **16**, Article No. 6284. https://doi.org/10.1038/s41598-026-37094-w

[68] Deng, M., Li, X. and Chen, G. (2026) LAGS-YOLO: An End-to-End Low-Light Vehicle Detection Model with Luminance-Adaptive Glare Suppression. *Symmetry*, **18**, Article No. 1729. https://doi.org/10.3390/sym18101729

[69] Jaiswal, M., Nagwanshi, K.K., Khan, M.H., Verma, U. and Taylor, A. (2026) YOLOv8s-WAMNet: Enhancing Robust Vehicle Detection under Adverse Weather via Hybrid Attention and Multi-Scale Fusion in Real Time. *Scientific Reports*, **16**, Article No. 16057. https://doi.org/10.1038/s41598-026-46527-5

[70] Jayaraman, R. (2026) Weather-Resilient Object Detection Framework for Autonomous Vehicles Using Conditional Preprocessing and YOLOv8. *International Journal of Computational Intelligence Systems*, **19**, Article No. 329. https://doi.org/10.1007/s44196-026-01413-w

[71] Qian, Z. and Xu, J. (2013) Oversaturation Traffic State Discrimination Based on Coil Detection. *Journal of South China University of Technology* (*Natural Science Edition*), **41**, 93-98. (In Chinese) https://doi.org/10.3969/j.issn.1000-565X.2013.08.015

[72] Zhan, J., Zhang, Y., Wang, C., *et al.* (2008) Video Traffic Parameter Detection Method Based on HALCON. *Highway Transportation Technology*, **25**, 126-130.

[73] Ren, J., Chen, Y., Xin, L., Shi, J., Li, B. and Liu, Y. (2016) Detecting and Positioning of Traffic Incidents via Video-Based Analysis of Traffic States in a Road Segment. *IET Intelligent Transport Systems*, **10**, 428-437. https://doi.org/10.1049/iet-its.2015.0022

[74] Coifman, B., McCord, M., Mishalani, R.G., Iswalt, M. and Ji, Y. (2006) Roadway Traffic Monitoring from an Unmanned Aerial Vehicle. *IEE Proceedings—Intelligent Transport Systems*, **153**, Article No. 11. https://doi.org/10.1049/ip-its:20055014

[75] Peng, Z., Liu, X., Zhang, L., *et al.* (2012) Research Progress and Prospects of Unmanned Aerial Vehicles in Traffic Information Collection. *Journal of Transportation Engineering*, **12**, 119-126. (In Chinese) https://doi.org/10.19818/j.cnki.1671-1637.2012.06.018

[76] Cao, X., Gao, C., Lan, J., Yuan, Y. and Yan, P. (2014) Ego Motion Guided Particle Filter for Vehicle Tracking in Airborne Videos. *Neurocomputing*, **124**, 168-177. https://doi.org/10.1016/j.neucom.2013.07.014

[77] Zhou, Y., Gong, Y. and Yao, L. (2015) Calculation and Analysis of Motor Vehicle Traffic Parameters in Drone Wide Area Video. *Transportation System Engineering and Information*, **15**, 67-73. (In Chinese) https://doi.org/10.16097/j.cnki.1009-6744.2015.06.011

[78] Su, A., Zhang, Y., Yang, X. and Yu, Q. (2017) Fast Circle Filter HOG for Car Detection from Aerial Images. *Journal of National University of Defense Technology*, **39**, 137-141.

[79] Peng, B., Cai, X., Zhang, Y. and Li, S. (2017) Automatic Vehicle Detection from UAV Videos Based on Symmetrical Frame Difference and Background Block Modeling. *Journal of Southeast University* (*Natural Science Edition*), **47**, 685-690. (In Chinese) https://teacher.cqjtu.edu.cn/caixiaoyu/zh_CN/lwcg/44856/list/index.htm

[80] Hua, S., Kapoor, M. and Anastasiu, D.C. (2018) Vehicle Tracking and Speed Estimation from Traffic Videos. 2018 *IEEE/CVF Conference on Computer Vision and Pat-*

tern Recognition Workshops (*CVPRW*), Salt Lake City, 18-22 June 2018, 153-1537. https://doi.org/10.1109/cvprw.2018.00028

[81] Liu, R. and Long, Z. (2026) A Hybrid SORT-CNN Algorithm for Multi-Target Real-Time Recognition of Urban Traffic Surveillance Images. *Alexandria Engineering Journal*, **145**, 207-219. https://doi.org/10.1016/j.aej.2026.04.054

[82] Guimarães, P. and da Silva, F.O. (2026) AI-Powered Vehicle Image Recognition Model for Sustainable and Smart Urban Traffic Management. *Procedia Computer Science*, **278**, 808-816. https://doi.org/10.1016/j.procs.2026.03.054

[83] Yin, J., Sun, J. and Tang, K. (2018) A Kalman Filter-Based Queue Length Estimation Method with Low-Penetration Mobile Sensor Data at Signalized Intersections. *Transportation Research Record: Journal of the Transportation Research Board*, **2672**, 253-264. https://doi.org/10.1177/0361198118798734

[84] Nigro, M., Cipriani, E. and del Giudice, A. (2018) Exploiting Floating Car Data for Time-Dependent Origin–Destination Matrices Estimation. *Journal of Intelligent Transportation Systems*, **22**, 159-174. https://doi.org/10.1080/15472450.2017.1421462

[85] Kong, X., Xu, Z., Shen, G., Wang, J., Yang, Q. and Zhang, B. (2016) Urban Traffic Congestion Estimation and Prediction Based on Floating Car Trajectory Data. *Future Generation Computer Systems*, **61**, 97-107. https://doi.org/10.1016/j.future.2015.11.013

[86] Wang, X. (2018) Research on Traffic Congestion Identification Based on Improved Hidden Markov Models. *Journal of Lanzhou Jiaotong University*, **37**, 29-34. (In Chinese)

[87] Wang, Y. (2010) Research on Road Traffic Jam Simulation Based on Cellular Automata. *Journal of System Simulation*, **22**, 2149-2154. (In Chinese)

[88] Wu, Y., Li, Y., Yu, W. and Yu, D. (2014) Simulation of Sporadic Traffic Congestion Behavior on Urban Roads Based on Cellular Automata. *Transportation Science and Engineering*, **30**, 72-78. (In Chinese)

[89] Tao, R., Xi, Y. and Li, D. (2016) Simulation Analysis on Urban Traffic Congestion Propagation Based on Complex Network. 2016 *IEEE International Conference on Service Operations and Logistics, and Informatics* (*SOLI*), Beijing, 10-12 July 2016, 217-222. https://doi.org/10.1109/soli.2016.7551690

[90] Xing, S., Gu, Y., Shen, L., Yang, X. and Zhuang, G. (2016) Research on Speed Based Traffic Congestion Prediction of Urban Expressways. *Traffic Information and Safety*, **34**, 48-54. (In Chinese)

[91] Jia, Y.H., Wu, J.P. and Du, Y.M. (2016) Traffic Speed Prediction Using Deep Learning Method. 2016 *IEEE* 19*th International Conference on Intelligent Transportation Systems* (*ITSC*), Rio de Janeiro, 1-4 November 2016, 1217-1222. https://doi.org/10.1109/itsc.2016.7795712

[92] Lv, Y., Duan, Y., Kang, W., Li, Z. and Wang, F. (2014) Traffic Flow Prediction with Big Data: A Deep Learning Approach. *IEEE Transactions on Intelligent Transportation Systems*, **16**, 865-873. https://doi.org/10.1109/tits.2014.2345663

[93] Soua, R., Koesdwiady, A. and Karray, F. (2016) Big-Data-Generated Traffic Flow Prediction Using Deep Learning and Dempster-Shafer Theory. 2016 *International Joint Conference on Neural Networks* (*IJCNN*), Vancouver, 24-29 July 2016, 3195-3202. https://doi.org/10.1109/ijcnn.2016.7727607

[94] Wang, X. and Xu, L. (2018) Research on Short Term Traffic Flow Prediction Based on Deep Learning. *Transportation System Engineering and Information*, **18**, 81-88.

(In Chinese) https://doi.org/10.16097/j.cnki.1009-6744.2018.01.012

[95] Du, S., Li, T., Gong, X., Yang, Y. and Horng, S.J. (2017) Traffic Flow Forecasting Based on Hybrid Deep Learning Framework. 2017 12*th International Conference on Intelligent Systems and Knowledge Engineering* (*ISKE*), Nanjing, 24-26 November 2017, 1-6. https://doi.org/10.1109/iske.2017.8258813

[96] Wu, Y., Tan, H., Qin, L., Ran, B. and Jiang, Z. (2018) A Hybrid Deep Learning Based Traffic Flow Prediction Method and Its Understanding. *Transportation Research Part C*: *Emerging Technologies*, **90**, 166-180. https://doi.org/10.1016/j.trc.2018.03.001

[97] Chen, C., Li, K., Teo, S.G., Chen, G., Zou, X., Yang, X., *et al.* (2018). Exploiting Spatio-Temporal Correlations with Multiple 3D Convolutional Neural Networks for Citywide Vehicle Flow Prediction. 2018 *IEEE International Conference on Data Mining* (*ICDM*), Singapore, 17-20 November 2018, 893-898. https://doi.org/10.1109/icdm.2018.00107

[98] Sun, Z., Li, Z. and Zhao, Y. (2016) Traffic Congestion Forecasting Based on Possibility Theory. *International Journal of Intelligent Transportation Systems Research*, **14**, 85-91. https://doi.org/10.1007/s13177-014-0104-1

[99] Chen, Y.-Y., Lv, Y., Li, Z. and Wang, F. (2016) Long Short-Term Memory Model for Traffic Congestion Prediction with Online Open Data. 2016 *IEEE* 19*th International Conference on Intelligent Transportation Systems* (*ITSC*), Rio de Janeiro, 1-4 November 2016, 132-137. https://doi.org/10.1109/itsc.2016.7795543

[100] Chen, M., Yu, X. and Liu, Y. (2018) PCNN: Deep Convolutional Networks for Short-Term Traffic Congestion Prediction. *IEEE Transactions on Intelligent Transportation Systems*, **19**, 3550-3559. https://doi.org/10.1109/tits.2018.2835523

[101] Nguyen, H., Liu, W. and Chen, F. (2017) Discovering Congestion Propagation Patterns in Spatio-Temporal Traffic Data. *IEEE Transactions on Big Data*, **3**, 169-180. https://doi.org/10.1109/tbdata.2016.2587669

[102] Wu, Y., Cai, X., Chen, M. and Li, J. (2018) Research on Traffic Congestion Propagation Model Based on Rectangle Method. *Urban Transportation*, **16**, 91-98. (In Chinese) https://doi.org/10.13813/j.cn11-5141/u.2018.0514

[103] Ultralytics (n.d.) Explore Ultralytics YOLOv8. Documentation. https://docs.ultralytics.com/models/yolov8/

[104] Ultralytics (n.d.) Ultralytics Source Code, Version 8.0.236. https://github.com/ultralytics/ultralytics/tree/v8.0.236

Supplementary Material for Intelligent Vehicle Traffic Recognition Using Deep Learning Algorithms: A Class-Specific Evaluation of YOLOv8n

Ashikur Rahman Mohammad, Li Jun, Yanfeng Han, and Sheikh Sakibul Hasan.

Supplementary Methods and Audit Evidence

This supplement accompanies the archive audit, conditional import experiment, and saved-output analysis. All image examples come from the supplied export or previously supplied predictions. The source archive includes filename, row, checksum, and selection records; the historical detector was not retrained or reevaluated.

S1. Exact Identity and Overlap Exposure

File-byte and decoded RGB SHA-256 hashing identified the same ten groups with the same image members. Four groups crossed partitions. Decoded RGB hashing added zero groups overall and zero across partitions. The shared groups expose two validation images (0.415% of 482) and two test images (0.847% of 236) to identical training counterparts. The validation counterparts contain two class-ID-4 rows. Parsed labels agree within these groups. The effect on AP is unknown.

Table S1 gives the duplicate group counts for both representations. The duplicate manifests list the relative filenames and hashes. Perceptual candidate counts depend on the declared screening threshold and require further review to establish scene identity.

Table S1. Duplicate image groups identified by SHA-256 of file bytes and decoded RGB data.

Partitions	File bytes	Decoded RGB
Within training	4	4
Within validation	2	2
Within test	0	0
Training-validation	2	2
Training-test	2	2
Validation-test	0	0
All three partitions	0	0
Total	10	10

S2. Fixed Annotation Sample

Within each label, a SHA-256 ordering of relative image path and annotation row selected one box per distinct validation image, up to twelve images per label. The same image could contribute to several labels. The 84 selected regions came from 57 distinct image files, and 16 image files contributed to multiple labels. The selection used neither model confidence nor visual-content filtering. Five-column

rows use center and size coordinates, and polygon rows use their enclosing rectangle. The crop includes a contextual margin.

Table S2 records descriptions of the visible contents. Figures S1-S7 display the full sample. An AI-assisted pass produced initial descriptions, and Ashikur Rahman Mohammad manually checked all 84 regions. The retained materials give the image and row identifier and the final observation for each region. They do not preserve the exact AI model or version, the prompts or their sequence, or a separate record of how uncertain cases were resolved. The observations concern visible contents, including character-sized regions, whole plates, rider groups, vehicles, and head-related regions. They support examples of content variation in the sample. Helmet use, rider association, inter-reviewer agreement, and dataset-wide prevalence were outside the scope of the review.

Table S2. Observed contents of the fixed class-balanced annotation sample (12 images per label).

Exported label	Images	Observed box contents (counts)
Triple riding	12	plate punctuation-like mark (10); rider group (2)
bike	12	plate character/mark (11); two-wheeler region (1)
motorcycle	12	motorcycle/rider region (12)
number plate	12	plate region (11); whole vehicle region (1)
vehicle	12	miniature vehicle (10); full-size vehicle photograph (2)
with helmet	12	head/helmet-related region (12)
without helmet	12	head-related region (11); scene/background region (1)

The sample contains one selected box per distinct image within each class. Sampling is fixed by hashes, not model confidence. These are descriptive observations, not original annotation definitions, an independent relabeling study, or population error rates. A file/row manifest accompanies the figures.

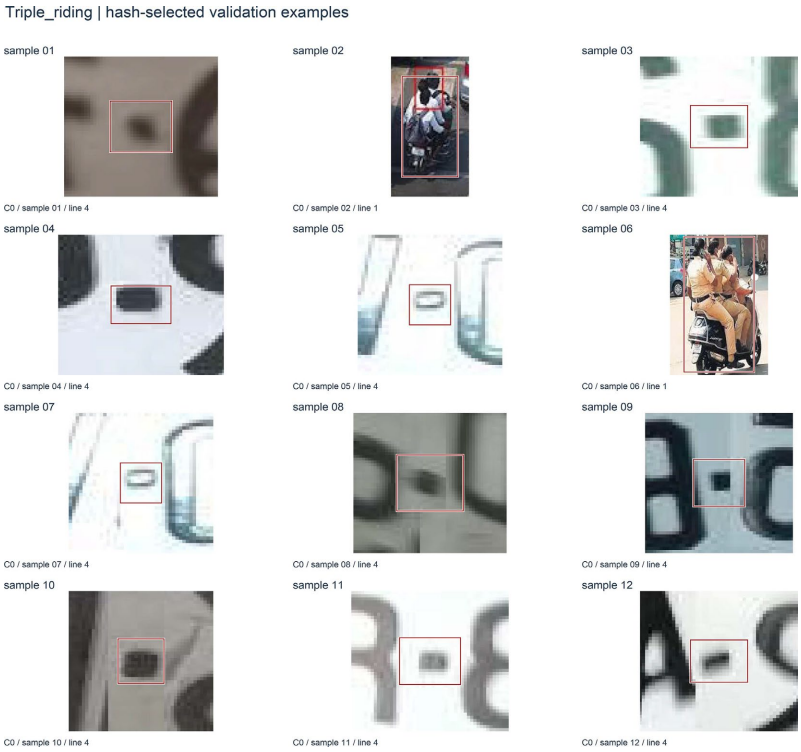


Figure S1. All twelve hash-selected validation examples for the exported label Triple riding. Rectangles follow row-wise source coordinates; sample and row IDs identify the manifest records.

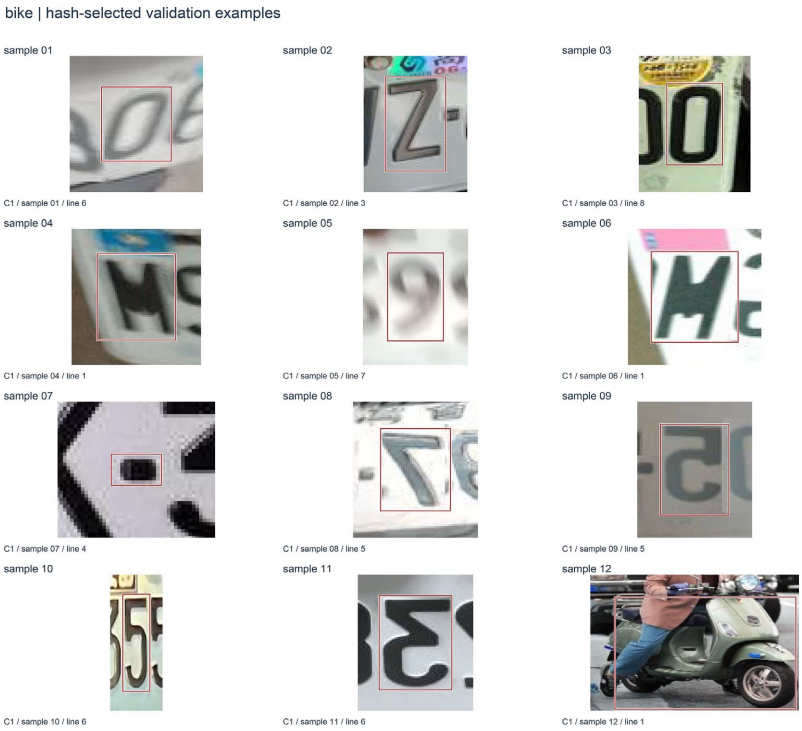


Figure S2. All twelve hash-selected validation examples for the exported label Bike. Rectangles follow row-wise source coordinates; sample and row IDs identify the manifest records.

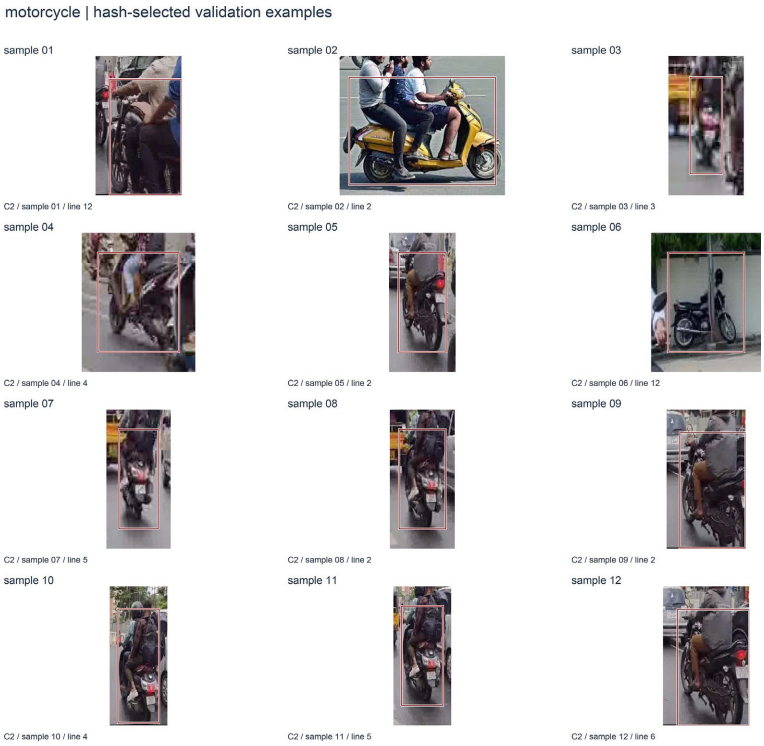


Figure S3. All twelve hash-selected validation examples for the exported label Motorcycle. Rectangles follow row-wise source coordinates; sample and row IDs identify the manifest records.



Figure S4. All twelve hash-selected validation examples for the exported label Number plate. Rectangles follow row-wise source coordinates; sample and row IDs identify the manifest records.

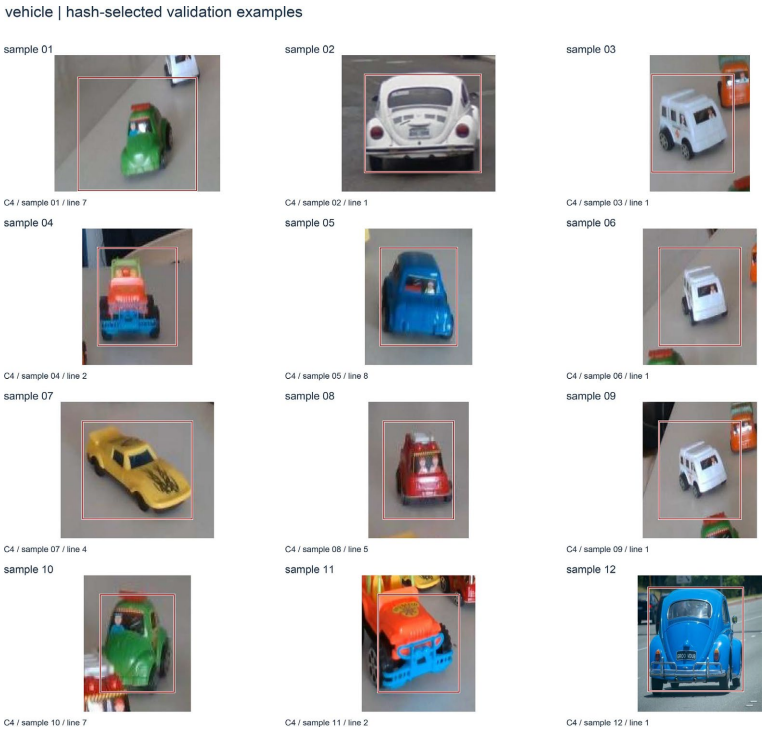


Figure S5. All twelve hash-selected validation examples for the exported label Vehicle. Rectangles follow row-wise source coordinates; sample and row IDs identify the manifest records.

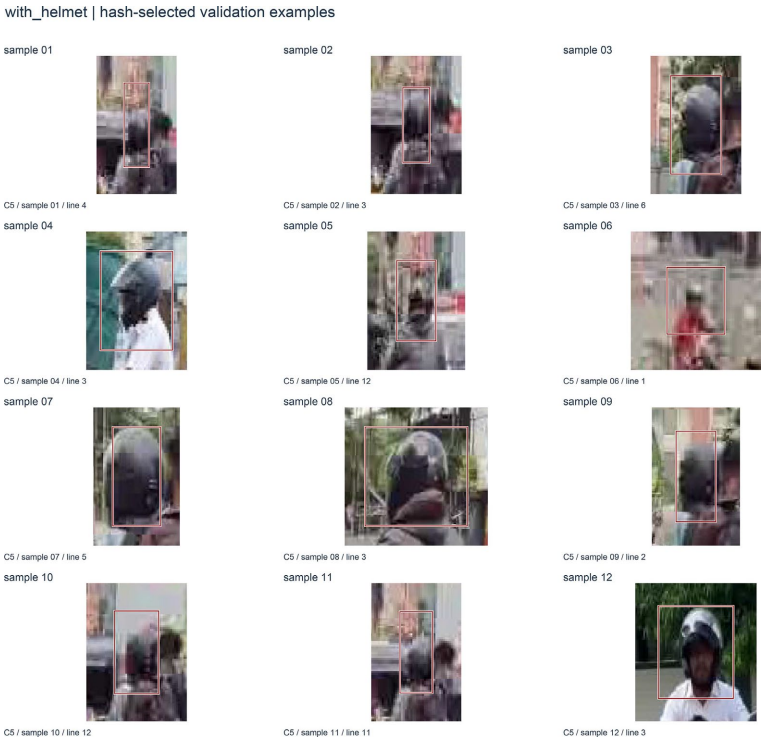


Figure S6. All twelve hash-selected validation examples for the exported label With helmet. Rectangles follow row-wise source coordinates; sample and row IDs identify the manifest records.

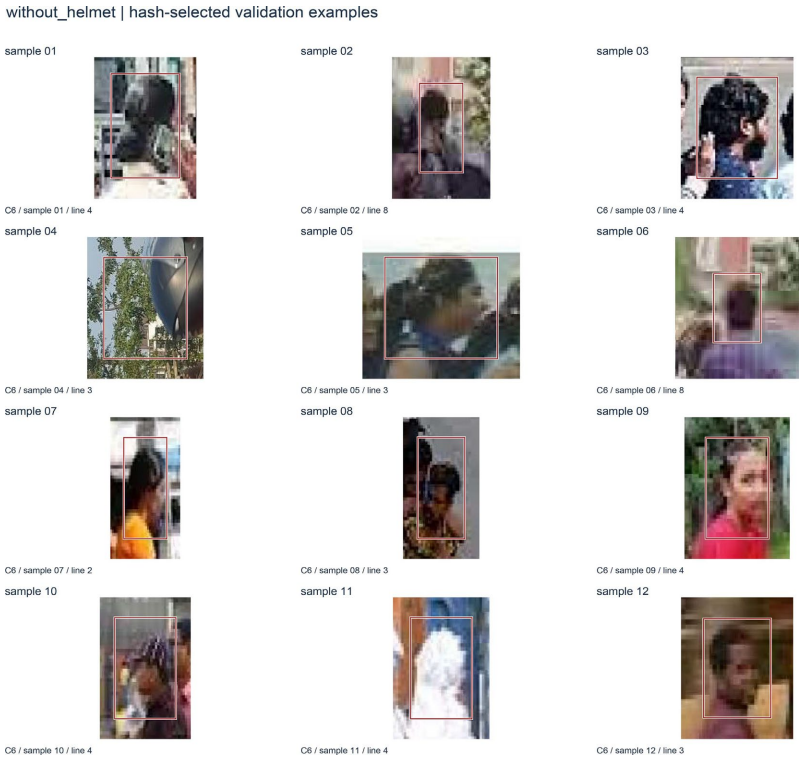


Figure S7. All twelve hash-selected validation examples for the exported label Without helmet. Rectangles follow row-wise source coordinates; sample and row IDs identify the manifest records.

S3. Coordinate Compatibility and Canonical Labels

The archived v8.0.236 source uses a polygon branch for a file whenever one row has more than six fields. The first experiment reproduced that dispatch and executed the original coordinate-conversion functions on NumPy arrays. The maximum absolute normalized-coordinate discrepancy between the reconstructed geometry and those functions was 6.82×10^{-8}.

The second experiment ran the unmodified v8.0.236 verify_image_label routine on a temporary copy of each source image paired with its canonical label file. The routine accepted all 2582 pairs and returned all 14,074 rows with the intended class IDs and coordinates. **Table S3** gives the per-split counts. Across the three partitions, the maximum absolute coordinate difference was zero at float32 precision. There were zero missing label files, rejected pairs, duplicate row removals, or image repairs. The canonical label ZIP contains five-column boxes. For detection, each original polygon is represented by its enclosing rectangle.

Table S3. Results from the pinned verifier on the canonical label export.

Partition	Pairs accepted	Intended rows	Rows returned	Empty labels
Training	1864	10,425	10,425	1
Validation	482	2443	2443	0
Test	236	1206	1206	0
Total	2582	14,074	14,074	1

The reproduction files include VerifyCanonicalLabels.py, VerifierSummary.json, and VerifierPairs.csv. The run used Python 3.12.14, NumPy 2.3.5, and Pillow 13.0.0. The source and canonical label archives were read directly. Temporary copies allowed the original routine to perform its image checks, including its JPEG-repair branch when needed.

S4. Reproduction and Provenance

The audit operates directly on the 125-MB supplied ZIP with SHA-256:

393e6c8b44be70a6f90c1ec01f4885f2e9902bcda7bcf98283bb21296a7cc1a6

The code package contains the dataset audit, coordinate compatibility/normalization, saved-notebook extractor, and figure/table generation programs. CSV files provide per-image identities, row-wise and reconstructed coordinates, class geometry, selected visual examples, and perceptual candidates. JSON records preserve thresholds, versions, invariants, and source hashes. The original notebook is read as saved data; its training cells are not executed by the extractor. The original checkpoints, historical label cache, and full per-image detector predictions are unavailable.

S5. Additional Supplied Qualitative Output

Figure S8 retains the rainy-road output provided with the study. It has no matched evaluation labels or recorded sampling protocol. Its confidence labels are illustrative output values and do not establish weather robustness.



Figure S8. Supplied qualitative prediction visualization on a rainy-road image.

Figure S8 preserves the available visual record. A condition-specific performance estimate requires a separate annotated evaluation.

S6. Standard Objectives for the Prior Detector Frameworks

The following equations document the standard mechanisms underlying the retained DETR and Faster R-CNN illustrations. They are background formulations, separate from the archived YOLOv8n experiment and the coordinate audit. The

attention operation used in the encoder and decoder can be summarized as

$$\begin{aligned}&\mathrm{Attn}\left(Q,K,V\right)=\mathrm{softmax}\left(\frac{QK^{\mathrm{T}}}{\sqrt{d_k}}\right)V,\\&H_j=\mathrm{Attn}\left(QW_j^Q,KW_j^K,VW_j^V\right),\\&\mathrm{MHA}\left(Q,K,V\right)=\mathrm{Concat}\left(H_1,\cdots,H_h\right)W^O.\end{aligned}\tag{S1}$$

Equation (S1) defines the query, key, and value matrices Q, K, and V, the key dimension d_k, and h learned attention heads. Encoder self-attention relates image tokens; decoder cross-attention relates object queries to encoded image features. The expression describes the DETR mechanism and does not imply that attention was added to the current YOLOv8n system [7] [44].

DETR uses bipartite matching to assign ground-truth objects to distinct query predictions. For M ground-truth objects and $N \geq M$ queries, the foreground matching can be written as

$$\hat{\pi}=\underset{\pi\in\mathcal{I}_{M,N}}{\arg\min}\sum_{i=1}^{M}\left[-\hat{p}_{\pi(i)}\left(c_i\right)+\lambda_{\mathrm{box}}\mathcal{L}_{\mathrm{box}}\left(b_i,\hat{b}_{\pi(i)}\right)\right],\tag{S2}$$

Equation (S2) selects an injective assignment from ground-truth objects to query indices; c_i and b_i denote a target class and box, while $\hat{p}$ and $\hat{b}$ are the predicted class probability and box. The box cost combines localization terms, and unassigned queries are supervised as no-object. Matching is an assignment step, not the complete training loss; class and box losses are applied after assignment [7].

A standard RPN objective combines binary objectness classification with regression on positive anchors:

$$\mathcal{L}_{\mathrm{RPN}}=\frac{1}{N_{\mathrm{cls}}}\sum_i\ell_{\mathrm{cls}}\left(p_i,p_i^*\right)+\frac{\lambda}{N_{\mathrm{reg}}}\sum_i p_i^*\ell_{\mathrm{reg}}\left(t_i,t_i^*\right).\tag{S3}$$

Equation (S3) uses predicted objectness p_i, binary target p_i^*, and predicted and target box offsets t_i and t_i^*; the indicator p_i^* restricts regression to positive anchors. The normalization terms and weight balance the two losses. This is the standard Faster R-CNN RPN objective, not a loss recovered from the authors' old training logs and not the YOLOv8n loss used in the present experiment [6].