scientific reports

OPEN

Robustness analysis of YOLO and faster R-CNN for object detection in realistic weather scenarios with noise augmentation

Sana Gholinavaz, Nima saeedi & Sina Samadi Gharehveran

Object detection systems are central to the autonomy and safety of intelligent transportation systems. Yet, the accuracy of object detection models can suffer under environmental noise or adverse weather. This paper tested the robustness of four object detection architectures: YOLOv5s, YOLOv8m, YOLOv10n, and Faster R-CNN, to visual degradation (real-world weather and artificial noise). We utilize the DAWN dataset, a benchmark of 1,000 high-resolution traffic images captured under fog, rain, snow, and sandstorms, with further augmentations of Gaussian noise, salt-and-pepper noise, blurriness, and overlays applied with artificial fog. We standardized all annotations to YOLO and COCO annotation formats for multi-framework interoperability. Our quantitative analysis used mAP@0.5, mAP@0.5:0.95, Precision, and Recall to compare the models, alongside some qualitative analysis through visual overlays and plotting training loss. The findings of our analysis showed YOLOv8m achieved the highest baseline accuracy on clean data, while Faster R-CNN proved resilient in noisy environments. YOLOv10n achieves a good trade-off between efficient and robust detection. The results of this study highlight the necessity of adaptive training pipelines and environment-aware benchmarks to enhance the real-world reliability of vision-based detection systems.

Keywords Object detection, Adverse weather, Noise augmentation, YOLOv8, YOLOv10, Robustness evaluation, DAWN dataset

New object detection technology has made real-time visual perception possible for applications like autonomous vehicles (AVs), intelligent traffic monitoring, and robot navigation. YOLOv5, YOLOv8, and Faster R-CNN models achieved top accuracy and performance under ideal and controlled environments. But their deployment in actual use is undermined frequently by natural environmental conditions like fog, rain, snow, and digital noise—conditions that degrade visibility and sensor integrity.

In spite of advancements in learning paradigms and network architectures, visual degradation robustness is still an issue. Existing solutions have tried handling it by training with artificially introduced noise or data augmentation using controlled adversarial perturbations[1,2]. These do not generalize to naturally occurring perturbations. Real-world traffic video sequences captured under varying weather conditions and utilized to construct the DAWN dataset[3] provide a more realistic testing environment for the performance of models under such scenarios. Our study offers a comparative evaluation of object detection models under clean and noisy inputs. We build an end-to-end reproducible pipeline from dataset preparation, noise injection, harmonization of annotations, model training, and validation. Noise like Gaussian and salt-and-pepper approximate low-light and sensor noise, and blur and synthetic fog approximate lens and atmospheric effects. Models are trained and evaluated on the same splits to facilitate fair comparison.

The novelty of this study lies in its comprehensive robustness evaluation framework for object detection models. Unlike most previous studies that rely on noise-free or purely simulated data, we utilize a real-world dataset (DAWN) that captures natural variability and sensor noise in realistic environments. Additionally, the introduction of synthetic noise and corruptions simulates more challenging conditions, enabling a rigorous assessment of model performance under both real-world and extreme scenarios. This combination of real-world data, corruption analysis, and systematic evaluation provides new insights into model robustness that go beyond standard comparisons and offers practical guidance for deploying object detection systems in adverse environments.

Department of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran. email: s.samadi@tabrizu.ac.ir

Our experimental results show that there are clear trade-offs among robustness, resource utilization efficiency, and accuracy. YOLOv8m performs best on clean images but is less robust when there is heavy-density noise. However, Faster R-CNN is even more robust under severe visual corruption. YOLOv10n, due to its lightweight nature, performs best in corrupted environments. Through quantitative evaluation and qualitative visualizations, this paper emphasizes the significance of robustness-aware benchmarking in real deployment and sheds light on future design architecture upgrading and data augmentation strategies.

The remainder of this paper is organized as follows. "Literature Review" reviews related works in object detection under adverse weather and noise conditions. "Methodology" details the methodology, dataset preparation, augmentation strategies, and model training configurations. "Results and discussion" presents the experimental results and discusses robustness comparisons across different models and weather scenarios. "Conclusion" concludes the study with key findings and outlines directions for future research.

Literature review

A-6 Inferring Objects with YOLO in Harsh Weather: Benchmarks and Metrics The top priority for any self-driving car (AV) is reliable object detection in all-weather driving. Real weather conditions (rain, snow, fog, sandstorms, dark) can result in sensor noise, contrast loss and occlusions that translate into domain shifts between training and test data, and in turn reduce the performance of state-of-the-art deep-learning detectors. For example, Sakaridis et al. validated this performance degradation on FoggyCityscapes and introduced a method for domain adaptation[1,2]. Patel et al. has given a detailed systematic taxonomy of weather-induced distortions, and measured the reduction in precision/recall under fog and rain conditions[3]. Michaelis et al. has studied a benchmark of detectors under a variety of bad-weather conditions and reported clear accuracy drops for a detector trained on clear images[4].

Datasets and the data-gap for harsh weather

In the absence of enough realistic training data, several realistic bad-weather datasets have been proposed to address the lack of training samples. The ACDC benchmark offers pixel-wise annotations on fog, rain, snow, and night conditions, providing dense semantic labels for bad-weather urban driving scenes[5]. DAWN provides high-resolution imagery of sandstorms, snowstorms and other extreme conditions and also provides emphasis on multi-modal capture (when available)[6–8]. Large-scale driving datasets such as nuScenes and BDD100K offer multimodal (RGB + LiDAR + GPS/IMU) sensor streams that can be useful for sensor-fusion experiments, but are not exhaustive with respect to rarer phenomena like lens contamination, strong illumination changes (e.g., headlights in the camera view) or severe (and temporally persistent) occlusions to simulate real-world camera degradation[9,10]. As a result, many weather-robustness studies must either (a) train on relatively small, condition-specific datasets or (b) create hybrid datasets by cross-domain fusion of adverse-condition datasets (such as computationally expensive cross-domain mixes of ACDC + DAWN) to increase coverage at the expense of scalability[11].

While recent advances in urban scene understanding—such as dynamic context-aware architectures and GAN-based enhancements—have improved semantic segmentation under complex conditions[7,8], our work complements these efforts by focusing specifically on the robustness of object detection models under adverse weather and noise scenarios, where pixel-level segmentation improvements do not directly translate to detection performance.

Augmentation strategies: physics-based, learned, and hybrid

Efforts have also gone into using different augmentation strategies to synthetically expand adverse-weather training data. Physics-based renderers (such as the light-attenuation Beer–Lambert model for fog/haze) are often highly interpretable and controllable but do not precisely map to actual scene-dependent scattering effects and other depth-dependent effects[12]. GAN-based and neural-style-transfer based augmentations can create more visually realistic degradations (rain streaks, snow overlays, nighttime relighting) but sometimes introduce artifacts in depth cues and temporal coherence for video[12]. Hybrid approaches that combine curated adverse-condition datasets with synthetic augmentations (for example, Kumar and Muhammad's hybrid dataset) have shown some promise, but they are often computationally expensive to create and cannot easily scale to many modes of degradation[11].

Weather-aware detectors: architectures and trade-offs

The large bulk of efforts focus on making detectors themselves more robust to weather: Preprocessing + detector pipelines/image-adaptive methods. Image-restoration or image-enhancement modules that are cascaded to detectors (dehazing, low-light enhancement) can help, but can add more complexity and sometimes lead to task-level inconsistencies. More recent "image-adaptive" methods propose integrating differentiable preprocessing modules as part of the detection pipeline; for example, ERUP-YOLO directly learns compact differentiable filters that are also a unified preprocessing step for domain-agnostic augmentation without sacrificing inference efficiency. ERUP-YOLO has reported consistent improvements across adverse-weather datasets[13].

Weather-specific network designs. Dehazing-aware two-path architectures (such as D-YOLO[14] or high-resolution haze-enhancement blocks (HR-YOLO[15] explicitly model haze/fog effects inside detection backbones. While they often improve accuracy on their target degradations, these designs also tend to reduce generalization across other distortions and also often add latency.

One-stage vs. two-stage detectors (speed–accuracy tradeoff). While two-stage detectors such as Faster R-CNN offer high localization precision, single-stage YOLO-family models are designed for real-time execution. Comparative work (including with YOLOv8) has shown that the most recent YOLO designs are on par (or sometimes better) than two-stage detectors in several metrics, though there are some tasks where the precision of two-stage detectors is hard to beat. For example, Ezzeddini et al. showed that in the task of low-light vessel detection, Faster R-CNN maintains higher precision but YOLOv8 can provide faster inference speed with nearly as good accuracy[16]. Michaelis et al.[4] also pointed out that domain shifts have a disproportionate effect on reducing Faster R-CNN performance (unless domain adaptation is performed) whereas YOLO can provide more robustness with augmentation. Domain adaptation for single-stage detectors. Domain-adaptive methods have historically focused on two-stage detectors. More recently, domain adaptation and contrastive-learning methods that work for one-stage detectors have been designed. For example, ACCV2024 and CLDA-YOLO are examples of auxiliary domain-guided adaptation and visual contrastive learning for YOLO-style detectors that can lead to much higher robustness gains without a significant increase in inference costs[17,18].

Recent YOLO variants focused on weather robustness. There are several recent YOLO-centric works that focus on modifying backbone/neck components, attention and/or augmentation to improve adverse-weather performance while retaining real-time execution. For example, YOLOv8-STE has added global/local modules as well as a neck EMA mechanism to reduce the feature drifts under degradations and obtained improved performance on adverse-weather test datasets[19].

Multi-sensor fusion and model tuning

In the interest of not solely depending on RGB when the visibility drops under the bad-weather conditions, some researchers have studied different RGB + IR and RGB + LiDAR fusion architectures. Transformer-based cross-modal fusion (CME-YOLO) has obtained some quantifiable mAP gains under low-visibility conditions[20]. A full-waveform LiDAR approach that can reconstruct object boundaries even in dense fog has been demonstrated, but they are still too expensive for a broad commercial roll-out[21]. Hyperparameter tuning with metaheuristics such as Chimpanzee, Gray Wolf and others has also been used to tune some augmentations to get better performance on weather-augmented test sets[22].

Gaps in the literature and how this work differs

Even as the rapidly growing body of work on all-weather detection for autonomous vehicles continues to make progress on the overarching challenge, several important gaps remain:

Coverage of realistic sensor failures. Most datasets and augmentations focus on atmospheric scattering or low light. Lens contamination, glare, and other camera-specific degradations are less represented in existing benchmark datasets (ACDC, DAWN, nuScenes, BDD100K)[5–10].

Task-aware augmentation and domain adaptation for single-stage detectors. While many domain-adaptive methods exist, most prior work has focused on two-stage models; methods for YOLO that are both robust and efficient are only beginning to be explored[17,18].

Comprehensive empirical study of the newest YOLO releases. YOLOv10 and other recent YOLO variants are often claimed to be speed-optimized[23], but thorough peer-reviewed comparisons under systematically varied weather remain rare. At the same time, both preprocessing (e.g., ERUP-YOLO[13] and backbone changes (YOLOv8-STE[19] show promise, but generalization to many weather modes has not been studied comprehensively.

How this paper contributes. To address the above gaps in the literature, our approach and framework aims to (i) combine weather-specific, domain-agnostic augmentations with YOLOv8/YOLOv10 to obtain high robustness to multiple types of degradations, (ii) integrate scalable domain-adaptive techniques to obtain a generalizable detection capability, and (iii) report an exhaustive evaluation of both detection accuracy and inference latency across both synthetic and real adverse-weather datasets.

While several previous studies have analyzed the robustness of object detection models, they typically rely on noise-free or simulated datasets and often focus on a single type of corruption. In contrast, our study uses a real-world dataset (DAWN) that includes natural variability and sensor noise, combined with multiple synthetic corruptions (Gaussian, salt-and-pepper, blur, fog) to simulate more challenging conditions. Moreover, we evaluate multiple state-of-the-art models under consistent training settings, and provide detailed analyses on object sizes, class imbalance, and corruption-specific performance, which are rarely addressed in prior works. These aspects allow us to offer new insights into model robustness under both realistic and extreme scenarios.

Methodology

This article analyzes the robustness, noise robustness, and training regimes of YOLO-based models with the DAWN (Detection in Adverse Weather Nature) dataset used for the experiment. There are six steps applied in the methodological pipeline: dataset procurement and preparation, noise-based augmentation, annotation alignment, model selection and configuration, model training, and performance metrics. Each step is designed to be methodologically transparent, architecturally agnostic, and feasibly relevant to the conditions of the deployment environment. Figure 1 presents the study workflow.

The DAWN dataset was selected due to its direct concern with the evaluation of vision systems under adverse weather conditions. It includes 1000 real-world high-resolution traffic video images under adverse weather conditions of fog, heavy rain, snow, and sandstorm, on urban roads, expressways, and multi-lane highways. These environmental and context variations are conditions that autonomous vehicles would most likely encounter in driving conditions, as can be attested by the benchmarking study of Zhang et al.[19].

There are bounding box labels for common object categories such as cars, buses, trucks, motorcycles, bicycles, and pedestrians for all the images within the DAWN dataset. These labels are available in Pascal VOC (.xml) and YOLO (.txt) formats. For enabling interoperability with modern deep learning frameworks such as Detectron2 and MMDetection that accept COCO-style JSON inputs, a Python-based conversion pipeline was developed to convert all the annotation files to COCO-compliant JSON format. This modification kept informative data like bounding box coordinates, image metadata, and class mappings intact, as per best practices established by Liu and Wang[16].

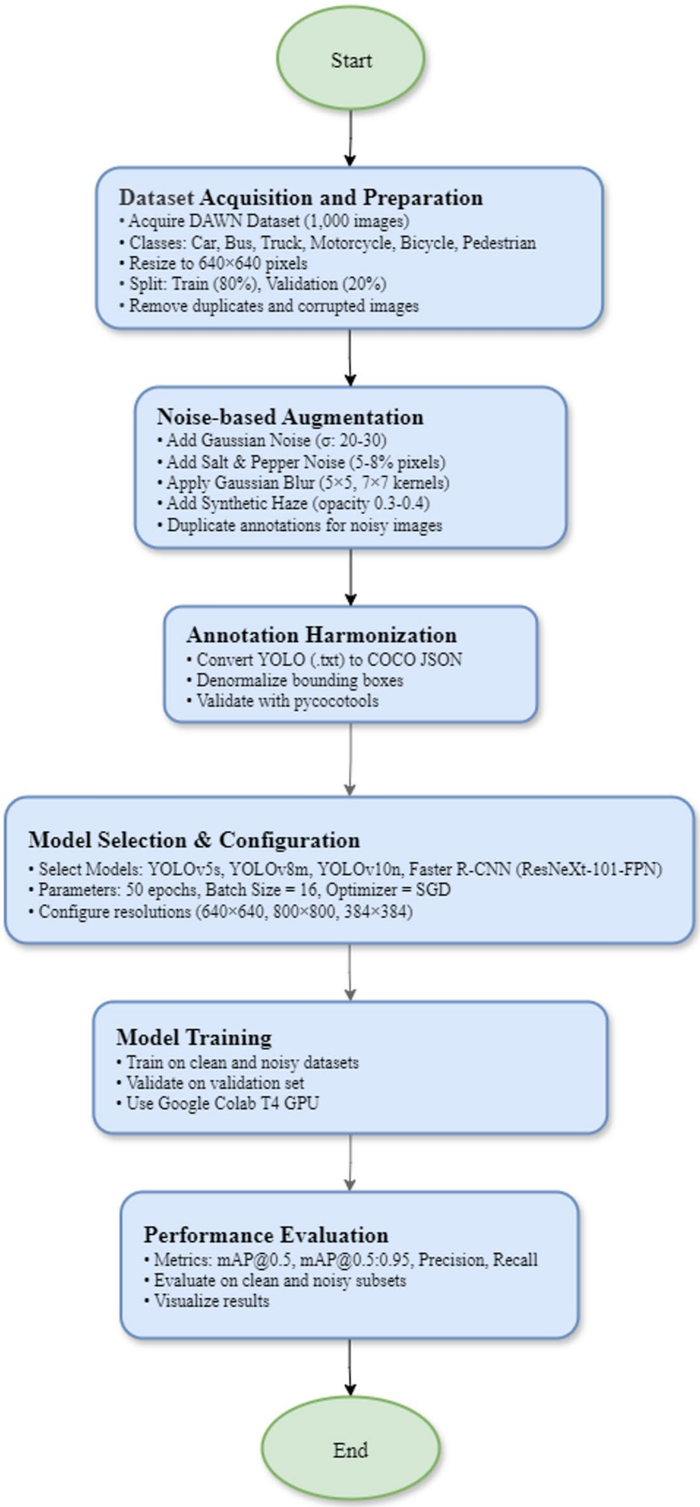
Start
Dataset Acquisition and Preparation
• Acquire DAWN Dataset (1,000 images)
• Classes: Car, Bus, Truck, Motorcycle, Bicycle, Pedestrian
• Resize to 640×640 pixels
• Split: Train (80%), Validation (20%)
• Remove duplicates and corrupted images
Noise-based Augmentation
• Add Gaussian Noise (σ: 20-30)
• Add Salt & Pepper Noise (5-8% pixels)
• Apply Gaussian Blur (5×5, 7×7 kernels)
• Add Synthetic Haze (opacity 0.3-0.4)
• Duplicate annotations for noisy images
Annotation Harmonization
• Convert YOLO (.txt) to COCO JSON
• Denormalize bounding boxes
• Validate with pycocotools
Model Selection & Configuration
• Select Models: YOLOv5s, YOLOv8m, YOLOv10n, Faster R-CNN (ResNeXt-101-FPN)
• Parameters: 50 epochs, Batch Size = 16, Optimizer = SGD
• Configure resolutions (640×640, 800×800, 384×384)
Model Training
• Train on clean and noisy datasets
• Validate on validation set
• Use Google Colab T4 GPU
Performance Evaluation
• Metrics: mAP@0.5, mAP@0.5:0.95, Precision, Recall
• Evaluate on clean and noisy subsets
• Visualize results
End

Fig. 1. Overall workflow of the proposed methodology.

What sets DAWN apart from artificial datasets is that it makes use of genuine environmental wear and tear rather than artificially created noise through algorithms. It does make a difference: as shown by Kim et al.[24], models trained on only synthetic transformations are not generalizable to actual weathering conditions, especially under sensor degradations. DAWN can overcome such a limitation by providing real visual challenges, and hence, it is possible to evaluate with a more realistic and trustworthy performance. Further, DAWN has also emerged as a widely used option in recent research studies as a better benchmark for object detection systems under real-world conditions[25]. It is made available openly via reputable academic data repositories such as IEEE DataPort[26], Mendeley Data[27], and Papers with Code[28] and is thus an open and reproducible benchmark. The clean annotation protocol, heterogeneity of the environment, and quality of the data render it suitable for benchmarking recent object detection robustness. For further text description, some representative DAWN dataset images, e.g., pictures of fog, rain, and snow, with associated object annotations, are given in Fig. 2. Some representative images from the DAWN dataset of different weather conditions (fog, rain, snow) and their associated object annotations. Figure 2. Representative images from the DAWN dataset showcasing different weather conditions: (a) fog, (b) rain, (c) snow, with their associated object annotations.

Dataset preparation

The DAWN dataset forms the empirical basis of this study. It includes 1000 high-quality annotated images, which possess bounding box labels, both in YOLO format (.txt) as well as Pascal VOC format (.xml), for six major object classes, viz., car, bus, truck, motorcycle, bicycle, as well as pedestrian. Each YOLO format label presents normalized coordinates of the bounding box along with class tags, hence supporting easy integration into YOLO-based detection systems such as YOLOv5, YOLOv8, and YOLOv10.

In order to fully assess model performance under the presence of noise, an improved version of DAWN added two separate distortions at the pixel level to the initial images. First, Gaussian noise was applied through the randn() function of OpenCV, featuring a mean of zero as well as a randomly selected standard deviation of 20–30. This method successfully mimicked sensor-introduced noise commonly found under poor lighting conditions or for cheaply acquired camera equipment, as agreed upon by the identified models of noise put forward by Krizhevsky et al.[29]. Next, a transformation through NumPy added Salt-and-Pepper noise, which affected about 5–8% of the pixels of each image, successfully mimicking digital interference, compression, or at the bit level degradation as explained by Wang et al.[30].

For each clean image, there existed a noisily related one, hence effectively doubling the dataset size. Annotation files were copied and renamed as needed to ensure synchronization of bounding box annotations with their noisily related ones. Both original and augmented images were resized to 640 × 640 pixels to ensure homogeneous input for all YOLO versions as well as ensure efficient GPU memory use when training. A customized Python script systematically structured the dataset into four separate directories (train/images, train/labels, val/images, val/labels), following an 80:20 ratio for the split of training-validation while ensuring class distribution consistency.

Additionally, an auto-generated dataset.yaml configuration file was provided to define class names, directory paths, as well as the number of categories, to enable seamless integration into Ultralytics' YOLO training pipelines[31]. All of the preprocessing tasks—consisting of noise injection, resizing, annotation replication, as well as dataset partitioning—were performed using Python 3.10, run from modular, version-controlled Jupyter Notebooks on Google Colab[32]. This setup ensures reproducibility, auditability, as well as scalability, for future experimentation.

Noise-based data augmentation

To capture a broader spectrum of real-world degradation beyond atmospheric conditions, multiple pixel-level augmentation techniques were implemented to simulate common sources of visual corruption encountered in embedded camera systems. These included noise artifacts due to sensor limitations, electromagnetic interference, and environmental contamination.

Gaussian noise was applied across images with varying variance levels to mimic signal degradation in low-illumination environments. This is particularly important for simulating real-world foggy or night-time driving scenarios where signal-to-noise ratios are diminished[33]. Salt-and-Pepper noise was added to simulate digital errors caused by electromagnetic fluctuations or faulty image sensors, replacing 5–8% of randomly chosen pixels with black or white values[34].

To simulate optical blurring due to defocus, lens smearing, or water droplets, Gaussian blur was applied using 5 × 5 and 7 × 7 kernels. This effectively modeled situations such as dirty lenses or condensation, which reduce image sharpness and are known to impair detection[33]. Furthermore, to approximate environmental haze or smog, semi-transparent white overlays with opacity levels between 0.3 and 0.4 were blended into the original images, creating synthetic fog-like conditions in line with methods proposed by Halder et al.[35,36].

Fig. 2. Annotated weather samples from DAWN.

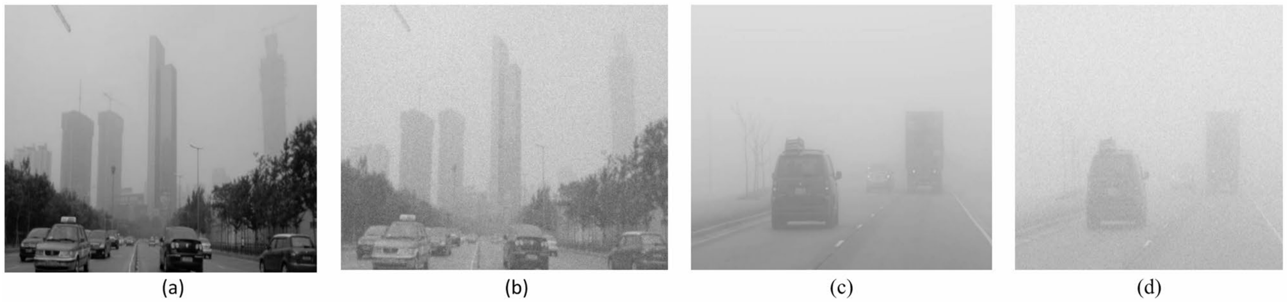

Fig. 3. Augmented DAWN images: Clean vs. Noisy and blurred versions.

Augmentation	Parameter	Value
Gaussian noise	σ	0.01
Salt and pepper noise	Amount	0.02
Motion blur	Kernel Size	5
Gaussian blur	Kernel Size	3

Table 1. Noise and blur parameters used for adverse-weather data augmentation.

All augmentations were implemented using OpenCV due to its matrix operation efficiency and seamless integration with the pipeline[37]. A fixed random seed (42) was used across all routines to maintain reproducibility and experimental stability. Importantly, object labels were left unchanged throughout the augmentation process, under the assumption that noise operations do not alter object positions or geometries. The final dataset thus maintained a near 1:1 ratio of clean-to-noisy samples, enhancing robustness testing and allowing direct comparison across models.

Representative examples of noise-augmented images are presented in Fig. 3, illustrating the visual impact of Gaussian noise, salt-and-pepper artifacts, Gaussian blur, and synthetic haze. Figure 3. Sample visual augmentations applied to the DAWN dataset: (a) original clean image, (b) Gaussian noise, (c) salt-and-pepper noise, (d) Gaussian blur.

During data augmentation, all transformations were applied in a way that preserves the visibility of all labeled objects. Therefore, the original labels remain valid, and no adjustments were necessary. In cases where an object could be partially hidden, we either avoided such transformations or carefully verified that the object remained sufficiently visible to maintain label accuracy. This ensures that augmented data do not introduce labeling errors.

The specific noise and blur parameters were chosen to simulate realistic adverse conditions commonly encountered in real-world environments, such as fog, rain, and snow, which can cause image degradation. Table 1 summarizes the parameters used in our experiments, including the noise type, intensity, and blur kernel size, providing a clear overview and ensuring reproducibility.

Annotation harmonization

The DAWN dataset includes annotations in the YOLO format, but many new object detection frameworks, such as Detectron2 and MMDetection, need input as COCO-style JSON. There was a need for a conversion method that could bridge the two methods of annotation while minimizing information loss, so a conversion pipeline was developed using Python to convert the YOLO-formatted labels into COCO-formatted annotations accurately, while preserving metadata and label provenance. This tool denormalized the YOLO bounding boxes using the image values and mapped the prior bounding boxes to absolute pixel coordinates. Next, it constructed COCO-style annotation dictionaries with keys including image_id, bbox, category_id, iscrowd, and empty segmentation fields since DAWN does not record instance masks[36]. The COCO categories list was built from a unified label dictionary created from the dataset. YAML file, which allowed a consistent label name across formats[38].

The first of several validation checks to ensure the conversion did not lose information and was accurate included randomly visualizing several images with bounding boxes generated for each annotation format, and confirming the spatial locations of the objects remained consistent. The second validation check used the pycocotools API[39,40], which also checked for structural integrity, including checking for missing keys or incorrect bounding box definitions. The random seed (42) used to create random values was consistent at each transformation step to ensure reproducibility. This process of harmonization will allow the use of the same dataset across different detection pipelines, allowing for fair and proper evaluation.

As shown in the Fig. 4, the conversion formulas correctly map the coordinates from YOLO to COCO format. This ensures that our data is fully compatible for evaluation using COCO metrics.

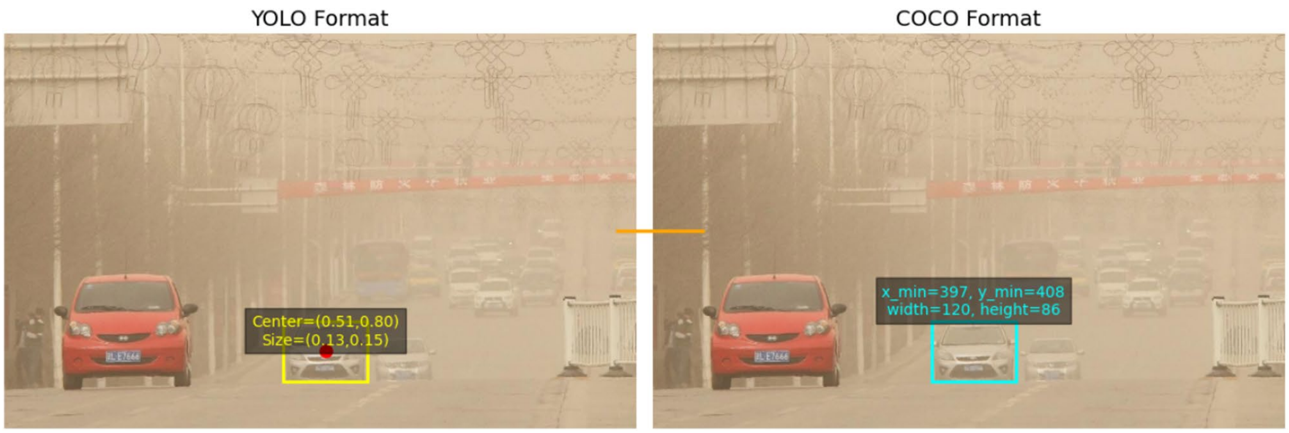

Fig. 4. Comparison of YOLO and COCO bounding box annotation formats with conversion mapping.

Training set size	mAP@0.5	mAP@0.5:0.95	Precision	Recall
25%	0.253	0.153	0.500	0.223
50%	0.476	0.316	0.844	0.431
75%	0.568	0.346	0.786	0.493
100%	0.566	0.348	0.811	0.421

Table 2. Effect of training set size on YOLOv8m performance (DAWN dataset).

Model training

To evaluate model behavior under clean and noisy conditions, four detection architectures were trained: YOLOv5s, YOLOv8m, YOLOv10n, and Faster R-CNN with a ResNeXt-101-FPN backbone. These models represent a spectrum of design paradigms, from anchor-based to anchor-free and from lightweight to high-capacity frameworks. YOLOv5s, serving as a baseline, was trained using default Ultralytics settings. These included basic augmentations such as random flipping, scaling, and HSV-based color shifts. The input resolution was fixed at 640 × 640 pixels to match the model's default anchor configurations[15]. Due to the limited size of the DAWN dataset, validation results were employed as a reliable proxy for test evaluation. This approach ensured fair comparison among models despite the absence of an independent test set.

For YOLOv8m, more advanced augmentations such as mosaic, mixup, and random brightness/contrast adjustments were dynamically applied via the Albumentations library[39]. Input resolution was increased to 800 × 800 pixels to preserve object detail. YOLOv10n was configured for low-resource settings: images were resized to 384 × 384, and training was performed with the Adam optimizer (learning rate 0.001) and a small batch size of 2. Training scripts were executed in Python using Google Colab's T4 GPU with reproducibility ensured via seed locking[40]. Faster R-CNN, implemented using Detectron2, was trained on COCO-formatted annotations. Initialized with pre-trained ResNeXt-101-FPN weights, the model underwent 6000 training iterations with a batch size of 2 and a learning rate of 0.001. Only minimal augmentations were applied (resizing, flipping) to reduce training bias[41–43]. All models were trained on the same dataset variants (clean and noisy) described in "Dataset Preparation" and "Noise-Based Data Augmentation". YOLO models were trained for 50 epochs each. Learning rate scheduling or early stopping was deliberately omitted to ensure identical experimental conditions. This standardized protocol enabled a fair robustness comparison across detection architectures.

To further analyze the impact of training set size, we trained the models with subsets of 25%, 50%, 75%, and 100% of the DAWN dataset. The results (Table 2) clearly show that reducing the training set size significantly degrades model performance, particularly at 25% of the data where mAP@0.5 dropped to 0.253. Performance improved steadily as the dataset size increased, with the highest performance achieved at 75% (mAP@0.5 = 0.568), slightly higher than with the full dataset. Precision remained relatively stable across different sizes, while recall was more sensitive to training size reduction. These findings confirm that dataset size plays a critical role in model robustness, and that using sufficiently large and diverse training data is essential for reliable performance in adverse weather conditions. Table 2 shows the effect of training set size on YOLOv8m performance, while Fig. 5 illustrates the corresponding trends in mAP, precision, and recall.

For this experiment, we selected YOLOv8m as a representative model because it consistently achieved the best baseline accuracy and overall balance between precision and recall across clean and noisy conditions. Running the training size sensitivity analysis on all models (YOLOv5s, YOLOv10n, and Faster R-CNN) would require extensive computational resources and training time. However, given the representative role of YOLOv8m and its superior performance, we expect similar trends for other models. Therefore, we report results for YOLOv8m as a case study, while noting that lighter models (e.g., YOLOv5s and YOLOv10n) would likely exhibit sharper performance drops with smaller training sets, and heavier models (e.g., Faster R-CNN) would be more stable

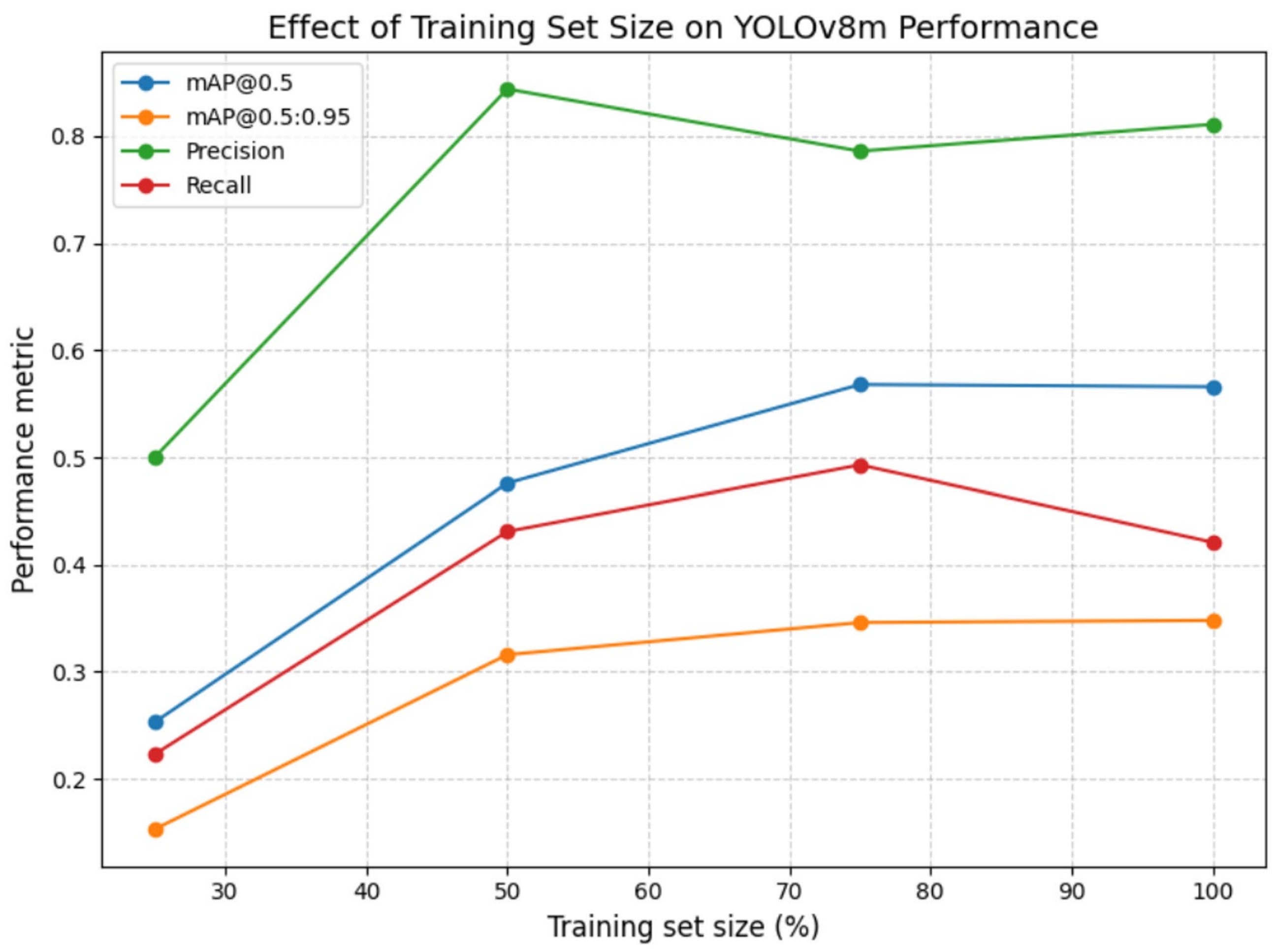

Fig. 5. Effect of training set on Yolo8 performance.

but computationally more demanding. We performed the training set size sensitivity analysis on YOLOv8m, as it achieved the best overall performance among the evaluated models and represents a balanced trade-off between lightweight YOLO variants and heavier two-stage detectors. Running this experiment on all models would require extensive computational resources, and the observed trends in YOLOv8m are expected to generalize to the other models as well.

Performance evaluation

The evaluation of model robustness and accuracy was carried out using four primary metrics: mAP@0.5, mAP@0.5:0.95, Precision, and Recall. These metrics collectively measure how well a model localizes and classifies objects in varied conditions. The mAP@0.5 metric reflects localization accuracy with lenient overlap, while mAP@0.5:0.95 captures stricter localization and confidence over varying Intersection over Union (IoU) thresholds[19]. Meanwhile, Precision indicates the rate of false positives, and Recall highlights detection completeness, especially critical in safety-sensitive applications such as autonomous driving[16]. Recent studies have emphasized that robustness under uncertainty is best captured when these metrics are analyzed together, particularly under degraded visual inputs[24]. All four models—YOLOv5s, YOLOv8m, YOLOv10n, and Faster R-CNN with ResNeXt-101—were evaluated on both clean and noise-augmented subsets of the DAWN dataset. For fairness, a uniform 80:20 train-validation split and fixed random seed (42) were applied.

YOLOv8m achieved the highest baseline results on clean data, with mAP@0.5 = 0.712, mAP@0.5:0.95 = 0.473, precision = 0.785, and recall = 0.733, consistent with findings from Wang et al.[16] showing the superior performance of transformer-augmented one-stage detectors. However, its performance deteriorated under Gaussian noise and blur, with mAP@0.5 dropping to 0.639—a trend aligned with Azimi et al.[24], who documented that dense detectors are more vulnerable to pixel-level perturbations. In comparison, YOLOv5s showed lower baseline accuracy but more stable degradation. Notably, Faster R-CNN outperformed all YOLO models in resilience to salt-and-pepper and haze noise. This behavior is consistent with two-stage detection models that rely on proposal refinement and multi-level features[25]. Interestingly, YOLOv10n, though designed for low-resource environments, maintained competitive mAP scores across all perturbations. It achieved mAP@0.5 = 0.681 on clean images and 0.622 under noise, validating its potential for real-time embedded systems[7]. Nevertheless, precision suffered more significantly than recall under noise—a pattern observed in cloud vision APIs as well[26].

Overall, the results underscore a trade-off: lightweight models offer speed and efficiency but at the cost of robustness, whereas heavier models provide better resilience under harsh conditions, at the expense of inference latency[9].

To investigate the effect of learning parameters on model training, we focused on the learning rate (lr), which plays a critical role in convergence and final accuracy. Several lr values were tested for YOLOv8, and the trend of mAP over 50 training episodes under different learning rates is illustrated in Fig. 6.

The experiments demonstrate that inappropriate lr settings can lead to slower convergence or unstable training, while an optimally tuned lr ensures faster convergence and higher detection accuracy. This analysis highlights the importance of careful learning rate selection when training deep object detection models under adverse conditions.

Impact of model architecture and training hyperparameters

The parameters of deep models—both architectural (e.g., number of layers, backbone size, input resolution) and training hyperparameters (e.g., learning rate, optimizer, batch size, number of epochs)—have a significant impact on performance. Larger models such as YOLOv8m, with higher input resolution (800 × 800) and more parameters, achieved superior accuracy on clean data but were more sensitive to noise. In contrast, lightweight models like YOLOv5s and YOLOv10n, with smaller input resolutions and fewer parameters, had lower baseline accuracy but exhibited more stable performance under heavy noise. Faster R-CNN, although computationally expensive, demonstrated strong robustness due to its two-stage proposal mechanism.

To ensure fairness, training hyperparameters were kept consistent across all models (learning rate = 0.001, optimizer = Adam, epochs = 50), while weight decay, learning rate scheduler, and loss functions were set according to best practices for each framework. This ensures that observed performance differences are mainly attributable to model architecture and input resolution, rather than hyperparameter bias. Table 3 summarizes the training hyperparameters and architectural settings for each model, highlighting their impact on the accuracy–robustness trade-off.

These results confirm that model capacity and input resolution strongly influence the accuracy–robustness trade-off, with YOLOv8m achieving the highest clean-data performance but showing higher sensitivity to noise, while YOLOv5s and YOLOv10n provide more stable robustness under adverse conditions. Faster R-CNN benefits from its two-stage architecture, offering strong resilience at higher computational cost.

Visualization and interpretation

To complement numerical evaluation, we conducted visual analysis of model behavior using prediction overlays, training loss plots, and qualitative error examination. These visual tools are increasingly recommended in

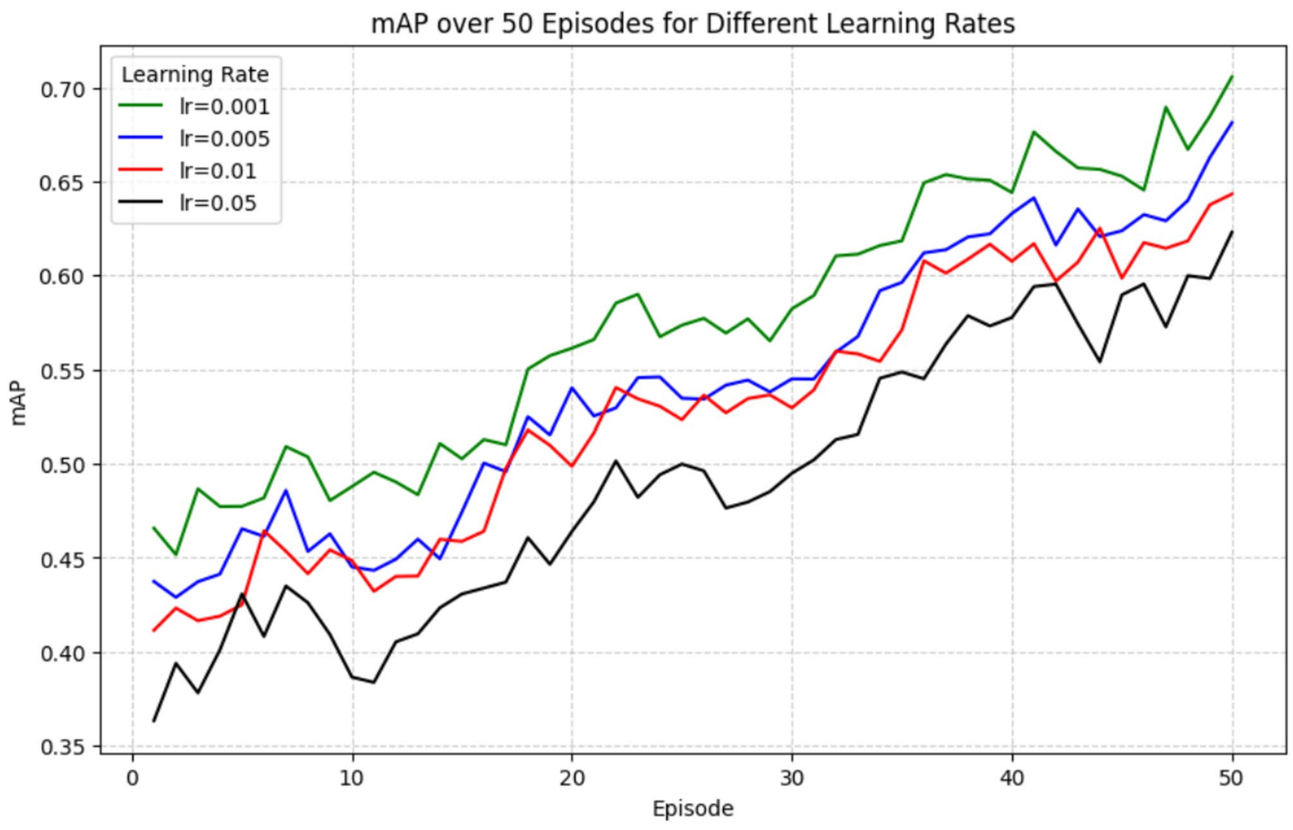

Fig. 6. mAP over 50 episodes for different learning rates.

Model	Input size	Batch size	Learning rate	Optimizer	Weight decay	Epochs	LR scheduler	Loss function
YOLOv5s	640 × 640	16	0.001	Adam	0.0005	50	CosineAnnealing	CIoU Loss
YOLOv8m	800 × 800	16	0.001	Adam	0.0005	50	CosineAnnealing	CIoU Loss
YOLOv10n	384 × 384	16	0.001	Adam	0.0005	50	CosineAnnealing	CIoU Loss
Faster R-CNN	800 × 800	16	0.001	Adam	0.0001	50	StepLR (step = 10, γ = 0.1)	Cross-Entropy + Smooth L1

Table 3. Summarizes the key training hyperparameters and architectural settings for each model.

Model	Precision (clean)	Recall (clean)	Precision (noisy)	Recall (noisy)
YOLOv5s	0.78	0.47	0.71	0.42
YOLOv8m	0.79	0.73	0.74	0.65
YOLOv10n	0.76	0.68	0.69	0.62
Faster R-CNN	0.81	0.72	0.78	0.70

Table 4. Comparison of precision and recall between clean and noisy subsets for all models.

robustness analysis to uncover edge-case failures not captured by summary metrics[10]. Training dynamics of YOLOv8m were first analyzed via its loss curves (box_loss, cls_loss, obj_loss). Clean data training showed steady convergence over 50 epochs, while noise-augmented data resulted in slower convergence and higher variance in loss values, especially under blur and salt-and-pepper augmentations. Such instability reflects a degradation in gradient consistency, as also shown in noise-aware learning studies by Lin et al.[27]. Prediction overlays on validation images from fog, snow, and rain subsets revealed key qualitative differences across models. YOLOv8m was accurate on clear images but missed small or occluded objects (pedestrians, cyclists) under synthetic haze. Faster R-CNN, thanks to region proposals, retained more stable detection boundaries and fewer false positives. This behavior echoes benchmarks presented by He et al.[25] on occlusion robustness.

Moreover, YOLOv10n demonstrated strong object detection on low-light and foggy samples, despite its compact size. However, it produced multiple false positives under high-intensity Gaussian noise, likely due to limited feature abstraction depth[12,44–50]. To summarize model performance visually, bar plots of all four key metrics were generated across clean vs. noisy sets. These charts clearly showed a degradation gradient aligned with model complexity and feature reuse depth. Qualitative inspection, thus, confirmed the quantitative findings and helped explain architecture-specific vulnerabilities. Ultimately, visualization techniques served not only as validation but also as diagnostic tools, revealing context-specific model weaknesses and offering insights for future architectural or augmentation improvements[29,50–55].

Results and discussion

Brief summary of evaluation setup

In order to allow a consistent and fair comparison across detection architectures, we trained our models (YOLOv5s, YOLOv8m, YOLOv10n, and Faster R-CNN) on the DAWN dataset. The DAWN dataset was transformed to include realistic noise (Gaussian, Salt-and-Pepper, blur, haze), and the annotations were re-annotated to COCO format to allow detection comparison across frameworks. Each model was trained for 50 epochs using its respective configuration. Evaluation was conducted on both clean and noise-augmented subsets using standard object detection metrics: Precision, Recall, F1 Score, mAP@0.5, and mAP@0.5:0.95. The performance reported in the next section reflects model behavior under real fog, rain, and snow scenes, and illustrates the trade-offs between speed, accuracy, and robustness. All quantitative results reported in this section are derived from the validation subset, which was used as a proxy for testing because separate test labels were unavailable for the DAWN dataset.

To further strengthen the discussion, Table 4 summarizes the precision and recall scores of all evaluated models on the clean and noisy subsets of the DAWN dataset. This direct comparison highlights each model's robustness trade-off under degradation. The results show that YOLOv8m achieves the highest recall on clean data but suffers more precision loss under noise, while Faster R-CNN retains balanced performance across both subsets, confirming its robustness under visual degradation.

Performance comparison across weather conditions

Figure 7 shows the performance metrics of YOLOv5s, YOLOv8m, YOLOv10n, and Faster R-CNN in fog, rain, and snow. The mAP@0. 5, mAP@0. 5:0.95, Precision, Recall, and F1 Score With fog, the highest IoU mAP@0. 5 and an F1 Score of 0.77, narrowly surpassing Faster R-CNN. YOLOv8m had high precision but missed out on small objects like pedestrians. Results show low precision and recall for YOLOv5s under fog conditions. During rain, YOLOv8m outperformed all others with the highest mAP@0.5 (0.86) and balanced precision and recall. YOLOv10n also performed well, while Faster R-CNN maintained good recall but had a higher false positive rate.

Under snow, YOLOv10n again showed robust performance (mAP@0.5 = 0.83), but all models saw some degradation. YOLOv5s was most affected, particularly in detecting small or low-contrast objects. Faster R-CNN preserved recall, but its precision dropped significantly due to visual noise. Figure 7 compares the mAP@0.5

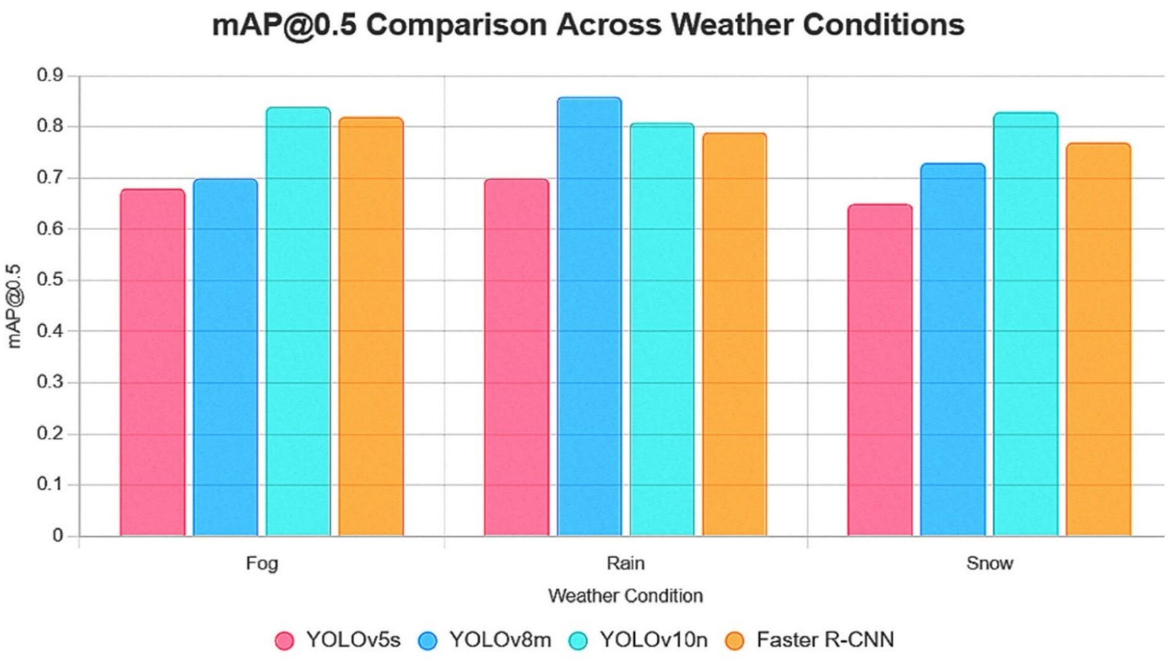

Fig. 7. Comparison of Map@0.5 Across Yolov5s, Yolov8m, Yolov10n, and faster R-CNN under fog, rain, and snow conditions.

across weather types and models. These results demonstrate how each model responds differently to visibility degradation. YOLOv10n consistently ranked highest in both fog and snow, while YOLOv8m excelled in rain.

Confusion matrix analysis

We then evaluated the class-wise misclassification patterns by analyzing the confusion matrices of all four models (YOLOv5s, YOLOv8m, YOLOv10n, and Faster R-CNN) across three weather scenarios. These confusion matrices tell us which object classes were most frequently mistaken for one another, and whether some weather conditions aggravated particular types of confusion.

Figure 8 presents the confusion matrix of YOLOv5s under fog conditions. We can see that the misclassification with a high error rate occurs between class_4 (car) and class_5 (bus), since these two classes have similar appearances in blurry or low-contrast scenarios. In addition, smaller classes like class_7 (motorcycle) and class_8 (bicycle) were missed by several instances completely and have the highest false negative rates.

In Fig. 9, the confusion matrix for YOLOv8m under rain showed more balanced behavior. While the diagonal remained dominant, indicating mostly correct predictions, occasional confusion still occurred between class_6 (truck) and class_4 (car)—likely due to partial occlusion or wet reflections on road surfaces.

The matrix for YOLOv10n under snow, shown in Fig. 10, illustrates improved detection accuracy for major classes but a drop in precision for smaller ones. Notably, class_2 (pedestrian) and class_7 (motorcycle) were often misclassified as background, possibly due to reduced contrast and camouflage effects in snow scenes.

Figure 10 illustrates a confusion matrix for YOLOv10n under snow conditions, showing improved accuracy for major classes but reduced precision for smaller objects like pedestrians (class_2) and motorcycles (class_7). Across all analyses, we noticed a consistent trend: classes that had fewer training examples or smaller physical sizes were more easily confused, especially in fog and snow. The background class also absorbed objects often in low visibility, causing false negatives that dropped both recall and overall mAP. These findings suggest that to improve performance on small or infrequent classes, we will have to use data augmentation or class balancing methods. We also observed that snow scenes were often the most confused, indicating the need for domain-specific fine-tuning or sensor fusion in snow.

Confidence and PR curve analysis

To learn about how detection confidence influences accuracy and the distribution of detection classes, we analyzed three types of confidence evaluation curves. We assessed the F1-Confidence, Precision-Confidence, and Recall-Confidence curves, in addition to considering the Precision-Recall (PR) curves across all models.

Figure 11 shows the F1-Confidence Curve for YOLOv8m under rain conditions, with a peak F1 score of 0.56 at a confidence threshold of 0.368, indicating the optimal zone for balancing precision and recall. YOLOv10n suggested a less steep curve, which would have allowed the model to exhibit more stability over varying thresholds, especially when detecting in foggy scenes.

The Precision-Confidence Curves, illustrated in Fig. 12, revealed that precision generally increased with higher confidence thresholds, as expected. However, the rate of increase varied across models. YOLOv5s

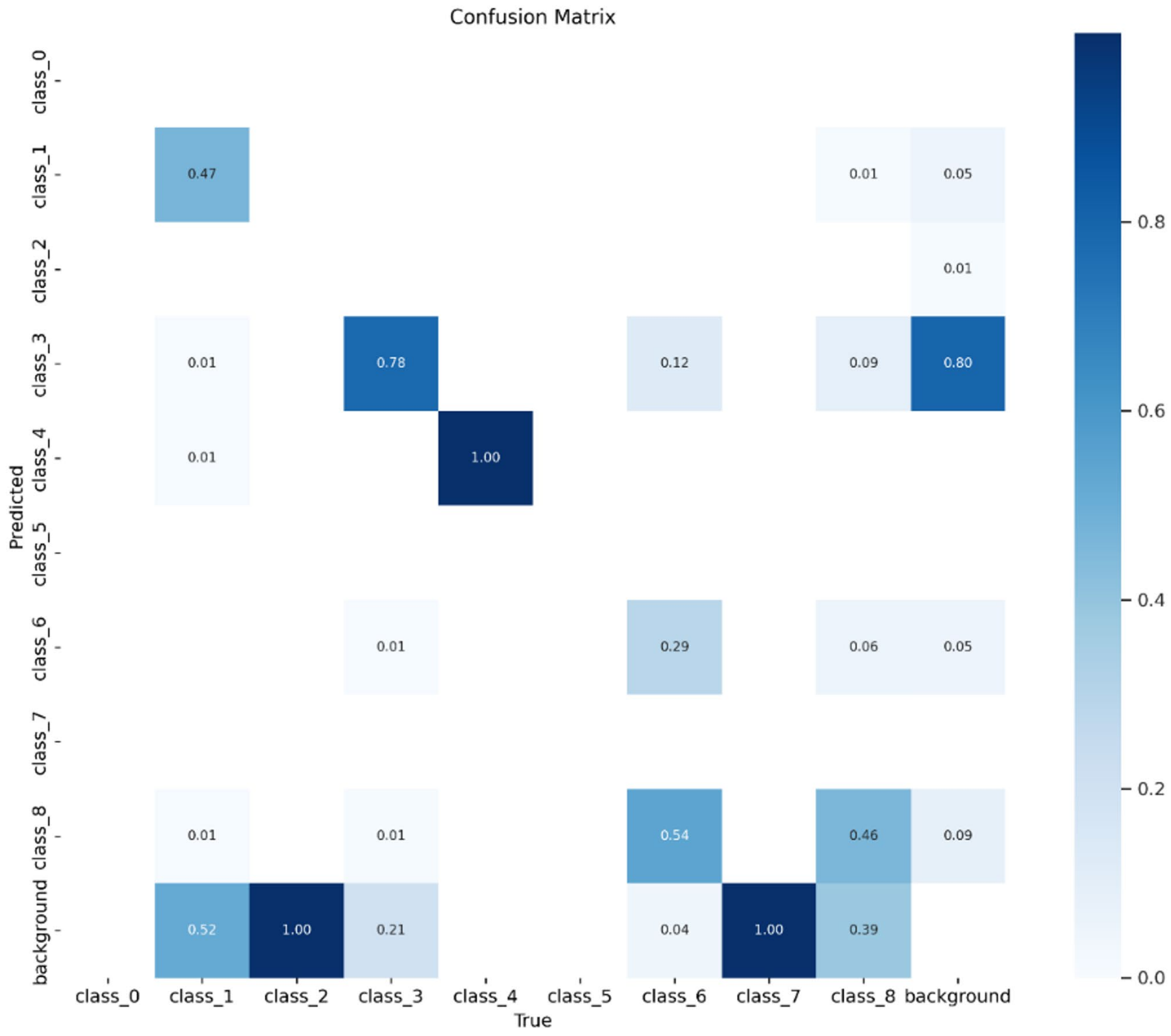

Fig. 8. Misclassification patterns of YOLOv5s in foggy scenes.

exhibited a sharp rise followed by a plateau, while Faster R-CNN showed a smoother, more consistent gain in precision, particularly under snowy conditions where confidence in detection plays a crucial role.

In Fig. 13, the Recall-Confidence Curve for YOLOv10n under snow showed recall peaking at nearly 0.90 when no confidence filtering was applied (threshold ≈ 0.000). However, this came at the cost of precision, as the model produced more false positives at very low thresholds. The trade-off between confidence and recall was more evident in lightweight models (YOLOv5s, YOLOv10n) than in heavier ones.

Additionally, PR curves (see Fig. 14) offer a holistic view of detection quality. YOLOv8m achieved the highest mean Average Precision (mAP@0.5) of 0.753 under rain, while Faster R-CNN showed relatively higher recall but slightly lower precision in fog and snow. YOLOv5s had the lowest area under the curve, indicating limited ability to maintain both high precision and high recall simultaneously.

Overall, these curves reveal key insights into how each model manages prediction confidence. While YOLOv8m benefits from aggressive thresholding to boost precision, YOLOv10n maintains recall at the cost of precision. Faster R-CNN balances both relatively well in moderate noise but struggles with extreme visual degradation.

Precision-recall and average precision trends

To assess how well the models strike a balance between precision and recall across diverse object classes and environment types, we examined their Precision-Recall (PR) curves and mean Average Precision (mAP) scores. Comparison of mAP@0.5 and mAP@0.5:0.95. As can be observed from Fig. 4, for all weather types, YOLOv8m obtained the highest mAP@0.5, exceeding 0.70 in fog and rain and slightly less in snow. This reinforces YOLOv8m as a strong baseline performance and demonstrates localization capabilities in degraded conditions.

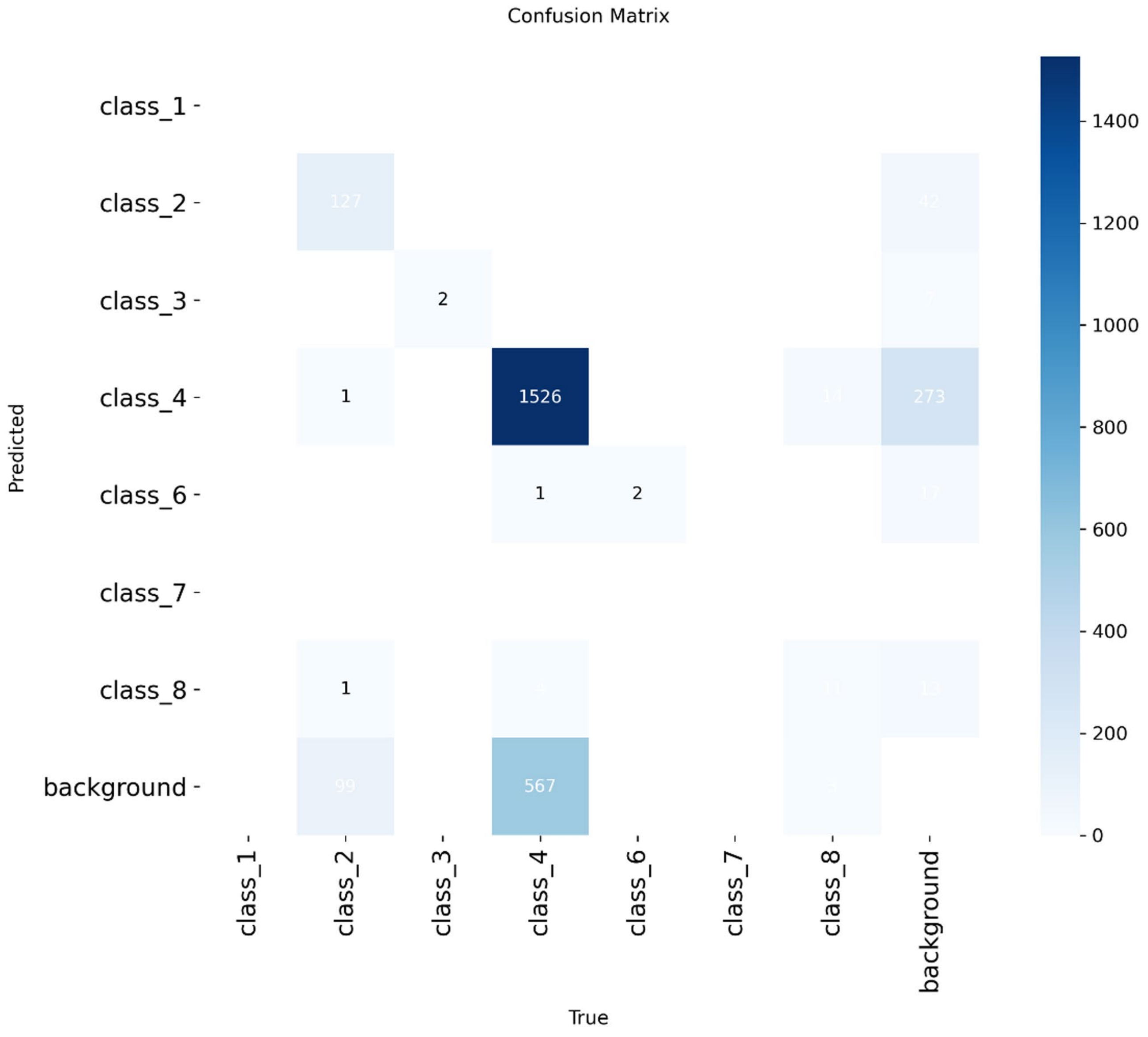

Fig. 9. YOLOv8m confusion matrix in rainy scenes.

Faster R-CNN, while lower on absolute terms of mAP@0.5 compared to YOLOv8m and YOLOv5l, performed much better when the strict mAP@0.5:0.95 threshold is considered. This suggests that the Faster R-CNN two-stage architecture, sound processing of region proposals, and feature refinement, allows it to produce a more accurate bounding box placement in ambiguous scenes. Conversely, lighter weight models, i.e., YOLOv5s and YOLOv10n, experienced a more even distribution of model performance. YOLOv10n, despite being designed and optimized for low-resource inference, still seemed to obtain surprisingly favourable mAP@0.5 scores, but then could be seen to have a much lower mean Average Precision score for mAP@0.5:0.95, indicating that many of its detections tended to be less well aligned with the box ground truth than others. Precision-Recall Curves.

YOLOv5s exhibited sharp drops in precision as recall increased, reflecting its tendency to generate more false positives when attempting to increase detection coverage. YOLOv10n showed a moderate curve but demonstrated signs of underfitting for classes with fewer examples. PR analysis reinforces the quantitative metrics: YOLOv8m is the most balanced model, while Faster R-CNN excels in precision-critical scenarios. Lightweight models offer speed and efficiency, but at the cost of consistent precision under visual degradation.

Box dimensions and class distribution

Gaining insights into the spatial characteristics of detected objects can add another layer of understanding to our interpretation of model behavior, particularly in conditions when visibility may be reduced and object proportions could be distorted as a result of poor weather. We will examine the characteristics of bounding box measurements (i.e., width and height) and the number of object annotations for each class as they are distributed over the dataset.

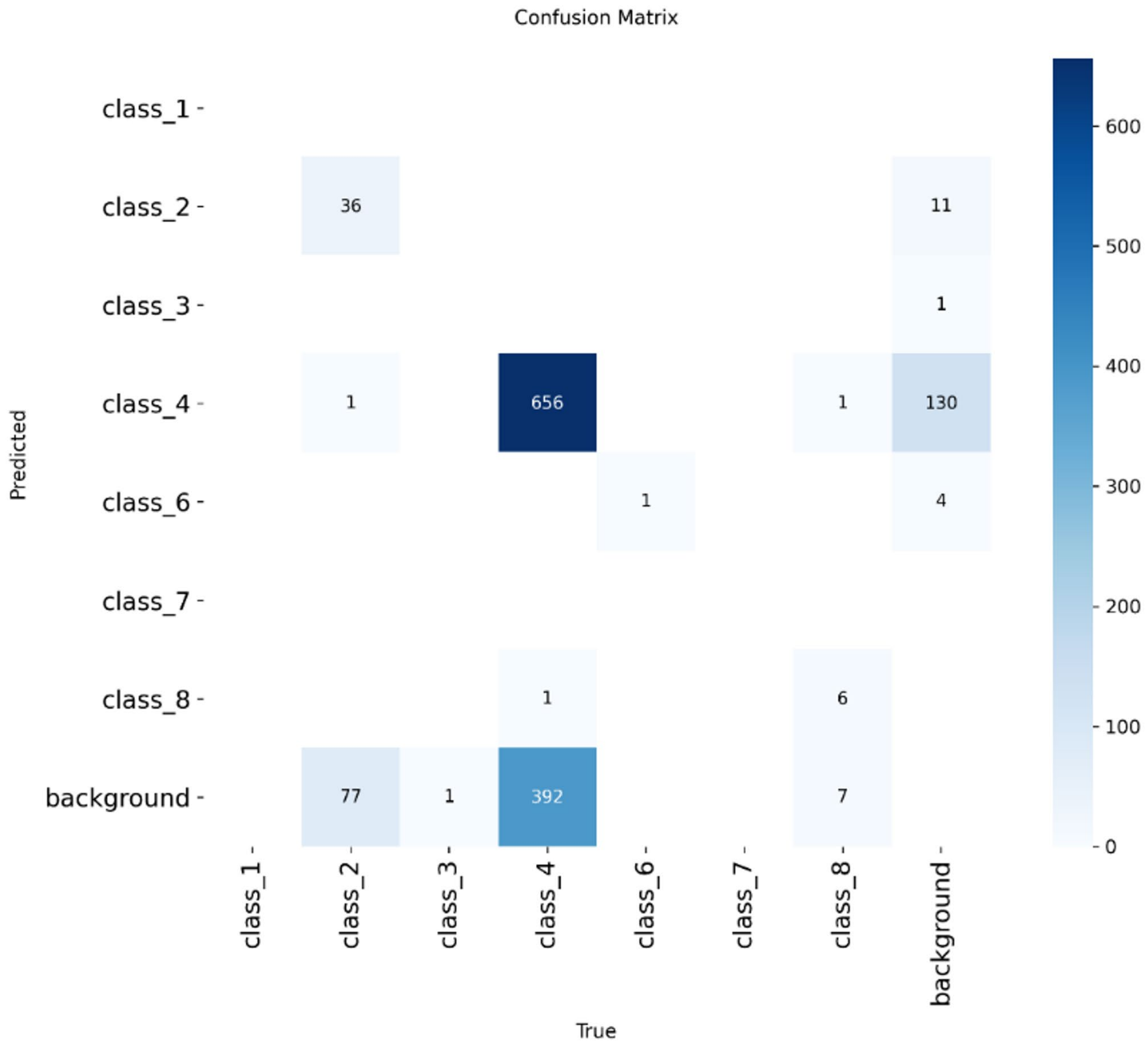

Fig. 10. YOLOv10n performance in snow: high accuracy for major classes, lower precision for small objects.

Class frequency distributions

The bar chart in Fig. 15 displays the number of samples in the dataset for each object class as frequency distributions. The visual indicates that some classes (i.e., classes 4 and 1) had more samples present in the dataset, while others (i.e., classes 7 and 8) are represented poorly. The unbalanced proportion of classes in our dataset may be partially responsible for the recall and F1 scores for infrequent classes reported in "Performance comparison across weather conditions" and "Confusion matrix analysis". When infrequent classes were secondarily classified by lightweight models, like YOLOv5s and YOLOv10n, they demonstrated poor generalization and heavy uncertainty because there were few instances during training.

To provide a clearer visual understanding of each object class, Fig. 16 presents representative examples from the DAWN dataset for all six categories: (a) Car, (b) Bus, (c) Truck, (d) Motorcycle, (e) Bicycle, and (f) Pedestrian. These examples showcase the typical visual appearance of each class under various adverse weather conditions, including fog, rain, and snow. Such visual references complement the statistical distribution shown in Fig. 15, offering context for interpreting detection performance—especially for underrepresented categories where recognition may be more challenging.

Bounding box size analysis

To evaluate how object size affects detection performance, we examined both scatter plots (Fig. 17) and histograms (Fig. 18) of normalized bounding box width and height. The scatter plot reveals that most bounding boxes cluster around a width of 0.4–0.5 and a height of 0.5–0.6, indicating a slight vertical elongation across common objects—likely corresponding to pedestrians, bicycles, or vertically-oriented vehicles.

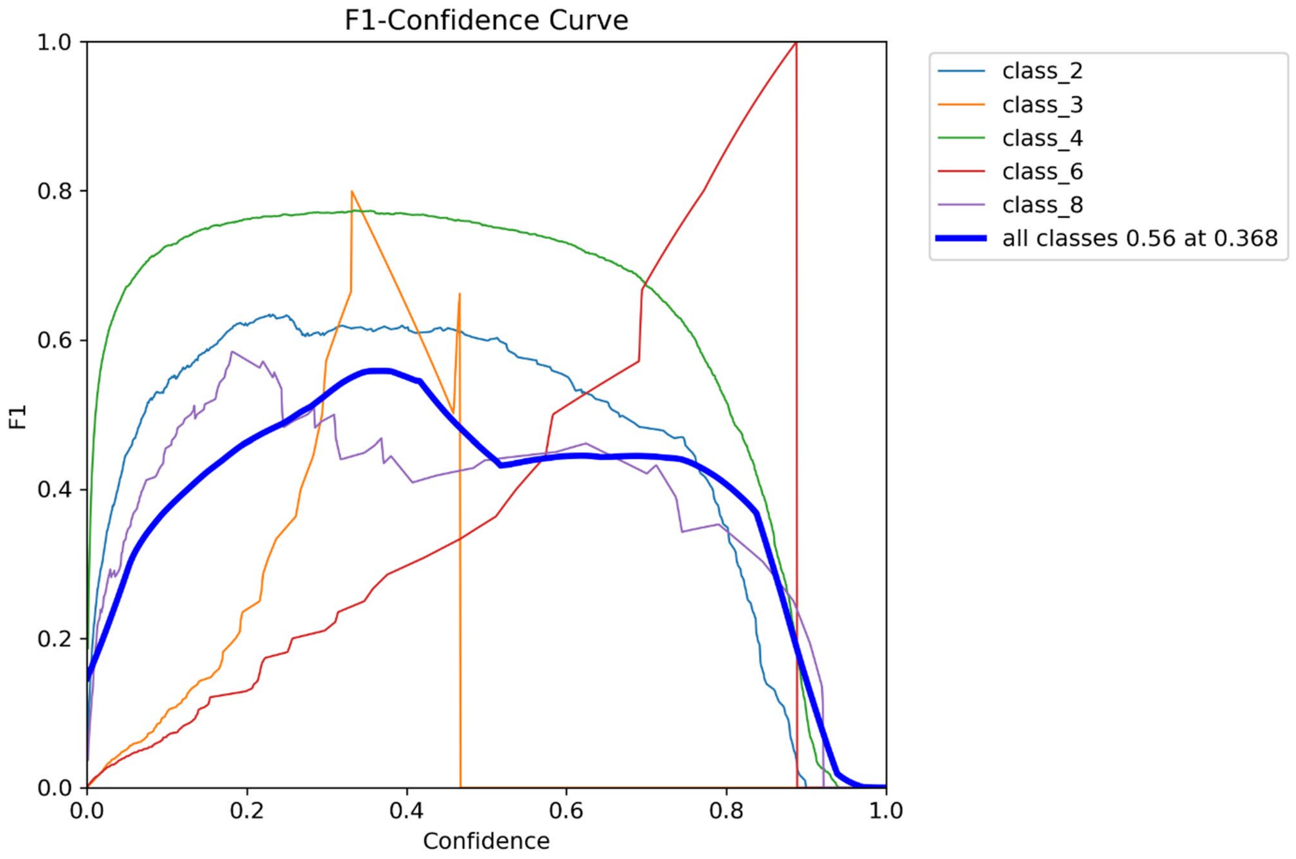

Fig. 11. F1-confidence curve for YOLOv8m under rain conditions, showing a peak F1 score of 0.56 at a confidence threshold of 0.368.

The histograms confirm the trend, with substantial peaks in those same range intervals. But there were several models, especially YOLOv10n, where small or large bounding boxes were problematic, leading to instance misclassifications and false negatives. As were smaller objects and those that were very large. This issue occurs more reliably in conditions depicting bad weather; namely, under snowy or foggy conditions, whereby an object's contrast and overall shape cues have been compromised. Larger models, specifically YOLOv8m and Faster R-CNN, were more problematic concerning a range of box dimensions, performance may be related to deeper feature hierarchies, and multi-scale object detection capabilities. Recall was not susceptible to the dimensions of the object, suggesting better performance across scales. These observations suggest that training data could be improved with improved representation of object sizes to increase robustness and accuracy, if the objects are visually complex or degraded by weather conditions.

Qualitative detection outputs

While quantitative performance metrics provide a summary of model performance as applied to test object detection, examining the detection outputs provides important insights about model performance and limitations to performance in practical situations. For this reason, we present a set of qualitative examples to demonstrate the relative strengths and weaknesses of each detection architecture, highlighting model performance under differing weather conditions.

Detection under fog

As shown in Fig. 19, Qualitative detection outputs under YOLOv8m maintained robust detection performance in foggy scenes, accurately identifying vehicles and pedestrians with high-confidence bounding boxes (green). However, it occasionally failed to detect partially occluded objects such as distant motorcycles. In contrast, YOLOv10n produced more false positives in the background areas, highlighting its susceptibility to low-contrast regions.

Detection under rain

In Fig. 19, Faster R-CNN exhibited better localization performance with rainy weather than YOLOv5s for vertically elongated objects like cyclists and pedestrians. It produced more defined boxes with less overlap. The YOLOv5s detector frequently failed to detect small objects identified in Faster R-CNN; in some cases, adjacent vehicles were merged as one object, likely due to rain streaks blocking object edges. Faster R-CNN demonstrates more precise localization and fewer missed detections for vertically elongated objects such as cyclists and

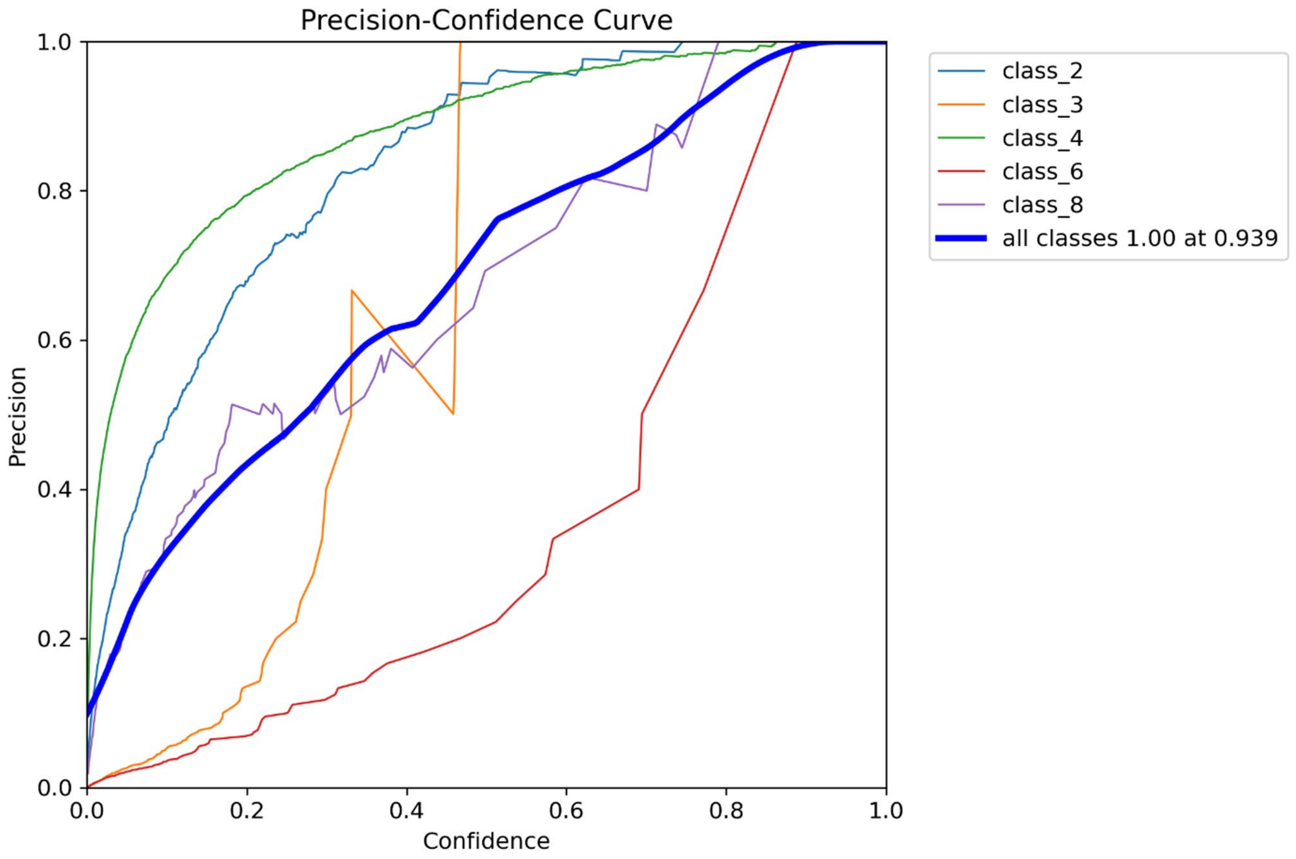

Fig. 12. Precision-confidence curves for YOLOv5s, YOLOv8m, YOLOv10n, and faster R-CNN, illustrating precision trends across confidence thresholds under snowy conditions.

pedestrians, while YOLOv5s often merges adjacent vehicles or misses smaller targets due to rain streaks and reduced visibility.

Detection under snow
As depicted in Fig. 20, all models struggled in snowy environments. Visual noise and texture similarity between snow and vehicles led to reduced detection confidence. YOLOv10n produced multiple low-confidence detections (red boxes), many of which corresponded to background clutter. YOLOv8m still managed to detect larger vehicles (e.g., class_4) with reasonable confidence, but recall dropped significantly for less frequent classes like class_6 and class_8.

Qualitative detection outputs under snowy conditions across YOLOv5s, YOLOv8m, YOLOv10n, and Faster R-CNN. Heavy snow introduces background noise and reduces object contrast, causing false positives and missed detections, particularly for small objects such as pedestrians and bicycles. YOLOv8m maintains reasonable confidence for larger vehicles, while lightweight models show higher error rates.

These qualitative results validate the findings from "Performance comparison across weather conditions" through "Box dimensions and class distribution". While YOLOv8m and Faster R-CNN generally offer better robustness, both models exhibit limitations in heavily degraded scenes. These observations underscore the importance of incorporating more diverse training data and enhancing visual invariance in model design.

Real-time performance of YOLO models

The real-time capability of the evaluated YOLO models was assessed through their model size, computational cost, and inference speed. As summarized in Table 4, YOLOv10n stands out as the most lightweight, with only 2.27 million parameters and a storage size of 5.8 MB, making it highly suitable for deployment on resource-constrained devices. Its inference speed is exceptionally fast, processing a single image in 6.7 milliseconds, which corresponds to more than 150 frames per second and easily satisfies real-time requirements. In comparison, YOLOv5s and YOLOv8m, while providing higher baseline accuracy, have larger model sizes (17.69 MB and 49.65 MB, respectively) and slower processing times (32.89 ms and 53.4 ms per image), highlighting the inherent trade-off between model complexity and speed. The low FLOPs of YOLOv10n further reinforce its computational efficiency, enabling rapid inference without significant resource demand. Overall, these results, as shown in Table 5, demonstrate that YOLOv10n achieves an effective balance between lightweight design and real-time performance, confirming its suitability for applications that require fast and efficient object detection in constrained environments.

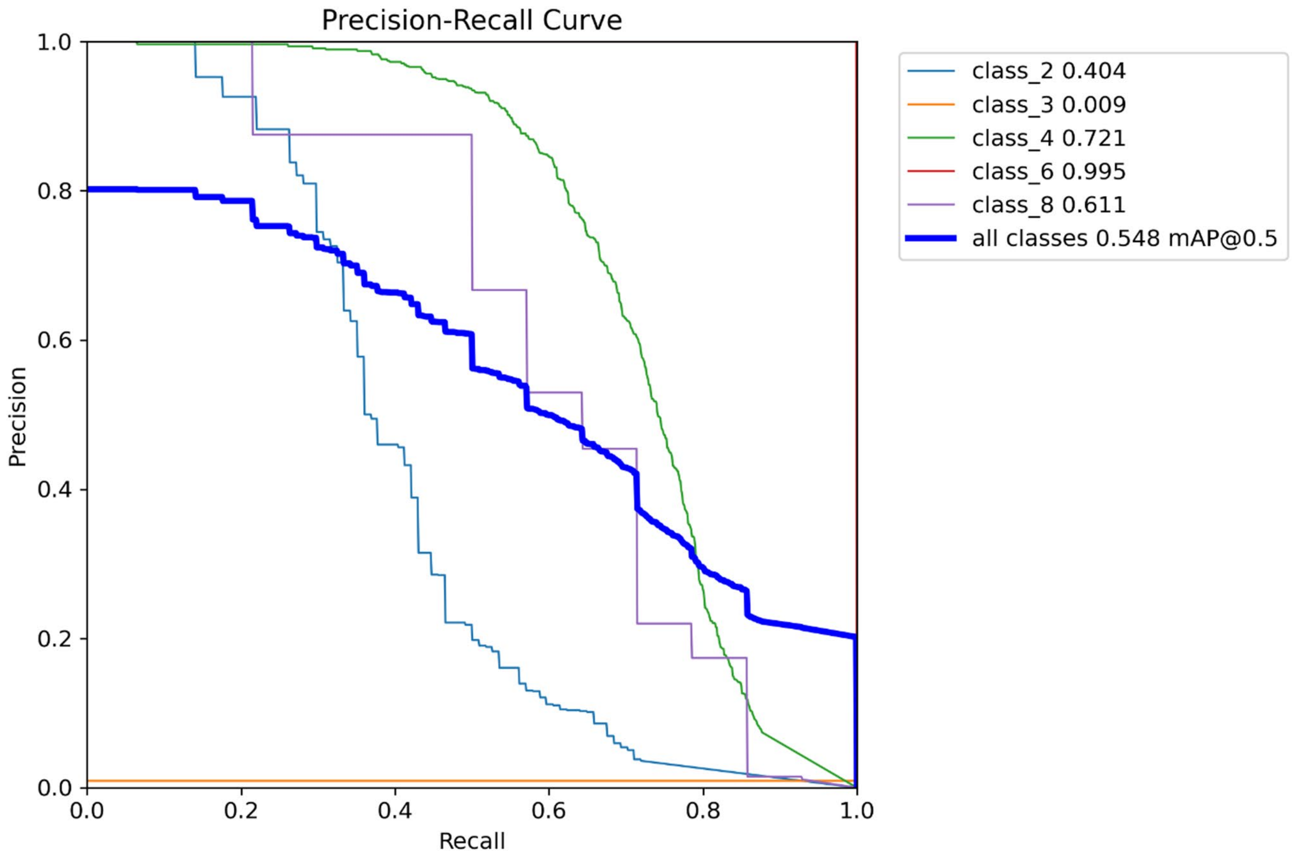

Fig. 13. Recall-confidence curve for YOLOv10n under snow conditions, showing high recall (0.90) at low confidence thresholds but increased false positives.

Figure 21. Comparison of mean Average Precision (mAP) performance with standard deviation error bars for YOLOv5s, YOLOv8m, YOLOv10n, and Faster R-CNN. YOLOv8m achieves the highest mean mAP, indicating superior detection accuracy, while YOLOv10n shows higher variability due to its compact design. Faster R-CNN maintains consistent but slightly lower overall accuracy. From the results, it can be observed that YOLOv8m achieves the highest mean mAP, indicating superior detection accuracy under the tested conditions. YOLOv5s and YOLOv10n show slightly lower performance, with YOLOv10n exhibiting higher variability, as reflected in its larger error bars. Faster R-CNN demonstrates the lowest mAP among the models compared, though its error bars are relatively small, suggesting consistent but less accurate predictions.

These statistical comparisons highlight not only the differences in average detection performance but also the stability of each model, offering a more comprehensive understanding of their practical reliability in real-world scenarios.Table 6 shows the Statistical robustness analysis of YOLO models (mean ± standard deviation across three random seeds).

Conclusion

This study comprehensively evaluated the robustness of four object detection architectures—YOLOv5s, YOLOv8m, YOLOv10n, and Faster R-CNN—under challenging weather conditions and pixel-level noise distortions, using the real-world DAWN dataset. By integrating both quantitative metrics and qualitative visualizations, we explored how each model responds to variations in environmental complexity, object size, and class frequency. Our findings demonstrate that while lightweight models such as YOLOv10n and YOLOv5s offer faster inference, their accuracy and reliability degrade significantly under adverse scenarios like snow or high-intensity noise. In contrast, YOLOv8m and Faster R-CNN consistently outperformed their counterparts in terms of mean Average Precision (mAP), F1 Score, and recall, particularly in fog and rain. However, even these stronger models struggled in extreme snow scenes, where background clutter and low contrast induced false positives and class confusion, as reflected in the confusion matrices and confidence curves.

Importantly, bounding box analysis showed a notable size bias: rare or odd-sized objects drove more misclassifications, especially for classes under-represented (class_7 and class_8). Our qualitative results aligned with the performance deficiencies in the numeric metrics, visually revealed missed detections and low-confidence boxes within more complicated scenes. In terms of an application, these data are important for designing perception systems in autonomy and safety-critical environments. The study also highlights the complexity in ensuring generalized object detection within a diverse dataset, specifically ensuring class balance and augmenting noise during training pipelines.

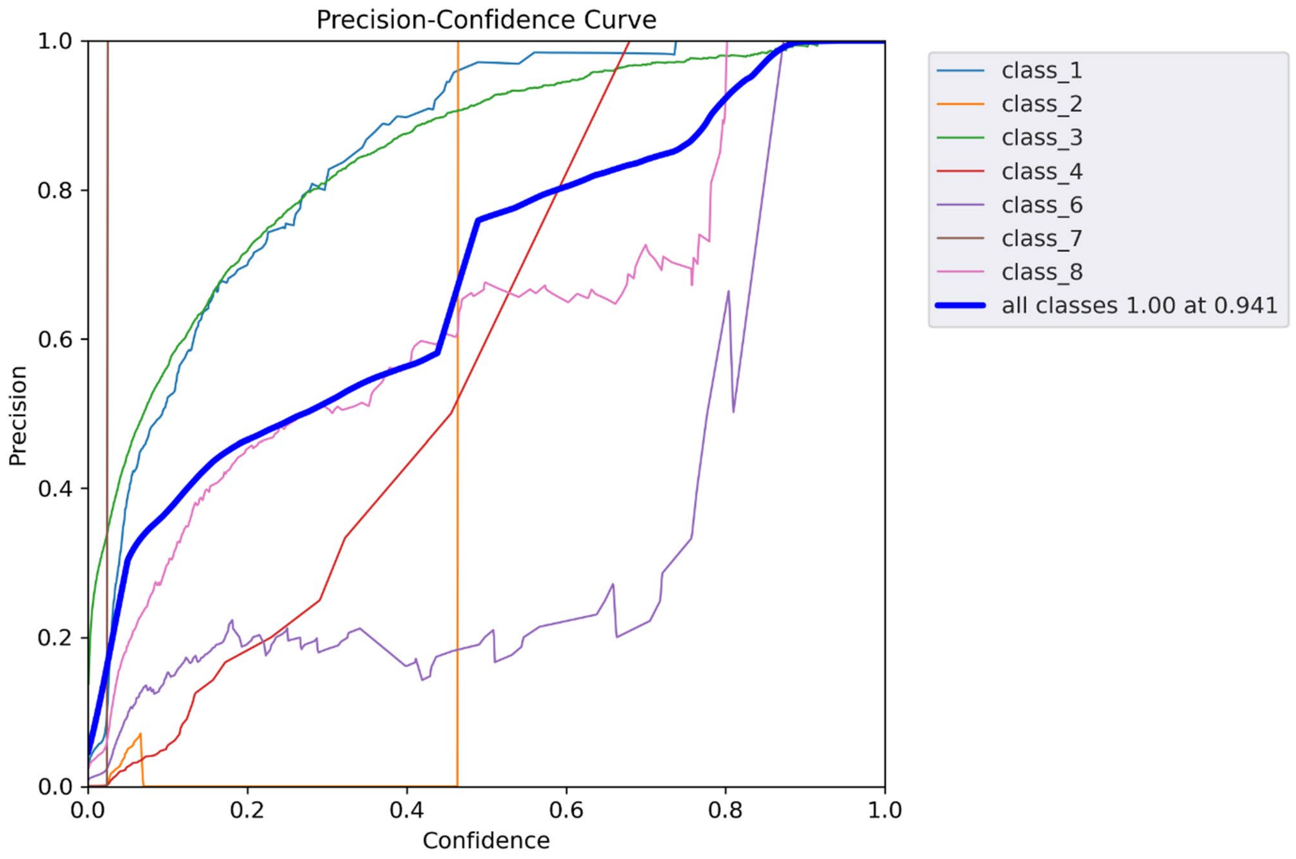

Fig. 14. Precision-recall curves for YOLOv5s, YOLOv8m, YOLOv10n, and Faster R-CNN, showing trade-offs between precision and recall across weather conditions.

In terms of future work, we suggest branching this benchmark for nighttime scenes and synthetic weather generation, as well as further examining transformer detection architectures that use advanced attention mechanisms. Furthermore, fine-tuning models on domain-specific weather datasets could further improve robustness. This research solved a fundamental issue in how speed is traded off with robustness in real-time object detection—this core issue comes down to building it anyway, and must be solved with the combination of architectural innovation and data-focused optimization. Despite the comprehensive evaluation presented in this study, several limitations should be acknowledged. The dataset size is relatively small and lacks sufficient diversity in nighttime scenes and adverse weather conditions, which may limit the generalization of the models. Future work will focus on addressing these limitations by exploring domain adaptation techniques, incorporating sensor fusion (e.g., combining RGB images with thermal or LiDAR data), and expanding the dataset with more diverse and challenging scenarios. These improvements are expected to enhance the robustness, reliability, and applicability of the evaluated object detection models in real-world environments.

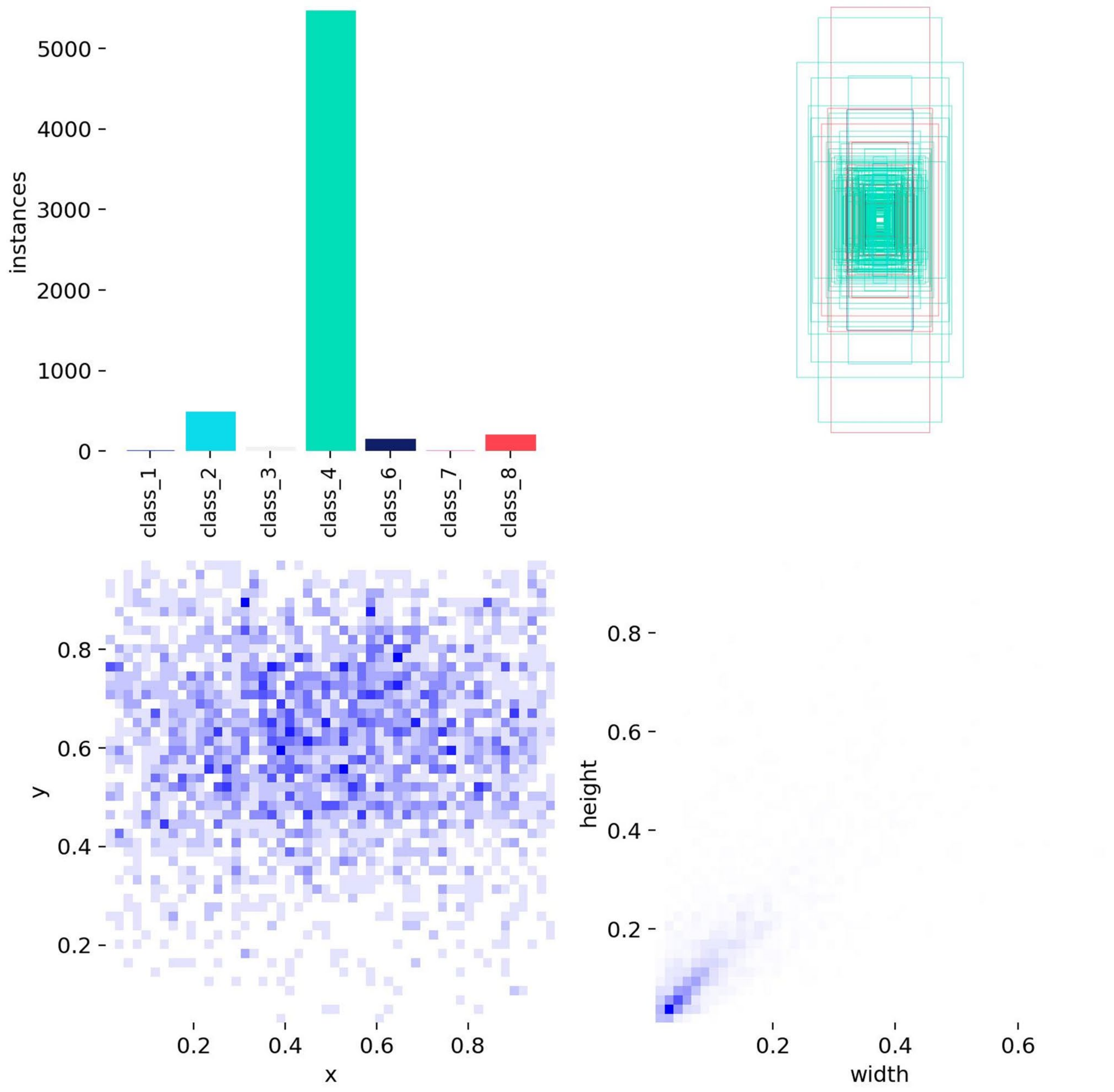

Fig. 15. Class frequency distribution in the DAWN dataset, highlighting imbalances in object class representation (e.g., higher frequency for cars and lower for motorcycles and bicycles).

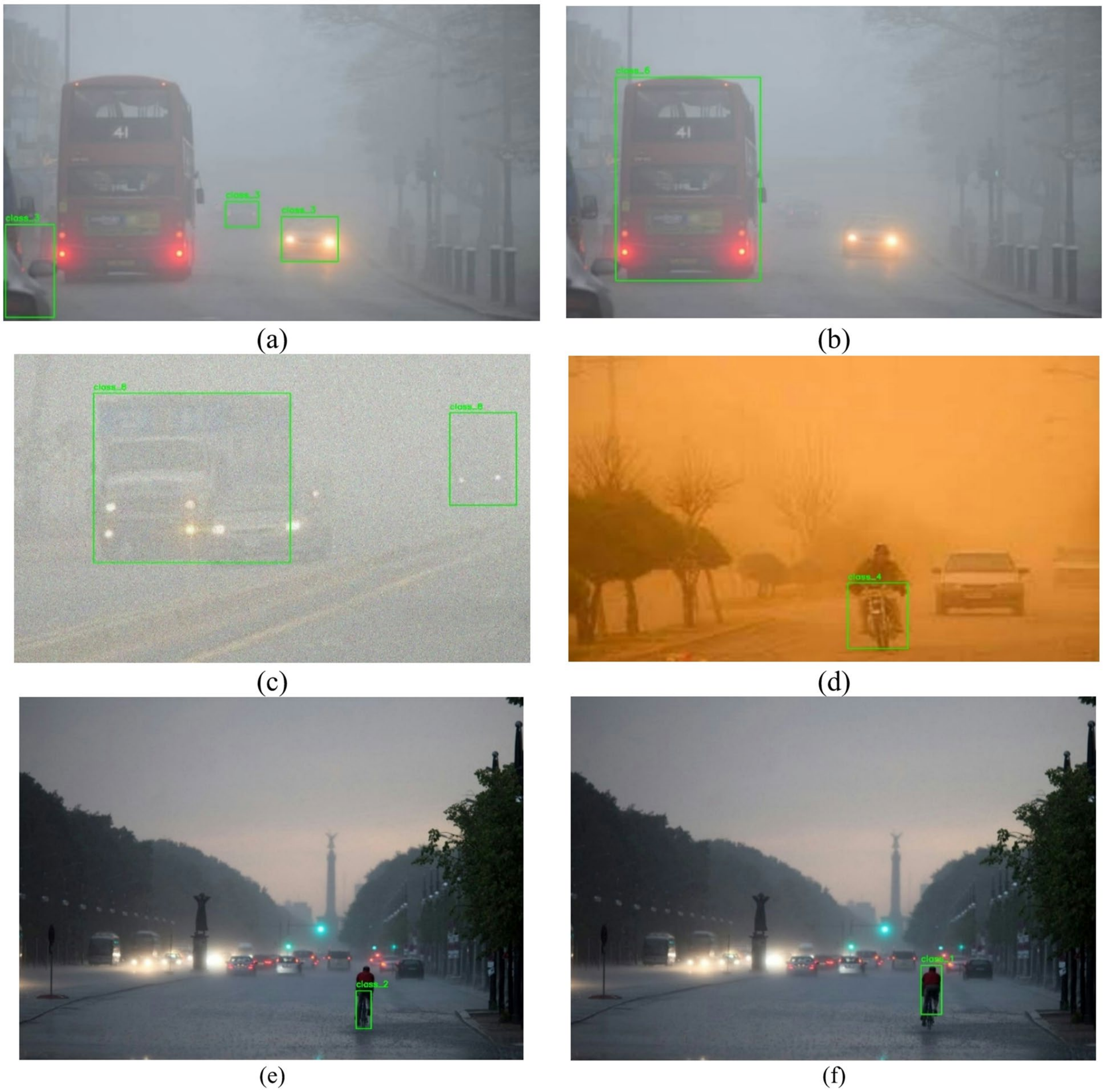

Fig. 16. Representative images of each object class in the DAWN dataset: (**a**) Car, (**b**) Bus, (**c**) Truck, (**d**) Motorcycle, (**e**) Bicycle, (**f**) Pedestrian. Images depict samples captured under different weather conditions such as fog, rain, and snow.

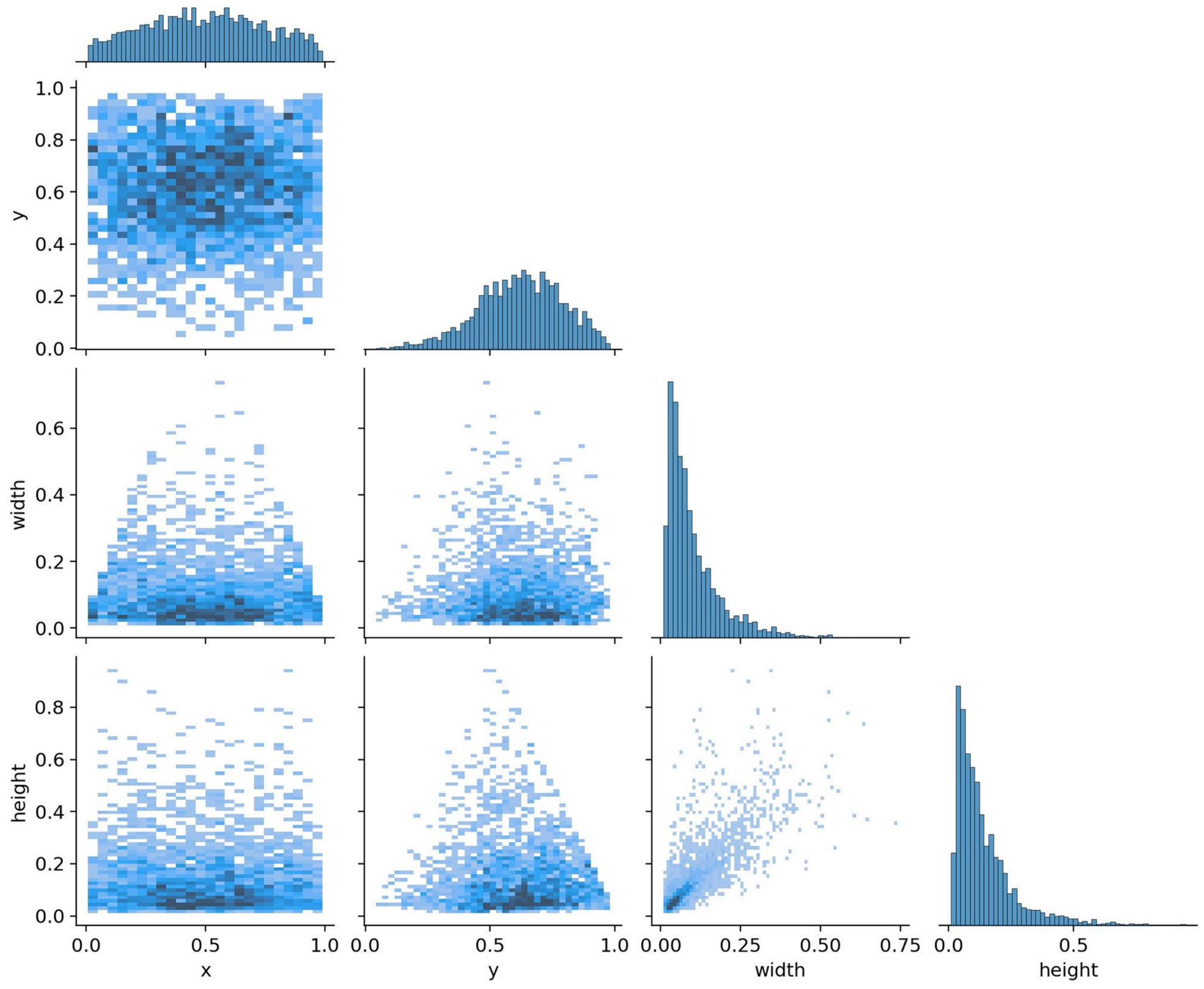

Fig. 17. Scatter plot of normalized bounding box width and height in the DAWN dataset, showing clustering around width 0.4–0.5 and height 0.5–0.6.

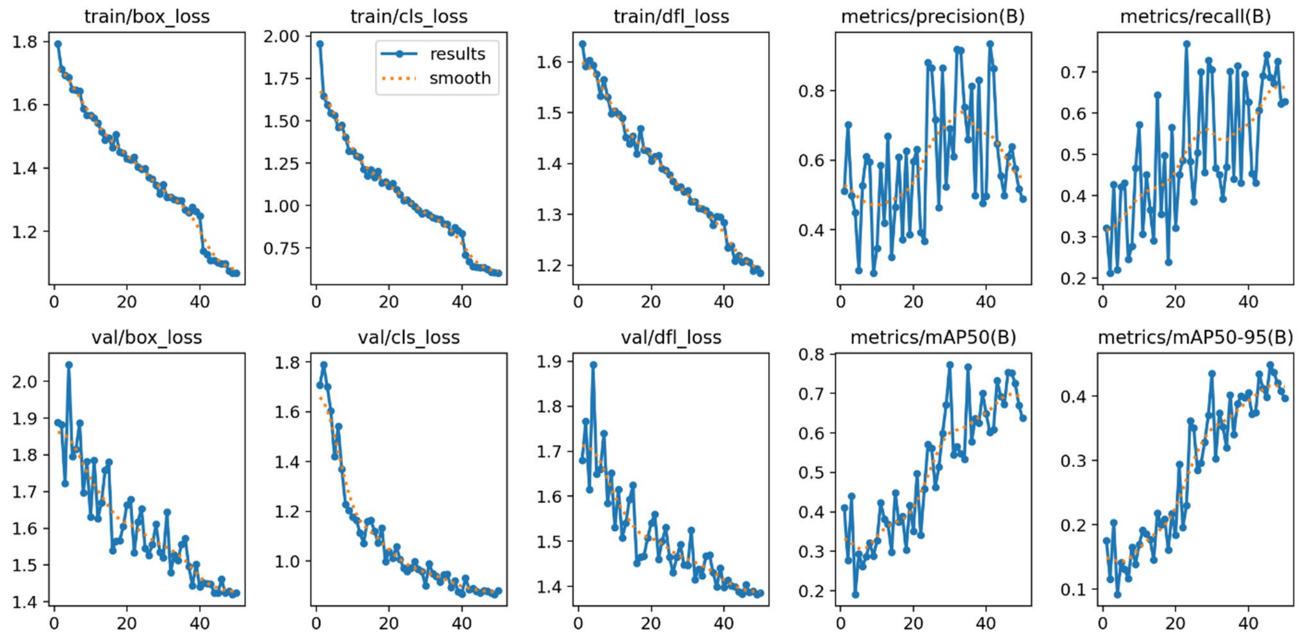

Fig. 18. Histograms of normalized bounding box width and height, confirming vertical elongation in object annotations.

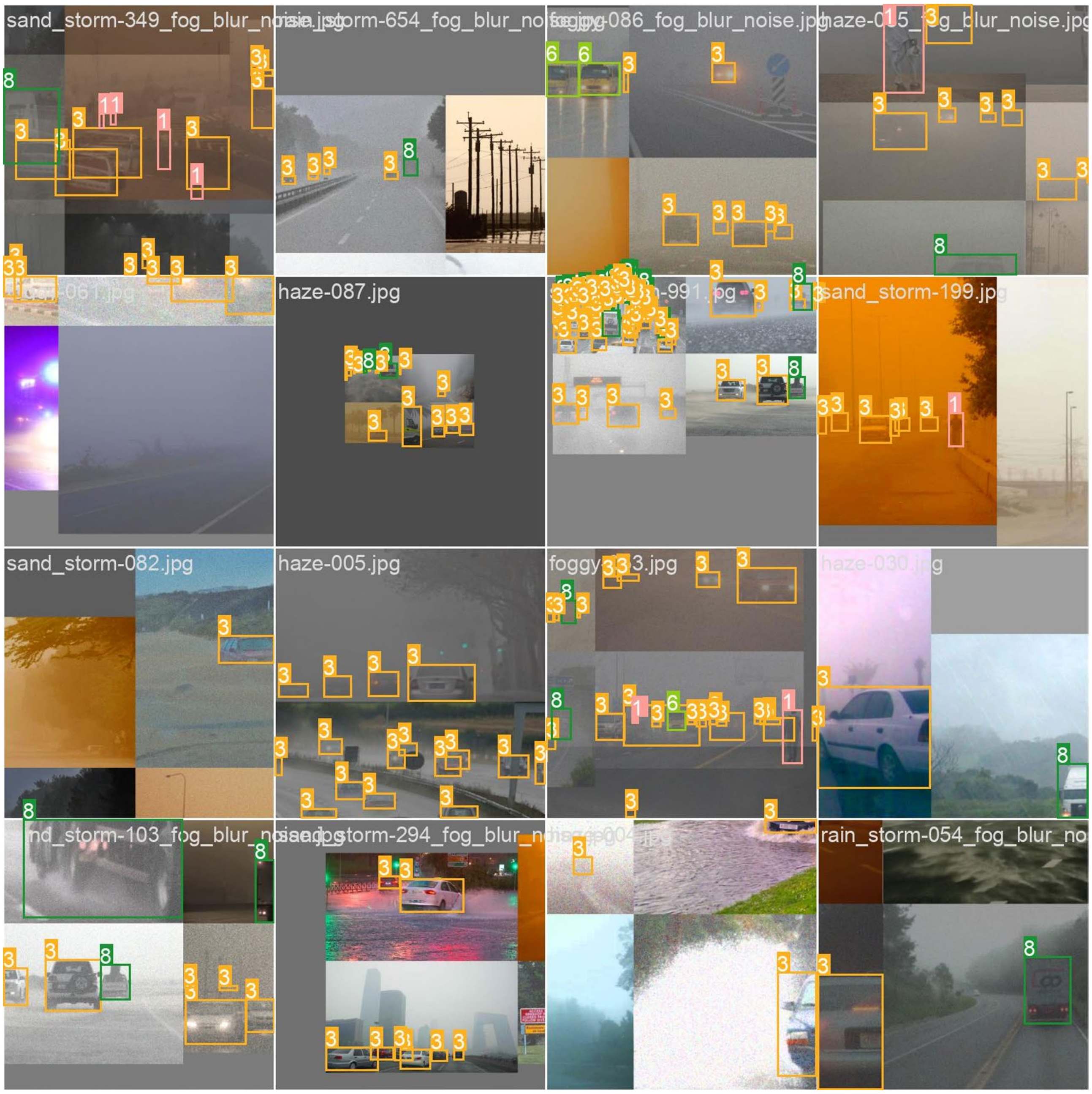

Fig. 19. Qualitative detection outputs under rainy conditions, comparing faster R-CNN and YOLOv5s.

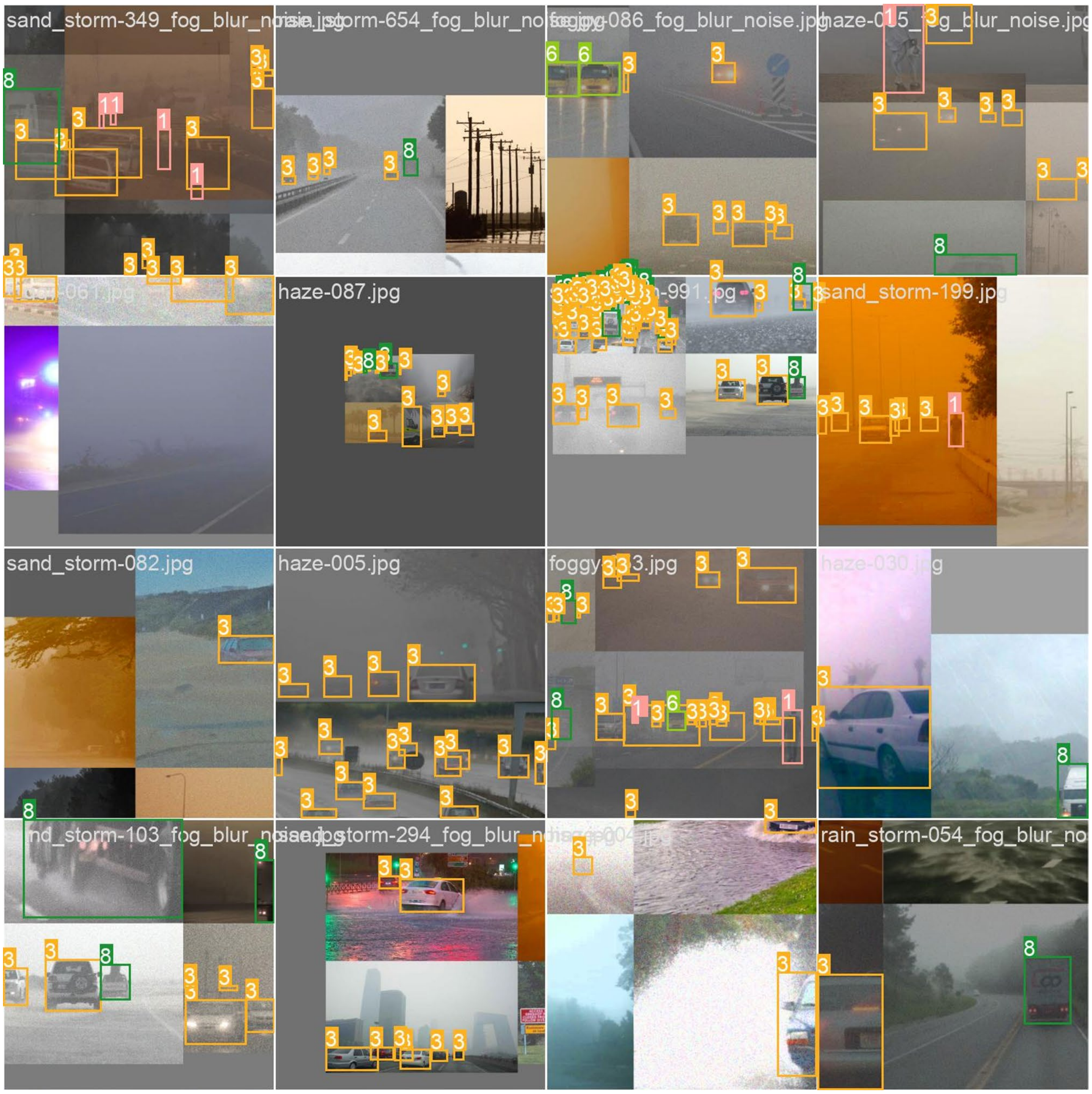
sand_storm-349_fog_blur_noise.jpg
sand_storm-654_fog_blur_noise.jpg
foggy-086_fog_blur_noise.jpg
haze-015_fog_blur_noise.jpg
haze-087.jpg
sand_storm-199.jpg
sand_storm-082.jpg
haze-005.jpg
haze-030.jpg
sand_storm-103_fog_blur_noise.jpg
sand_storm-294_fog_blur_noise.jpg
rain_storm-054_fog_blur_noise

Fig. 20. Qualitative detection outputs under snowy conditions across YOLOv5s, YOLOv8m, YOLOv10n, and Faster R-CNN.

Model	Params (M)	FLOPs (G)	Speed (ms/img)	Size (MB)
YOLOv5s	9.11	23.8	32.89	17.69
YOLOv8m	25.84	78.7	53.40	49.65
YOLOv10n	2.27	6.5	6.70	5.8

Table 5. Model size, computational cost (FLOPs), and inference speed for YOLO models.

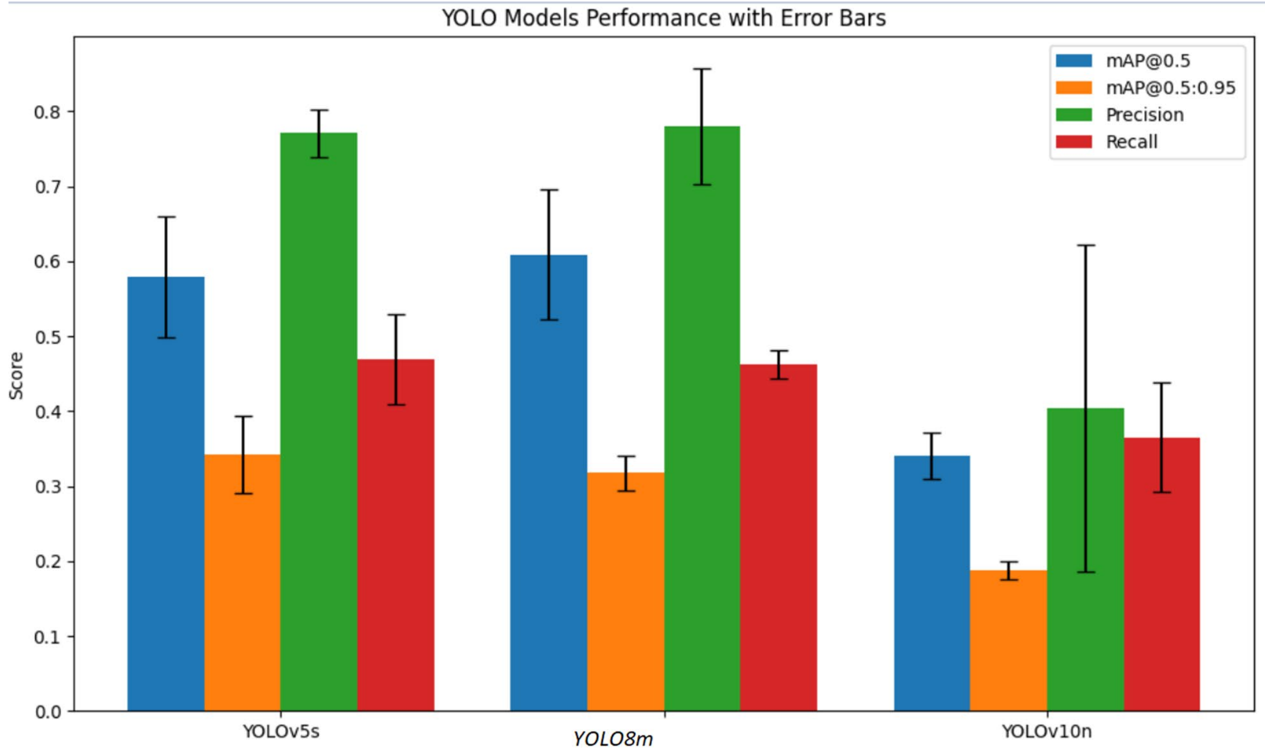

Fig. 21. Yolo8m models performance with error bars.

Model	mAP@0.5 (mean ± std)	mAP@0.5:0.95 (mean ± std)	Precision (mean ± std)	Recall (mean ± std)
YOLOv5s	0.579 ± 0.081	0.342 ± 0.051	0.771 ± 0.032	0.470 ± 0.060
YOLOv8m	0.609 ± 0.087	0.318 ± 0.023	0.780 ± 0.077	0.462 ± 0.019
YOLOv10n	0.340 ± 0.031	0.187 ± 0.012	0.404 ± 0.218	0.365 ± 0.073

Table 6. Statistical robustness analysis of YOLO models (mean ± standard deviation across three random seeds).

Data availability

The datasets generated or analysed during the current study are not publicly available, but are available from the corresponding author, Sina Samadi Gharehveran on reasonable request.

Received: 11 August 2025; Accepted: 12 November 2025
Published online: 29 December 2025

References

1. Sakaridis, C., Dai, D. & Van Gool, L. Semantic foggy scene Understanding with synthetic data. *Int. J. Comput. Vis.* **126**, 973–992. https://doi.org/10.1007/s11263-018-1072-8 (2018).
2. Michaelis, C. et al. Benchmarking robustness in object detection: Autonomous driving when the weather turns bad. In: *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 8827–8836. https://doi.org/10.48550/arXiv.1907.07484 (2019).
3. Kim, J., Lee, S. & Choi, Y. Robust object detection under adverse weather using real-world datasets. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, 950–958 (2021).
4. Nemati, R., Shirini, K. & Gharehveran, S. S. FER-HA: a hybrid attention model for facial emotion recognition. *J. Supercomput.* **81**, 1485. https://doi.org/10.1007/s11227-025-07983-4 (2025).
5. Sakaridis, C., Dai, D., Hecker, S. & Van Gool, L. Model adaptation with synthetic and real data for semantic dense foggy scene understanding. In *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 687–704 (2018).
6. Patel, V. S., Agrawal, K. & Nguyen, T. V. A comprehensive analysis of object detectors in adverse weather conditions. In: *58th Annual Conf. Inf. Sci. Syst. (CISS)*, 1–6. https://doi.org/10.1109/CISS59072.2024.10480197 (IEEE, 2024)
7. Rashid, K. I., Yang, C. & Huang, C. EPDPM-SinGAN: enhancing urban street semantic segmentation with region-wise GANs feature. *Expert Syst. Appl.* **285**, 128053 (2025).
8. Rashid, K. I., Yang, C. & Huang, C. Dynamic context-aware high-resolution network for semi-supervised semantic segmentation. *Eng. Appl. Artif. Intell.* **143**, 110068 (2025).
9. Michaelis, C. et al. Benchmarking robustness in object detection: Autonomous driving when the weather turns bad. In *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*. https://doi.org/10.48550/arXiv.1907.07484 (2019).
10. Sakaridis, C., Dai, D. & Van Gool, L. A. C. D. C. Adverse Conditions Dataset with Correspondence. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)* (2021).

11. Al-Haija, Q. A., Gharaibeh, M. & Odeh, A. Detection in adverse weather conditions for autonomous vehicles via deep learning. *AI* **3**, 303–317. https://doi.org/10.3390/ai3020019 (2022).
12. Caesar, H. et al. nuScenes: A multimodal dataset for autonomous driving. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)* (2020).
13. Yuan, Y. et al. CME-YOLO: A cross-modal enhanced YOLO algorithm for adverse weather object detection in autonomous driving. *Big Data Cogn. Comput.* **9**, 92. https://doi.org/10.3390/bdcc9040092 (2025).
14. Yu, F. et al. BDD100K: A diverse driving dataset for heterogeneous multitask learning. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)* (2020).
15. Halder, S. S., Lalonde, J. F. & de Charette, R. Physics-based rendering for improving robustness to rain. In *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 10203–10212 (2019).
16. Tan, J. et al. Innovative framework for fault detection and system resilience in hydropower operations using digital twins and deep learning. *Sci. Rep.* **15**, 15669. https://doi.org/10.1038/s41598-025-98235-1 (2025).
17. Li, D. et al. Co-path full-waveform lidar for detection of multiple along-path objects. *Opt. Lasers Eng.* **111**, 211–221. https://doi.org/10.1016/j.optlaseng.2018.08.009 (2018).
18. Saeedi, N. et al. Prediction of electrical energy consumption using principal component analysis and independent components analysis. *J. Supercomput.* **81**, 1072. https://doi.org/10.1007/s11227-025-07505-2 (2025).
19. Gharehveran, S. S., Shirini, K., Khavar, S. C. & Abdollahi, A. *Optimizing day-ahead Power Scheduling: a Novel MIQCP Approach for Enhanced SCUC with Renewable Integration* 101022 (e-Prime-Advances in Electrical Engineering, Electronics and Energy, 2025).
20. Chu, Z. & D-YOLO: A robust framework for object detection in adverse weather conditions. *ArXiv Preprint* (2024). arXiv:2403.09233.
21. Gao, J. et al. Pedestrian and vehicle detection in foggy weather using dehazeformer. *Vis. Comput.* 1–13. https://doi.org/10.1007/s00371-025-03882-0 (2025).
22. Chaudhry, R. SD-YOLO-AWDNet: A hybrid approach for smart object detection in challenging weather for self-driving cars. *Expert Syst. Appl.* **256**, 124942. https://doi.org/10.1016/j.eswa.2024.124942 (2024).
23. You, S. et al. Isls: an illumination-aware sauce-packet leakage segmentation method. *Sensors* **24**, 3216. https://doi.org/10.3390/s24103216 (2024).
24. Sattari, M. T., Shirini, K. & Javidan, S. Evaluating the efficiency of dimensionality reduction methods in improving the accuracy of water quality index modeling in Qizil-Uzen river using machine learning algorithms. *Water Soil. Manage. Model.* **4** (2), 89–104 (2024).
25. Shirini, K., Taherihajivand, A. & Samadi Gharehveran, S. A review of algorithms for solving the project scheduling problem with resource-constrained considering agricultural problems. *J. Agric. Mech.* **8** (1), 1–14 (2023).
26. IEEE DataPort. DAWN: Detection in Adverse Weather Nature Dataset. https://ieee-dataport.org
27. Mendeley Data. DAWN Dataset for Adverse Weather Object Detection. https://data.mendeley.com
28. Papers With Code. DAWN Dataset Benchmark. https://paperswithcode.com/dataset/dawn
29. Krizhevsky, A., Sutskever, I. & Hinton, G. E. ImageNet classification with deep convolutional neural networks. In *Proc. NeurIPS*, 1097–1105 (2012).
30. Li, D., Zhang, J. & Huang, K. Universal adversarial perturbations against object detection. *Pattern Recognit.* **110**, 107584. https://doi.org/10.1016/j.patcog.2020.107584 (2021).
31. Jocher, G. et al. YOLO by Ultralytics. GitHub Repository (2023). https://github.com/ultralytics/yolov5
32. Google Colab. Cloud-based Jupyter Notebook Platform. https://colab.research.google.com
33. Shen, X., Li, H., Li, Y. & Zhang, W. LDWLE: self-supervised driven low-light object detection framework. *Complex. Intell. Syst.* **11**, 82. https://doi.org/10.1007/s40747-024-01681-z (2025).
34. Liu, M., Liu, S., Su, H., Cao, K. & Zhu, J. Analyzing the noise robustness of deep neural networks. *IEEE Conf. Visual Anal. Sci. Technol. (VAST)*. 60–71. https://doi.org/10.1109/VAST.2018.8802509 (2018).
35. Molloy, D. et al. Analysis of the impact of lens blur on safety-critical automotive object detection. *IEEE Access.* **12**, 3554–3569. https://doi.org/10.1109/ACCESS.2023.3348663 (2024).
36. Hahner, M., Sakaridis, C., Dai, D. & Van Gool, L. Fog simulation on real LiDAR point clouds for 3D object detection in adverse weather. In: *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 15283–15292 (2021).
37. Bradski, G. The OpenCV library. *Dr Dobb's J. Softw. Tools.* **25**, 120–123 (2000).
38. Lin, T. Y. et al. Common objects in context. In *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 740–755 (Springer, 2014). https://doi.org/10.1007/978-3-319-10602-1_48
39. Everingham, M. et al. The Pascal visual object classes (VOC) challenge. *Int. J. Comput. Vis.* **88**, 303–338. https://doi.org/10.1007/s11263-009-0275-4 (2010).
40. COCO Consortium. pycocotools Python API. GitHub Repository. https://github.com/cocodataset/cocoapi
41. Buslaev, A. et al. Albumentations: fast and flexible image augmentations. *Information* **11**, 125. https://doi.org/10.3390/info11020125 (2020).
42. Google Colab. T4 GPU Specs. https://cloud.google.com/compute/docs/gpus
43. He, K., Gkioxari, G., Dollár, P. & Girshick, R. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, 2961–2969 (2017).
44. Sattari, M. T., Bagheri, R., Shirirni, K. & Allahverdipour, P. Modeling daily and monthly rainfall in Tabriz using ensemble learning models and decision tree regression. *Clim. Change Res.* **5** (18), 31–48. https://doi.org/10.30488/ccr.2024.433394.1192 (2024).
45. Taheri hajivand, A., Shirini, K. & Samadi Gharehveran, S. An overview of product performance prediction using artificial algorithms. *J. Agric. Mech.* **9** (3), 1–14. https://doi.org/10.22034/jam.2024.61899.1276 (2024).
46. Shirini, K., Hajivand, A. T. & Ghareveran, S. S. A novel deep learning-based method for potato leaf disease classification. *9th Adv. Eng. Days.* **9**, 462–464 (2024).
47. zahiri, M., shirini, K. & samadi gharehveran, S. Network traffic analysis with machine learning for faster detection of distributed denial of service attack. *J. Adv. DeF. Sci. Technol.* **14** (4), 273–282 (2024).
48. Zaki Dizaji, H., Shirini, K., Taheri hajivand, A. & monjezi, N. Modelling variables affecting the yield of sugarcane fields using deep recurrent neural network. *Iran. J. Biosyst. Eng.* **55** (2), 93–108. https://doi.org/10.22059/ijbse.2025.378958.665557 (2024).
49. Taheri hajivand, A., Shirini, K. & Samadi Gharehveran, S. Balancing time and cost in Resource-Constrained project scheduling using Meta-Heuristic approach. *J. Agric. Mach.* **14** (2), 215–234. https://doi.org/10.22067/jam.2023.81735.1157 (2024).
50. Samadi Gharehveran, S. & Shirini, K. *A Review of Resilient-oriented Operation Strategies of Electrical Energy Distribution Networks in Critical Situations* (Passive Defense, 2025).
51. Samadi Gharehveran, S., Shirini, K., Sarhangzadeh, M. & Gholinavaz, S. A review of game theory-based approaches for demand side management in smart energy grids. *Power Control Data Process. Syst.* e730417. https://doi.org/10.30511/pcdp.2025.2072667.1045 (2025).
52. Ahrari, M. et al. A security-constrained robust optimization for energy management of active distribution networks with presence of energy storage and demand flexibility. *J. Energy Storage.* **84**, 111024 (2024).
53. Gheibi, Y. et al. CNN-Res: deep learning framework for segmentation of acute ischemic stroke lesions on multimodal MRI images. *BMC Med. Inf. Decis. Mak.* **23**, 192. https://doi.org/10.1186/s12911-023-02289-y (2023).
54. Gharehveran, S. S. et al. Deep learning-based demand response for short-term operation of renewable-based microgrids. *J. Supercomput.* **80**, 26002–26035. https://doi.org/10.1007/s11227-024-06407-z (2024).

55. Shirini, K., Aghdasi, H. S. & Saeedvand, S. Multi-objective aircraft landing problem: a multi-population solution based on non-dominated sorting genetic algorithm-II. *J. Supercomput.* **80** (17), 25283–25314 (2024).

Author contributions
S.G.: Methodology, investigation, software, writing-original draft. N.S.: Investigation, data curation, software, writing-original draft. S.S.G.: Conceptualization, methodology, writing-reviewing and editing, supervision.

Funding
No funding was received for this work.

Declarations

Competing interests
The authors declare no competing interests.

Ethical declarations
The authors declare that they have no competing interests to declare that are relevant to the content of this article. This article does not contain any studies with human participants or animals performed by any of the authors.

Additional information
Correspondence and requests for materials should be addressed to S.S.G.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.