

Resource-Aware Small-UAV Perception under Annotation and Compute Constraints

Xufeng Zhou

Information Support Force Engineering University

Jun Li

PLA Unit 31648

Hua Hu

PLA Army Special Operations Academy

Yan Liu

PLA Army Special Operations Academy

Xiaodong Wang

PLA Army Special Operations Academy

Kebin Chen

Information Support Force Engineering University

Chao Jiang

Information Support Force Engineering University

Quantong Lv

Information Support Force Engineering University

Xuping Jiang

jiangxuping_jxp@163.com

Information Support Force Engineering University

Article

Keywords: small UAV detection, resource-aware perception, adaptive inference, hard-example replay, limited annotation, edge computing

Posted Date: September 15th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10704994/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.
[Read Full License](#)

Additional Declarations: No competing interests reported.

Resource-Aware Small-UAV Perception under Annotation and Compute Constraints

Xufeng Zhou^{1,3}, Jun Li², Hua Hu³, Yan Liu³, Xiaodong Wang³
Kebin Chen¹, Chao Jiang¹, Quantong Lv¹ and Xuping Jiang^{1,*}

¹Information Support Force Engineering University, Wuhan 430035, China

²PLA Unit 31648, Nanning 530221, China

³PLA Army Special Operations Academy, Guilin 541000, China

*Correspondence: Xuping Jiang, jiangxuping_jxp@163.com

Abstract

Small unmanned aerial vehicle detection is constrained by two resources that are often studied separately: computation at inference time and annotated images during adaptation. We evaluated a resource-aware perception strategy on the visible-light detection subset of the DUT-Anti-UAV benchmark without additional UAV-domain pretraining. The study first established detector and input-resolution baselines, then tested confidence-guided routing from YOLOv8n to YOLOv8m and difficulty-ranked replay under reduced annotation budgets. At input size 640, YOLOv8s achieved the highest observed mAP50 of 0.9052. On the 416-input test protocol, validation-selected routing sent 35.68% of images from YOLOv8n to YOLOv8m and increased F1 from 0.8132 to 0.8446. At the same route fraction, repeated random-routing assignments produced an average F1 of 0.8237, indicating that the gain was associated with confidence-based sample selection rather than routing volume alone. The resulting 16.00 FPS is reported as an analytical cascade throughput estimate derived from measured standalone CPU timings. Under the 50% label setting, difficulty-ranked replay produced a moderate, label-dependent benefit over compute-matched uniform replay, increasing mAP50 from 0.7575 to 0.7621 and tiny-target recall from 0.5730 to 0.5861. The results support detector-first resource allocation under annotation and compute constraints.

Keywords: small UAV detection; resource-aware perception; adaptive inference; hard-example replay; limited annotation; edge computing

1 Introduction

Vision-based detection of small unmanned aerial vehicles is important for low-altitude monitoring, inspection, and safety management. Cameras are attractive because they are inexpensive and provide direct visual evidence. The sensing problem is nevertheless difficult. A distant UAV may occupy only a few pixels, while sky, buildings, vegetation, water, haze, compression artifacts, and illumination changes create visually similar background patterns. These conditions make detector choice and operating-point selection as important as the nominal model architecture.

The practical constraints are not limited to visual ambiguity. A monitoring system may need to process images on a CPU or a compact edge computer, where running a large detector on every frame is costly. At the same time, a new UAV sensing task may begin with only a fraction of the annotations required for full supervised training. This setting is a domain-specific cold start: models are initialized from generic public pretrained weights, but no additional UAV-domain pretraining or external UAV training data are used. A useful evaluation must therefore consider both inference computation and annotation availability.

Existing UAV detection studies and general object detection benchmarks provide strong model families and datasets [1, 2, 3, 4, 5, 6, 7, 8]. The YOLO family offers a practical range of model scales [9, 10, 11, 12, 13, 14, 15, 16, 17]. However, rankings obtained on general benchmarks do not determine the best configuration for tiny UAVs. Input resolution can change the amount of target evidence that survives resizing, and multi-scale feature processing and small-object augmentation can also affect the result [18, 19].

Adaptive computation provides a relevant conceptual background for this setting. Conditional inference, dynamic routing, cascaded models, and early-exit networks reduce average computation by allocating deeper processing only to selected inputs [20, 21, 22, 23]. Confidence-based model selection is therefore a practical operating rule rather than a claim of a new routing primitive. A related line of work studies hard-example learning through online hard-example mining, difficulty-aware sampling, curriculum learning, few-annotation learning, annotation-efficient detection, and small-object augmentation [24, 25, 26, 27, 28]. These studies motivate the controls used here: a routing comparison must separate sample selection from route volume, and a replay comparison must separate sample ranking from additional optimization and duplicated training exposure.

This study asks how a small-UAV detector can use limited computation and limited annotation more deliberately. We address the question through a controlled evaluation with two primary method components. First, confidence-guided detector routing uses the confidence of a lightweight detector to route uncertain images to a larger verifier. Second, difficulty-ranked replay ranks labeled images using missed objects, localization error, low matched confidence, tiny-target exposure, and false-positive pressure. Plain continuation and compute-matched uniform replay provide controls for additional training and replay volume. Resolution, detector scale, robustness, object size, operating thresholds, random seeds, and bootstrap uncertainty are treated as supporting analyses rather than separate contributions.

The study makes three contributions. First, it provides confidence-guided adaptive computation with a matched random-routing control at the same routed fraction. Second, it evaluates difficulty-ranked hard-example replay with plain-continuation and compute-matched uniform controls, with the clearest aggregate result at the 50% label setting. Third, it supplies a resource-aware evaluation that integrates detector scale, input resolution, annotation budget, target size, robustness, and statistical uncertainty. An exploratory multimodal diagnostic is retained to examine the applicability boundary of detector-conditioned candidate review; it is not a primary method contribution and does not determine detector training or operating points. The overall evaluation structure is summarized in Figure 1.

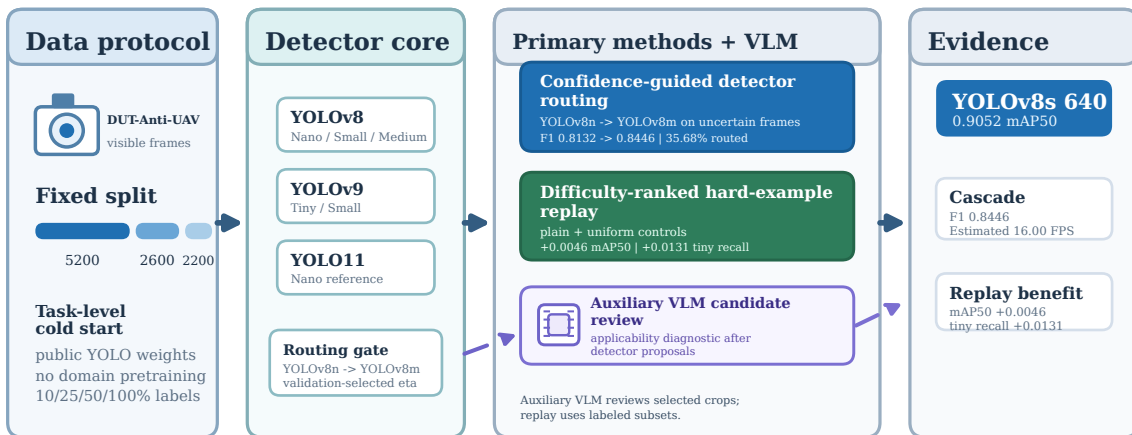


Figure 1: Resource-aware evaluation framework. The detector and replay branches provide the primary evidence. The small auxiliary VLM branch is restricted to an applicability-boundary diagnostic after detector proposals.

2 Results

2.1 Detector and resolution foundations

We first compared representative detector scales and recent YOLO families at input size 416. The models were trained and evaluated under the same *DUT-Anti-UAV* split and task definition. The three-seed YOLOv8n reference obtained a mean mAP50 of 0.8023 with a standard deviation of 0.0013. Increasing the YOLOv8 scale improved mAP50 to 0.8391 for YOLOv8s and 0.8465 for YOLOv8m. YOLOv9s reached 0.8176, whereas YOLO11n reached 0.7928 under the same protocol (Figure 2 and Table 1).

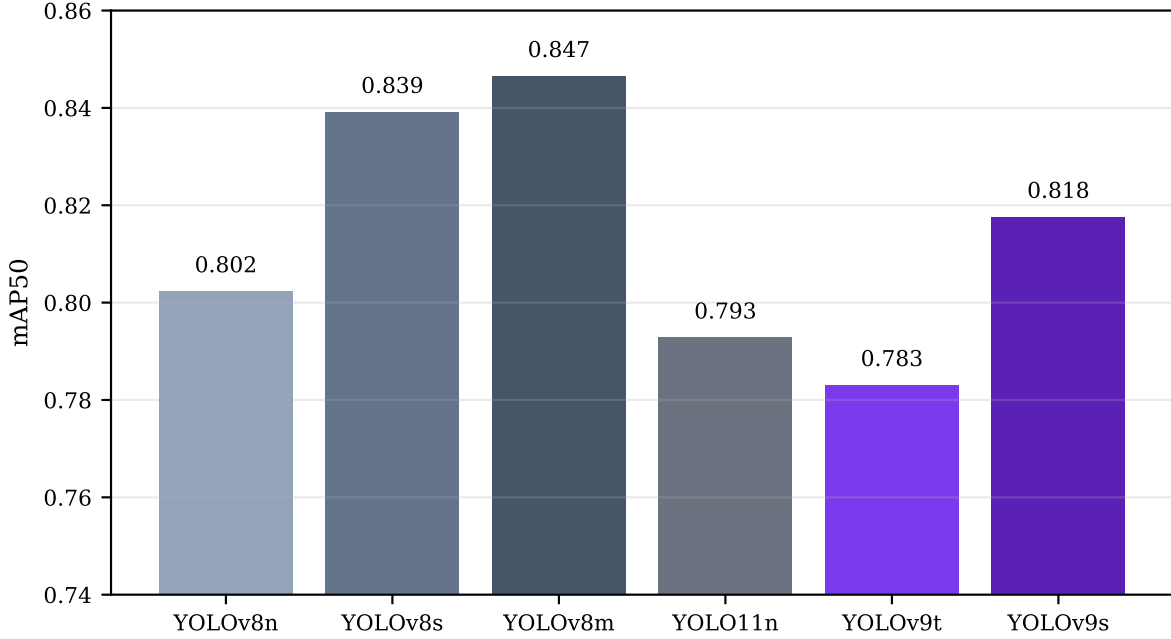


Figure 2: Representative detector comparison at input size 416. YOLOv8s and YOLOv8m provided the strongest results among the evaluated configurations.

Table 1: Main *DUT-Anti-UAV* test results at input size 416. YOLOv8n values are mean $\pm$ SD across seeds 42, 123, and 2024; the other rows are designated seed-42 runs.

Model	mAP50	mAP50-95	Precision	Recall
YOLOv8n, three seeds	0.8023 $\pm$ 0.0013	0.4557 $\pm$ 0.0021	0.9060 $\pm$ 0.0073	0.7388 $\pm$ 0.0080
YOLOv8s	0.8391	0.4857	0.9047	0.7779
YOLOv8m	0.8465	0.4911	0.9138	0.7817
YOLO11n	0.7928	0.4450	0.8951	0.7318
YOLOv9s	0.8176	0.4814	0.9075	0.7518
YOLOv9t	0.7830	0.4513	0.8914	0.7265

Input resolution had a larger effect than changing detector family at a fixed low resolution. Raising the input size from 416 to 640 increased mAP50 from 0.8040 to 0.8875 for YOLOv8n, from 0.8391 to 0.9052 for YOLOv8s, from 0.8465 to 0.9025 for YOLOv8m, and from 0.7928 to 0.8890 for YOLO11n (Figure 3 and Table 2). YOLOv8s at 640 was the best observed single-detector configuration. It reached 0.9052 mAP50 and 0.8543 recall at 8.71 FPS under the reported CPU timing protocol.

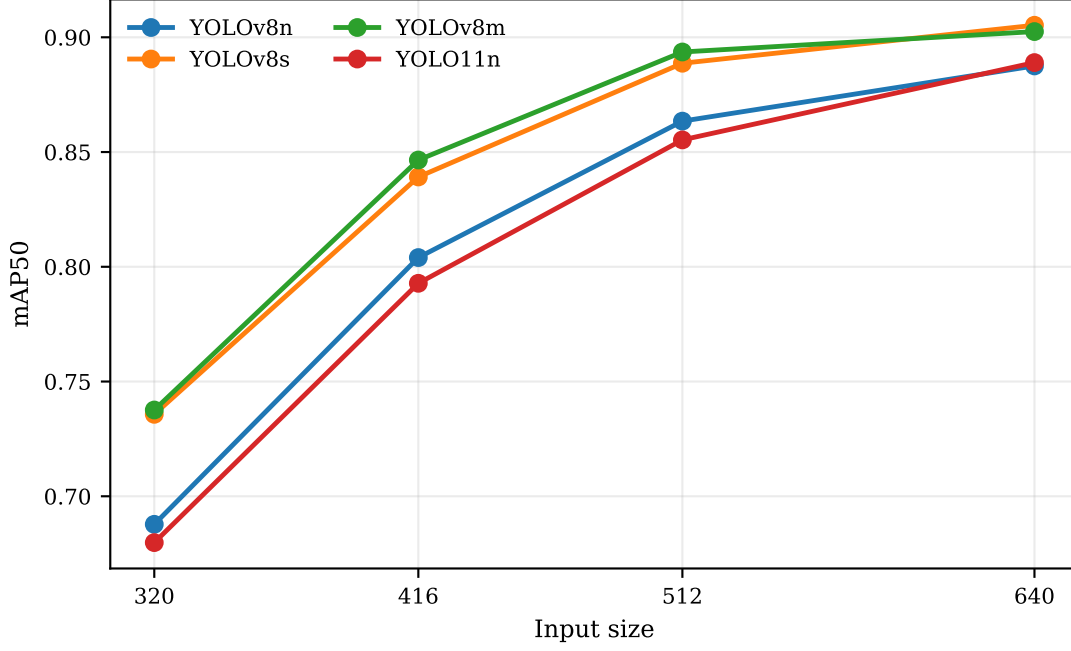


Figure 3: Input-resolution sensitivity on the *DUT-Anti-UAV* test split. Increasing resolution consistently improved mAP50 for the evaluated detector configurations.

Table 2: Resolution sensitivity on the *DUT-Anti-UAV* test split. FPS values are standalone CPU timings measured with batch size 1, including preprocessing, model inference, and postprocessing but excluding image loading and visualization. The resolution sweep uses the seed-42 YOLOv8n run, whereas Table 1 reports the three-seed mean.

Model	Input	mAP50	mAP50-95	Recall	FPS
YOLOv8n	320	0.6878	0.3727	0.6316	74.93
YOLOv8n	416	0.8040	0.4527	0.7477	53.60
YOLOv8n	640	0.8875	0.5452	0.8236	25.80
YOLOv8s	416	0.8391	0.4857	0.7779	18.50
YOLOv8s	640	0.9052	0.5647	0.8543	8.71
YOLOv8m	416	0.8465	0.4911	0.7817	8.14
YOLOv8m	640	0.9025	0.5575	0.8383	4.06
YOLO11n	416	0.7928	0.4450	0.7318	60.97
YOLO11n	640	0.8890	0.5424	0.8187	23.25

2.2 Confidence-guided adaptive computation

For image j , let q_j denote the maximum retained confidence produced by the first-pass YOLOv8n detector. The validation-selected routing threshold η determines whether an image is sent to the larger verifier: images with $q_j < \eta$ are routed to YOLOv8m. The resulting route fraction α is the proportion of test images sent to YOLOv8m. In the primary experiment, $\eta = 0.60$ and $\alpha = 35.68\%$. Relative to YOLOv8n alone, the cascade increased recall from 0.7318 to 0.7893 and F1 from 0.8132 to 0.8446. The cascade F1 was also slightly higher than YOLOv8m alone, while its analytical throughput was approximately twice that of running YOLOv8m on every image (Table 3).

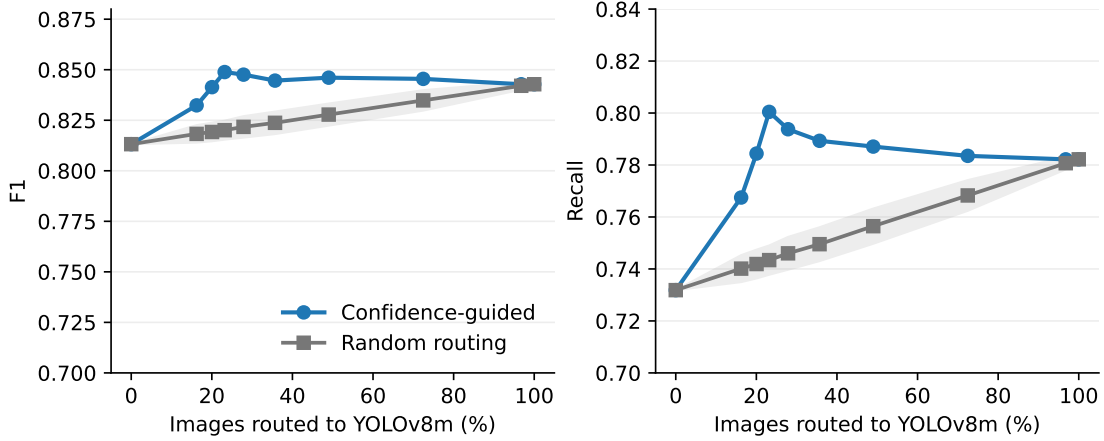


Figure 4: Confidence-guided routing compared with repeated random routing at matched routed fractions. The shaded band denotes the empirical 2.5th–97.5th percentile range across the random-routing assignments.

To test whether the gain came from selecting uncertain images rather than from the number of verifier calls, we compared confidence-guided routing with repeated random assignments at matched route fractions. At the selected 35.68% route rate, random routing achieved an average F1 of 0.8237 ± 0.0028 and recall of 0.7495 ± 0.0033 . Confidence-guided routing reached F1 0.8446 and recall 0.7893. The corresponding differences were 0.0209 in F1 and 0.0398 in recall. The confidence-guided curve remained above the random-routing mean across the intermediate route rates in Figure 4. The shaded regions show empirical 2.5th–97.5th percentiles across the repeated assignments, so they are repeated-assignment ranges rather than parametric confidence intervals.

Table 3: Adaptive routing at the validation-selected route rate. Random routing used the same number of routed images and repeated assignments.

Method	Route rate	Recall	F1	FP/image	Throughput
YOLOv8n	0	0.7318	0.8132	0.0695	53.60 FPS
YOLOv8m	0	0.7822	0.8428	0.0755	8.14 FPS
Random routing	35.68%	0.7495 ± 0.0033	0.8237 ± 0.0028	0.0718 ± 0.0035	16.00 FPS ^a
Confidence-guided routing	35.68%	0.7893	0.8446	0.0814	16.00 FPS ^a

^aAnalytical estimate based on measured standalone CPU latencies and the observed route rate, not an end-to-end hardware deployment measurement.

The benefit of routing was accompanied by a modest increase in false positives per image relative to YOLOv8n. This tradeoff is expected when the routing policy prioritizes uncertain images for recall recovery. The route threshold was selected on validation data and then fixed before test evaluation, so the reported test result does not use test-set threshold tuning.

2.3 Learning with limited annotation

The reduced-label experiment used nested training subsets containing 10%, 25%, and 50% of the available training images. The validation and test splits remained unchanged. YOLOv8n obtained 0.6242 mAP50 with 10% labels, 0.7094 with 25%, and 0.7337 with 50%. The full 100% reference reached 0.8040 mAP50 (Figure 5 and Table 4). These values describe practical low-budget training behavior because the reduced-label runs used a 20-epoch schedule, whereas the full reference used 30 epochs. They should not be interpreted as a strict annotation-only causal comparison.

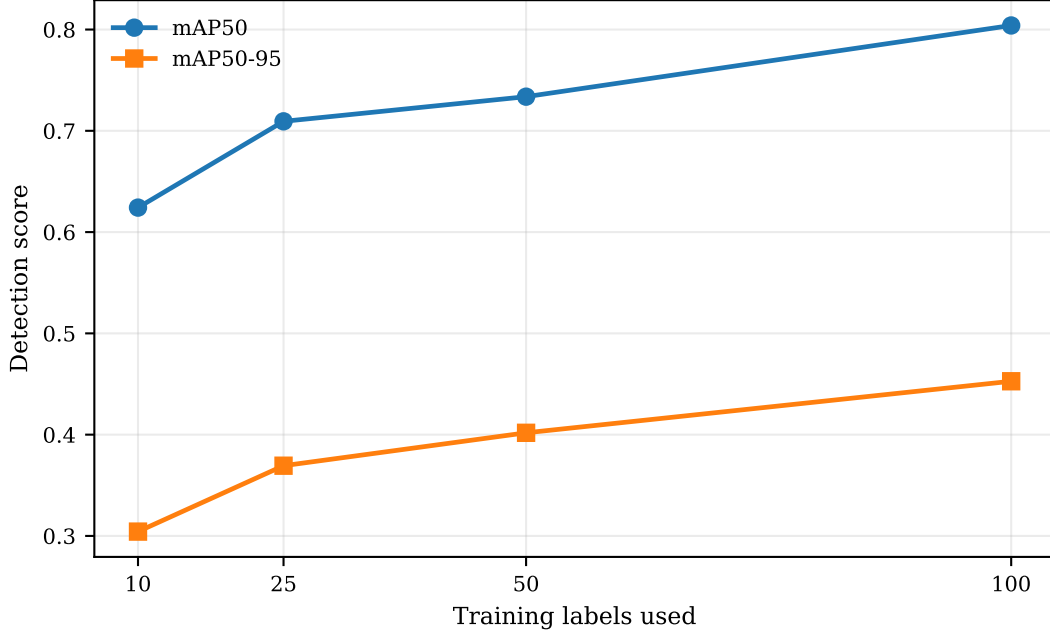


Figure 5: Test performance of YOLOv8n under reduced annotation budgets. The reduced-label runs use a 20-epoch schedule and the full-data reference uses 30 epochs.

Table 4: Practical low-budget training behavior of YOLOv8n on *DUT-Anti-UAV*.

Training images	Epochs	mAP50	mAP50-95	Recall
10%	20	0.6242	0.3043	0.5728
25%	20	0.7094	0.3693	0.6414
50%	20	0.7337	0.4018	0.6635
100%	30	0.8040	0.4527	0.7477

2.4 Difficulty-ranked replay

Difficulty-ranked replay was evaluated after the reduced-label supervised checkpoints. The 50% setting included three controls: the original supervised checkpoint, plain continuation with one copy per source image, and compute-matched uniform replay. The difficulty-ranked replay used the same source images, copy-count histogram, number of replay samples, optimizer, learning rate, batch size, and five-epoch schedule as uniform replay. Only the association between image difficulty and replay multiplicity differed.

Plain continuation increased mAP50 from 0.7337 to 0.7538, showing that additional low-rate training accounted for much of the gain. Uniform replay reached 0.7575 mAP50 and 0.7738 F1. Difficulty-ranked replay reached 0.7621 mAP50 and 0.7797 F1. Relative to uniform replay, the difficulty-ranked strategy

added 0.0046 mAP50 and 0.0059 F1. Its tiny-target recall increased from 0.5730 to 0.5861, while aggregate recall was slightly lower than uniform replay, 0.6962 versus 0.7029 (Table 5).

Table 5: Low-label post-training results for YOLOv8n at input size 416. Uniform and difficulty-ranked replay are compute-matched.

Label ratio	Stage	mAP50	mAP50-95	Precision	Recall	F1
10%	SFT	0.6242	0.3043	0.8079	0.5728	0.6704
10%	Uniform replay	0.6315	0.3146	0.8066	0.5721	0.6694
10%	Difficulty-ranked replay	0.6289	0.3095	0.8232	0.5726	0.6754
25%	SFT	0.7094	0.3693	0.8671	0.6414	0.7374
25%	Uniform replay	0.7249	0.3791	0.8605	0.6601	0.7471
25%	Difficulty-ranked replay	0.7239	0.3793	0.8664	0.6641	0.7519
50%	SFT	0.7337	0.4018	0.8848	0.6635	0.7583
50%	Plain continuation	0.7538	0.4124	0.8647	0.6931	0.7695
50%	Uniform replay	0.7575	0.4131	0.8606	0.7029	0.7738
50%	Difficulty-ranked replay	0.7621	0.4138	0.8859	0.6962	0.7797

The difficulty-ranked advantage was not stable in mAP50 at 10% or 25% labels. At those budgets, the strategy produced small F1 or tiny-recall shifts with accompanying false-positive changes. The evidence therefore supports a moderate, label-dependent benefit rather than a universal improvement. The difficulty weights were heuristic and recall-oriented. They were chosen to prioritize missed detections and small targets, not obtained by a complete weight-search procedure.

2.5 Target size, robustness, and statistical reliability

Object-size analysis showed that model scaling mainly benefited the smallest targets. At confidence 0.25 and IoU 0.5, tiny-target recall increased from 0.6089 for YOLOv8n to 0.6651 for YOLOv8s and 0.7010 for YOLOv8m. Small-target recall increased from 0.7383 to 0.8012 and 0.8215, respectively (Figure 6 and Table 6).

Table 6: Object-size-stratified recall on the *DUT-Anti-UAV* test split.

Model	Tiny	Small	Medium	Large
YOLOv8n	0.6089	0.7383	0.8797	0.9367
YOLOv8s	0.6651	0.8012	0.8956	0.9467
YOLOv8m	0.7010	0.8215	0.8418	0.9333

Representative qualitative detections are shown in Fig. 7. The examples cover marine, water, open-sky, building, vegetation, tiny-target, low-light, and dim-background scenes.

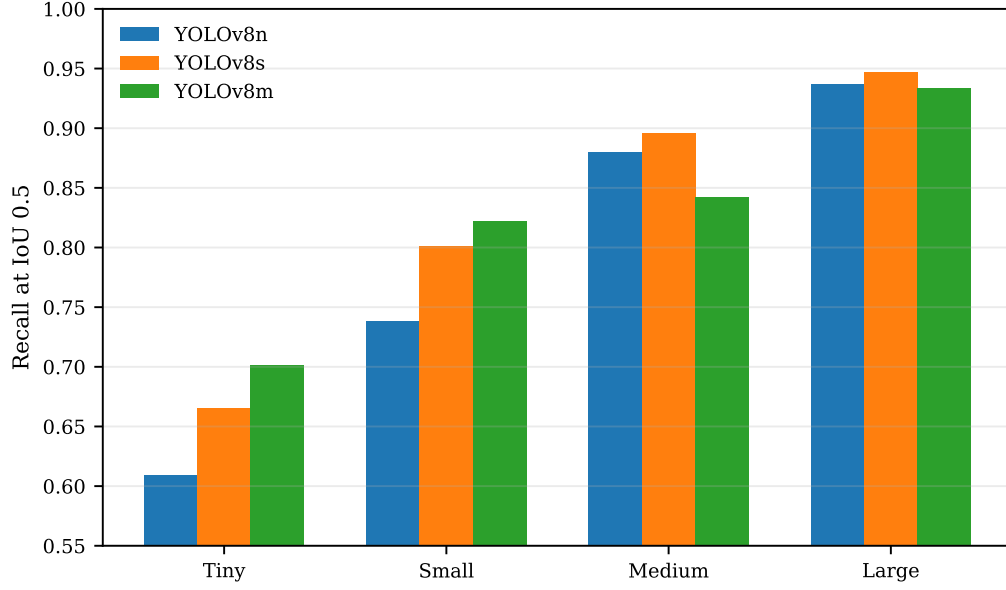


Figure 6: Object-size-stratified recall at IoU 0.5 and confidence 0.25. Model scaling provided the largest gains for tiny and small UAVs.

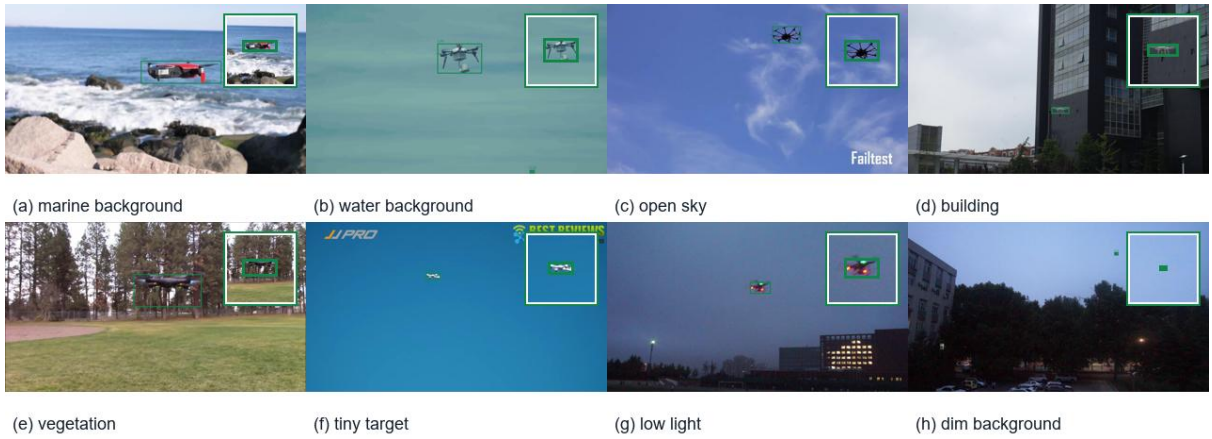


Figure 7: Representative qualitative detections of YOLOv8s at 640 input size on the *DUT-Anti-UAV* test split. The examples cover marine, water, open-sky, building, vegetation, tiny-target, low-light, and dim-background scenes. Each inset enlarges the detected UAV region.

The robustness analysis applied eight image perturbations to the test images, following common corruption evaluation practice [29, 30]. Gaussian noise and overexposure caused the largest mAP50 reductions. YOLOv8s and YOLOv8m retained mean corrupted mAP50 values of 0.8096 and 0.8174, respectively, with mean retention close to 96.5% relative to clean images (Figure 8 and Table 7).

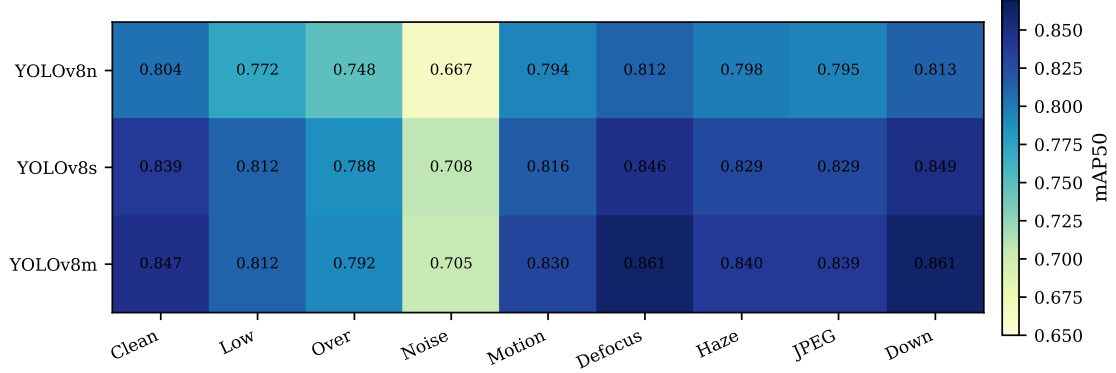


Figure 8: Robustness to synthetic image corruptions. Gaussian noise and overexposure were the most challenging tested conditions.

Table 7: Robustness summary under eight synthetic image corruptions.

Model	Clean mAP50	Mean corrupted mAP50	Mean retention	Worst retention
YOLOv8n	0.8040	0.7747	96.36%	82.91%
YOLOv8s	0.8391	0.8096	96.48%	84.36%
YOLOv8m	0.8465	0.8174	96.56%	83.25%

Validation-selected confidence thresholds were applied once to the held-out test split. Confidence thresholds are operational choices because they determine the balance between missed detections and false alarms [31]. At the selected thresholds, YOLOv8n, YOLOv8s, and YOLOv8m obtained F1 values of 0.8132, 0.8343, and 0.8428, respectively. For a separate model-level uncertainty comparison, image-level bootstrap intervals were computed at a common confidence threshold of 0.25, rather than at each model’s validation-selected deployment threshold. These intervals gave mean F1 values of 0.8050 for YOLOv8n, 0.8197 for YOLOv8s, 0.8309 for YOLOv8m, and 0.7887 for YOLO11n (Figure 9 and Table 8). The intervals were close enough to motivate reporting the detector choice as a practical tradeoff rather than as a claim of statistically separated model families.

Table 8: Bootstrap confidence intervals for model-level uncertainty comparison at a common confidence threshold of 0.25. This threshold was not the validation-selected deployment operating point.

Model	Precision	Recall	F1
YOLOv8n	0.8519 [0.8368, 0.8682]	0.7630 [0.7462, 0.7793]	0.8050 [0.7910, 0.8185]
YOLOv8s	0.8375 [0.8217, 0.8547]	0.8027 [0.7863, 0.8188]	0.8197 [0.8065, 0.8343]
YOLOv8m	0.8533 [0.8369, 0.8700]	0.8097 [0.7926, 0.8250]	0.8309 [0.8169, 0.8446]
YOLO11n	0.8405 [0.8248, 0.8567]	0.7430 [0.7248, 0.7603]	0.7887 [0.7740, 0.8032]

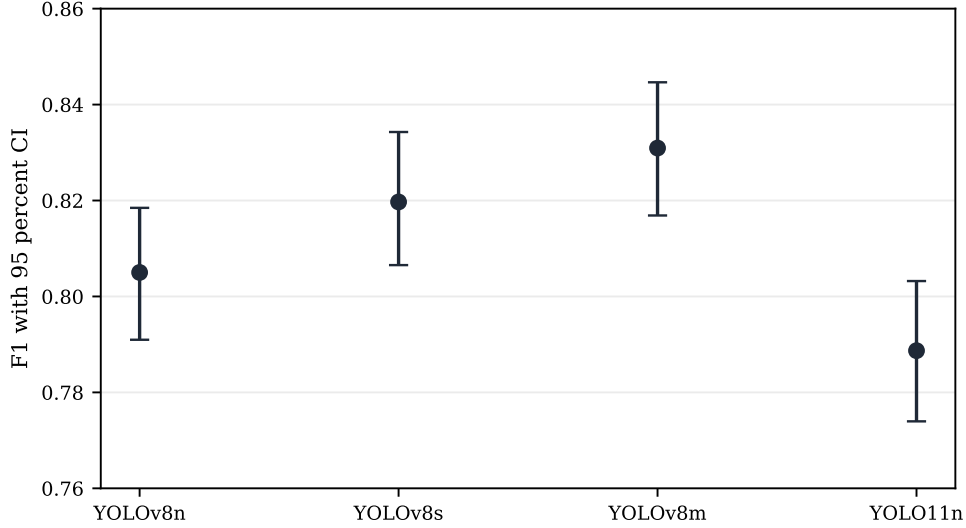


Figure 9: Image-level bootstrap 95% confidence intervals for model-level uncertainty comparison at a common confidence threshold of 0.25. This threshold was not the validation-selected deployment operating point.

2.6 VLM applicability boundary

The VLM branch was evaluated only as a small, controlled applicability-boundary diagnostic. Whole-image localization was unreliable, whereas detector-guided proposal crops produced limited nonzero grounding. The sample was selected for interface testing rather than detector benchmarking. VLM outputs were not used for detector training, threshold selection, replay ranking, or the primary metrics.

3 Discussion

The central result is that resource constraints should be treated as part of the perception problem. For this dataset, preserving spatial detail had a larger effect on mAP50 than changing among several detector families at input size 416. YOLOv8s at 640 provided the highest observed mAP50, while YOLOv8n remained useful as a lightweight first-pass model. The result does not imply that one detector scale is universally optimal. It indicates that input resolution and model capacity should be selected jointly with the target-size distribution and available computation.

The routing experiment provides the clearest evidence for adaptive computation. Confidence-guided routing outperformed random routing at the same route fraction, especially in recall. This pattern is consistent with the intended role of the confidence score as a difficulty signal. It also clarifies the meaning of the cascade result: the improvement did not arise merely because a larger detector was called on more images. The throughput value remains an analytical estimate based on standalone component timings. A deployment implementation would additionally need to measure device-specific scheduling, memory transfer, batching, and thermal behavior.

The replay results show why low-label adaptation should be separated into training volume and sample selection. Plain continuation explained much of the improvement over the original 50% checkpoint. Uniform replay then provided a further gain through matched additional optimization. Difficulty-ranked replay added a smaller but targeted improvement in aggregate F1 and tiny-target recall. Its advantage was not stable in mAP50 at 10% or 25% labels. One plausible interpretation is that difficulty ranking becomes more useful after the detector has learned a sufficiently stable UAV representation. At very low label budgets, the detector’s own errors may be too noisy for reliable ranking. This interpretation is consistent

with the results but was not directly tested as a causal mechanism.

The VLM diagnostic defines an applicability boundary rather than a new performance source. Whole-image localization was unreliable, while detector-guided crops produced limited nonzero grounding in a small controlled sample. A detector-first workflow is therefore more defensible for this setting. The detector supplies localized candidates, while a VLM may provide optional semantic review or human-in-the-loop triage.

The combined evidence supports a two-level resource allocation view. Detector configuration and resolution determine the quality of the base representation. Confidence-guided routing allocates inference computation across easy and uncertain images. Difficulty-ranked replay allocates limited training effort across labeled samples. These mechanisms address different resources and can be evaluated independently, which makes the framework easier to reproduce and adapt to other small-object sensing tasks.

4 Conclusion

This study evaluated small-UAV target detection when both inference computation and annotated images are limited. The results showed that input resolution and detector scale should be considered jointly, while a lightweight YOLOv8n first pass can route uncertain images to YOLOv8m without applying the larger detector to every image. At the validation-selected 35.68% route rate, confidence-guided routing achieved an F1 of 0.8446, compared with 0.8237 for matched random routing. Under the 50% label setting, difficulty-ranked replay produced a moderate additional gain over compute-matched uniform replay, including higher tiny-target recall.

The findings support a detector-first resource allocation strategy for the tested visible-light UAV task. Confidence-guided routing addresses inference cost, and difficulty-ranked replay addresses limited annotation during adaptation. These conclusions are bounded by the single dataset, recorded-image evaluation, analytical cascade timing, and limited multimodal diagnostic sample.

5 Methods

5.1 Dataset and task definition

All experiments used the detection subset of *DUT-Anti-UAV*, which contains 10,000 annotated visible-light images split into 5200 training images, 2600 validation images, and 2200 test images [1]. The task was formulated as single-class UAV target detection. Image annotations were converted to YOLO format. The validation and test splits were kept unchanged across all training-budget and replay experiments.

The term domain-specific cold start refers to initialization from generic public pretrained weights without additional UAV-domain pretraining or external UAV training data. The main runs used public YOLO weights and fine-tuned only on the designated *DUT-Anti-UAV* training split. The main YOLOv8n training schedule used 30 epochs, input size 416, batch size 8, the AdamW optimizer selected by the Ultralytics training pipeline, and random seed 42. YOLOv8s and YOLOv8m used physical batch sizes 4 and 2, respectively, because of memory requirements. No gradient accumulation or nominal-batch correction was used to equalize these settings. The detector-scale comparison should therefore be interpreted as a practical configuration comparison rather than a strict batch-controlled architectural ablation. Additional seeds 123 and 2024 were used for the YOLOv8n stability analysis.

5.2 Detector training and evaluation

The detector comparison included YOLOv8n, YOLOv8s, and YOLOv8m. Additional comparisons used YOLOv9t, YOLOv9s, and YOLO11n. YOLOv8n and YOLOv8s were evaluated at 320, 416, 512, and

640 pixels, whereas YOLOv8m and YOLO11n were evaluated at 416 and 640 pixels. The primary metrics were mAP50, mAP50-95, precision, and recall. F1, false positives per image, and miss rate were calculated at confidence thresholds selected from the validation split. A generic YOLO detection objective can be written as

$$\mathcal{L}(\theta) = \lambda_{\text{box}} \mathcal{L}_{\text{box}} + \lambda_{\text{cls}} \mathcal{L}_{\text{cls}} + \lambda_{\text{dff}} \mathcal{L}_{\text{dff}}, \quad (1)$$

Here, θ denotes the trainable detector parameters and $\mathcal{L}(\theta)$ is the total training objective. The terms $\mathcal{L}_{\text{box}}$, $\mathcal{L}_{\text{cls}}$, and $\mathcal{L}_{\text{dff}}$ are the box-regression, classification, and distribution-focal losses, respectively. The coefficients λ_{box} , λ_{cls} , and λ_{dff} are the corresponding nonnegative loss weights. Their exact values and implementation were inherited from the Ultralytics model version used for each detector.

5.3 Confidence-guided routing and random control

Let j index an input image, let q_j denote the maximum retained YOLOv8n confidence for that image, and let η denote the routing threshold selected on the validation split. For images with no retained YOLOv8n detections, q_j was set to 0. The routing decision was

$$r_j = \mathbf{1}(q_j < \eta), \quad (2)$$

where $\mathbf{1}(\cdot)$ is the indicator function, equal to 1 when its condition is true and 0 otherwise. Thus, $r_j = 1$ routes image j to YOLOv8m, whereas $r_j = 0$ keeps the YOLOv8n prediction. The final prediction for image j was

$$\hat{\mathcal{Y}}_j = (1 - r_j) \hat{\mathcal{Y}}_j^n + r_j \hat{\mathcal{Y}}_j^m. \quad (3)$$

Here, $\hat{\mathcal{Y}}_j$ is the final set of predicted boxes for image j , while the superscripts n and m identify the outputs of YOLOv8n and YOLOv8m, respectively. The route fraction is $\alpha = N^{-1} \sum_{j=1}^N r_j$, where N is the number of evaluated images. Detector confidence thresholds were selected on the validation split from the predefined grid [0.05, 0.10, 0.15, 0.20, 0.25, 0.30, 0.40, 0.50, 0.60, 0.70, 0.80, 0.90] by maximizing F1. Ties were resolved by higher recall and then lower FP/image. The selected detector thresholds were fixed before test evaluation. The routing threshold η was selected independently on the validation split from the predefined trigger grid [0.20, 0.30, 0.40, 0.50, 0.60, 0.70, 0.80] using the same F1-first, recall-second, lower-FP/image tie-breaking rule. Cascade detections were evaluated at a fixed protocol confidence of 0.40, specified before test evaluation and not tuned on the test split. The selected value for the primary YOLOv8n-to-YOLOv8m cascade was 0.60. A random-routing control selected the same number of test images for the larger detector without using confidence. Repeated random assignments were summarized by their mean, standard deviation, and empirical percentile range. After validation selection, thresholds were fixed, and the test split was used only for the final evaluation.

Standalone CPU timing was measured on an Intel Core Ultra 5 125H platform at batch size 1 using the same Ultralytics validation timing protocol. Timing included preprocessing, model inference, and postprocessing, but excluded image loading and visualization. Efficient edge inference depends on model scale and implementation details, including compression and quantization choices [32, 33, 34]. The cascade throughput was calculated analytically as

$$T_{\text{cascade}} = T_n + \alpha T_m, \quad (4)$$

where T_{cascade} is the estimated mean per-image cascade latency, T_n and T_m are the measured standalone per-image latencies of YOLOv8n and YOLOv8m, respectively, and α is the observed route fraction. The corresponding analytical throughput is obtained from the reciprocal of this latency after unit conversion.

The reported 16.00 FPS is therefore an analytical cascade throughput estimate, not an end-to-end hardware deployment measurement.

5.4 Difficulty-ranked hard-example replay

Each image in a low-label training subset was scored by the current YOLOv8n checkpoint. Candidate predictions were retained with confidence at least 0.05 for scoring. For an image containing ground-truth boxes, the difficulty score was

$$D_i = 2.35M_i + 0.75L_i + 0.65C_i + 0.65T_i + 0.12F_i, \quad (5)$$

Here, i indexes a training image and D_i is its difficulty score. The coefficients 2.35, 0.75, 0.65, 0.65, and 0.12 are fixed weighting coefficients for the five difficulty components, rather than learned parameters. The missed-object fraction is M_i , the localization error is $L_i = 1 - \overline{\text{IoU}}_i$, C_i penalizes low matched confidence, T_i is the fraction of tiny ground-truth boxes, and F_i measures false-positive pressure. IoU denotes intersection over union. For each ground-truth box, the prediction with the highest IoU was selected independently. A match required IoU at least 0.5 and confidence at least 0.30. Thus, M_i is the fraction of ground-truth boxes whose best prediction failed either condition, $\overline{\text{IoU}}_i$ is the mean of the selected best IoUs, and c_i is the mean confidence of those selected predictions. The confidence term is $C_i = \max(0, 0.30 - c_i)/0.30$. The tiny-target term T_i uses normalized box area below 0.0015. For positive images, F_i is the sum of confidences, clipped to 1, for predictions whose best IoU with any ground-truth box is below 0.30 and whose confidence is at least 0.30. If a positive image has no predictions, its mean best IoU and mean best confidence are zero, $M_i = 1$, and $F_i = 0$. For images without ground-truth boxes, $D_i = 1.25F_i$, where F_i is the clipped confidence sum of predictions with confidence at least 0.30.

Difficulty percentiles were computed within each label subset using the nonzero D_i values. Samples with zero difficulty received one copy. Nonzero samples below the 50th percentile received one copy, samples at or above the 50th percentile received two copies, and samples at or above the 75th percentile received three copies. The same rule was applied with a maximum of three copies in every label setting. The 10%, 25%, and 50% subsets contained 520, 1300, and 2600 source images and produced 910, 2275, and 4549 replay samples, respectively. A compute-matched uniform control randomly permuted the identical copy-count histogram across the identical source images. Therefore, source images, copy-count histogram, total replay samples, optimizer, learning rate, batch size, epochs, input size, and seed were identical between the two replay strategies.

Both replay methods used five additional epochs, input size 416, batch size 8, AdamW, initial learning rate 2×10^{-5} , final learning-rate factor 0.5, weight decay 0.01, and seed 42. The following augmentation parameters were set to zero: mosaic, mixup, copy-paste, horizontal flipping, translation, scaling, HSV hue, HSV saturation, HSV value, and erasing. The 50% setting also included plain continuation with one copy per source image and the same five-epoch schedule. Object-size recall was evaluated at confidence 0.25 and IoU 0.5.

5.5 Robustness and uncertainty analysis

Robustness was evaluated using low light, overexposure, Gaussian noise, motion blur, defocus blur, haze, JPEG compression, and downsample-upsample transformations. Mean retention was defined as

$$\rho_m = \frac{1}{|C|} \sum_{c \in C} \frac{mAP50_{m,c}}{mAP50_{m,\text{clean}}}, \quad (6)$$

where m indexes a detector model, c indexes one corruption condition, ρ_m is the model’s mean mAP50 retention, C is the set of eight transformations, $mAP50_{m,c}$ is the mAP50 of model m under

condition c , and $mAP50_{m, \text{clean}}$ is its clean-image mAP50. Bootstrap confidence intervals used 1000 image-level resamples with seed 42. The bootstrap analysis used a common confidence threshold of 0.25 only for model-level uncertainty comparison and not as the validation-selected deployment operating point. No formal significance testing was used for the replay differences, so the manuscript reports them as descriptive effect sizes.

5.6 VLM diagnostic

The VLM diagnostic used Qwen2-VL-72B on 20 positive whole-image validation samples and 10 detector-guided proposal crops containing 11 annotated UAV instances [35]. The model used deterministic decoding, maximum image side 448, and 4-bit NF4 quantization with double quantization on a 96 GB RTX PRO 6000 GPU. Parsed boxes were matched to official annotations at IoU 0.5. VLM outputs were excluded from detector training, threshold selection, replay ranking, and the primary held-out test metrics.

Data Availability

The DUT-Anti-UAV dataset used in this study is publicly accessible at <https://www.kaggle.com/datasets/akaashb001/dut-anti-uav>. The original project repository and dataset information are available at <https://github.com/wangdongdut/DUT-Anti-UAV>. The processed split files, training configurations, evaluation scripts, detector routing code, replay code, analysis scripts, and result summaries used in this study are available from the corresponding author on reasonable request.

References

- [1] Zhao, J., Zhang, J., Li, D. & Wang, D. Vision-based anti-UAV detection and tracking. *IEEE Trans. Intell. Transp. Syst.* **23**, 25323–25334 (2022).
- [2] Jiang, N. *et al.* Anti-UAV: a large multi-modal benchmark for UAV tracking. arXiv:2101.08466 (2021).
- [3] Zhu, P., Wen, L., Bian, X., Ling, H. & Hu, Q. Vision meets drones: a challenge. arXiv:1804.07437 (2018).
- [4] Du, D. *et al.* The unmanned aerial vehicle benchmark: object detection and tracking. In *Proc. ECCV Workshops* (2018).
- [5] Coluccia, A. *et al.* Drone-vs-bird detection challenge at IEEE AVSS2021. *Sensors* **21**, 2824 (2021).
- [6] Yu, M. & Leung, H. Small-object detection for UAV-based images. In *Proc. 2023 IEEE International Systems Conference (SysCon)*, 1–6 (2023). doi:10.1109/SYSCON53073.2023.10131084.
- [7] Zhou, Y. & Wei, Y. UAV-DETR: an enhanced RT-DETR architecture for efficient small object detection in UAV imagery. *Sensors* **25**, 4582 (2025). doi:10.3390/s25154582.
- [8] Hu, H., Wang, H.-Z., Chen, S.-B. & Tang, J. YOFOR: you only focus on object regions for tiny object detection in aerial images. *Neural Networks* **198**, 108571 (2026). doi:10.1016/j.neunet.2026.108571.
- [9] Redmon, J. & Farhadi, A. YOLOv3: an incremental improvement. arXiv:1804.02767 (2018).
- [10] Bochkovskiy, A., Wang, C.-Y. & Liao, H.-Y. M. YOLOv4: optimal speed and accuracy of object detection. arXiv:2004.10934 (2020).

- [11] Ge, Z., Liu, S., Wang, F., Li, Z. & Sun, J. YOLOX: exceeding YOLO series in 2021. arXiv:2107.08430 (2021).
- [12] Li, C. *et al.* YOLOv6: a single-stage object detection framework for industrial applications. arXiv:2209.02976 (2022).
- [13] Wang, C.-Y., Bochkovskiy, A. & Liao, H.-Y. M. YOLOv7: trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proc. IEEE/CVF CVPR*, 7464–7475 (2023).
- [14] Ultralytics. YOLOv8 documentation (2023).
- [15] Wang, C.-Y., Yeh, I.-H. & Liao, H.-Y. M. YOLOv9: learning what you want to learn using programmable gradient information. arXiv:2402.13616 (2024).
- [16] Wang, A. *et al.* YOLOv10: real-time end-to-end object detection. arXiv:2405.14458 (2024).
- [17] Ultralytics. YOLO11 documentation (2024).
- [18] Lin, T.-Y. *et al.* Feature pyramid networks for object detection. In *Proc. IEEE CVPR*, 2117–2125 (2017).
- [19] Kisantal, M. *et al.* Augmentation for small object detection. arXiv:1902.07296 (2019).
- [20] Teerapittayanon, S., McDanel, B. & Kung, H.-T. BranchyNet: fast inference via early exiting from deep neural networks. In *Proc. 23rd Int. Conf. Pattern Recognition (ICPR)*, 2464–2469 (2016). doi:10.1109/ICPR.2016.7900006.
- [21] Wang, X., Yu, F., Dou, Z.-Y., Darrell, T. & Gonzalez, J. E. SkipNet: learning dynamic routing in convolutional networks. In *Proc. European Conf. Computer Vision (ECCV)*, 420–436 (2018). doi:10.1007/978-3-030-01261-8_25.
- [22] Zhang, H., Hao, C., Song, W., Jiang, B. & Li, B. Adaptive slicing-aided hyper inference for small object detection in high-resolution remote sensing images. *Remote Sens.* **15**, 1249 (2023). doi:10.3390/rs15051249.
- [23] Muzammul, M., Li, X. & Li, X. Enhancing tiny object detection using guided object inference slicing (GOIS): an efficient dynamic adaptive framework for fine-tuned and non-fine-tuned deep learning models. *Neurocomputing* **640**, 130327 (2025). doi:10.1016/j.neucom.2025.130327.
- [24] Shrivastava, A., Gupta, A. & Girshick, R. Training region-based object detectors with online hard example mining. In *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, 761–769 (2016). doi:10.1109/CVPR.2016.89.
- [25] Bengio, Y., Louradour, J., Collobert, R. & Weston, J. Curriculum learning. In *Proc. 26th Annual Int. Conf. Machine Learning (ICML)*, 41–48 (2009). doi:10.1145/1553374.1553380.
- [26] Bouniot, Q., Loesch, A., Habrard, A. & Audigier, R. Towards few-annotation learning for object detection: are transformer-based models more efficient? In *Proc. 2023 IEEE/CVF Winter Conf. Applications of Computer Vision (WACV)*, 75–84 (2023). doi:10.1109/WACV56688.2023.00016.
- [27] Liu, Z., Hu, X. & Nevatia, R. Efficient feature distillation for zero-shot annotation object detection. In *Proc. 2024 IEEE/CVF Winter Conf. Applications of Computer Vision (WACV)*, 893–902 (2024). doi:10.1109/WACV57701.2024.00094.
- [28] Nisa, U. Image augmentation approaches for small and tiny object detection in aerial images: a review. *Multimedia Tools Appl.* **84**, 21521–21568 (2025). doi:10.1007/s11042-024-19768-7.

- [29] Hendrycks, D. & Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. In *Proc. ICLR* (2019).
- [30] Michaelis, C. *et al.* Benchmarking robustness in object detection: autonomous driving when winter is coming. arXiv:1907.07484 (2019).
- [31] Guo, C., Pleiss, G., Sun, Y. & Weinberger, K. Q. On calibration of modern neural networks. In *Proc. ICML*, 1321–1330 (2017).
- [32] Howard, A. G. *et al.* MobileNets: efficient convolutional neural networks for mobile vision applications. arXiv:1704.04861 (2017).
- [33] Tan, M. & Le, Q. V. EfficientNet: rethinking model scaling for convolutional neural networks. In *Proc. ICML*, 6105–6114 (2019).
- [34] Jacob, B. *et al.* Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proc. IEEE CVPR*, 2704–2713 (2018).
- [35] Wang, P. *et al.* Qwen2-VL: enhancing vision-language model’s perception of the world at any resolution. arXiv:2409.12191 (2024).

Author Contributions

X.Z. and X.J. conceptualized the study. X.Z. developed the experimental software, performed the experiments, and analyzed the data. J.L., H.H., Y.L., X.W., K.C., C.J., and Q.L. contributed to implementation, data curation, interpretation, and manuscript revision. X.Z. and X.J. prepared the manuscript. All authors reviewed and approved the final manuscript.

Additional Information

Funding. This research received no external funding.

Code availability. The code used to reproduce the main detector, routing, replay, robustness, and uncertainty analyses will be made publicly available upon publication. The Qwen2-VL-72B diagnostic was executed on external cloud hardware, and its summary records are available from the corresponding author on reasonable request.

Acknowledgements. The authors thank the maintainers of the *DUT-Anti-UAV* dataset and the open-source object detection community for providing reusable resources for vision research.

Competing interests. The authors declare no competing interests.