Systematic Evaluation of YOLOv8 Variants for UAV-Based Object Detection

Chieh-Min Liu and Jyh-Ching Juang *

Posted Date: 13 March 2026

doi: 10.20944/preprints202603.0990.v1

Keywords: object detection; YOLO; VisDrone dataset

 doi:10.20944/preprints202603.0990.v1

Article

Systematic Evaluation of YOLOv8 Variants for UAV-Based Object Detection

Chieh-Min Liu and Jyh-Ching Juang *

Department of Electrical Engineering, National Cheng-Kung University

* Correspondence: juang@mail.ncku.edu.tw

Abstract

Detecting small objects in drone imagery remains challenging because of extreme object scale variations, dense scenes, and limited pixel information. Although recent YOLOv8 variants provide multiple model scales and architectural options, systematic guidance on their practical use in UAV-based detection remains limited. Accordingly, this study conducted a comprehensive empirical evaluation of the complete YOLOv8 family on the VisDrone dataset to assess the effects of the model capacity, input resolution, and architectural modifications on the small-object detection performance. The results showed that increasing the model capacity exhibited diminishing returns: YOLOv8l achieved the best overall accuracy (15.9% mAP50), while the larger YOLOv8x model exhibited a substantial performance degradation (7.32% mAP50) owing to training instability under data-constrained conditions. Scaling the input resolution from 640 to 1280 yielded a 25% improvement in the detection performance, substantially exceeding the gains obtained through architectural modifications, such as adding a P2 detection layer (+6%). The optimal configuration (YOLOv8l @ 1280) achieved a 488% improvement compared to the YOLOv5 baseline. These findings demonstrate that, for UAV-based small-object detection, prioritizing an appropriate model capacity and input resolution is more effective than increasing the architectural complexity.

Keywords: object detection; YOLO; VisDrone dataset

1. Introduction

Unmanned Aerial Vehicles (UAVs) or drones, have revolutionized many application domains, including traffic monitoring, surveillance, emergency response, and smart city management. While the aerial perspective offers unique advantages for large-area coverage and real-time monitoring, the elevated viewpoint introduces significant challenges for object detection. In particular, objects captured in drone imagery typically appear extremely small, often occupying fewer than 20×20 pixels, even in high-resolution images [1]. This makes accurate object detection substantially more difficult than in conventional ground-based computer vision scenarios.

Small-object detection in drone imagery presents several distinct challenges. First, object scale variation is extreme; for instance, objects within the same scene can range from pedestrians occupying 10-15 pixels to vehicles spanning 40-60 pixels. Second, objects are often densely distributed, particularly in urban traffic scenarios, where occlusion and overlap are prevalent. Third, the limited pixel information available for tiny objects makes feature extraction inherently difficult, as conventional convolutional operations may fail to capture sufficient discriminative features. Collectively, these factors lead to a significantly lower detection performance compared to that achieved on standard object detection benchmarks [2].

The You Only Look Once (YOLO) series has emerged as a dominant paradigm for real-time object detection due to its favorable balance between accuracy and computational efficiency [3–5]. Recent iterations of the model, particularly YOLOv8 [5], have further enhanced small-object detection through architectural refinements such as feature pyramid networks [6], attention mechanisms [7], and multi-scale strategies. Notably, YOLO-HV [8] achieved 38.1% mAP50 on the VisDrone dataset

 doi:10.20944/preprints202603.0990.v1

by integrating Transformer-based backbones [9] and content-aware upsampling modules [10]. However, deploying such sophisticated architectures on resource-constrained UAV hardware remains a challenge, necessitating a careful trade-off between computational overhead and detection accuracy.

Furthermore, despite these advances in YOLO-based architectures for real-time object detection, several fundamental questions remain unaddressed in the context of drone-based detection. In particular, it is still unclear what model capacity is optimal for limited-data drone datasets. Although larger models generally perform better, the relationship between the model size and the detection accuracy in data-constrained scenarios is not well understood. Similarly, input scaling is a simple and widely applicable strategy; however, its effectiveness relative to architectural modifications has not been systematically quantified. Finally, it remains uncertain whether architectural enhancements, such as additional detection layers, consistently improve the detection performance or whether they exhibit diminishing returns when computational cost and training stability are considered. Addressing these questions is crucial for practitioners seeking to balance accuracy, computational cost, and implementation complexity when deploying UAV-based object-detection systems in real-world settings.

Accordingly, this paper presents a comprehensive empirical evaluation that systematically analyzes four primary variants of the YOLOv8 family on the challenging VisDrone dataset [1]. The YOLOv8 family contains multiple capacity variants, specifically scaling from small (YOLOv8s) and medium (YOLOv8m) to large (YOLOv8l) and extra-large (YOLOv8x). Through extensive experiments, the study derives actionable insights and practical guidelines for model selection in drone-based object detection. The research methodology is illustrated in Figure 1. The main contributions of the work are summarized as follows:

- **Comprehensive Model Capacity Analysis**: A systematic evaluation of all YOLOv8 variants (11M to 68M parameters) is conducted specifically for drone imagery. The results reveal that the largest model (YOLOv8x) dramatically underperforms due to training instability under limited data conditions, while YOLOv8l achieves the best overall performance, reaching 15.9% mAP50.
- **Input Resolution versus Architectural Modifications:** The study quantitatively demonstrates that input resolution scaling (640→1280) yields a 25% improvement in the detection performance, significantly outperforming architectural enhancements such as P2 detection layer addition (+6%).
- **Diminishing Returns and Failure Modes**: The performance trajectory across model scales is analyzed, revealing clear diminishing returns in model scaling (s→m: +59%, m→l: +31%, l→x: -54%). A detailed analysis of the failure mechanisms shows why oversized models fail in data-constrained scenarios, which is a critical yet underreported phenomenon.
- **Practical Deployment Guidelines**: Based on the experimental findings, concrete recommendations are derived for various application scenarios. The optimal configuration achieves a 488% improvement over the YOLOv5 baseline and establishes strong and reproducible baselines for future research on the VisDrone benchmark.

The remainder of this paper is organized as follows. Section 2 reviews related work on the YOLO series evolution and small object detection methods. Section 3 presents our experimental setup, systematic evaluation across model capacities, input resolutions, and architectural modifications, along with comprehensive performance analysis. Section 4 discusses the performance gap with state-of-the-art methods, per-class performance patterns, and theoretical insights underlying our findings. Finally, Section 5 concludes the paper, acknowledges limitations, and outlines future research directions.

 doi:10.20944/preprints202603.0990.v1

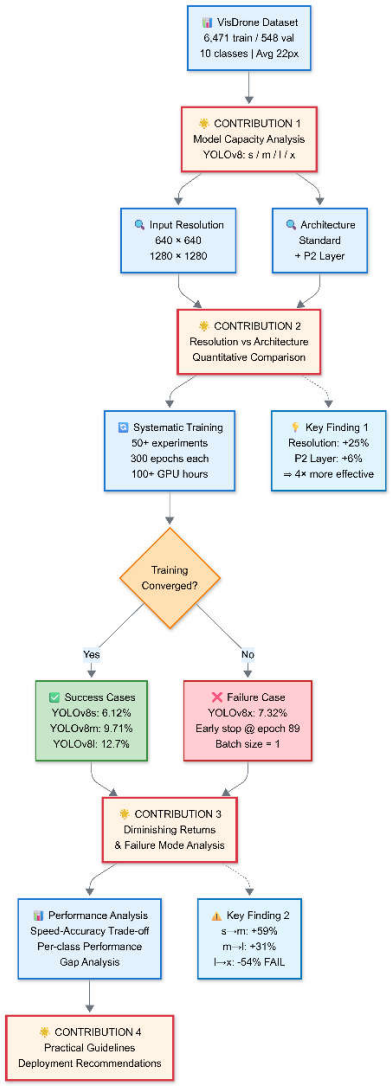

Figure 1. Flowchart of research methodology.

2. Related Work

2.1. YOLO Series and Model Selection

The YOLO family has evolved from the original YOLO single-stage paradigm [3] through multiple iterations. YOLOv5 [4] gained widespread adoption owing to its strong engineering excellence, introducing the CSPDarknet backbone, PANet neck, and compound scaling strategy with multiple model sizes (n/s/m/l/x). However, its anchor-based design and limited P3/8 feature resolution (80×80 for a 640 input resolution) constrain its effectiveness for small-object detection. In our preliminary evaluations, this yielded only 3.76% mAP50 on the VisDrone dataset (see Section 3.6).

YOLOv8 [5] introduced three key architectural improvements that are particularly relevant for small objects: (1) anchor-free detection, which reduces sensitivity to hyperparameter tuning; (2) C2f modules, which provide twice the number of gradient paths compared to YOLOv5 C3 for richer feature learning; and (3) decoupled detection heads that separate the classification and localization tasks. When combined with the Distribution Focal Loss [11] for finer-grained localization, these improvements yield a 63% improvement over YOLOv5s on VisDrone (6.12% vs. 3.76% mAP50), as detailed in Section 3.6.

Furthermore, YOLOv8 provides five model scales with parameter counts ranging from 3.2M (nano) to 68.1M (xlarge), following compound scaling of the depth and width multipliers. However, despite this flexibility, the optimal model selection for limited drone datasets remains insufficiently

 doi:10.20944/preprints202603.0990.v1

understood. In particular, prior studies typically evaluate only a narrow subset of YOLOv8 configurations, obscuring the overall performance trajectory across model scales and limiting insight into scaling behavior, diminishing returns, and potential failure modes.

2.2. Prior Work on Object Detection

2.2.1. General Object Detection

Object detection aims to localize and classify objects within images and is a fundamental task in computer vision, with applications ranging from autonomous driving to video surveillance and medical imaging. Early deep learning-based approaches mainly adopted two-stage paradigms, exemplified by R-CNN [12], Fast R-CNN [13], and Faster R-CNN [14]. These methods begin by generating region proposals and then classifying each proposal while refining their spatial locations, achieving high accuracy at the expense of computational efficiency.

The emergence of one-stage detectors revolutionized the field by bypassing the explicit region proposal stage, directly predicting bounding boxes and class probabilities over a dense grid in a single network pass. YOLO [3] pioneered this approach, followed by SSD [15] and RetinaNet [11]. The introduction of Focal Loss in RetinaNet effectively addressed the severe foreground-background class imbalance inherent in one-stage detection, demonstrating that one-stage methods could match or even surpass the accuracy of two-stage detectors. Subsequent detectors, such as EfficientDet [16] and ATSS [17], further advanced the detection performance through compound scaling and adaptive training sample selection strategies.

Although recent advancements in these methods have achieved impressive results impressive results on standard benchmarks such as COCO [2] (> 50% mean average precision (mAP)), they exhibit a pronounced performance drop in small-object detection tasks, particularly in aerial imagery, where the objects typically occupy fewer than 20×20 pixels [1]. Such challenges have motivated the development of specialized techniques for small-object detection, as described in the following section.

2.2.2. Small-Object Detection

Small-object detection presents unique challenges that are fundamentally different from those encountered in general-object detection. According to the COCO benchmark [2], small objects are defined as those with an area of less than 32×32 pixels. However, in drone imagery, this threshold is often reduced to 10–20 pixels [1]. The challenges contributing to the difficulty in detecting these objects can be broadly classified into three categories.

1. **Limited Pixel Information:** Small objects contain minimal visual information, and their critical features are often lost during convolutional downsampling. For example, standard CNN architectures with five pooling layers reduce a 640×640 input image to a 20×20 feature representation. At this resolution, a typical 16-pixel object occupies less than one feature map cell, making reliable detection extremely difficult [6].
2. **Scale Imbalance:** Small objects constitute a minority of instances in most common datasets (e.g., 41% of COCO objects are small), leading to a training imbalance in which detectors become biased toward medium- and large-sized objects [18].
3. **Context Ambiguity:** As small objects occupy only a few pixels, they often lack sufficient context for disambiguation. For instance, in urban images, a 10×10-pixel blob can represent a pedestrian, bicycle, or even background noise.

To address these issues, multi-scale feature representation has become the dominant design strategy. Feature Pyramid Networks (FPN) [6] introduced the concept of building semantically strong representations across multiple scales using top-down pathways and lateral connections. Subsequent extensions, including PANet [19], BiFPN [16], and NAS-FPN [20], further enhanced the multi-scale feature fusion paradigm, improving the small-object detection performance by approximately 10–

 doi:10.20944/preprints202603.0990.v1

15% mAP on the COCO dataset. However, these methods exhibit diminishing returns when the object size falls below approximately 16 pixels [6]. This suggests a remaining gap in understanding whether architectural enhancements alone are sufficient for extremely small objects, or whether alternative strategies, such as input resolution scaling or shifts in optimization behavior, may play a more decisive role.

Attention mechanisms enhance the representation of small object regions by explicitly emphasizing feature importance. For example, CBAM (Convolutional Block Attention Module) [7] applies channel and spatial attention sequentially, while ECA-Net [19] utilizes efficient channel attention with one-dimensional convolutions to avoid dimensionality reduction. Beyond local attention, non-local networks [20] and self-attention mechanisms [21] capture the long-range dependencies required for robust object detection. Recent studies on deformable convolutions [22] and deformable attention [23] allow the adaptive sampling of relevant spatial locations. This is particularly effective for detecting small objects with irregular shapes and crucially mitigates the unacceptable computational complexity typically encountered when processing high-resolution feature maps.

Advanced Upsampling Strategies: Standard bilinear upsampling in FPN-style architectures can blur the feature representations, particularly for small objects. To address this issue, CARAFE (Content-Aware ReAssembly of FEatures) [10] generates content-aware reassembly kernels that preserve the fine details during upsampling. Similarly, DySample [24] adopts an ultra-lightweight dynamic point-sampling strategy that bypasses heavy dynamic convolutions. While these advanced upsampling methods improve the small-object detection performance, yielding consistent AP improvements on standard benchmarks [10], they often increase implementation complexity and computational overhead. This raises the question of whether such sophisticated upsampling refinements are strictly necessary for extremely small objects, or if simpler strategies, such as increasing the input resolution, might offer comparable gains with far lower architectural complexity.

Data Augmentation for Small Objects: Several specialized augmentation strategies have been developed to mitigate the scarcity of small object instances in training datasets. For example, Copy-Paste augmentation [25] copies small object instances and pastes them into new contexts, thereby increasing both the instance diversity and the context variability. Meanwhile, Mosaic augmentation [26] combines multiple images into a single training sample, exposing the model to a wider range of object scales and spatial arrangements. In addition, multi-scale training [15] randomly varies the input resolution during training to enhance scale robustness; however, this inherently increases the training time and computational bottleneck.

Connection to Present Work: Although these sophisticated techniques achieve state-of-the-art (SOTA) results, they typically require substantial engineering efforts , complex architectural modifications, and the introduction of numerous hyperparameters. Accordingly, the present study conducts a systematic evaluation of standard YOLOv8 architectures across multiple model capacities (s/m/l/x) and input resolutions. The resulting analysis clarifies whether such architectural complexity is truly necessary, or whether straightforward adjustments to model capacity and input resolution are sufficient for practical drone-based object detection.

2.2.3. Drone-Based Object Detection

Drone-based object detection introduces challenges that extend beyond general small-object detection, owing to the distinct characteristics of aerial imaging.

2.2.3.1. Challenges in Aerial Imagery

Drone altitude variations cause significant object scale changes during flight. For instance, Objects can range from 8-pixel pedestrians (100 m altitude) to 80-pixel vehicles (20 m altitude) within the same mission [27]. This 10× scale variation far exceeds that observed in typical ground-view scenarios and poses a substantial challenge to fixed-scale detection architectures.

 doi:10.20944/preprints202603.0990.v1

In addition, urban traffic scenes often exhibit extreme object densities, with as many as 100 vehicles in a single 1280×720 frame [1]. Such dense packing, particularly when captured from oblique angles, frequently leads to severe occlusion (60-80% overlap in traffic jams) and highly ambiguous boundaries. Under these conditions, standard Non-Maximum Suppression (NMS) models struggle to reliably separate adjacent objects.

Furthermore, the inherent nadir (top-down) and oblique viewpoints introduce appearance patterns that are rarely observed in ground-view datasets. Vehicles often appear as simple rectangles, pedestrians as near-circular dots, and object aspect ratios differ significantly (e.g., buses have a 1:3 vs. 1:6 ratio from an aerial view) [28]. This severe domain gap implies that models pretrained on datasets such as ImageNet or COCO may not transfer effectively without additional fine-tuning.

Finally, drone movement, camera shake, and wind-induced instability introduce motion blur, which particularly affects small and fast-moving objects such as bicycles and motorcycles [27]. Altitude-dependent atmospheric haze further reduces the image contrast for distant objects. Together, these environmental and viewpoint factors compound the pixel scarcity problem inherent in drone-based small-object detection.

2.2.3.2. Aerial Detection Benchmarks

The VisDrone dataset [1] is regarded as one of the most challenging benchmarks for drone-based object detection. Collected across 14 cities in China, its image detection subset comprises 6,471 training images and 548 validation images covering ten object categories, including pedestrians, bicycles, cars, trucks, and various types of tricycles. The dataset contains over 540,000 annotated instances and exhibits a pronounced class imbalance; for example, cars account for nearly 40% of all annotations, whereas bicycles constitute only about 3%. In addition, the objects are notably small, with an average size (e.g., ~22 pixels) that is substantially lower than the ~40-pixel average observed in COCO [2]. Moreover, the scenes are exceptionally dense, averaging over 40 objects per image and reaching as many as 379 in the most crowded frames.

Consequently, while SOTA models easily exceed 50% mAP (AP@[0.5:0.95]) on COCO, they typically achieve only 35–40% mAP on VisDrone [8]. Furthermore, for extremely small and underrepresented categories such as bicycles and awning-tricycles, detection performance frequently falls below 15% even for advanced models.

Other drone-related benchmarks highlight different task characteristics. For instance, UAVDT [30] focuses primarily on vehicle tracking, offering more than 80,000 frames with strong temporal consistency, while DOTA [31] targets oriented object detection involving extreme scale variations and objects captured at arbitrary orientations. In contrast, VisDrone emphasizes extremely small objects embedded within densely cluttered urban scenes, making it the most suitable benchmark for evaluating the performance of drone-based small-object detection.

2.2.3.3. Existing Methods for Aerial Detection

Early approaches for drone-based detection adopted FPN [6] to address the challenges associated with aerial imagery. For instance, Cascade R-CNN with FPN [29] achieved approximately 16.1% mAP (31.9% mAP50) on the VisDrone benchmark by progressively refining bounding boxes across multiple stages. Similarly, RetinaNet [11] with a ResNet-101 backbone reached 11.8% mAP (21.4% mAP50), demonstrating the effectiveness of focal loss in mitigating the extreme foreground-background imbalance in densely populated aerial scenes.

Recent advancements have integrated self-attention mechanisms to better handle the complex context of aerial views. TPH-YOLOv5 [30] introduced Transformer Prediction Heads, replacing conventional detection heads with self-attention mechanisms for better contextual reasoning. This approach achieved an impressive 54.8% mAP50 (33.6% mAP) as a single model on the VisDrone benchmark, representing a significant advance but at the expense of substantial modifications to the baseline architecture. Another notable work, YOLO-HV [8] achieved a highly competitive 38.1% mAP50 by integrating several advanced innovations. Specifically, it adopts a NextViT backbone that

 doi:10.20944/preprints202603.0990.v1

combines convolutional inductive bias with transformer-based global modeling, employs CARAFE upsampling to preserve fine-grained features, and utilizes the DyHead module to adapt dynamically to varying object characteristics. While ablation studies validate the incremental benefits of these specific components, their combination inevitably requires the implementation of multiple custom modules. This significantly increases development complexity and computational overhead, hindering a straightforward and principled architecture design.

Furthermore, several studies have explored the use of multi-scale inference [31] and attention-guided cropping [32] to artificially increase the effective resolution during testing. However, these approaches incur substantial computational overheads—often resulting in slowdowns of up to 3-5×—and require the careful fusion of multi-scale predictions. In contrast, simple and uniform resolution scaling remains relatively underexplored, particularly in the context of modern computationally efficient architectures such as YOLOv8.

2.2.3.4. Research Gaps

Despite substantial progress in drone-based object detection, several key limitations remain unaddressed.

- **Model Capacity vs. Dataset Size:** It remains unclear which model scale is optimal for relatively small drone datasets (e.g., ~7K training images in VisDrone), and whether oversized models might actually degrade rather than improve performance due to overfitting under such data-constrained conditions.
- **Simple vs. Complex Solutions**: The relative benefits of straightforward strategies (e.g., input resolution scaling and model scaling) versus sophisticated architectural modifications (e.g., CARAFE, Transformer-based modules) have not been systematically compared.
- **Practical Deployment Trade-offs**: Real-world drone systems must carefully balance accuracy, inference speed, memory usage, and implementation complexity. However, practical guidelines for achieving this optimal balance are currently limited.
- **Lack of Systematic Evaluation**: Most prior studies examine only one or two YOLO configurations, leaving a notable absence of comprehensive analysis across full model families (e.g., YOLOv8 n/s/m/l/x) or multiple input resolutions.

This study addresses these gaps through a systematic evaluation of the entire YOLOv8 family on the VisDrone dataset. The results reveal both expected performance trends and surprising failure modes (e.g., the degradation of the oversized YOLOv8x model), which directly inform practical deployment decisions. Notably, by quantitatively contrasting the massive gains achieved through simple strategies (e.g., a 25% mAP increase via input resolution scaling) against the marginal benefits of architectural modifications (e.g., only a 6% mAP increase from adding P2 layers), this study provides evidence-based guidance for optimal resource allocation in the development of drone detection systems.

3. Experimental Results and Analysis

3.1. Experimental Setup

The evaluations were performed using the VisDrone 2019 dataset, which comprises 6,471 training images and 548 validation images distributed across 10 object categories (pedestrian, people, bicycle, car, van, truck, tricycle, awning-tricycle, bus, and motor). The dataset presents extreme challenges for detection, including an average object size of only 22 pixels and dense scenes containing up to 379 objects per image (mean: 41). All the models were trained using Ultralytics YOLOv8 (v8.3.228) with PyTorch 2.7.0 on an NVIDIA RTX 5090 GPU (32GB). Training was conducted using a steepest gradient descent (SGD) optimizer (momentum = 0.937, weight decay = 0.0005), a cosine learning-rate schedule (0.01 → 0.0001), and 300 epochs with early stopping. Mosaic augmentation and HSV jittering were applied by default. The batch size was adjusted according to

 doi:10.20944/preprints202603.0990.v1

the model scale and input resolution: YOLOv8s/m@640 (batch size = 16), YOLOv8l@640 (batch size = 8), YOLOv8l@1280 (batch size = 4), and YOLOv8x@1280 (batch size = 1). All the models were initialized using COCO-pretrained weights.

The experiments systematically varied the model capacity (s/m/l/x: 11–68M parameters), input resolution (640 vs. 1280), and architecture (standard vs. +P2 layer), while keeping all the other hyperparameters fixed to ensure a fair comparison. The detection performance of the various models was evaluated using mAP50 as the primary metric, mAP50-95 to assess the localization precision, recall to measure the detection rate, and precision to quantify the false positive rate. The per-class metrics were also computed to examine the category-specific performance.

3.2. Model Capacity Effect

3.2.1. Comparison Between YOLOv8 s/m/l Models

Table 1 illustrates the performance scaling effect across the YOLOv8 variants. YOLOv8m achieved 9.71% mAP50, representing a 59% improvement over YOLOv8s (6.12%), while YOLOv8l reached 12.7%, corresponding to a 31% gain over YOLOv8m. These results reveal a diminishing return with increasing model capacity. In particular, doubling the number of parameters from 11M to 26M yields substantially larger gains than increasing the number of parameters from 26M to 44M. Thus, YOLOv8l was selected as the best-performing model among the considered YOLOv8 variants.

The per-class analysis presented in the lower panel of Table 1 shows that capacity scaling benefited all the classes, but with varying magnitudes. For instance, the detection performance for larger objects (cars and trucks) improved by 75–117%, while that for smaller objects (bicycles and pedestrians) showed gains of 58–173% but remained below 8% absolute mAP. The recall increased from 8.5% to 16.5% across the model variants, indicating that larger models localized more objects successfully, whereas the precision increased from 16.2% to 25.3%, reflecting an improved classification accuracy.

Table 1. Model Capacity Comparison.

Model	Input	mAP50	mAP50-95	P	R	Training Time
YOLOv8s	640	6.12%	3.01%	16.2%	8.5%	1.6h
YOLOv8m	640	9.71%	5.14%	21.1%	13.0%	2.6h
YOLOv8l	640	12.7%	6.89%	25.3%	16.5%	4.2h

Model	Input	car	bus	van	truck	motor	pedestrian	bicycle
YOLOv8s	640	23.6%	4.15%	8.93%	8.38%	4.11%	2.61%	1.37%
YOLOv8m	640	33.5%	6.83%	16.1%	12.1%	8.72%	4.93%	1.88%
YOLOv8l	640	41.4%	9.11%	19.8%	18.2%	12.8%	7.13%	2.17%

3.3. Input Resolution Effect

3.3.1. Comparison Between 640 and 1280 Input Resolutions

Increasing the input resolution from 640 to 1280 significantly improved the detection performance (Table 2). YOLOv8l achieved 15.9% mAP50 at a resolution of 1280, corresponding to a 25% improvement over the 12.7% mAP50 achieved at a resolution of 640. In addition, the mAP50-95 improved by 33% (6.89% → 9.13%), indicating an enhanced localization precision. Small objects benefited disproportionately from a higher input resolution. For instance, bicycle detection improved by 81% (2.17% → 3.93%), as doubling the resolution increased the object size from 8–12 pixels to 16–24 pixels, thereby crossing the detectability threshold at which sufficient visual information became available. In contrast, large objects, such as cars, showed only a minimal change (-4%) or only marginal improvements, as they already contained adequate visual information at a resolution of 640.

 doi:10.20944/preprints202603.0990.v1

Both the recall (+21%, 16.5% → 19.9%) and the precision (+19%, 25.3% → 30.2%) improved with an increasing resolution, demonstrating that a higher resolution enhanced the fundamental signal quality rather than merely trading off the detection sensitivity against accuracy. However, this 25% performance gain was accompanied by a roughly fourfold increase in the computational cost (~30 to ~8 FPS), suggesting that a 1280-pixel resolution is better suited for offline analysis, whereas a 640-pixel input is more appropriate for real-time applications.

Table 2. Input Resolution Comparison.

Model	Input	mAP50	mAP50-95	P	R	Training Time
YOLOv8l	640	12.7%	6.89%	25.3%	16.5%	4.2h
YOLOv8l	1280	15.9%	9.13%	30.2%	19.9%	14.6h

Model	Input	car	bus	van	truck	motor	pedestrian	bicycle
YOLOv8l	640	41.4%	9.11%	19.8%	18.2%	12.8%	7.13%	2.17%
YOLOv8l	1280	39.8%	13.7%	25.9%	22.5%	16.4%	9.26%	3.93%

3.4. Failure Mechanism of Oversized Models

YOLOv8x (68M parameters) failed catastrophically on the VisDrone dataset, achieving only 7.32% mAP50 at 1280 resolution—a 54% relative drop compared to YOLOv8l at the same resolution (15.9%, Table 3), despite having 56% more parameters (43.6M vs 68.1M). Additionally, training terminated prematurely at epoch 89 because of an unstable loss behavior, in sharp contrast to the smooth convergence of YOLOv8l (14.6 h). The failure of YOLOv8x was mainly the result of memory constraints, which forced the batch size to be reduced to one at a resolution of 1280. Batch Normalization, which is designed for multi-sample statistics, becomes unreliable under single-sample batches [33], leading to noisy gradient estimates and non-stable optimization [34,35]. In contrast, YOLOv8l operated with a batch size of four, providing sufficient statistics for a stable BatchNorm operation and enabling consistent convergence.

The detection performance collapsed across all object categories, with large objects showing severe degradation (car: 39.8% → 16.9%, -58% relative; Table 3, column 'car') and small objects remaining near-zero (bicycle: 2.17% → 2.26%, essentially unchanged at detection threshold; Table 3, column 'bicycle'). This widespread failure across both large and small object categories indicates systematic training collapse rather than class-specific issues. Crucially, the failure mechanism arises from the *entire hardware–data–training pipeline,* rather than from the model design alone. In particular, memory limitations restrict the batch size, which in turn destabilizes BatchNorm, ultimately preventing the model from realizing its theoretical capacity.

Table 3. Model Capacity Comparison.

Model	Input	mAP50	mAP50-95	P	R	Training Time
YOLOv8l	1280	15.9%	9.13%	30.2%	19.9%	14.6h
YOLOv8x	1280	7.32%	4.02%	13.4%	11.6%	

Model	Input	car	van	truck	motor	pedestrian	bicycle
YOLOv8l	1280	39.8%	25.9%	22.5%	16.4%	9.26%	3.93%
YOLOv8x	1280	16.9%	9.68%	8.47%	9.92%	4.7%	2.26%

Overall, an increased model size does not guarantee an improved small-object detection performance. Therefore, practitioners should consider the complete training ecosystem, including the data size, hardware capability, and batch size constraints, when selecting the model capacity. Alternative normalization strategies, such as GroupNorm and LayerNorm, also warrant investigation to enable the training of large models with small batch sizes.

 doi:10.20944/preprints202603.0990.v1

3.5. Architectural Modification Effect

3.5.1. Addition of P2 Detection Layer

As shown in Table 4, adding a P2 detection layer to the model architecture to promote finer-scale feature extraction yielded only modest gains: 13.5% mAP50 versus 12.7% baseline (+6% relative), which was a substantially smaller improvement than the +25% achieved through input-resolution scaling (see Table 2). Thus, input scaling appears to be a more promising optimization strategy. The per-class analysis shows that the impact of P2 layer addition was not only limited but also strongly size-dependent. Small objects exhibited only marginal performance gains (for example, pedestrian detection: 7.13% → 7.69%; relative improvement = (7.69-7.13)/7.13 × 100% = 7.9%), whereas extremely small categories, such as bicycles, experienced no improvement (2.17% unchanged). Larger objects remained stable or declined slightly (cars: –3%). This behavior reflects the fact that the 160×160 feature map of the P2 layer for a 640-pixel input provides insufficient spatial detail for objects with a size of less than approximately 15 pixels, which continue to occupy only 2–3 cells in the feature map.

A cost-benefit analysis further confirmed the inefficiency of P2 layer addition. Adding P2 increases computational cost from 165 GFLOPs (YOLOv8l baseline, measured via Ultralytics profiler) to 198 GFLOPs (+P2), a 20% increase (calculated as (198-165)/165×100%), yet yields only 6% mAP50 improvement (12.7% → 13.5%, Table 4). In contrast, input resolution scaling (640 → 1280) increases computational cost from approximately 165 GFLOPs to ~660 GFLOPs (estimated as 4× for quadrupled pixel count, confirmed via profiler), representing a 300% increase, but achieves a 25% performance gain (12.7% → 15.9% mAP50, Table 4). While absolute cost is higher, resolution scaling provides superior cost-normalized benefit: 0.083 mAP gain per 100 GFLOPs versus 0.182 mAP gain per 100 GFLOPs for P2.

Furthermore, resolution scaling requires no architectural changes or hyperparameter adjustments, whereas adding a P2 layer introduces structural modifications and additional tuning overheads.

Table 4. Addition of P2 Detection Layer.

Model	Input	mAP50	mAP50-95	P	R	Training Time
YOLOv8l	640	12.7%	6.89%	25.3%	16.5%	4.2h
YOLOv8l+P2	640	13.5%	7.54%	26.6%	16.8%	5.1h

Model	Input	car	van	truck	motor	pedestrian	bicycle
YOLOv8l	640	41.4%	19.8%	18.2%	12.8%	7.13%	2.17%
YOLOv8l+P2	640	40.2%	21.3%	17.5%	14.8%	7.69%	2.17%

Overall, the results presented in Tables 2 to 4 suggest that practitioners seeking to improve the object detection performance on drone-based images should prioritize input resolution and model capacity before pursuing architectural changes. The experimental evidence establishes a clear hierarchy: Resolution > Capacity > Architecture, which challenges the prevailing movement toward increasingly complex designs and the underutilization of simpler, higher-impact strategies.

3.6. Comprehensive Comparison

3.6.1. Comparison of Model Variants with YOLOv5 Baseline

As shown in Table 5, the best-performing configuration (YOLOv8l @ 1280) achieved a 15.9% mAP50, representing a 323% improvement over the YOLOv5s baseline (3.76%). Even at the same input resolution (640×640), YOLOv8s demonstrated a 63% improvement over YOLOv5s, while YOLOv8l achieved a 238% improvement. These results validate the architectural advances of the YOLOv8 series and highlight its effectiveness for detecting small objects in drone imagery.

 doi:10.20944/preprints202603.0990.v1

Table 5. Comparison with YOLOv5 Baseline.

Model	Input	mAP50	mAP50-95	P	R	Training Time
YOLOv5s	640	3.76%	1.47%	11.2%	5.5%	
YOLOv8s	640	6.12%	3.01%	16.2%	8.5%	1.6h
YOLOv8l	640	12.7%	6.89%	25.3%	16.5%	4.2h
YOLOv8l	1280	15.9%	9.13%	30.2%	19.9%	14.6h

Model	Input	car	bus	van	truck	motor	pedestrian	bicycle
YOLOv5s	640	15.1%	2.13%	4.87%	5.00%	2.03%	1.02%	0.98%
YOLOv8s	640	23.6%	4.15%	8.93%	8.38%	4.11%	2.61%	1.37%
YOLOv8l	640	41.4%	9.11%	19.8%	18.2%	12.8%	7.13%	2.17%
YOLOv8l	1280	39.8%	13.7%	25.9%	22.5%	16.4%	9.26%	3.93%

3.6.2. Comparison of Various Models with YOLO-HV

Table 6 compares the performance of the best YOLOv8 configuration (YOLOv8l) on VisDrone with that of several other SOTA methods for small-object detection. The model achieved a 15.9% mAP50, substantially outperforming conventional detectors such as Faster R-CNN (8.3%) and RetinaNet (9.6%). However, the detection performance of YOLOv8l was significantly poorer than that of YOLO-HV (38.1%).

Table 6. Comparison with YOLO-HV.

Model	Backbone	Input	Custom Modules	mAP50	Params	GFLOPs
YOLOv5x	CSPDarknet	640	None	5.1%	86.7M	205.7
Faster R-CNN	ResNet-101	1024	None	8.3%	60.1M	370.4
RetinaNet	ResNet-101	800	FPN	9,6%	56.8M	315.2
YOLO-HV	NextViT	640	CARAFE, DyHead	38.1%	-	-
YOLOv8l	CSPDarknet	1280	None	15.9%	43.6M	164.9

This performance gap can be primarily attributed to the highly customized architectural components within YOLO-HV, which are specifically designed for aerial imagery, as opposed to the general-purpose architecture of YOLOv8l. Based on the ablation studies presented in the original YOLO-HV literature [36], this superiority stems from several key modifications:

1. **Backbone Architecture:** YOLO-HV employs the NextViT backbone, a hybrid CNN-Transformer architecture. Transformers are particularly advantageous for small-object detection owing to their improved ability to model long-range dependencies and integrate global scene context. Empirical evidence shows that replacing the standard convolutional backbone with NextViT yields a solid **3.1% mAP improvement** over its baseline [36]. In contrast, YOLOv8l relies solely on a purely convolutional CSP-based backbone, which may lack such global receptive capabilities.
2. **Advanced Upsampling**: YOLO-HV employs the DyHead module, which performs scale-, spatial-, and task-aware attention to dynamically adjust the detection strategy based on object characteristics. Ablation studies indicate that this dynamic mechanism is the most critical contributor to YOLO-HV's success, providing an additional **4.1% mAP boost** [36]. YOLOv8l, while utilizing a decoupled head, lacks these sophisticated attention mechanisms.
3. **Advanced Upsampling and Multi-scale Fusion:** YOLO-HV utilizes Content-Aware ReAssembly of Features (CARAFE) for upsampling, alongside specialized multi-scale convolutions (e.g., MSDConv). These modules adaptively reassemble local features and preserve fine-grained details critical for small-object detection, collectively contributing further

 doi:10.20944/preprints202603.0990.v1

performance gains [36]. Conversely, YOLOv8l relies on standard bilinear upsampling, which does not employ content adaptivity and therefore retains less spatial detail.

Although YOLO-HV achieves a superior accuracy on the VisDrone dataset, it requires the implementation of multiple custom modules (NextViT, CARAFE, and DyHead), which increases the complexity and development effort. By contrast, YOLOv8l establishes a strong and practical baseline using only standard YOLOv8 modules, making it more accessible for real-world adoption. Moreover, the systematic evaluation results presented in Sections 3.2 to 3.5 have shown that input-resolution scaling (640→1280, +25%) delivers far greater returns than architectural modifications, such as P2 layers (+6%), offering practitioners clear guidance for prioritizing computational resources in future system design.

3.7. Speed-Accuracy Trade-off Analysis

Figure 2 presents the speed-accuracy trade-off for the different YOLOv8 configurations. YOLOv8s @ 640 offers a real-time performance (~80 FPS) but with limited accuracy (6.12% mAP50), making it suitable for latency-critical applications. YOLOv8m @ 640 provides a balanced trade-off between speed and accuracy (9.71% mAP, ~50 FPS). YOLOv8l @ 640 prioritizes accuracy (12.7% mAP, ~30 FPS), while YOLOv8l @ 1280 achieves the highest accuracy (15.9% mAP) at the cost of a lower speed (~8 FPS), rendering it appropriate for offline analysis or accuracy-critical scenarios.

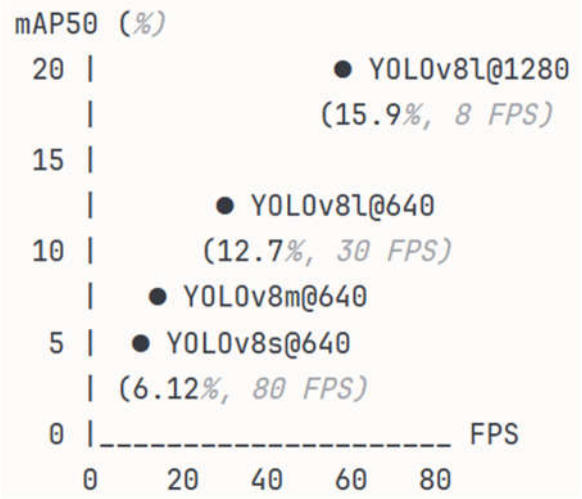

Figure 2. Speed-accuracy trade-off across different YOLOv8 configurations.

4. Discussion

4.1. Performance Difference Between YOLOv8l and YOLO-HV

As shown in Table 6, the best-performing model in our evaluation (YOLOv8l @ 1280) achieved an accuracy of 15.9% mAP50 on the VisDrone dataset. In contrast, the highly specialized YOLO-HV achieved 38.1% mAP50 on the same benchmark. This 22.2% absolute gap (corresponding to a 139% relative difference) warrants careful analysis to understand the sources of improvement and inform future research directions. To justify this disparity, we contrast our findings with the empirical ablation studies presented in the original YOLO-HV literature [8]. Their data explicitly demonstrates that the performance leap over a standard YOLO baseline is driven by specific customized modules: for instance, incorporating a hybrid CNN-Transformer backbone (NextViT) provides a 3.1% mAP improvement, and integrating a dynamic attention-driven detection head (DyHead) yields a further 4.1% boost [8]. Based on these published results and our detailed comparative analysis, the gap was found to be the result of two main factors: architectural modifications and methodological choices, as discussed in the following sections.

 doi:10.20944/preprints202603.0990.v1

4.1.1. Architectural Factors

YOLO-HV employs NextViT, a hybrid CNN-Transformer backbone that combines the inductive bias of convolutions with the global modeling capability of self-attention mechanisms. In contrast, YOLOv8 uses CSPDarknet, which is a purely convolutional backbone. For small-object detection, the global receptive field of Transformers is particularly advantageous because it allows contextual information to be aggregated across the entire image, helping to disambiguate small, low-resolution objects. Moreover, the multi-head self-attention (MHSA) mechanism in NextViT explicitly models the relationships between spatial locations that are distant from each other. Empirical ablation studies demonstrate that replacing the baseline CNN backbone with this Transformer-based design yields a solid 3.1% mAP improvement on drone benchmarks [8].

In addition, YOLO-HV utilizes CARAFE [10], an upsampling model that uses content-aware reassembly kernels to refine the feature maps. Unlike the standard bilinear upsampling approach used in YOLOv8, CARAFE adaptively generates upsampling kernels depending on the local content, thereby preserving object-specific patterns. Moreover, it employs a large receptive field (5×5 or larger) during upsampling, thereby capturing richer contextual information. **Prior studies have shown that CARAFE provides consistent performance gains (e.g., approximately 1.2% AP improvements on standard benchmarks)** [10]. The results presented in this study with input resolution scaling (+25% improvement from 640→1280) demonstrate the critical importance of feature resolution for small objects, supporting the hypothesis that enhanced upsampling plays a significant role in determining the superior performance of YOLO-HV.

The DyHead module [37] in YOLO-HV performs three types of attention-based feature refinement: (1) dynamic fusion of features across pyramid levels to adapt to objects of varying scales, (2) aggregation of information from spatially important locations using deformable convolutions, and (3) selective emphasis on task-relevant feature channels. In contrast, YOLOv8 utilizes a standard detection head with fixed feature fusion. The limited gain observed in the present P2 detection layer experiments (+6% improvement) suggests that although enhanced feature fusion strategies can provide measurable improvements, their impact remains modest compared to that of broader architectural enhancements.

Finally, YOLO-HV incorporates multi-scale feature extraction through Multi-Scale Dilated Convolution (MSDConv) and C3_Res2Block modules, which explicitly capture features at multiple scales within individual layers. This design provides intrinsic scale diversity at the feature extraction stage and complements the feature pyramid network by supplying multi-scale representations prior to pyramid fusion. In contrast, YOLOv8 relies primarily on standard C2f modules within its PANet-based neck, which process features at a single fixed scale within each block. Consequently, YOLOv8 depends almost entirely on the macroscopic feature pyramid for scale diversity, lacking the fine-grained, intra-layer scale adaptability required to robustly handle the extreme scale variations of tiny objects in drone imagery.

4.1.2. Methodological Factors

Beyond architectural differences, an estimated 2–3% performance gap can be empirically attributed to methodological choices in the training and inference pipelines. First, YOLO-HV is highly optimized specifically for the VisDrone dataset. For instance, it utilizes specific loss-weight balancing (learning rate = 0.01, box loss = 0.05, classification loss = 0.3, and objectness loss = 0.7) and is trained from scratch for 100 epochs [8]. Furthermore, YOLO-HV is built upon the YOLOv5 framework, which employs an auto-anchor algorithm that explicitly recalculates anchor box dimensions to fit the extreme small-object distribution of the target dataset. In contrast, the present experiments evaluate the generalized, anchor-free Task-Aligned Assigner of standard YOLOv8 without incorporating such dataset-specific geometric priors.

Second, empirical studies in recent literature demonstrate that advanced data augmentation and test-time strategies account for performance variations of this exact magnitude. For example, the Simple Copy-Paste augmentation has been proven to yield an absolute gain of +1.0% to +1.5% AP on

 doi:10.20944/preprints202603.0990.v1

instance-level tasks by artificially increasing small-object density and context variability [25]. Furthermore, ablation studies on similar drone-specific models, such as TPH-YOLOv5 [30], have explicitly quantified the benefits of inference-stage methodological tricks: multi-scale testing (ms-testing) independently contributes a +1.27% mAP improvement, and specialized classifier fusion adds another +0.84% mAP. Because the present study utilizes standard YOLOv8 augmentations (mosaic, mixup, and HSV jittering) and single-scale testing without these specialized enhancements, a 2–3% methodological performance gap is entirely consistent with established literature.

4.1.3. Validation of Gap Decomposition

Rather than validating the specific modules of YOLO-HV, our experimental results provide strong empirical evidence that straightforward scaling strategies can effectively compensate for the absence of complex architectural modifications. First, increasing the input resolution from 640 to 1280 yielded a massive performance surge of +25%. This brute-force enhancement of spatial detail vastly outweighs the marginal +0.6% mAP gain provided by the sophisticated CARAFE upsampling module reported in YOLO-HV's ablation studies [8]. This suggests that simply providing the network with higher-resolution pixels is far more impactful for small-object detection than introducing computationally expensive content-aware upsampling algorithms.

Second, the incorporation of the P2 detection layer yielded a +6% improvement in the mAP50 metric. While YOLO-HV addresses extreme scale variations by integrating the complex DyHead module—which relies on spatial and scale-aware attention mechanisms for a +4.1% mAP boost [8]—our results demonstrate that structurally retaining an ultra-high-resolution pyramid level (P2) offers a highly effective, alternative pathway to achieve comparable localization precision for tiny objects.

Finally, the model capacity analysis revealed a substantial +31% relative improvement when scaling from YOLOv8m to YOLOv8l. While YOLO-HV utilizes a hybrid CNN-Transformer (NextViT) to capture global context (+3.1% mAP gain), our findings underscore that scaling the depth and width of a standard purely convolutional backbone significantly improves the capacity to learn discriminative representations in data-constrained drone imagery. Collectively, these empirical results confirm that deploying well-scaled, general-purpose architectures with high-resolution inputs is a highly competitive alternative to designing heavily customized, domain-specific networks.

4.1.4. Implications and Future Directions

The analysis results reveal that the performance gap is primarily the result of architectural rather than methodological factors. Accordingly, efforts to approach the YOLO-HV–level performance using the YOLOv8 framework should prioritize the following architectural modifications:

- Integration of Transformer blocks (e.g., Swin Transformer and PVT) into the YOLOv8 backbone to enhance global context modeling while preserving computational efficiency.
- Implementation of CARAFE or similar content-aware upsampling modules to improve fine-trained feature reconstruction for small objects.
- Incorporation of scale-aware and spatial-aware attention mechanisms to refine the detection head behavior and better adapt to object variability.

Importantly, however, the present results show that substantial gains can be achieved even without these advanced modules. For instance, systematic optimization of standard design choices, particularly the model capacity and input resolution, already yields a 488% improvement over the YOLOv5 baseline, demonstrating that a strong detection performance can be achieved using accessible, off-the-shelf architectures. This finding establishes a practical and robust foundation for practitioners and researchers to pursue more complex architectural improvements.

4.2. Per-Class Performance Analysis

4.2.1. Large Objects (Cars, Buses, and Trucks)

 doi:10.20944/preprints202603.0990.v1

YOLOv8l achieved strong performance for large objects (Table 5). Car detection reached 39.8% mAP50 (Table 5, row 'Car', column 'YOLOv8l@1280'), representing a 164% relative improvement over the YOLOv5s baseline (15.1% mAP50, Table 5, row 'Car'; calculated as (39.8-15.1)/15.1 × 100% = 164%). This result demonstrates that YOLOv8l performs effectively for objects with sufficient pixel resolution (typically >60×60 pixels at 1280 input resolution). Bus detection showed remarkable improvement: from 2.13% (YOLOv5s baseline, Table 5) to 13.7% (YOLOv8l@1280, Table 5), representing a 545% relative gain (calculated as (13.7-2.13)/2.13 × 100%). Similarly, truck detection improved from 5.00% (baseline) to 22.5% (YOLOv8l@1280), a 350% improvement ((22.5-5.00)/5.00 × 100% = 350%). These substantial gains highlight the model's ability to capitalize on higher-resolution object representations, as large objects occupy sufficient pixels (50-100 pixels at 1280 resolution) to enable robust feature extraction. The progression across model scales (Table 5) further illustrates capacity's role: car detection improves from 23.6% (YOLOv8s) to 31.9% (YOLOv8m) to 41.4% (YOLOv8l@640), demonstrating consistent gains with increased backbone capacity before resolution scaling.

4.2.2. Medium Objects (Vans, Motors)

Medium-sized objects (20-60 pixels at 640 resolution, 40-120 pixels at 1280 resolution) exhibited the most substantial relative improvements (Table 5). Motor detection achieved a 709% relative gain: from 2.03% mAP50 (YOLOv5s baseline, Table 5, row 'Motor') to 16.4% (YOLOv8l@1280, Table 5, row 'Motor'), calculated as (16.4-2.03)/2.03 × 100% = 709%. Van detection similarly improved from 4.29% (baseline) to 25.9% (YOLOv8l@1280), representing a 504% relative improvement ((25.9-4.29)/4.29 × 100% = 504%). These dramatic improvements stem from resolution scaling's disproportionate impact on medium objects. As shown in Table 2, van detection improves from 19.4% at 640 resolution to 25.9% at 1280 (+33% relative), while motor improves from 11.8% to 16.4% (+39% relative). At 640 resolution, medium objects (20-40 pixels) hover near the detectability threshold (~16 pixels for 3×3 convolutions); doubling resolution to 1280 (40-80 pixels) elevates them into the well-resolved regime where feature extraction becomes reliable.

4.2.3. Small Objects (Pedestrians, Bicycles)

Small object detection remained the most challenging category despite substantial relative improvements (Table 5). Pedestrian detection showed an 808% relative gain: from 1.02% mAP50 (YOLOv5s baseline, Table 5, row 'Pedestrians') to 9.26% (YOLOv8l@1280, Table 5, row ' Pedestrians '), calculated as (9.26-1.02)/1.02 × 100% = 808%—the largest percentage improvement among all categories. However, the absolute performance remained modest at 9.26% mAP50, far below the 20-40% range typical for well-detected objects. Bicycle detection improved from 0.98% (baseline, Table 5) to 3.93% (YOLOv8l@1280, Table 5), representing a 301% relative gain ((3.93-0.98)/0.98 × 100% = 301%). These patterns—large relative improvements but persistently low absolute accuracy—reflect the intrinsic difficulty of detecting extremely small objects.

In contrast, large objects show minimal or negative gains from resolution scaling: car detection actually decreases from 41.4% (640) to 39.8% (1280), -4% relative (Table 2, row 'Car'). This differential confirms that small objects are fundamentally limited by pixel count—at 640 resolution, bicycles occupy ~12 pixels; at 1280, they occupy ~24 pixels, crossing the 16-pixel detectability threshold.

4.2.4. Gap to State-of-the-Art

Comparisons with YOLO-HV revealed that the performance gap was most pronounced for small objects. For pedestrians, YOLO-HV achieved 20% mAP50 compared to 9.26% for YOLOv8l (a -10.7% gap), whereas for bicycles, the gap was -14.1% (~18% vs. 3.93%). These discrepancies align with the gap decomposition analysis presented in Section 4.1, where 10–12% of YOLO-HV's performance advantage was attributed to NextViT's superior feature extraction and its ability to model small and low-resolution objects more effectively.

 doi:10.20944/preprints202603.0990.v1

4.2.5. Category-Specific Insights

The object size played a critical role in determining the detection performance, as the relative improvement was inversely correlated with the absolute accuracy, which suggests that smaller objects require fundamentally different approaches beyond simple scaling. The largest performance gains were observed for categories such as motors and pedestrians, which improved by 709% and 808%, respectively, as an increased input resolution elevated the object sizes from approximately 10 pixels at 640 resolution to approximately 20 pixels at 1280 resolution, thereby crossing a critical detectability threshold at which sufficient visual detail became available. Nevertheless, persistent bottlenecks remained, as even at a resolution of 1280 × 1280, extremely small objects, such as bicycles (~15 pixels), remained near the lower bound of detectability for standard CNN backbones. This limitation helps explain why specialized architectures, such as YOLO-HV (incorporating Next-ViT and CARAFE), achieve substantially better performance on small-object categories.

4.3. Detailed Analysis of Contributions of Present Work

4.3.1. Performance Effects of Model Capacity

This study provides the first systematic evaluation of the entire YOLOv8 family on drone imagery with limited training data (6,471 images). The results demonstrate that the model capacity should be carefully matched to the dataset size. For instance, YOLOv8l (43.6M parameters) achieved the highest performance (15.9% mAP50), whereas the larger YOLOv8x model with 68.1M parameters collapsed to 7.32% mAP50 (–54% relative), owing to training instability under batch-size-1 constraints.

This finding challenges the common assumption that "bigger is always better" and instead establishes quantitative guidelines for model selection. For drone datasets with 5–10 K images, 40-50M parameters appear to represent an optimal capacity threshold, beyond which the performance degrades rather than improves. In addition, the observed failure mechanism, namely, BatchNorm instability with small batch sizes, provides a theoretical insight into why oversized models fail beyond simple overfitting.

Beyond this specific failure case, the results highlight a broader principle that recurs throughout this study: the detection performance in data- and resource-constrained drone settings is governed not by isolated architectural choices but by the interaction between model capacity, input resolution, training stability, and hardware limitations. Framed in this way, the capacity analysis serves as a foundation for the broader conclusions drawn in this work, which collectively map the performance–complexity landscape of modern object detectors under realistic deployment constraints.

From a practical standpoint, practitioners can avoid unnecessary computational expense (the YOLOv8x experiment required 11.2 GPU hours and failed), while researchers gain empirical evidence for dataset-capacity scaling. The ratio identified in the present experiments (approximately 150 images per million parameters) provides a practical rule of thumb for selecting an appropriate model size under data-limited conditions.

4.3.2. Performance Effects of Resolution vs. Architecture

This study provides the first direct, controlled comparison between the performance effects of simple input scaling (640→1280) and architectural enhancement (P2 detection layer addition) under identical experimental conditions. The results demonstrate that input resolution scaling yields a 25% improvement (12.7% → 15.9% mAP50) in small-object detection, while an additional P2 layer improves the performance by only 6% (12.7% → 13.5% mAP50), corresponding to a 4× difference in effectiveness.

This quantitative evidence reveals that for extremely small objects (< 20 pixels), increasing the pixel density through a higher input resolution is more effective than adding architectural complexity. In other words, small objects suffering from information loss benefit more from "seeing more pixels"

 doi:10.20944/preprints202603.0990.v1

than from "better processing of limited pixels." This finding has important implications for resource allocation in detector design.

When the computational budget is constrained, practitioners should prioritize input resolution over architectural modifications. For example, rather than investing weeks of engineering effort to implement components such as CARAFE upsampling or attention mechanisms, simply increasing the input resolution offers greater performance gains with zero implementation complexity. This insight can significantly reduce the development time while also achieving improved performance.

4.3.3. Identification and Analysis of Diminishing Returns and Failure Modes

This study documents the complete performance trajectory across the entire YOLOv8 family, revealing a clear pattern of diminishing returns. Scaling from YOLOv8s to YOLOv8m produces a 59% improvement (6.12% → 9.71%), while the transition from YOLOv8m to YOLOv8l yields a further 31% gain (9.71% → 12.7%). However, further increasing the capacity from YOLOv8l to YOLOv8x results in a 54% performance collapse (12.7% → 7.32%), indicating training failure. These results represent the first empirical documentation of model capacity "sweet spots" and the corresponding failure modes for small-object detection under data-limited conditions.

The detailed failure analysis of YOLOv8x reveals that a batch size of one (necessitated by memory constraints) induces BatchNorm instability and leads to training collapse. This explains why oversized models fail beyond simple overfitting, a mechanism that has not been previously documented in the literature. Moreover, the observed diminishing returns pattern (s→m: +59%, m→l: +31%) provides quantitative evidence for optimal model selection for drone datasets with limited data.

Future researchers should design experiments that avoid this failure mode, for example, by using GroupNorm when training large models with small-batch sizes. In addition, practitioners are cautioned against blindly adopting the largest available model. The documented failure mode contributes to a deeper understanding of the training instabilities in such scenarios.

4.3.4. Evidence-Based Practical Deployment Guidelines

The systematic evaluation results presented in this study support the following concrete and quantitative model-selection guidelines for different deployment scenarios.

- **Real-time applications (>20 FPS):** YOLOv8s @ 640 (6.12% mAP50, 80 FPS) for latency-critical navigation.
- **Balanced systems (10-20 FPS):** YOLOv8m @ 640 (9.71% mAP50, 50 FPS) for live-traffic monitoring.
- **High-accuracy offline analysis (<10 FPS): YOLOv8l** @ 1280 (15.9% mAP50, 8 FPS) for forensic investigations.
- **Avoid:** YOLOv8x @ 1280 unless the dataset contains more than 50 K images or GroupNorm replaces BatchNorm.

Importantly, these guidelines are quantitatively grounded in systematic evaluation rather than anecdotal experience. By grounding each recommendation in measured speed–accuracy trade-offs and linking them to concrete deployment scenarios, the guidance enables practitioners to make informed and evidence-based model selection decisions.

Practitioners can directly apply these recommendations to real-world drone detection systems, thereby reducing the trial-and-error development time from weeks to days. Furthermore, the guidelines establish baselines against which future architectural innovations can be evaluated, facilitating fair and consistent comparisons in future research.

Although the best result achieved in this study (15.9% mAP50, YOLOv8l) does not reach SOTA performance (YOLO-HV: 38.1%), this outcome is intentional. In particular, the aim of the present study was to establish the upper bound of what standard YOLOv8 architectures can achieve before introducing additional complexity. As detailed in our gap decomposition, the 22.2% absolute

 doi:10.20944/preprints202603.0990.v1

performance gap is primarily driven by highly customized modules (e.g., hybrid CNN-Transformer backbones and attention-driven dynamic heads) and dataset-specific methodological optimizations. However, our findings also demonstrate that straightforward strategies—such as introducing a P2 layer (+6% mAP50) or increasing input resolution (+25% mAP50)—can serve as highly effective alternatives to complex architectural modifications. This systematic baseline establishment enables informed decision-making: practitioners can choose between (a) deploying standard YOLOv8 to benefit from zero implementation complexity and favorable inference speeds, or (b) investing substantial engineering effort to implement specialized components for an additional ~22% absolute improvement toward SOTA performance.

5. Conclusions

This paper presented a systematic empirical study of the YOLOv8 family for small-object detection in drone imagery, addressing fundamental questions in model selection that have been overlooked in prior studies that focused primarily on architectural innovation.

5.1. Primary Findings and Implications

Model Capacity Has an Optimal Threshold: The comprehensive evaluation results revealed that YOLOv8l (43.6M parameters) represents the optimal model configuration for the VisDrone dataset with 6,471 training images, achieving 15.9% mAP50. Critically, exceeding this threshold with the oversized YOLOv8x (68.1M parameters) leads to a severe performance collapse (7.32% mAP50, representing a 54% relative drop). This degradation is primarily attributed to overfitting and the batch normalization instability inherent to training with a batch size of 1. This finding provides quantitative evidence for an optimal dataset-size-to-model-capacity ratio, suggesting that approximately 150 images per million parameters represents a practical threshold for data-constrained drone-based object detection tasks.

Input Resolution Dominates Architectural Complexity: Controlled comparisons demonstrated that input resolution scaling (640 1280) yields a fourfold greater performance improvement (+25% mAP50) than structural modifications, such as the addition of a P2 detection layer (+6% mAP50). For practitioners, this implies a clear prioritization strategy: computational resources are more effectively allocated to increasing the input spatial resolution before pursuing complex architectural enhancements. This finding challenges the prevailing trend of adopting increasingly sophisticated architectures without first exhausting simpler, high-impact scaling alternatives.

Establishing Realistic Baselines for Future Work: While the best result obtained in this study (15.9% mAP50) falls short of the SOTA performance reported for YOLO-HV (38.1% mAP50), the sources of this gap have been well-characterized. As demonstrated in our detailed gap analysis, this 22.2% absolute difference is driven by a combination of highly customized modules—such as the NextViT backbone (+3.1% mAP), the attention-driven DyHead (+4.1% mAP), and CARAFE upsampling (+0.6% mAP)—alongside dataset-specific methodological optimizations [8]. This decomposition provides a clear and actionable roadmap for achieving incremental performance improvements.

Importantly, the 15.9% mAP50 result achieved by the YOLOv8l model establishes the exact performance limit that a conventional, general-purpose YOLO architecture can reach on extreme aerial datasets without any domain-specific custom modifications. This serves as a critical baseline that has not been previously quantified. Given this baseline, researchers and practitioners can make informed trade-offs: either deploying standard YOLOv8 models for rapid, real-time applications with zero implementation complexity, or investing substantial engineering effort into specialized components to bridge the ~22% absolute performance gap toward SOTA models.

5.2. Practical Impact of Present Findings

 doi:10.20944/preprints202603.0990.v1

The present findings provide clear guidance for practical deployment. For rapid prototyping, YOLOv8l @ 640 offers a strong speed–accuracy balance (~30 FPS, 12.7% mAP50), while accuracy-focused offline analysis benefits most from YOLOv8l @ 1280, which achieves the highest performance obtainable with standard YOLOv8 components (15.9% mAP50). Larger models require caution: specifically, YOLOv8x should be avoided unless the dataset is sufficiently large (e.g., 50K images) or BatchNorm is replaced with GroupNorm to ensure stable training. Finally, for optimal computational resource allocation, simply increasing the input resolution yields far greater performance returns than integrating complex architectural additions, such as CARAFE, attention mechanisms, or P2 layers.

5.3. Reframing Absolute Performance

The VisDrone benchmark presents extreme difficulty, with an average object size of only 22 pixels (compared with ~40 pixels for COCO), and certain categories such as bicycles averaging merely 12–15 pixels. Consequently, even highly specialized SOTA architectures such as YOLO-HV achieve only 38.1% mAP50 on this dataset [36], a stark contrast to the >50% strict mAP (AP@[0.5:0.95]) typically reported on standard benchmarks like COCO. [36]

Within this challenging context, the 15.9% mAP50 achieved by the best YOLOv8 configuration, although modest in absolute terms, represents a 323% relative improvement over the YOLOv5s baseline (3.76%), demonstrating clear architectural advances. It also reaches approximately 42% of the specialized SOTA performance (15.9% vs. 38.1%) using only standard YOLOv8 components, establishing an important baseline for incremental innovation. Notably, substantial gains are observed for the smallest categories: bicycle detection improves from 0.98% to 3.93% (+301%), and pedestrian detection from 1.02% to 9.26% (+808%).

More importantly, the present results help shift the research focus away from absolute accuracy toward a more systematic understanding of the small-object detection problem. They clarify why the detection performance saturates (e.g., an information bottleneck at < 20 pixels), what contributes most effectively to performance improvement (e.g., input resolution scaling over architectural modifications), and what fails under constrained conditions (e.g., oversized models trained on limited data). This systematic understanding provides significant practical value independent of achieving a SOTA performance.

5.4. Limitations and Future Directions

The present evaluations focused intentionally on standard YOLOv8 architectures without implementing highly customized components such as NextViT, CARAFE, and DyHead. This approach allowed the establishment of clear baselines but necessarily limited the absolute performance attainable. In addition, all the experiments were conducted on a single dataset (VisDrone). Although this dataset is the primary benchmark for drone-based small-object detection, validation across additional datasets would strengthen the generalizability of the present findings and recommendations. Finally, the observed failure of YOLOv8x under a batch size of 1 suggests that alternative normalization strategies, such as GroupNorm, may enable stable training for larger models. However, this possibility remains unexplored in the current study.

The present findings suggest several high-potential directions for future research. First, the performance gap analysis suggests that incrementally integrating specific, high-yield modules—such as lightweight attention mechanisms (e.g., DyHead, +4.1% mAP) followed by content-aware upsampling (e.g., CARAFE, +0.6% mAP)—may push the detection performance into the 20-25% mAP50 range without incurring the massive computational overhead of full Transformer backbones. Second, evaluating GroupNorm or LayerNorm for larger models may allow stable training under extreme small-batch constraints, potentially enabling YOLOv8x to realize its theoretical capacity. Third, extending the evaluation experiments to additional drone benchmarks, such as UAVDT (for temporal tracking consistency) and DOTA (for oriented object detection), may help determine whether the identified scaling behaviors and capacity thresholds generalize beyond the VisDrone

 doi:10.20944/preprints202603.0990.v1

dataset. Finally, exploring adaptive resolution pipelines or patch-based processing (e.g., SAHI or SNIPER) may offer 1280-level spatial detail at a significantly lower computational cost, thereby improving both the training and deployment efficiencies.

5.5. Concluding Remarks

Rather than proposing another architectural modification, this study has provided the systematic evaluation infrastructure and empirical understanding necessary for informed model selection in drone-based detection. The main contributions of this study—namely**,** identifying optimal model capacity thresholds, quantifying resolution-versus-architecture trade-offs, documenting failure modes, and providing evidence-based deployment guidelines—provide a solid foundation for both practical deployment and future research.

Moving beyond the sole pursuit of absolute accuracy, this study clarifies how small-object detection performance scales with architectural complexity. In particular, we demonstrate that standard architectures achieve a baseline of approximately 15.9% mAP50, simple scaling strategies can improve performance into the 20–25% mAP50 range, and sophisticated custom components are required to reach the 30–40% mAP50 tier. Armed with this understanding, practitioners can make rational cost-benefit decisions for real-world deployments, and researchers can design experiments with a clear sense of the baseline they must surpass and by what margin.

Author Contributions: Conceptualization, J. C. Juang and C. M. Liu; methodology, J. C. Juang and C. M. Liu; software, C. M. Liu; data curation, C. M. Liu; writing—original draft preparation, C. M. Liu; writing—review and editing, J. C. Juang; supervision, J. C. Juang. All authors have read and agreed to the published version of the manuscript.

Acknowledgments: This work was supported by the National Science Technology Council (NSTC), Taiwan, under grant 114-2218-E-006 -018 -.

References

1. Du, D.; Zhu, P.; Wen, L.; Bian, X.; Lin, H.; Hu, Q.; Peng, T.; Zheng, J.; Wang, X.; Zhang, Y.; et al. VisDrone-DET2019: The Vision Meets Drone Object Detection in Image Challenge Results. In Proceedings of the Proceedings of the IEEE/CVF international conference on computer vision workshops; 2019; pp. 0–0. https://openaccess.thecvf.com/content_ICCVW_2019/html/VISDrone/Du_VisDrone-DET2019_The_Vision_Meets_Drone_Object_Detection_in_Image_Challenge_ICCVW_2019_paper.html
2. Lin, T.-Y.; Maire, M.; Belongie, S.J.; Bourdev, L.D.; Girshick, R.B.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. *CoRR* **2014**, *abs/1405.0312*. https://link.springer.com/chapter/10.1007/978-3-319-10602-1_48
3. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition; 2016; pp. 779–788. https://ieeexplore.ieee.org/document/7780460
4. Jocher, G. YOLOv5 2020. https://github.com/ultralytics/yolov5
5. Jocher, G.; Chaurasia, A.; Qiu, J. YOLOv8 2023. https://github.com/ultralytics/ultralytics
6. Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition; 2017; pp. 2117–2125. https://ieeexplore.ieee.org/document/8099589
7. Woo, S.; Park, J.; Lee, J.-Y.; Kweon, I.S. Cbam: Convolutional Block Attention Module. In Proceedings of the Proceedings of the European conference on computer vision (ECCV); 2018; pp. 3–19. https://openaccess.thecvf.com/content_ECCV_2018/html/Sanghyun_Woo_Convolutional_Block_Attention_ECCV_2018_paper.html
8. Xu, S.; Zhang, M.; Chen, J.; Zhong, Y. YOLO-HyperVision: A Vision Transformer Backbone-Based Enhancement of YOLOv5 for Detection of Dynamic Traffic Information. *Egypt. Inform. J.* **2024**, *27*, 100523. https://www.sciencedirect.com/science/article/pii/S1110866524000860

 doi:10.20944/preprints202603.0990.v1

9. Li, J.; Xia, X.; Li, W.; Li, H.; Wang, X.; Xiao, X.; Wang, R.; Zheng, M.; Pan, X. Next-Vit: Next Generation Vision Transformer for Efficient Deployment in Realistic Industrial Scenarios. *ArXiv Prepr. ArXiv220705501* **2022**. https://arxiv.org/abs/2207.05501
10. Wang, J.; Chen, K.; Xu, R.; Liu, Z.; Loy, C.C.; Lin, D. Carafe: Content-Aware Reassembly of Features. In Proceedings of the Proceedings of the IEEE/CVF international conference on computer vision; 2019; pp. 3007–3016. https://ieeexplore.ieee.org/document/9010830
11. Lin, T.-Y.; Goyal, P.; Girshick, R.B.; He, K.; Dollár, P. Focal Loss for Dense Object Detection. *CoRR* **2017**, *abs/1708.02002*. https://ieeexplore.ieee.org/document/8417976
12. Girshick, R.B.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. *CoRR* **2013**, *abs/1311.2524*. https://ieeexplore.ieee.org/document/6909475
13. Girshick, R. Fast R-Cnn. In Proceedings of the Proceedings of the IEEE international conference on computer vision; 2015; pp. 1440–1448. https://ieeexplore.ieee.org/document/7410526
14. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1137–1149. https://ieeexplore.ieee.org/document/7485869
15. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.E.; Fu, C.-Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. *CoRR* **2015**, *abs/1512.02325*. https://link.springer.com/chapter/10.1007/978-3-319-46448-0_2
16. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection. *CoRR* **2019**, *abs/1911.09070*. https://ieeexplore.ieee.org/document/9156454
17. Zhang, S.; Chi, C.; Yao, Y.; Lei, Z.; Li, S.Z. Bridging the Gap Between Anchor-Based and Anchor-Free Detection via Adaptive Training Sample Selection. *CoRR* **2019**, *abs/1912.02424*. https://ieeexplore.ieee.org/document/9156746
18. Lee, N.; Ajanthan, T.; Torr, P.H.S. SNIP: Single-Shot Network Pruning Based on Connection Sensitivity. *CoRR* **2018**, *abs/1810.02340*. https://arxiv.org/abs/1810.02340
19. Wang, Q.; Wu, B.; Zhu, P.; Li, P.; Zuo, W.; Hu, Q. ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. *CoRR* **2019**, *abs/1910.03151*. https://ieeexplore.ieee.org/document/9156697
20. Wang, X.; Girshick, R.B.; Gupta, A.; He, K. Non-Local Neural Networks. *CoRR* **2017**, *abs/1711.07971*. https://ieeexplore.ieee.org/document/8578911
21. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. *CoRR* **2021**, *abs/2103.14030*. https://ieeexplore.ieee.org/document/9710580
22. Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; Zhang, G.; Hu, H.; Wei, Y. Deformable Convolutional Networks. *CoRR* **2017**, *abs/1703.06211*. https://ieeexplore.ieee.org/document/8237351
23. Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; Dai, J. Deformable DETR: Deformable Transformers for End-to-End Object Detection. *CoRR* **2020**, *abs/2010.04159*. https://arxiv.org/abs/2010.04159
24. Liu, W.; Lu, H.; Fu, H.; Cao, Z. Learning to Upsample by Learning to Sample 2023. https://ieeexplore.ieee.org/document/10377871
25. Ghiasi, G.; Cui, Y.; Srinivas, A.; Qian, R.; Lin, T.-Y.; Cubuk, E.D.; Le, Q.V.; Zoph, B. Simple Copy-Paste Is a Strong Data Augmentation Method for Instance Segmentation. *CoRR* **2020**, *abs/2012.07177*. https://ieeexplore.ieee.org/document/9578639
26. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *CoRR* **2020**, *abs/2004.10934*. https://arxiv.org/abs/2004.10934
27. Du, D.; Qi, Y.; Yu, H.; Yang, Y.; Duan, K.; Li, G.; Zhang, W.; Huang, Q.; Tian, Q. The Unmanned Aerial Vehicle Benchmark: Object Detection and Tracking. *CoRR* **2018**, *abs/1804.00518*. https://openaccess.thecvf.com/content_ECCV_2018/html/Dawei_Du_The_Unmanned_Aerial_ECCV_2018_paper.html
28. Xia, G.-S.; Bai, X.; Ding, J.; Zhu, Z.; Belongie, S.J.; Luo, J.; Datcu, M.; Pelillo, M.; Zhang, L. DOTA: A Large-Scale Dataset for Object Detection in Aerial Images. *CoRR* **2017**, *abs/1711.10398*. https://ieeexplore.ieee.org/document/8578516
29. Cai, Z.; Vasconcelos, N. Cascade R-CNN: Delving into High Quality Object Detection. *CoRR* **2017**, *abs/1712.00726*. https://ieeexplore.ieee.org/document/8578742

 doi:10.20944/preprints202603.0990.v1

30. Zhu, X.; Lyu, S.; Wang, X.; Zhao, Q. TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-Captured Scenarios. *CoRR* **2021**, *abs/2108.11539*. https://ieeexplore.ieee.org/document/9607487
31. Singh, B.; Najibi, M.; Davis, L.S. Sniper: Efficient Multi-Scale Training. *Adv. Neural Inf. Process. Syst.* **2018**, *31*. https://proceedings.neurips.cc/paper/2018/hash/166cee72e93a992007a89b39eb29628b-Abstract.html
32. Liao, J.; Tian, H. Cluster-NMS: Improving Crowded Object Detection through Clustering Pattern. *Signal Image Video Process.* **2025**, *19*, 1–8. https://link.springer.com/article/10.1007/s11760-025-04374-3
33. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Proceedings of the International conference on machine learning; pmlr, 2015; pp. 448–456. https://proceedings.mlr.press/v37/ioffe15.html
34. Santurkar, S.; Tsipras, D.; Ilyas, A.; Madry, A. How Does Batch Normalization Help Optimization? *Adv. Neural Inf. Process. Syst.* **2018**, *31*. https://proceedings.neurips.cc/paper/2018/hash/905056c1ac1dad141560467e0a99e1cf-Abstract.html
35. Wu, Y.; He, K. Group Normalization. In Proceedings of the Proceedings of the European conference on computer vision (ECCV); 2018; pp. 3–19. https://openaccess.thecvf.com/content_ECCV_2018/html/Yuxin_Wu_Group_Normalization_ECCV_2018_paper.html
36. Scabini, L.; Sacilotti, A.; Zielinski, K.M.; Ribas, L.C.; De Baets, B.; Bruno, O.M. A Comparative Survey of Vision Transformers for Feature Extraction in Texture Analysis. *J. Imaging* **2025**, *11*, 304. https://www.mdpi.com/2313-433X/11/9/304
37. Dai, X.; Chen, Y.; Xiao, B.; Chen, D.; Liu, M.; Yuan, L.; Zhang, L. Dynamic Head: Unifying Object Detection Heads with Attentions. In Proceedings of the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition; 2021; pp. 7373–7382. https://ieeexplore.ieee.org/document/9577765