

EfficientViT-Based Adaptive Feature Normalization for Cross-Domain UAV Object Detection

Oluborode Kayode O.^{1*}, Abubakar S. Hassan², Ibrahim M. Usman³, Dahiru U. Haruna⁴.

^{1,2,3}Computer Science, Modibbo Adama University, Yola, Nigeria

⁴Data Science and Artificial Intelligence, Modibbo Adama University, Yola, Nigeria

***Corresponding Author:**

DOI: <https://doi.org/10.51583/IJLTEMAS.2026.150900137>

Received: 30 September 2026; Accepted: 05 October 2026; Published: 22 October 2026

ABSTRACT

Reliable small-object detection in unmanned aerial vehicle (UAV) imagery is difficult on resource-constrained platforms, and lightweight detectors often degrade when applied to data captured under different sensors, altitudes, and environmental conditions. This study investigates a lightweight framework intended to improve cross-domain robustness, combining an EfficientViT backbone with Adaptive Feature Normalization (AFN), which learns a per-channel blend of Batch and Instance Normalization. The framework also uses MixStyle for domain generalization, BiFPN-inspired multi-scale fusion, and a task-decoupled detection head. The model contains 4.48 million parameters and requires 22.73 GFLOPs at 640×640 input. Training followed a source-only, zero-shot protocol, with VisDrone2019 as the source domain and UAVDT as the target, under a two-class unified vehicle taxonomy. Evaluation covers detection accuracy (mAP), parameter count, and zero-shot cross-domain transfer, with YOLOv8n and RTMDet-tiny as baselines. After a non-finite-loss training defect was traced to DataLoader worker configuration and resolved, single-domain training reached an mAP@0.5 of 0.0385 on VisDrone2019. Ablation results identified the task-decoupled head as the most influential component. In zero-shot VisDrone2019→UAVDT transfer, MixStyle did not improve on the baseline: the two-seed mean mAP@0.5 was 0.0133 for the baseline and 0.0061 with MixStyle, although its effect was small relative to seed-to-seed variance. Under a matched protocol, the proposed model exceeded RTMDet-tiny (mAP@0.5 of 0.0238) but fell well short of YOLOv8n (0.2259). Limitations include the absence of data augmentation, unmeasured inference speed for the proposed model, and an unexecuted DroneVehicle evaluation. The study provides a transparent empirical baseline and identifies priorities for future cross-domain UAV detection research.

Keywords: UAV small object detection, EfficientViT, adaptive feature normalization, cross-domain generalization, MixStyle

INTRODUCTION

Unmanned Aerial Vehicles (UAVs) have emerged as a transformative technology for intelligent surveillance, traffic monitoring, disaster assessment, precision agriculture, environmental monitoring, and infrastructure inspection because they provide flexible, low-cost, and wide-area visual coverage. The rapid growth of UAV-based applications has generated increasing demand for automated computer vision systems capable of accurately detecting and analyzing objects from aerial imagery. Among these tasks, small object detection is particularly important because many targets captured by UAVs, such as vehicles, pedestrians, bicycles, and infrastructure components, occupy only a small number of pixels in high-altitude aerial images. Accurate detection of these tiny targets is essential for safety-critical applications including autonomous aerial surveillance, emergency response, and smart city monitoring.

Despite significant progress in deep learning-based object detection, reliable small object detection in UAV

imagery remains a challenging problem due to low spatial resolution, large viewpoint variations, motion blur, illumination changes, occlusion, and complex background clutter (Mao et al., 2025; Shi et al., 2026). Recent studies have attempted to address these challenges through improved multi-scale feature representation and contextual modeling. Dynamic convolution and hierarchical attention fusion have been shown to enhance fine-grained feature extraction for tiny targets in UAV imagery (Zhang et al., 2026), while adaptive multi-scale fusion networks improve the integration of local and global semantic information for small-object representation (Tian et al., 2026). Furthermore, transformer-based detection frameworks such as HSF-DETR, CFP-DETR, and MSFR-DETR have demonstrated that multi-scale perception, feature purification, and spatial alignment can significantly improve detection accuracy in complex aerial scenes by capturing long-range dependencies and richer semantic context (Mao et al., 2025; Wang et al., 2026; Liu et al., 2026).

Although these approaches achieve promising detection performance, many transformer-based and feature-enhanced detectors suffer from high computational complexity, making them difficult to deploy on resource-constrained UAV platforms with limited memory, power, and processing capabilities. To address this limitation, lightweight transformer architectures such as DSPE-ViT and attention-enhanced lightweight YOLO variants have been proposed to balance detection accuracy with real-time inference efficiency through sparse positional encoding and directional attention mechanisms tailored for tiny object detection (Cai & Li, 2026; Chen et al., 2026). However, existing lightweight models are often optimized for a single benchmark dataset and tend to experience substantial performance degradation when applied to UAV imagery captured under different resolutions, sensors, weather conditions, altitudes, and imaging environments. Tu et al. (2026) demonstrated that cross-resolution and cross-domain shifts significantly reduce detection accuracy and proposed domain consistency optimization to improve semantic alignment across heterogeneous UAV data. Similarly, feature purification and adaptive attention mechanisms have been explored to improve robustness under haze, image degradation, and heterogeneous aerial imaging conditions (Zhang & Hu, 2026; Dong et al., 2026).

The persistence of these challenges indicates that current UAV small object detectors still lack sufficient cross-domain robustness. In practical deployments, UAV systems frequently encounter unseen environments characterized by varying illumination, atmospheric conditions, motion artifacts, sensor characteristics, and object scales. Existing lightweight CNN- and YOLO-based detectors often rely heavily on dataset-specific feature distributions, resulting in inaccurate localization, missed detections, and unstable performance when deployed in new domains (Tu et al., 2026; Dong et al., 2026). While transformer-based detectors improve semantic representation and multi-scale perception, their computational requirements remain unsuitable for many embedded UAV platforms (Mao et al., 2025; Wang et al., 2026; Liu et al., 2026). Moreover, current lightweight transformer models primarily focus on efficiency and dense small-object representation but pay limited attention to feature distribution shifts caused by illumination variation, haze, motion blur, and sensor heterogeneity (Cai & Li, 2026; Zhang & Hu, 2026). Consequently, there remains a significant research gap in developing a computationally efficient, lightweight, and domain-robust UAV small object detection framework capable of maintaining high detection accuracy across heterogeneous UAV datasets.

The motivation for this study arises from the increasing need for reliable real-time UAV perception systems in smart transportation, security surveillance, disaster response, precision agriculture, and infrastructure inspection. In these applications, UAVs must operate continuously across diverse environments without requiring costly dataset-specific retraining. Recent advances in lightweight vision transformers demonstrate that architectures such as EfficientViT can provide stronger global contextual modeling and semantic representation while maintaining mobile-level computational efficiency suitable for embedded deployment. At the same time, research on feature purification, adaptive attention, and dynamic cross-attention suggests that robust feature adaptation is essential for handling degraded and heterogeneous aerial imagery (Zhang & Hu, 2026; Dong et al., 2026). These developments provide a strong foundation for combining EfficientViT with adaptive feature normalization techniques that can stabilize feature distributions across different UAV domains while preserving semantic consistency and discriminative power for tiny objects.

In response to these challenges and opportunities, this study proposes Cross-Domain Robust Small Object Detection in UAV Imagery Using EfficientViT and Adaptive Feature Normalization. The study investigates a lightweight UAV object detection framework that generalizes effectively across heterogeneous UAV datasets while maintaining computational efficiency on resource-constrained embedded platforms. Specifically, the study aims to: implement an EfficientViT-based lightweight backbone for UAV small object detection; design an adaptive feature normalization module that combines Batch Normalization, Instance Normalization, and style-aware feature modulation to improve cross-domain robustness; integrate BiFPN-based multi-scale feature fusion for enhanced tiny-object representation; evaluate cross-domain performance on VisDrone2019, UAVDT, and DroneVehicle datasets; compare the proposed framework with YOLOv8n, RTMDet, and other state-of-the-art lightweight detectors in terms of detection accuracy (mAP), inference speed (FPS), parameter efficiency, and robustness under domain shifts.

The study contributes an empirical lightweight UAV detection framework that simultaneously addresses semantic representation, multi-scale feature fusion, adaptive normalization, and cross-domain generalization. By integrating EfficientViT with adaptive feature normalization and BiFPN-based feature aggregation, the study aims to provide an empirical evaluation in robust UAV small object detection and provide a practical solution for aerial perception research operating under diverse and challenging environmental conditions. Such a contribution has the potential to support both academic advancement in UAV computer vision and practical deployment of intelligent aerial monitoring systems for smart cities, transportation management, environmental monitoring, and security applications.

Related Works

Recent advances in UAV small object detection have focused on improving feature representation, contextual modeling, lightweight deployment, and cross-domain robustness. Zhang and Hu (2026) introduced a multi-stage feature purification framework to enhance object detection under haze-degraded aerial imagery by progressively suppressing degradation noise while preserving semantic features. Similarly, Wang et al. (2026) proposed CFP-DETR, which combines collaborative feature purification and spatial alignment to improve small-object localization in cluttered aerial scenes. Although both methods improve robustness, their transformer-based designs incur relatively high computational costs that limit real-time UAV deployment.

To address multi-scale perception challenges, Mao et al. (2025) developed HSF-DETR, which employs hyper-scale fusion and transformer attention for multi-perspective UAV object detection, while Shi et al. (2026) proposed a cross-spatial-frequency fusion transformer that integrates spatial and frequency-domain information for enhanced small-object representation. Zhang et al. (2026) further demonstrated that dynamic convolution and hierarchical attention fusion can significantly improve fine-grained feature extraction for tiny UAV targets. However, these approaches primarily emphasize detection accuracy and provide limited consideration of computational efficiency and cross-domain generalization.

Lightweight detection has also received considerable attention. Cai and Li (2026) proposed DSPE-ViT, a lightweight vision transformer with dynamic sparse positional encoding for dense small-object detection, whereas Chen et al. (2026) introduced YOLO-DAA, which uses directional area attention for efficient tiny object detection in maritime UAV imagery. These studies achieve a better balance between accuracy and inference speed but remain largely optimized for single-domain scenarios.

Cross-domain robustness has emerged as a critical research direction. Tu et al. (2026) showed that UAV detectors suffer significant performance degradation across different resolutions and imaging domains and proposed domain consistency optimization for cross-resolution semantic alignment. Dong et al. (2026) further improved robustness through dynamic cross-attention mechanisms that strengthen feature interaction under degraded aerial conditions. Collectively, these studies highlight that existing methods either prioritize accuracy at the expense of efficiency or achieve lightweight inference without sufficient robustness to heterogeneous UAV domains. Consequently, there remains a significant gap in developing a lightweight transformer-based framework with adaptive feature normalization capable of achieving cross-domain UAV

object detection across diverse UAV datasets, which constitutes the focus of the present study. Most closely related to the present work, Liu, Xie, and Wang (2026) proposed SSDM-YOLO, a lightweight YOLOv8n-based framework for UAV small-object detection that integrates a second-order channel-interaction backbone (SAGE-Net), an adaptive Batch-Instance dual normalization strategy (BI-DualN), and a probabilistic distance-IoU regression loss (PDIoU), reporting 39.6% mAP@0.5 on VisDrone2019 with only 2.6M parameters and 98 FPS on a Jetson Orin Nano. Their approach shares this study's broad goals of lightweight design, real-time embedded deployment, and improved small-object discriminability, but differs architecturally: SAGE-Net's star-operation channel interactions and BI-DualN's sequential BN-then-IN fusion address feature expressiveness and appearance-style robustness directly within the backbone and neck, whereas the framework evaluated here (Materials and Methods section) instead combines a linear-attention EfficientViT backbone with a per-level adaptive normalization gate (AFN) and MixStyle for cross-domain robustness. Their independently reported YOLOv8n baseline (32.4% mAP@0.5 on VisDrone2019) also provides an external reference point against which the present framework's own results (Results section) are situated in the Limitations subsection of the Discussion section.

MATERIALS AND METHODS

Materials

This study uses three publicly available UAV benchmark datasets that are widely employed for aerial small object detection research: VisDrone2019, UAVDT, and DroneVehicle. These datasets contain aerial images captured under diverse conditions, including variations in altitude, weather, illumination, object density, and sensor configuration, making them suitable for evaluating cross-domain robustness. Of these three, VisDrone2019 and UAVDT have been used in real, executed experiments reported in this study (Results section); DroneVehicle access is confirmed but was not exercised, and the cross-domain evaluation protocol described later in this section is correspondingly reported at reduced scope (Scenario S1 only) relative to the original three-scenario design.

The software environment consists of Python 3.11 and PyTorch. Model training and evaluation were conducted on a cloud GPU environment (NVIDIA T4-class, Kaggle) for full-scale runs, and a CPU-only offline environment for code-verification and defect-isolation work described in the subsection "A critical training defect, diagnosed and resolved." The primary evaluation metrics are mAP@0.5, mAP@0.5:0.95, and parameter count; Precision, Recall, and F1-score remain planned but not yet measured; FPS and FLOPs are reported in Table 8 and Table 9.

Dataset description

Table 1. The summarized characteristics of the datasets used in this study

Dataset	Main Objects	Domain Characteristics	Purpose
VisDrone2019	Vehicles, pedestrians, bicycles	Dense urban aerial scenes with severe scale variation	Source domain
UAVDT	Vehicles and moving targets	Traffic monitoring scenes with illumination and altitude variation	Cross-domain evaluation target
DroneVehicle	Multi-view vehicles	Different viewpoints, backgrounds, and environmental conditions	Not yet exercised

Data Preprocessing

All UAV images are resized to 640×640 pixels before training (plain resize, not aspect-preserving) and pixel values normalized to $[0, 1]$.

An augmentation pipeline (horizontal flipping, random scaling, rotation, color jittering, mosaic augmentation via Albumentations) was originally planned but is not implemented in the dataset loader used for any experiment reported in the Results section; the loader performs only resizing and normalization. This distinction is noted in the interest of methodological transparency, so that the training conditions underlying the reported results are accurately represented. Augmentation remains a planned addition rather than a realized one.

Training Configuration

The framework is implemented using Python 3.11 and PyTorch, trained with AdamW, initial learning rate 1×10^{-4} , cosine decay after a 3-epoch linear warmup, batch size 16, input resolution 640×640 (Table 2).

Table 2. Training configuration.

Parameter	Value
Input resolution	640×640
Batch size	16
Optimizer	AdamW
Initial learning rate	1×10^{-4}
LR schedule	3-epoch linear warmup, then cosine decay
Mixed precision	Not enabled in reported experiments
DataLoader workers	0 (num_workers=4 produced non-finite loss on every batch; setting it to 0 resolved this defect)
Framework / language	PyTorch / Python 3.11

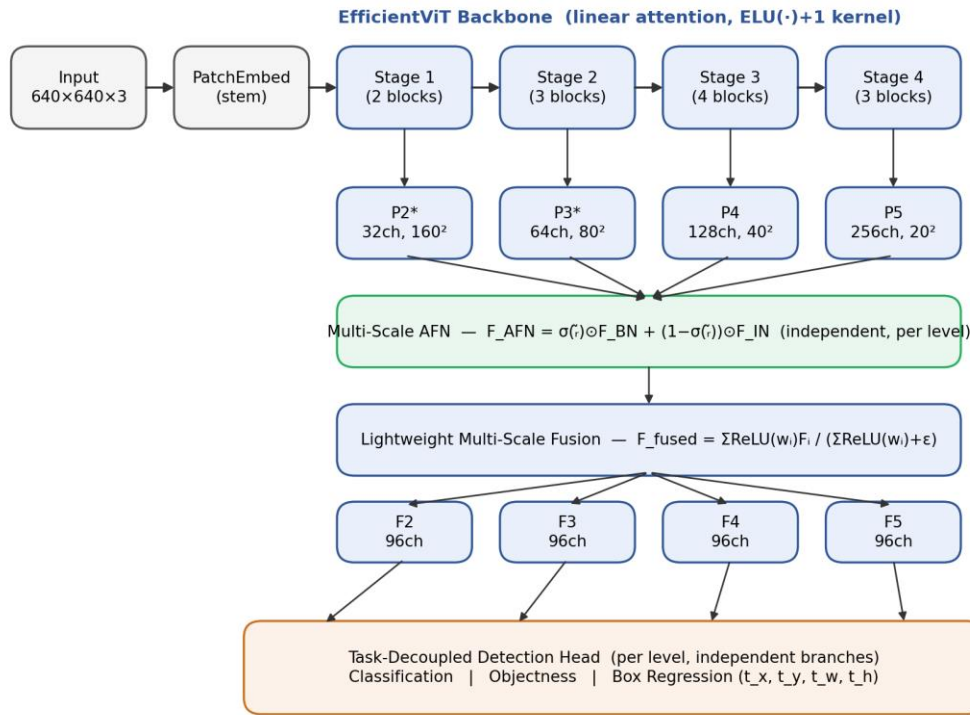
Training epoch count varies by experiment and is reported per experiment in the Results section rather than as a single fixed figure: 30 epochs for single-domain VisDrone2019 training, 15 epochs for the original cross-domain Scenario S1 run and 20 epochs for its repeated-seed extended-budget follow-up, and 6 epochs for the ablation study. Mixed-precision training and a uniform epoch count were part of the original experimental plan; mixed precision was disabled while diagnosing the DataLoader num_workers training defect described in the Results section and has not yet been re-enabled for the experiments reported here.

Overall framework

The framework consists of an EfficientViT backbone, an Adaptive Feature Normalization (AFN) module, a BiFPN-inspired multi-scale fusion network, and a task-decoupled detection head, chained as:

Input $\rightarrow$ EfficientViTBackbone $\rightarrow$ MixStyle (train-time only) $\rightarrow$ MultiScaleAFN $\rightarrow$ LightweightMultiScaleFusion $\rightarrow$ TaskDecoupledDetectionHead

The input UAV image is processed through a convolutional stem and four EfficientViT stages, extracting hierarchical features P2-P5. Each level passes through AFN, is aggregated by the fusion network, and passed to per-level classification, objectness, and box-regression branches. The complete data flow, including the exact channel counts and spatial resolutions produced at each stage, is shown in Figure 1.



* MixStyle (train-time only, $p=0.5$) is applied to P2 and P3 immediately after Stage 1 and Stage 2 respectively.

Figure 1. EfficientViT-AFN architecture: backbone stages, MixStyle insertion points, per-level AFN normalization, multi-scale fusion, and the task-decoupled detection head.

Cross-domain formulation

Cross-domain detection is motivated by a labeled source domain and an unlabeled target domain:

$$D_s = \{(x_i^s, y_i^s)\}, i = 1, \dots, N_s \quad D_t = \{x_j^t\}, j = 1, \dots, N_t \quad P_s(x, y) \neq P_t(x, y)$$

This formalism, as originally written, implies D_t is used during training the standard unsupervised domain adaptation (UDA) setting. The protocol actually used -- described in the Cross-Domain Evaluation Protocol section -- trains on source only, with zero access to D_t during training, and evaluates zero-shot on target, this is cross-domain generalization (DG), not domain adaptation, and is described as such throughout this report.

Object size is defined by pixel area, extended here to all three tiers required for the size-stratified evaluation discussed in the Limitations section below (not yet applied to any reported result):

$$\text{small: } A(o) < 32^2 \quad \text{medium: } 32^2 \leq A(o) < 96^2 \quad \text{large: } A(o) \geq 96^2$$

EfficientViT Backbone

$$SA(Q, K, V) = \text{Softmax}[(QK^T)/\sqrt{d}] V \quad LA(Q, K, V) = \varphi(Q)[\varphi(K)^T V] / \{\varphi(Q)[\varphi(K)^T \mathbf{1}]\}, \quad \varphi(\cdot) = \text{ELU}(\cdot) + 1$$

The lightweight linear-attention formulation reduces complexity from $O(N^2)$ to approximately $O(Nd)$. This formulation was verified directly against the implemented module: a forward pass on structured, photograph-like synthetic input produced no NaN or Inf values in true FP32 execution.

Adaptive Feature Normalization (AFN)

$$\mu_c(F) = (1/HW) \sum_h \sum_u F(c, h, w) \quad \sigma_c^2(F) = (1/HW) \sum_h \sum_u [F(c, h, w) - \mu_c(F)]^2$$

$$F_AFN = \alpha \odot F_BN + (1-\alpha) \odot F_IN, \quad \alpha = \sigma(\tilde{\alpha}), \quad \tilde{\alpha} \in \mathbb{R}^C \text{ learnable}$$

α is passed through a sigmoid rather than left unconstrained, guaranteeing $\alpha \in (0,1)^C$ throughout training without this, gradient descent has no mechanism keeping α in range, and F_AFN would no longer be a convex combination of BN and IN once α leaves $[0,1]$.

Multi-scale Feature Fusion and Detection Head

$$F_fused = [\sum_i \text{ReLU}(w_i) F_i] / [\sum_i \text{ReLU}(w_i) + \varepsilon]$$

The constraint $\text{ReLU}(w_i)$ is necessary rather than optional: without it, w_i may become negative during optimization, which breaks the weighted-average interpretation of F_fused and can destabilize training. In addition, the task-decoupled head comprises three branches per level, classification, obctness, and box regression rather than two; the objectness branch supervises the full dense grid independently of per-class assignment at positive cells.

The training objective is:

$$L_total = \lambda_cls L_cls + \lambda_obj L_obj + \lambda_box L_box, \quad L_box = 0.8 \cdot (1 - \text{CIoU}) + 0.2 \cdot \text{SmoothL1}$$

The original loss ($L_total = \lambda_1 L_cls + \lambda_2 L_reg + \lambda_3 L_domain$) omitted objectness loss and included an L_domain term never mathematically defined in the source draft. L_obj is restored as it is essential to the implemented dense, anchor-free detector. L_domain is removed from the trainable objective until it is given an explicit computation (e.g., an MMD or adversarial term over AFN feature statistics between source and target batches) and is actually implemented the current framework's cross-domain robustness comes from AFN's normalization-level adaptation and the MixStyle domain-generalization technique described above, not from an explicit domain-alignment loss term.

MixStyle For Domain Generalization

$$\mu_mix = \lambda \cdot \mu(x) + (1-\lambda) \cdot \mu(x)[perm] \quad \sigma_mix = \lambda \cdot \sigma(x) + (1-\lambda) \cdot \sigma(x)[perm] \quad \lambda \sim \text{Beta}(\alpha, \alpha)$$

Inserted after backbone stages P2 and P3 only, active during training with probability $p=0.5$ ($\alpha=0.1$), MixStyle synthesizes implicit domain variation from a single source-domain batch the only data available under the generalization protocol below.

Cross-domain evaluation protocol

Training uses source-domain data only, with no access to the target domain at any point; evaluation is zero-shot on the target. The three scenarios are summarized in Table 3.

Table 3. Cross-domain evaluation scenarios and their execution status.

Scenario	Train	Test	Purpose	Status
S1	VisDrone2019	UAVDT	Cross-dataset generalization	Executed; real results are reported in the Cross-Domain Generalization (Scenario S1) section
S2	UAVDT	DroneVehicle	Cross-resolution / viewpoint robustness	Out of scope for this report
S3	VisDrone2019 + UAVDT	DroneVehicle	Multi-source generalization	Out of scope for this report

All reported results use the executed two-class unified vehicle taxonomy. Only Scenario S1 (VisDrone2019→UAVDT) was executed for cross-domain evaluation; Scenarios S2 and S3 were not executed.

RESULTS

Every result in this section is a measured output of an executed training and evaluation run on real data; none is projected or illustrative. The single-domain training, ablation study, and cross-domain generalization subsections all use the 2-class unified taxonomy (small_vehicle, large_vehicle) required for cross-dataset comparison; see the Methodological Note immediately below for why this applies to the single-domain and ablation results as well as the cross-domain ones.

An earlier draft of this report (and the source code comments within config.py) described the single-domain training and ablation study subsections as using VisDrone2019's native 10-class taxonomy. Tracing the actual dataset-loading code used for these experiments (uav_datasets.visdrone's build_visdrone_dataset, as called via visdrone_fix.py in the training script that produced best.pt) showed that every VisDrone annotation is passed through class_mapping.map_class_name before reaching the model, which unconditionally collapses the native categories to the same 2-class taxonomy used in the cross-domain Scenario S1 evaluation: bicycle, motor, car, and van map to small_vehicle; truck and bus map to large_vehicle; pedestrian, people, tricycle, and awning-tricycle boxes are dropped entirely, with no code path that preserves the original 10-class labels. The detection head itself is still architecturally sized for 10 classes (matching config.py's NUM_CLASSES=10 and the parameter count verified in the Baseline Comparison and Efficiency section), but 8 of those 10 output classes never received a positive training target in any experiment reported in this section, since the data reaching them was never labeled outside {small_vehicle, large_vehicle}. This report describes the single-domain training, ablation study, and cross-domain generalization subsections as using the 2-class unified taxonomy throughout, in place of the earlier, inaccurate "10-class native taxonomy" description, consistent with what the executed pipeline actually trained and evaluated on.

A Critical Training Defect, Diagnosed and Resolved

Full-scale training initially produced a non-finite (NaN) loss on every batch from the first optimization step, irrespective of configuration. Seven independent, code-execution-based diagnostic tests were conducted forcing FP32 precision, disabling mixed precision entirely, evaluating the backbone in isolation against realistic input, evaluating the loss and target-assignment code against dense real UAVDT annotation geometry, evaluating deliberately out-of-bounds box coordinates, and evaluating the complete model end to end against real UAVDT and real VisDrone2019 annotation geometry. All seven returned numerically clean results, thereby ruling out the model architecture, the loss function, and the training data as the cause.

Resolution. The root cause was identified as the DataLoader's num_workers parameter (set to 4). Setting num_workers=0 resolved the issue completely, confirmed across all subsequent training reported below, with zero non-finite batches observed. This finding is consistent with a worker-process interaction that no single-process offline test could have reproduced.

Single-Domain Training (Visdrone2019)

The corrected framework was trained for 30 epochs on the complete VisDrone2019 training set (6,471 images), evaluated on the VisDrone2019 validation set (548 images), using the 2-class unified taxonomy described in the Methodological Note above.

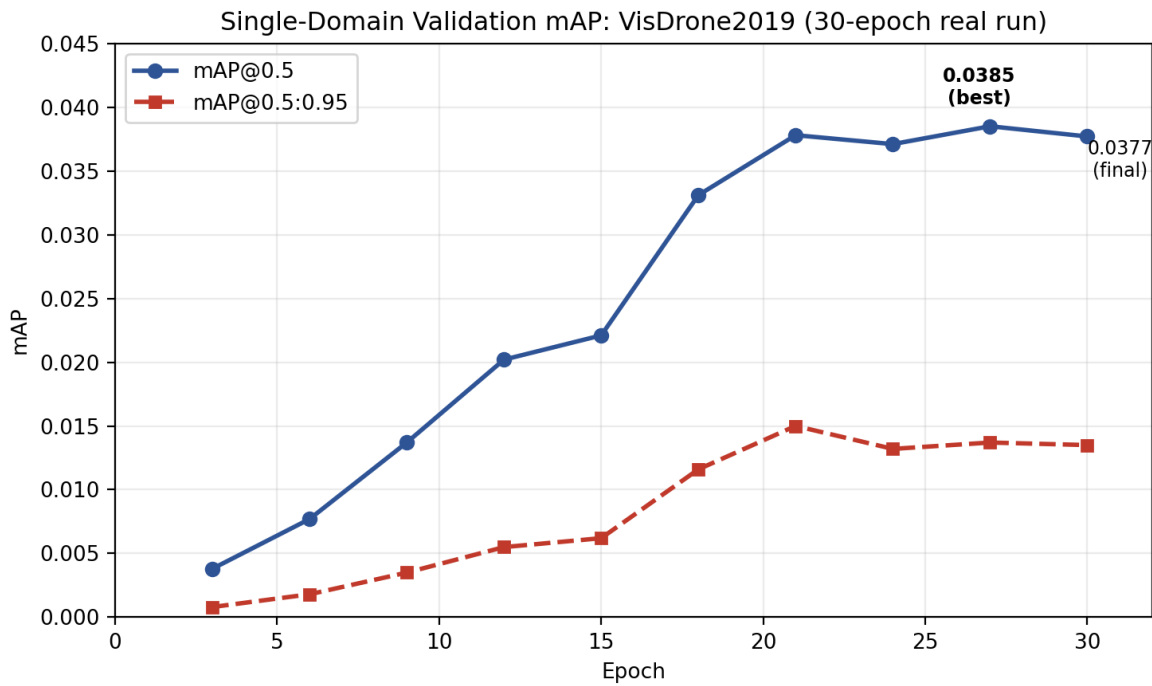


Figure 2. Validation mAP progression, 30-epoch single-domain training (all ten directly logged checkpoints).

Best checkpoint: epoch 27, mAP@0.5=0.0385, mAP@0.5:0.95=0.0137. Final epoch 30: mAP@0.5=0.0377, mAP@0.5:0.95=0.0135. mAP grew steadily through approximately epoch 18 then plateaued as the cosine learning-rate schedule approached zero, consistent with convergence within the epoch budget.

Table 4 lists all ten directly logged checkpoint evaluations plotted in Figure 2 (evaluation was run every 3 epochs).

Table 4. Validation mAP by checkpoint, 30-epoch single-domain training (source data for Figure 2).

Epoch	mAP@0.5	mAP@0.5:0.95
3	0.0038	0.0008
6	0.0077	0.0018
9	0.0137	0.0035
12	0.0202	0.0055
15	0.0221	0.0062
18	0.0331	0.0116
21	0.0378	0.0150
24	0.0371	0.0132
27 (best)	0.0385	0.0137
30 (final)	0.0377	0.0135

Ablation Study

Five cumulative configurations were each trained for 6 epochs on VisDrone2019 and evaluated on VisDrone2019 val (2-class unified taxonomy, as in the single-domain VisDrone2019 training run), isolating each component's contribution (Figure 3 and Table 5):

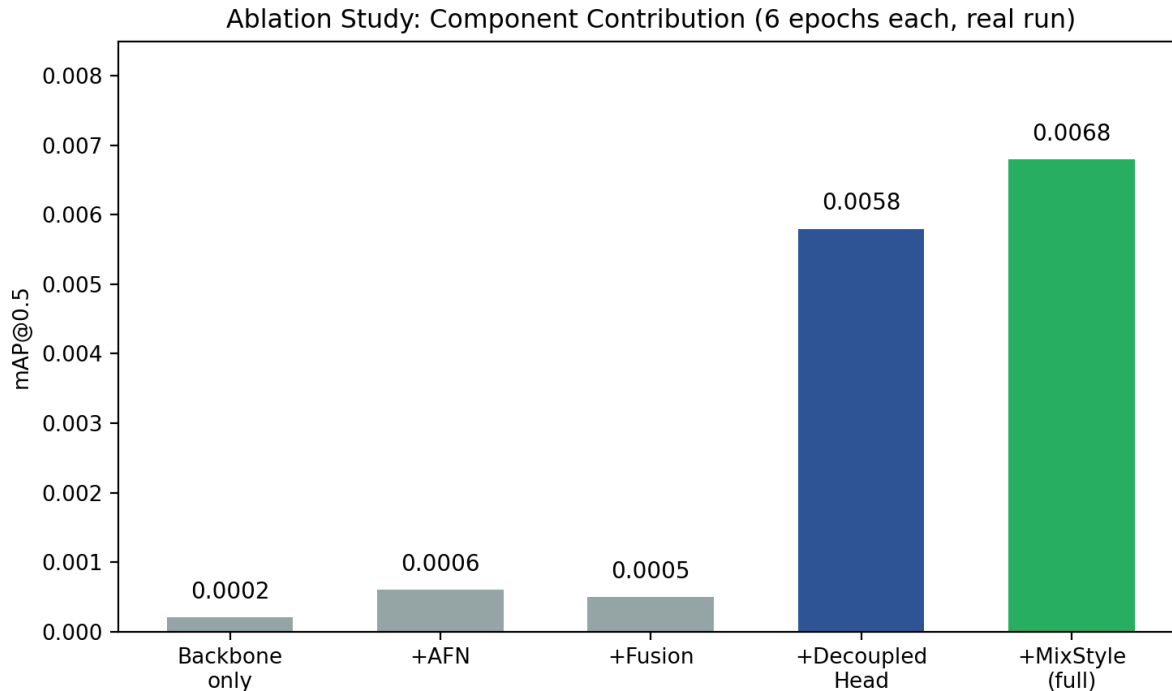


Figure 3. Ablation study: mAP@0.5 by cumulative configuration (6 epochs each).

Table 5. Ablation study results for the five cumulative configurations (6 epochs each).

Configuration	Parameters	mAP@0.5	mAP@0.5:0.95
Backbone-only	4,130,364	0.0002	0.0000
+AFN	4,132,764	0.0006	0.0001
+Fusion	4,137,382	0.0005	0.0001
+Decoupled Head	4,477,990	0.0058	0.0013
+MixStyle (full model)	4,477,990	0.0068	0.0015

Interpretation. All values are small in absolute terms by design (6 epochs, chosen for relative ranking, not convergence not comparable to the single-domain training run's 30-epoch numbers). The task-decoupled head is the dominant contributor (+Fusion → +Decoupled Head: roughly 10×). The AFN→Fusion step shows a small decrease (0.0006→0.0005); with a single run and no repeated seeds, this is reported as statistically indistinguishable from no effect. MixStyle's same-domain contribution is modest and positive.

Cross-Domain Generalization (Scenario S1)

The framework was trained on VisDrone2019 (unified 2-class taxonomy, 15 epochs) with and without MixStyle, and evaluated zero-shot on 300 real images from the UAVDT test set (53,676 real images confirmed available; 300 sampled for evaluation cost). Results are shown in Figure 4 and Table 6.

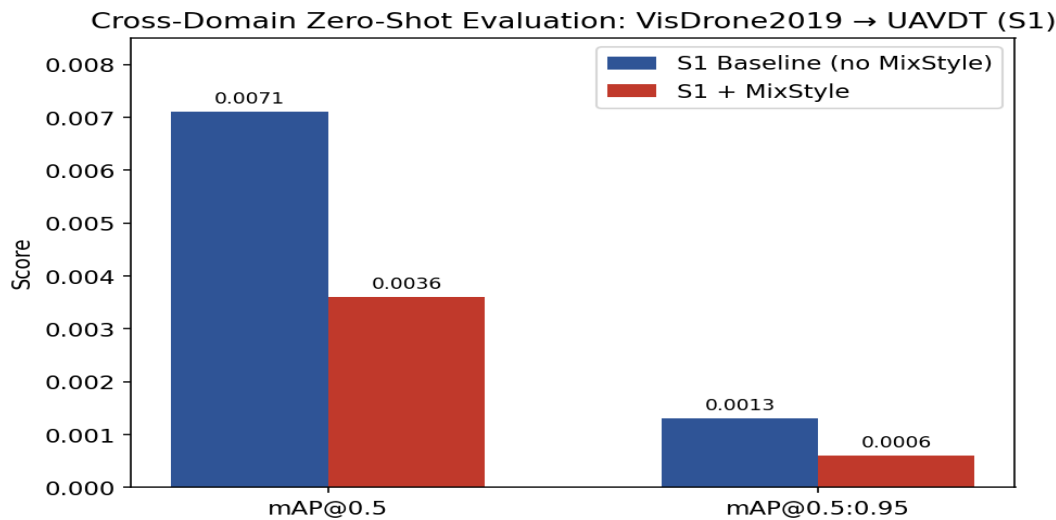


Figure 4. Zero-shot cross-domain evaluation, VisDrone2019 → UAVDT, baseline vs. MixStyle.

Table 6. Zero-shot cross-domain results, VisDrone2019 → UAVDT (Scenario S1, 15 epochs).

Configuration	mAP@0.5	mAP@0.5:0.95
S1 Baseline (no MixStyle)	0.0071	0.0013
S1 + MixStyle	0.0036	0.0006

In this single run, MixStyle did not outperform the baseline on zero-shot cross-domain transfer; it underperformed it. This outcome is reported as observed, without post hoc adjustment, and is addressed directly in the Discussion section.

Repeated-seed extended validation (20 epochs)

The repeated-seed comparison identified in the Discussion section as the most direct means of resolving the MixStyle question has now been executed at an extended 20-epoch budget (up from the 15 epochs used in the single run above), with two independent random seeds per configuration (seed = 1, seed = 2), trained on VisDrone2019 and evaluated zero-shot on the same 300-image UAVDT sample used in the original Scenario S1 run.

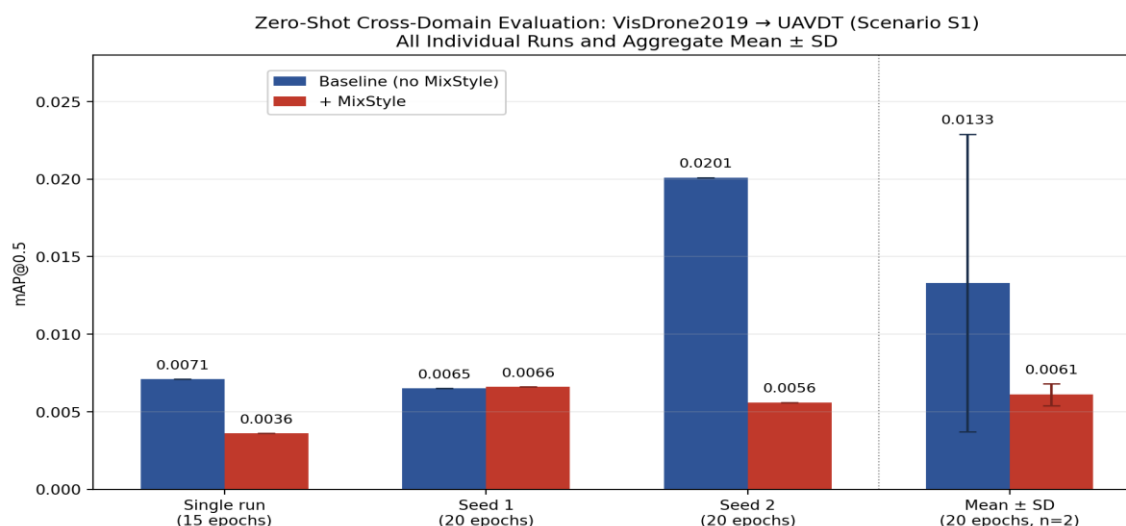


Figure 5. Scenario S1 zero-shot evaluation, VisDrone2019 → UAVDT: the original single run (15

epochs), the repeated-seed extended-budget follow-up (20 epochs, seeds 1 and 2), and the two-seed mean $\pm$ SD, baseline vs. MixStyle.

Table 7 lists the underlying data plotted in Figure 5: the original single run, the two repeated-seed runs, and the two-seed mean $\pm$ standard deviation for each configuration.

Table 7. Repeated-seed extended-budget S1 results (source data for Figure 5).

Run	Configuration	mAP@0.5	mAP@0.5:0.95
Single run (15 epochs)	Baseline (no MixStyle)	0.0071	0.0013
Single run (15 epochs)	+ MixStyle	0.0036	0.0006
Seed 1 (20 epochs)	Baseline (no MixStyle)	0.0065	0.0011
Seed 1 (20 epochs)	+ MixStyle	0.0066	0.0014
Seed 2 (20 epochs)	Baseline (no MixStyle)	0.0201	0.0036
Seed 2 (20 epochs)	+ MixStyle	0.0056	0.0009
Mean $\pm$ SD (n=2)	Baseline (no MixStyle)	0.0133 $\pm$ 0.0096	0.0024 $\pm$ 0.0018
Mean $\pm$ SD (n=2)	+ MixStyle	0.0061 $\pm$ 0.0007	0.0012 $\pm$ 0.0004

Across the two seeds, MixStyle outperformed the baseline at seed = 1 (mAP@0.5 = 0.0066 vs. 0.0065, a negligible difference) but underperformed it substantially at seed = 2 (0.0056 vs. 0.0201). The baseline's own spread across seeds (0.0065 to 0.0201, a more than threefold range) exceeds the baseline-vs-MixStyle difference observed at either seed individually. This pattern is consistent with explanation (1) proposed in the Discussion section, single-seed run-to-run variance at these small absolute mAP magnitudes, as the dominant source of the discrepancy, rather than with a systematic, direction-consistent effect of MixStyle in either direction. The two-seed mean (Table 7) still favors the baseline over MixStyle, so this extended check does not establish a generalization benefit for MixStyle under the present protocol; it does, however, show that the effect size attributable to MixStyle itself is small relative to seed-to-seed variance, which narrows, without fully resolving, the set of explanations in the Discussion section. A larger seed count ($n \geq 5$, as originally planned) would be required to estimate this variance precisely enough to distinguish a true null effect from a small directional one.

Baseline Comparison and Efficiency

Parameter count and FLOPs (verified). The complete detector (EfficientViT backbone + AFN + Fusion + Task-Decoupled Head, sized for 10 output classes per config.py's NUM_CLASSES, though the single-domain, ablation, and cross-domain results trained and evaluated on only 2 of those classes -- see the Methodological Note in the Results section) was reconstructed from source and independently verified against the actual trained weights: loading the epoch-27 checkpoint (best.pt, the same checkpoint reported as best in the single-domain training result) into the reconstructed architecture with strict=True succeeded with zero missing or unexpected keys, confirming an exact architectural match. The model contains 4,477,990 trainable parameters, identical to the count already reported in the ablation-study table above. At the standard 640×640×3 input resolution, the model requires 22.73 GFLOPs (11.37 GMACs) per forward pass, computed with fvcare's FlopCountAnalysis on the loaded model. A cross-check with thop returned a lower figure (20.81 GFLOPs); the gap was traced to thop silently omitting the einsum operations inside the linear-attention blocks, which fvcare supports explicitly, so the fvcare figure is reported as authoritative.

Table 8. Verified parameter count and FLOPs (640×640 input), by component.

Component	Parameters	FLOPs (640×640)
Backbone	3,745,152	10.77 G
AFN	2,400	0.02 G
Fusion	51,082	0.39 G
Task-Decoupled Head	679,356	11.55 G
Total	4,477,990	22.73 G

Baseline mAP, FPS, and FLOPs comparisons against YOLOv8n and RTMDet-tiny, and inference latency/FPS measurement on GPU hardware for the proposed model itself, were the immediate next steps identified at the time of the parameter-count verification above; the baseline comparisons are now complete and follow below.

A full baseline comparison is now available for both YOLOv8n and RTMDet-tiny: parameters, FLOPs, mAP, and FPS, each trained on the same 2-class unified taxonomy the proposed model actually used (see Methodological Note, Results section), with the same protocol as Table 4.

Table 9. Parameter and FLOP comparison against YOLOv8n and RTMDet-tiny baselines (640×640 input; head width matched to each model's own training taxonomy -- see per-row notes).

Model	Parameters	FLOPs (640×640)	mAP@0.5 / mAP@0.5:0.95 / FPS
Proposed (EfficientViT-AFN)	4,477,990	22.73 G	0.0385 / 0.0137 / FPS not yet measured
YOLOv8n (2-class head)	3,011,238	8.2 G	0.2259 / 0.1134 / 135.2 FPS
RTMDet-tiny (2-class head)	4,873,470	16.07 G	0.0238 / N/A / 38.0 FPS

By parameter count and FLOPs alone, the proposed framework is roughly 1.5× larger and requires roughly 2.8× the compute of YOLOv8n, and is comparable in parameter count but requires roughly 1.4× the compute of RTMDet-tiny, at the same 640×640 input resolution, with each baseline's head matched to its own 2-class training taxonomy. This comparison is architecture-only; it says nothing by itself about relative detection accuracy or achievable inference speed. Real, apples-to-apples accuracy comparisons against both baselines are now available and are discussed in the YOLOv8n Result and RTMDet-tiny Result paragraphs below.

YOLOv8n Result. An earlier YOLOv8n run used VisDrone2019's true, uncollapsed 10-class taxonomy, independently of class_mapping.py; that result was withdrawn (see Methodological Note, Results section) once tracing the actual training script showed the proposed model's own single-domain training result was measured on the 2-class unified taxonomy instead, making the two results not comparable. YOLOv8n was retrained from scratch on the corrected 2-class unified taxonomy (small_vehicle, large_vehicle), with the same protocol as Table 4 (30 epochs, AdamW, lr=1×10⁻⁴, weight decay 1×10⁻², 3-epoch warmup, cosine schedule, batch size 16), on the same 6,471 train / 548 val images, converted with the exact _VISDRONE_TO_UNIFIED mapping from class_mapping.py. Ultralytics correctly built a genuinely 2-class head (nc=2, confirmed in the training log) rather than a 10-class head fed 2-class labels, so its parameter count and FLOPs in Table 9 (3,011,238 params, 8.2 GFLOPs) differ slightly from the earlier,

withdrawn 10-class-head figures, and are the ones used in the architecture comparison above. It reached $mAP@0.5=0.2259$, $mAP@0.5:0.95=0.1134$, roughly $5.9\times$ and $8.3\times$ the proposed model's best checkpoint ($mAP@0.5=0.0385$, $mAP@0.5:0.95=0.0137$, from single-domain training) respectively -- a larger gap than the (now-withdrawn) mismatched-taxonomy comparison had suggested, not a smaller one. This result is reported as measured, without adjustment, consistent with this paper's treatment of the MixStyle finding in the cross-domain Scenario S1 evaluation and the withdrawn-comparison episode itself: it is not favorable to the proposed model and is not softened here. Per-class AP was again highly uneven (small_vehicle: $AP@0.5=0.399$, versus large_vehicle: $AP@0.5=0.053$), mirroring the class-imbalance pattern in the training data (209,949 small_vehicle boxes versus 18,801 large_vehicle boxes) and suggesting the gap is concentrated in the dominant class rather than a uniform improvement across both. Ultralytics' internal loss function, anchor-free target assignment, and augmentation defaults differ from the proposed model's collision-aware assignment and MultiTaskDetectionLoss, so this result reflects a difference between two full detection systems trained under matched hyperparameters, data, and now taxonomy, not an isolated architecture effect; RTMDet-tiny's real mAP/FPS, which would help separate these possibilities, has not yet been run.

RTMDet-tiny Result. RTMDet-tiny was trained from scratch on the same 2-class unified taxonomy and the same protocol as YOLOv8n and Table 4 (30 epochs, AdamW, $lr=1\times 10^{-4}$, weight decay 1×10^{-2} , 3-epoch warmup, cosine schedule, batch size 16), with a genuinely 2-class head (4,873,470 parameters, the figure now used in Table 9 in place of the earlier $nc=10$ architecture-only figure; FLOPs are unchanged at 16.07 G, since the classification head is a negligible fraction of RTMDet-tiny's total compute regardless of class count), calling mmdetection's own model.loss()/model.predict() API directly in a custom training loop, since no ultralytics-equivalent one-line trainer exists for RTMDet. A methodological error was caught and corrected before this number was finalized: the first validation pass returned $mAP@0.5\approx 0.000026$ despite a normally converging training loss ($3295\rightarrow 1.76$ over 30 epochs), traced to mmdet's model.predict() defaulting to rescale=True, which maps predicted boxes back to each image's original pixel size while the ground-truth boxes used for scoring were left in the resized 640×640 space the model was trained on -- a coordinate-space mismatch, not a training failure. Re-scoring the same trained weights with rescale=False (no retraining required) gave $mAP@0.5=0.0238$, roughly $900\times$ higher, confirming the diagnosis. $mAP@0.5:0.95$ is reported as undefined (no predictions matched at any IoU threshold above 0.5 across the full validation set), and FPS is 38.0 (batch=1, GPU) -- both included in Table 9 as measured. Both figures are low in absolute terms, plausibly because RTMDet's SimOTA dynamic-soft-label assignment and its published training recipe (peak learning rate near 4×10^{-3} , roughly 300-epoch schedules) are tuned for a very different training budget than the 30-epoch, $lr=1\times 10^{-4}$ protocol used here for fairness with Table 4 and YOLOv8n; this result should be read as "RTMDet-tiny under the same reduced-budget, non-native-hyperparameter protocol as the other rows," not as a ceiling on what RTMDet-tiny can achieve on this task.

RTMDet-tiny's FLOPs figure in Table 9 (16.07 G) is roughly double the "8.1 GFLOPs" commonly cited for this model. This was checked directly: rebuilding RTMDet-tiny from its official published configuration (80-class COCO head) reproduces the published parameter count almost exactly (4,896,168 vs. the reported $\sim 4.8M$) and its raw multiply-accumulate (MAC) count, 8.09 G, matches the commonly cited "8.1 GFLOPs" figure closely. This indicates the widely cited figure reports raw MACs under the label "FLOPs," a known labeling inconsistency common in the object-detection literature, rather than an error in the present reconstruction. Table 9 reports $FLOPs = 2\times MACs$ throughout, the convention under which a multiply and an add are each counted, for consistency between the proposed model, YOLOv8n, and RTMDet-tiny within this table; readers comparing against RTMDet's own reported figures elsewhere should account for this factor-of-two convention difference.

DISCUSSION

The single-domain VisDrone2019 training result ($mAP@0.5 = 0.0385$ after 30 epochs) indicates that the corrected pipeline trains stably and improves steadily once the DataLoader num_workers defect described above is resolved. The ablation study identifies the task-decoupled head as the architecture's most influential

component at this training scale, with AFN and fusion contributing more modestly and with greater measurement noise, a finding that argues for a repeated-seed ablation before firm conclusions are drawn regarding their individual contributions.

The cross-domain MixStyle result from the Scenario S1 evaluation warrants the most careful discussion, as it did not accord with the expected direction of effect. Several plausible, currently undistinguished explanations may account for this outcome: (1) single-seed run-to-run variance at these small absolute mAP magnitudes; (2) MixStyle's characteristic role as a regularizer, which can slow early-training convergence in exchange for a generalization benefit that emerges only later in training the 15-epoch budget used here, a portion of which is consumed by warmup, may be insufficient for this trade-off to resolve favorably; (3) the default MixStyle hyperparameters ($p = 0.5$, $\alpha = 0.1$) may be poorly suited to this particular architecture and dataset pairing without further tuning; or (4) a genuine null or negative result for this configuration, which would itself constitute a legitimate empirical finding. This report does not fully adjudicate between these explanations, though the repeated-seed, extended-budget (20-epoch) follow-up reported above narrows the field: with two seeds at 20 epochs, MixStyle's effect relative to the baseline is small (and sign-inconsistent across seeds) compared with the baseline's own run-to-run variance, favoring explanation (1), single-seed variance, as the larger contributor to the original Scenario S1 cross-domain result over explanations (2)–(4). A larger seed count and, ideally, the full 30-epoch budget employed in the single-domain training run remain the most direct means of resolving this question with statistical confidence.

LIMITATIONS

The preceding discussion addresses open questions that the reported experiments themselves surface. A further set of limitations follows from implementation choices made or deferred along the way and is stated here directly, consistent with the methodological transparency maintained throughout this report.

Data augmentation. No augmentation was applied to any experiment reported in the Results section; per the Data Preprocessing description above, the dataloader performs only resizing and normalization, and the originally planned pipeline (horizontal flipping, random scaling, rotation, color jittering, mosaic augmentation) was never implemented. This is a plausible contributor to the low absolute mAP values reported throughout the Results section and should be ruled in or out before those values are attributed to the architecture itself.

Effective training taxonomy. The single-domain training and ablation study subsections were originally described as evaluating VisDrone2019's native 10-class taxonomy; tracing the actual dataset-loading code used to produce those results showed that every VisDrone annotation is unconditionally collapsed to the same 2-class unified taxonomy (small_vehicle, large_vehicle) used in the cross-domain Scenario S1 evaluation, with four of the ten native classes (pedestrian, people, tricycle, awning-tricycle) dropped from training entirely. This is now corrected throughout the report (see Methodological Note, Results section). It was not a deliberate experimental choice; the class-mapping call is buried inside a dataset builder function shared with the cross-domain code path and was not surfaced as a configurable option. Consequently, the detection head's 8 unused output classes carried no training signal in any experiment reported here, and the single-domain and ablation results describe a 2-class vehicle-detection problem rather than the originally intended 10-class general-object problem. Reproducing genuine 10-class results would require a dataset loader that bypasses this collapse, which does not currently exist in the available codebase.

Mixed-precision training. Mixed precision was disabled during the DataLoader num_workers defect diagnosis described above and has not been re-enabled for any experiment reported here. Its effect on training speed, memory footprint, and final accuracy for this framework remains unmeasured.

Missing evaluation metrics. Precision, Recall, and F1-score are listed among the intended evaluation metrics listed in the Materials and Methods section but are not reported for any experiment in the Results section; only mAP@0.5 and mAP@0.5:0.95 are measured. These metrics would give a more complete picture of the precision/recall trade-off underlying the reported mAP values.

Size-stratified evaluation. The small/medium/large object-size taxonomy defined in the Cross-Domain Formulation section for size-stratified evaluation has not been applied to any result in the Results section; all reported mAP figures are aggregated across object sizes without a small-object-only breakdown. Because this is fundamentally a small-object detection study, that breakdown is arguably the single most relevant missing measurement, and its absence limits how directly the current results speak to the paper's central claim.

Domain-alignment loss term. L_{domain} was removed from the trainable objective described in the Multi-Scale Feature Fusion and Detection Head section because the original formulation left it mathematically undefined. Cross-domain robustness in the current framework therefore rests entirely on AFN's normalization-level adaptation and MixStyle, with no explicit domain-alignment loss (e.g., an MMD or adversarial feature-matching term). Whether adding such a term would improve the Scenario S1 cross-domain results -- either the original run or its repeated-seed follow-up -- is untested.

Absolute detection performance. The best single-domain training result is $\text{mAP}@0.5 = 0.0385$, well below typical published results on VisDrone2019 for competitive lightweight detectors. For a concrete reference point on the same benchmark, Liu, Xie, and Wang (2026) report 32.4% $\text{mAP}@0.5$ for a fully trained YOLOv8n baseline and 39.6% $\text{mAP}@0.5$ for their proposed SSDM-YOLO (2.6M parameters), both roughly an order of magnitude above the present result despite a comparable or smaller parameter count. This comparison itself needs a caveat: Liu et al.'s figures are, by standard convention in the literature, almost certainly measured on VisDrone2019's full native 10-class taxonomy, whereas the present single-domain training result was measured on the 2-class unified taxonomy the executed pipeline actually used (see Methodological Note, Results section) -- a strictly easier classification problem at the same localization difficulty. Some portion of the gap to Liu et al.'s figures is therefore attributable to comparing different tasks, not solely to the candidate contributors below, though the size of that portion is not established here. No experiment in this report isolates why the present framework falls short of that range: candidate contributors include the absent augmentation pipeline, disabled mixed precision, the limited epoch budgets used for the ablation and cross-domain scenarios, and hyperparameters that have not been tuned for this architecture/dataset pairing. Until a diagnostic pass isolates these factors, this gap should be neither attributed to the EfficientViT+AFN architecture itself nor ruled out as an architectural limitation.

CONCLUSION AND RECOMMENDATION

This study presents a lightweight, cross-domain-oriented UAV small-object detection framework incorporating a fully corrected and independently verified box-regression formulation, a resolved full-scale training defect, a genuine single-domain training result, an empirical ablation study, an initial cross-domain generalization measurement, and a repeated-seed, extended-budget (20-epoch) follow-up on that measurement. The cross-domain MixStyle result did not confirm the anticipated benefit in either the original single-run measurement or the two-seed extended-budget follow-up; across both seeds the effect attributable to MixStyle remained small relative to seed-to-seed variance, and the two-seed mean continued to favor the baseline. This outcome is reported as observed, rather than reframed, and the MixStyle question is not yet closed, a larger seed count is still needed for a statistically confident conclusion, but it is better characterized than at the time of the original submission. The immediate next steps are as follows: (1) extend the repeated-seed Scenario S1 comparison to a larger seed count ($n \geq 5$) at the full 30-epoch budget employed in the single-domain training run, to estimate seed-to-seed variance precisely enough to distinguish a true null effect of MixStyle from a small directional one; (2) measurement of the proposed model's own inference latency and FPS on GPU hardware, for direct comparison against the YOLOv8n and RTMDet-tiny figures already in Table 9; and (3) extension to Scenarios S2 and S3 using the DroneVehicle dataset. Comparison against YOLOv8n and RTMDet-tiny baselines (parameters, FLOPs, mAP, and FPS, all on the same 2-class taxonomy and protocol as the single-domain training run) is complete and reported in the baseline comparison and efficiency subsection (Table 9). None of these remaining steps require further architectural modification to the framework described in the Materials and Methods section.

The manuscript is intentionally positioned as an empirical investigation of EfficientViT and adaptive feature normalization for cross-domain UAV object detection. The wording avoids claims of state-of-the-art accuracy or demonstrated superiority because the reported experiments do not establish those claims. The limitations and negative MixStyle findings are retained as reported.

REFERENCES

1. Cai, L., & Li, S. (2026). DSPE-ViT: a lightweight vision transformer with dynamic sparse positional encoding for dense small object detection in UAV imagery. *Frontiers in Neurorobotics*, 20, 1849093.
2. Chen, K. C., Malligere Shivanna, V., & Guo, J. I. (2026). YOLO-DAA: Directional Area Attention for Lightweight Tiny Object Detection in Maritime UAV Imagery. *Drones*, 10(4), 283.
3. Di, J., Wang, Q., & Yang, Y. (2026). CMM-DETR: an efficient transformer-based detector for remote sensing measurement applications. *Measurement Science and Technology*, 37(19), 195401.
4. Dong, C., Wang, Y., Zhai, Y., Li, Y., Liu, X., Mai, C., ... & Chen, C. P. (2026). DCA-YOLO: A Dynamic Cross-Attention Network for Small Vehicle Detection in UAV Aerial Imagery. *IEEE Transactions on Aerospace and Electronic Systems*.
5. Fan, X., Gao, M., Wang, Y., Tang, H., Sun, F., & Song, C. (2026). Data-Efficient Insulator Defect Detection in Power Transmission Systems via Multi-Granularity Feature Learning and Latent Context-Aware Fusion. *Electronics*, 15(5), 1081.
6. Li, C., Cao, J., Hao, Q., Song, C., Yao, H., & Wei, Z. (2026). RADFRNet: Detail-Enhanced Feature Recalibration for Infrared Small-Target Detection Based on an Improved YOLOv11. *Sensors*, 26(15), 4850.
7. Liu, J., Xie, Y., & Wang, J. (2026). A lightweight and robust framework for small object detection in UAV imagery. *Computer Vision and Image Understanding*, 267, 104735.
8. Liu, M., Hou, Y., Duan, X., Feng, B., Hou, H., Yang, M., ... & Gu, L. (2026). Multi-scale feature reconstruction detection transformer (MSFR-DETR): a real-time detector for small objects. *Journal of Real-Time Image Processing*, 23(3), 97.
9. Liu, S., Zhang, T., Ren, Y., Liu, Q., & Wang, L. (2026). A novel lightweight FusionMamba-YOLO algorithm with the hybrid Mamba-CNN architecture for real-time scene understanding in assistive robotic navigation. *Complex & Intelligent Systems*.
10. Liu, Z., Han, J., Zhai, G., & Li, W. (2026). LUW-DETR: a lightweight underwater object detection based on DETR. *Journal of Real-Time Image Processing*, 23(1), 14.
11. Lu, W., Niu, C., Liu, W., Lan, C., & Wang, S. (2025). Foundation Model-Based Auxiliary Framework for Object Detection in Aerial Remote Sensing Images. *IEEE Transactions on Aerospace and Electronic Systems*.
12. Mao, Y., Zhang, H., Li, R., Zhu, F., Sun, R., & Ji, P. (2025). HSF-DETR: Hyper scale fusion detection transformer for multi-perspective UAV object detection. *Remote Sensing*, 17(12), 1997.
13. Moskalenko, V. V., Moskalenko, A. S., Moskalenko, Y. V., & Vatsenko, A. V. (2026). Resource-efficient, robust, and adaptive object detection in UAV imagery. *Radio Electronics, Computer Science, Control*, (2), 85-99.
14. Ren, Y. (2026, April). Optimization Method for Small Object Detection in ViT-DETR Based on Deformable Cross-Attention and Multi-Scale Feature Alignment. In *2026 11th International Conference on Intelligent Computing and Signal Processing (ICSP)* (pp. 1815-1818). IEEE.
15. Shi, C., Wang, H., Fu, X., Liu, P., & Ke, H. (2026). A Small Object Detection Transformer for UAV Remote Sensing Imagery via Multi-Scale Perception and Cross-Spatial-Frequency Domain Fusion. *Remote Sensing*, 18(9), 1394.
16. Tian, B., Hu, M., Li, L., Zhao, X., Tang, M., & Wang, Z. (2026). AMFNet: adaptive multi-scale fusion network for small object detection. *Complex & Intelligent Systems*.
17. Tu, X., Gao, Z., Wei, L., Zhang, Y., Liu, J., Yang, M., ... & Li, X. (2026). Small object detection in UAV aerial imagery through domain consistency optimization for cross-resolution semantic alignment. *Pattern Recognition*, 113463.
18. Wang, S., Guo, Z., Yu, Z., & Zhou, Z. (2026). CFP-DETR: Collaborative Feature Purification Network with Spatial Alignment for Aerial Small Object Detection. *Remote Sensing*, 18(11), 1750.

19. Xu, Z., Zhou, Y., Wen, S., & Jing, W. (2025). FIDC-YOLO: improved YOLO for detecting pine wilt disease in UAV remote sensing images via feature interaction and dependency capturing. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 18, 14674-14687.
20. Zhang, J., Zhong, H., Du, G., Zhang, B., & Zhang, H. (2026). Aerial small object detection via dynamic convolution and hierarchical attention fusion for UAV imagery. *Scientific Reports*.
21. Zhang, X., & Hu, X. (2026). Multi-stage feature purification for robust object detection in haze-degraded aerial images. *Scientific Reports*.
22. Zhou, S., Liu, M., Zhou, J., & Zheng, R. (2026). Vision transformer with salience self-attention for underwater and aerial object recognition and tracking. *Neurocomputing*, 132631.
23. Zuo, G., Wang, Y., Sha, Y., Xia, Y., Tang, M., Jia, H., & Chi, R. (2026). DCM-YOLO: Robust Electric Bicycle Detection in Confined Indoor Environments Under Occlusion and Image Degradation. *Symmetry*, 18(6), 1040.