ISSN: 2754-6659

Journal of Artificial Intelligence & Cloud Computing

SCIENTIFIC Research and Community

Review Article

Open Access

End-To-End Framework for Real-Time Drone Detection and Alerting Using Lightweight Deep Learning

Orwa Aladaileh[1*], Izz AlDrrass[2], Reem Shtaiwi[1], Mona Obaisat[2] and Mallak Albarari[1]

[1]Department of Computing and Informatics AlHussein Technical University, Amman, Jordan

[2]Department of Computer Science Tafila Technical University Tafila, Jordan

ABSTRACT

Real-time drone detection is a significant issue in airspace security and privacy because of the increasing use of UAVs; this issue affects airport security directly as it results in various inclusions such as airport incursions and unauthorized surveillance. We proposed end-to-end framework for real time drone detection and alerting integrating multiple YOLO architectures, our system trained on a largescale composite dataset of over 77,000 labeled images for drones, expanded to approximately 90,000 through targeted data augmentation. The Yolo models first trained and evaluated independently, then combined using ensemble learning strategies to reduce false detections while preserving real-time performance. The final ensemble of (YOLOv8, YOLOv11, and YOLOv12) achieves 96.1% accuracy and 93.9% mAP@0.5 Index Terms—Drone Detection, YOLO, Ensemble Learning, Real-Time Alert Systems, Unmanned Aerial Vehicles (UAVs), Lightweight DL Architecture.

***Corresponding author**

Orwa Aladaileh, Department of Computing and Informatics AlHussein Technical University, Amman, Jordan.

Received: August 04, 2026; **Accepted:** August 08, 2026; **Published:** August 18, 2026

Introduction

The widespread adoption of Unmanned Aerial Vehicles (UAVs), commonly known as drones, has significantly influenced various sectors, including agriculture, logistics, media, and infrastructure inspection. Nevertheless, this rapid growth has also created a number of challenges in terms of airspace security, privacy, and safety. Incursions into airports, surveil-lance of restricted regions, and drone smuggling activities have highlighted the need for intelligent, automated counter-UAV (C-UAV) systems that can track drones in real time. As shown in Figure 1, the current state of Unmanned Aerial Vehicles (UAVs) has grown to include a large range of applications, from high-speed racing and consumer photography to industrial inspection drones [1].

Conventional C-UAV detection systems are based on radar, radio frequency (RF) analysis, or acoustic detection. Although these systems work well in some controlled settings, they can be costly, hardware-intensive, and susceptible to interference in dense or urban areas. Computer vision-based systems, on the other hand, provide a passive, scalable, and cost-effective solution by utilizing image information for UAV detection. Currently, vision-based systems have their own set of challenges such as: (1) the drones are usually small in size, and (2) they are fast-moving with highly varying appearances and backgrounds. Consequently, the research problem of creating models that strike a balance between detection accuracy, speed, and efficiency still exists [2,3].

Recent breakthroughs in deep learning, especially in real-time object detection models, have greatly enhanced the effectiveness of computer vision in drone surveillance. Among these, the You Only Look Once (YOLO) series of models has proven to be a leading approach in real-time object detection, thanks to its unified end-to-end design. The earlier models of YOLO, namely YOLOv4 and YOLOv5, were highly effective in object detection but computationally intensive for edge devices in many cases. The YOLOv8, YOLOv11, and YOLOv12 series have brought significant architectural enhancements, such as the incorporation of adaptive spatial features and efficient designs with anchor-free detection heads. These models promise real-time processing without compromising object detection accuracy, especially when used in lightweight models such as Nano, Small, and Medium [4].

Based on these breakthroughs, this research work proposes an all-encompassing, end-to-end solution for real-time drone detection and notification, leveraging multiple YOLO models and configurations to investigate the trade-offs between speed and accuracy. These models claim to process in real-time without affecting the accuracy of object detection, particularly when applied to lightweight models such as Nano, Small, and Medium. Each series of models (namely, YOLOv8 [5], YOLOv11, and YOLOv12) was trained on a hybrid dataset of 90,000 labeled images of drones from various public domains, including various environments, distances, and types of drones. After the evaluation, the most accurate models from each series were then combined using an Ensemble Learning (EL) method specifically designed to combine the predictions of the models and remove false positives while still being able to process in real-time.

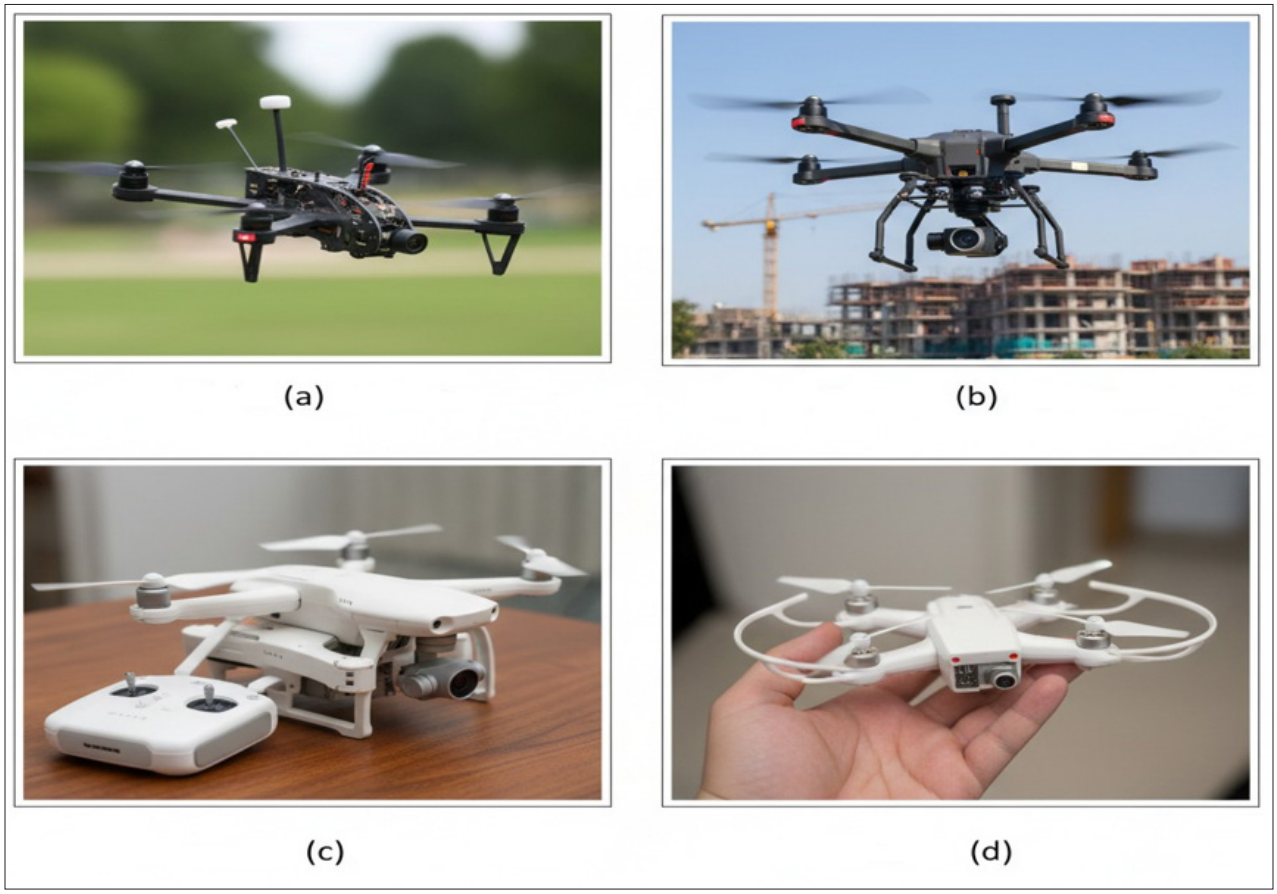

Figure 1: Various types of Unmanned Aerial Vehicles (UAVs): (a) FPV racing drone for speed; (b) Industrial inspection drone with high-resolution sensors; (c) Photography drone for consumers; (d) Miniature toy UAV for indoor flying

In addition to the development of the models, the proposed framework also includes a monitoring interface and alert system using PyQt5 that sends detection snapshots and coordinates to security personnel via Telegram. This approach helps ensure that the result of detection is not only precise but also actionable, which is a critical requirement for implementation. The major contributions of the proposed work are listed below:

- Comprehensive Dataset Aggregation: Development of a diverse dataset with approximately 90,000 images of drones from VisDrone, Anti-UAV, and UAV123 datasets, thus ensuring a high level of variability in terms of environmental and contextual factors.
- Comparative Evaluation of Modern Yolo ArChitectures: Experimental evaluation of YOLOv8 [5], YOLOv11, and YOLOv12 architectures for Nano, Small, and medium settings to understand the trade-offs between performance.
- Ensemble Learning for Improved Detection: Development of an ensemble inference system that combines predictions from multiple optimized YOLO models to improve robustness and reduce false alarms.
- End-to-End Operational Integration: Development of a real-time visual interface and alerting system linking detection outputs to immediate security notifications.

The rest of this paper is organized as follows. Section II introduces the related work on traditional and deep learning-based drone detection methods. Section III describes the proposed approach in detail, including data preparation, model training, and ensemble method. Section IV shows the experimental results and analysis. Finally, Section V concludes this paper and provides future work, such as edge device implementation and multi-sensor fusion.

Related Work

In the early days of counter-Unmanned Aerial Vehicle (C-UAV) technology, the primary sensors used were radar, radio-frequency (RF), and acoustic sensors. While these sensors have the advantage of long-range detection and are independent of lighting conditions, they also have the disadvantages of high deployment cost, RF interference, and difficulty in distinguishing between drones and other small objects. These issues have led to the development of passive vision-based detection systems that utilize recent advances in deep learning.

Vision-Based Drone Detection Methods

Visual detection based on deep learning has become the core of modern UAV monitoring systems. Redmon et al. proposed the YOLO system, which combined object localization and classification into one real-time task [6]. Later, improved models like YOLOv4 and YOLOv5 were developed to enhance the accuracy of object detection using feature fusion modules, cross-stage partial networks, and improved data augmentation techniques [3,7]. However, their computational complexity made it difficult to implement them on low-power embedded systems.

To overcome this problem, some research works have been done to develop lightweight versions of YOLO for real-time UAV object detection. Nguyen Tien Tai et al. introduced YOLOXpress, where the C2f block is replaced by RepConv and RepC3 modules,

resulting in faster detection with acceptable accuracy [8]. However, the accuracy of the model decreased slightly compared with the baseline of YOLO architecture.

Titu et al. proposed a real-time fire detection system based on a drone platform using YOLOv8 as a teacher model and light-weight distilled student models (YOLOv8n and DETR) [9]. The system reported a detection accuracy of 95.1% and F1-score of 0.985 on a dataset of 7,187 images, running on a Raspberry Pi 5 drone platform. However, the system performed poorly under varying illumination and weather conditions.

Bakirci compared the performance of YOLOv7-tiny, YOLOv9-tiny, YOLOv10-nano, and YOLOv10-small models on the NVIDIA Jetson platform in terms of trade-offs between detection accuracy, inference time, GPU usage, and energy efficiency [10]. YOLOv10-nano demonstrated the best balance be-tween precision and speed for edge applications. Ciccone and Ceruti analyzed YOLO-based detection models for search and rescue applications. Their most accurate model (YOLOv5s-PB-FPN-Deconv) reported a mean Average Precision (mAP@50) of 0.802 on the Heridal dataset with real-time processing on the Jetson Nano [11]. However, the model only utilized RGB images, making it less robust in low-visibility Ntousis et al. introduced a dual-stage framework that integrates both onboard YOLOv5 detection and ground-server tracking and trajectory prediction [12]. The model, trained on the VisDrone2018-DET dataset, reported a recall rate of about 70% while remaining in real-time on a Raspberry Pi, although better hardware may be beneficial for performance. Liu et al [13]. introduced a UAV aerial image small-object detection method that improves the preservation of fine-grained features and adopted a well-balanced multi-scale feature pyramid to better represent small targets in the spatial domain. The method greatly improved aerial image detection accuracy while keeping a moderate computational complexity, proving the significance of multi-scale fusion in UAV vision tasks. Svystun et al [14]. introduced an ensemble framework based on YOLO for multispectral defect detection using UAVs for wind turbine parts. Their method combines the predictions of different YOLO models to improve the accuracy of defect detection in different spectral bands, showing the effectiveness of the ensemble learning approach to improve the robustness of the features extracted under different lighting conditions and surface types. The proposed method showed higher accuracy than single-model YOLO models, thus proving the effective-ness of the ensemble learning approach for inspection and monitoring applications using UAVs. Lastly, Tian et al [15]. presented NFE-YOLO, a lightweight and efficient network that incorporates the EOrthoNet attention module to improve the accuracy of small drone detection. The model, trained on a dataset of 10,099 images, showed high accuracy in detection while being edge device-friendly, although more experiments are required to validate its generalization capability.

Research Gap and Novelty

Collectively, these contributions show substantial advancement towards achieving real-time UAV detection in embedded and edge platforms. Nonetheless, most of these contributions are based on a single generation of YOLO or a particular architectural optimization, without a comprehensive assessment within the latest YOLOv8-YOLOv11 series. In addition, very few of these contributions have a complete operational pipeline that includes real-time notification and visualization.

This contribution aims to address these issues by comprehensively evaluating the YOLOv8, YOLOv11, and YOLOv12 models with different scales (Nano, Small, and Medium) trained on a single dataset. The best models are then combined using an ensemble approach to enhance robustness while maintaining fast processing speed. Moreover, an end-to-end system is proposed to combine visualization and real-time notification seamlessly, thus bridging the gap between theoretical model performance and practical UAV monitoring systems.

Methodology

This section describes the methodology employed in the proposed real-time drone detection and alert system frame-work. The entire process involves a number of key steps, starting with data preparation and processing for diversity in the data, followed by model selection and optimization using different generations of YOLO. Later, an ensemble learning approach is used to combine the predictions of the best models to achieve better robustness and eliminate false positives. Finally, the system design integrates the trained models with a graphical user interface and real-time alert system to develop a complete functional pipeline for the resource-constrained UAV surveillance environment.

Dataset Preparation

A comprehensive dataset for UAV detection was developed by aggregating publicly available image datasets sourced from different online platforms. The aggregated dataset comprises a total of **77,470 labeled images**, covering different environmental settings, heights, and UAV models. The datasets were chosen to provide as much diversity as possible in terms of UAV models, backgrounds, and lighting conditions, which play a very significant role in enhancing the generalization capability of the model. Table I provides a brief summary of all the datasets used.

Augmentation for Visual Robustness: Apart from the standard YOLO preprocessing techniques, a data augmentation strategy was employed to enhance the robustness of the model to various visual conditions. A random sample of **1,600 images** was selected from the combined dataset and used to generate **13,000 derived samples** using a set of carefully designed geometric and photometric transformations. Unlike the standard data augmentation technique, which is mainly focused on augmenting the size of the dataset, this technique was intended to artificially create samples that simulate **real-world environmental changes** such as lighting, weather, orientation, and camera viewpoints.

The techniques used for data augmentation of drone images included random rotations, horizontal and vertical flips, brightness and contrast adjustments, shadow simulation, and HSV value translations simulating day/night changes and cloudy weather. The logic behind this carefully designed visual diversity was to counter **visual confusion** due to low contrast, glare, or complex backgrounds, thus enabling the model to better generalize to new aerial images. By exposing the detector to drones captured from different angles and lighting conditions, the resulting models were more robust to occlusions, motion blurs, and seasonal variations. In conclusion, the final dataset comprises **90,470 image**s, including the original 77,470 labeled images and 13,000 images.

Data Composition and Splitting: The combined data was split into three portions to ensure equal representation during training: **70% for training, 20% for validation**, and **10% for**

Table 1: Summary of Aggregated UAV Datasets Used for Model Training

Dataset Name	Platform	Image Count	Access Link
UAVs Dataset	Roboflow	9.8k	https://universe.roboflow.com/uavs-7l7kv/uavs-vqpqt
Accurate Drone	Dataset Ninja	9.0k	https://datasetninja.com/accurate-drone[Access Link]
Drones Dataset	Roboflow	6.6k	https://universe.roboflow.com/customyolo-k3lvg/drones-rom1a
Deadly Flying Bois	Roboflow	9.86k	https://universe.roboflow.com/dronerboner/deadly-flying-bois-yt-2
Drone Detection	Kaggle	12.0k	https://www.kaggle.com/datasets/cybersimar08/drone-detection
Drone YOLO Detection	Kaggle	4.0k	https://www.kaggle.com/datasets/sshikamaru/drone-yolo-detection
Drone Dataset UAV	Kaggle	2.5k	https://www.kaggle.com/datasets/dasmehdixtr/drone-dataset-uav
Drone Type Classification	Kaggle	8.0k	https://www.kaggle.com/datasets/balajikartheek/drone-type-classification
Drone Dataset UAV (Alt.)	Dataset Ninja	1.35k	https://datasetninja.com/drone-dataset-uav
YOLO Drone Detection	Kaggle	1.5k	https://www.kaggle.com/datasets/muki2003/yolo-drone-detection-dataset
DJI Tello Drone Detection	Kaggle	0.86k	https://www.kaggle.com/datasets/luispreciado99/dji-tello-drone-object-detection-dataset
Drone Dataset Practice	Roboflow	7.6k	https://universe.roboflow.com/samreen-practice/drone-dataset-741s2
UAVs Hunting	Roboflow	4.4k	https://universe.roboflow.com/uavs-hunter/uavs-hunting
Total	—	**77.47k**	—

Note: All the datasets were collected from publicly available sources hosted on Roboflow, Kaggle, and Dataset Ninja. All the datasets contain YOLO-compatible bounding box annotations. This ensures consistency in the training and testing process. The diversity of the datasets increases the robustness of the model to environmental changes and variations in the appearance of the drone.

Testing

The proportion of data split was determined to ensure that there are enough data points for optimizing the model while also having unseen data for unbiased model performance assessment.

Preprocessing and Normalization: During the preprocessing phase, all input images were automatically resized to a standard size of **640 × 640 pixels** to ensure standardization of input image size during batch training. The preprocessing phase was necessary since the original datasets contained images with varying aspect ratios and resolutions. Other pre-processing techniques included pixel value normalization and color space transformation to the RGB color space. All pre-processing techniques were performed within the Ultralytics YOLO training framework to ensure consistency in different model designs.

Model Architectures

Our proposed framework uses multiple generations of the You Only Look Once (YOLO) family of object detectors. In our framework, we used the following: (1) YOLOv8 (2) YOLOv11, and (3) YOLOv12. These architectures mark significant milestones in the development of real-time vision systems. The architectures are equipped with convolutional backbones and efficient detection heads that are optimized for end-to-end training and inference.

Yolo Families Overview

YOLOv8, released by Ultr-alytics1 in 2023, represented a paradigm shift from anchor-based to **anchor-free detection**, where bounding box centers and object scales could be directly predicted without anchor priors. This represented a major simplification of the training process, enhanced small object localization, and decreased model complexity. The architecture features C2f blocks (Cross-Stage Partial with faster gradient flow) in a CSPDarknet backbone and PAN-FPN neck for multi-scale feature fusion [16].

[1]https://docs.ultralytics.com/models/yolov8/

YOLOv11[2] further enhances this architecture by adding decoupled heads, optimized RepConv modules for inference, and a lightweight semantic/spatial interaction module. It further enhances normalization consistency and connectivity to the backbone, providing improved gradient stability across multiple scales, which is essential for the detection of small, fast-moving drones [17].

YOLOv12[3], the latest version used for comparison in this work, combines Efficient Layer Aggregation Networks (ELAN++) and dynamic reparameterization for optimal computational efficiency. With enhanced bottleneck connections and adaptive input scaling, YOLOv12 provides the best possible trade-off between accuracy and speed for edge applications [18].

Each version of YOLO has been made available in various model sizes: Nano (n), Small (s), Medium (m), and Large (l), which vary based on the number of parameters, depth, and feature width [18].

In this study, the Nano and Small variants of YOLOv8, YOLOv11, and YOLOv12 were selected and trained to balance computational efficiency with detection accuracy, ensuring real-time performance suitable for edge deployment.

Anchor-Free Detection Head: In contrast to conventional anchor-based detectors, which utilize pre-defined bounding box anchors, anchor-free detection heads predict the object center coordinates (x, y), as well as the width-height (w, h), and class probabilities for each location on the feature map. The anchor-free approach eliminates the computational cost of anchor matching and generation, and also enhances the detection accuracy of small or irregularly shaped objects, such as drones viewed from different altitudes. In aerial images, where the size and aspect ratio of objects vary greatly, the anchor-free approach generalizes better and is easier to optimize, with fewer false negatives and localization errors.

2https://docs.ultralytics.com/models/yolo11/
3https://docs.ultralytics.com/models/yolo12/

Justification use for Lightweight Model Variants: In light of the operational requirement for real-time UAV monitoring and edge device deployment with limited computational re-sources, this research focuses on the Nano and Small variants of the YOLO models. The Nano (n) models have less than 3.5 million parameters, with inference rates of over 70 FPS on mid-range GPUs with only a slight loss of accuracy. The Small (s) variants, although slightly larger, have better accuracy and recall in difficult scenarios like low contrast, cluttered backgrounds, or fast motion.

Ensemble Learning Strategy
To enhance the accuracy of detection within the framework, various ensemble learning (EL) approaches were employed by combining the predictions of different YOLO models trained using different architectures and scales [19]. Two approaches were used for ensemble learning: (1) Soft-Voting Ensemble and (2) Weighted Boxes Fusion (WBF). Each ensemble group aggregates the outputs of three models (i.e. YOLOv8, YOLOv11, and YOLOv12) in their corresponding Nano (n) and Small (s) variants. The ensembles are represented as

- EL_n = YOLOv8-n + YOLOv11-n + YOLOv12-n
- EL_s = YOLOv8-s + YOLOv11-s + YOLOv12-s

Soft-Voting Ensemble: In the soft-voting scheme, model predictions are aggregated by averaging their class probability distributions. Let pi(c) denote the predicted probability of class c by model i, where i ∈ {1, 2, 3}. The ensemble probability for class c is given by:

$$P(c) = \frac{1}{N}\sum_{i=1}^{N} p_i(c) \qquad (1)$$

where *N* is the number of models in the ensemble. The final class label ˆc is calculated as:

$$\hat{c} = \arg\max_{c} P(c) \qquad (2)$$

This approach improves stability and mitigates the over confidence of individual models by leveraging consensus among multiple detectors [20].

Weighted Boxes Fusion (WBF): The *Weighted Boxes Fusion* method [19] fuses bounding boxes from different detectors based on their confidence scores rather than suppressing them as in Non-Maximum Suppression (NMS). Given M boxes {B_1,B_2, ...,B_M} with confidence scores
{s1, s2, ..., sM} predicted for the same object, the final fused box Bf is calculated as:

$$B_f = \frac{\sum_{i=1}^{M} s_i \cdot B_i}{\sum_{i=1}^{M} s_i} \qquad (3)$$

and the aggregated confidence score S_f as:

$$S_f = \frac{1}{M}\sum_{i=1}^{M} s_i \qquad (4)$$

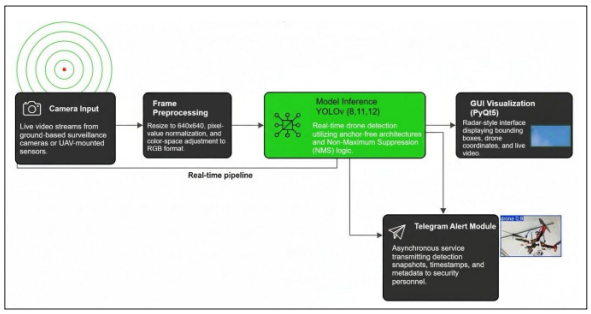

Figure 2: System Architecture of the Real-Time Drone Detection & Alerting framework

This approach maintains the contribution of detection for all models and provides better localization accuracy than the traditional NMS approach, especially when there is overlapping detection of drones or when the target is occluded.

Comparison of Ensemble Strategies: The differences among the various ensemble strategies implemented in this work and their essential characteristics when used for YOLO-based UAV detection are presented in Table I

System Architecture
The proposed end-to-end system combines computer vision, graphical visualization, and real-time communication modules for autonomous UAV monitoring and alerting. As illustrated in Figure 2, the entire process can be summarized as follows.

Camera Input → Frame Preprocessing → Model Inference → GUI Visualization → Telegram Alert

The video frames are then processed through a preprocessing stage, where they are resized, normalized, and converted to a compatible format for processing by the YOLO engine. To ensure optimal visual performance and output, the GUI is designed to incorporate the best performing detection model that had the highest accuracy and reliability during the evaluation stage.

The post-inference threshold logic filters out the predictions with low confidence based on confidence and IoU thresholds, thereby retaining only the accurate drone detection.

A graphical user interface (GUI) has been developed using PyQt5 for real-time visualization. The operational GUI, as shown in Figure 3, is a dashboard that combines various monitoring streams

into one interface. The GUI has a radar visualization panel that shows the detected UAVs with bounding boxes, confidence levels, and approximate location. The GUI has the functionality of playing back frame by frame, live monitoring, and taking snapshots. For remote notifications, the system combines the Telegram API, which automatically sends detection snapshots, timestamps, and confidence data to the concerned security personnel or groups of monitors. This ensures the operators have immediate situational awareness even when they are not actively monitoring the GUI. Through the use of edge-optimized inference and asynchronous communication, the proposed system ensures high responsiveness and operational feasibility.

Table 2: Comparison of Ensemble Learning Strategies

Method	Input Models	Fusion Type	Advantages
Soft-Voting Ensemble	YOLOv8, YOLOv11, YOLOv12 (n/s)	Probability Averaging	Reduces false negatives; robust to class uncertainty
Weighted Boxes Fusion (WBF)		YOLOv8, YOLOv11, YOLOv12 (n/s) Box	
Confidence Weighting		Improves localization accuracy; preserves overlapping detections	

Note: Both ensemble strategies were evaluated for Nano (n) and Small (s) model configurations, achieving a balance between inference efficiency and detection reliability

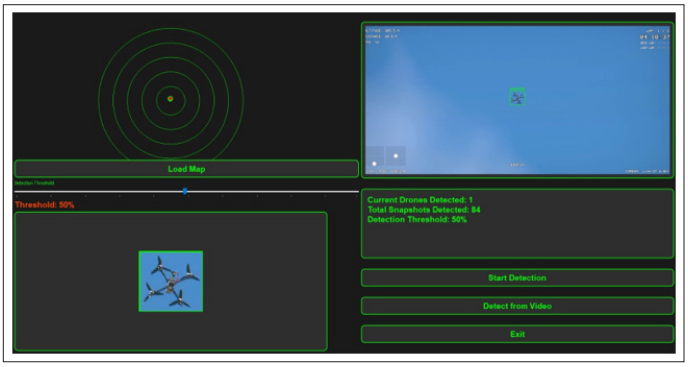

Figure 3: The developed PyQt5-based Graphical User Interface (GUI) for real-time drone monitoring, featuring a radar-style tracking display (top-left), a live detection feed with bounding box overlays (top-right), and a snapshot log of confirmed detections (bottom-left)

Experiments and Results

In the following section, we will explain the experimental setup, evaluation method, and performance results of the proposed UAV detection framework. A series of controlled experiments have been designed to assess the accuracy, inference speed, and efficiency of different YOLO architectures and ensemble configurations. The evaluation method involves both quantitative and qualitative analyses of the performance of individual models and ensembles in varying environmental conditions.

Experimental Setup

All experiments were performed on NVIDIA RTX 4060 GPU, which has 8 GB of VRAM, accompanied by 32 GB of RAM. The system has enough processing power for training small and medium-scale deep learning models and is capable of real-time inference for deployment validation.

Our framework was developed using PyTorch with Ultralytics YOLO repository as the base implementation for YOLOv8, YOLOv11, and YOLOv12 models. Other supporting libraries include OpenCV for image processing and video frame handling, and PyQt5 for the graphical user interface and radar display. System integration and notification functions were developed using Python 3.10 with the Telegram API for asynchronous message sending.

The training procedure was set up with the following hyperparameters: 100 epochs, batch size of 16 & 32 for Nano and Small models, and an adaptive learning rate initialized at $1 \times 10{-}3$ with a cosine annealing scheduler to ensure stable convergence and avoid overfitting.

All models were developed and evaluated using the con-solidated dataset outlined in Section III-A, with the data divided into 70% for training, 20% for validation, and 10% for testing. Model checkpoints and performance metrics were logged using the integrated Ultralytics tracking utilities for subsequent analysis and ensemble evaluation.

Evaluation Metrics and Results

The proposed UAV detection framework was evaluated using standard object detection metrics, namely Precision, Recall, and mean Average Precision (mAP) [21]. Performance is reported at two IoU settings: mAP@0.5 and mAP@0.5:0.95, following the COCO-style evaluation protocol [22]. In addition, we report a detection accuracy score under a fixed matching rule to provide an intuitive summary of overall correctness.

Metric Definitions A predicted bounding box is considered a true positive (TP) if it matches a ground-truth UAV box with Intersection over Union (IoU) greater than or equal to a threshold and with confidence score above a fixed threshold τ. Unmatched predictions are counted as false positives (FP), and unmatched ground-truth objects are counted as false negatives (FN).

Intersection over Union (IoU) between a predicted box Bp and a ground-truth box Bgt is defined as [23]:

$$\mathrm{IoU}(B_p, B_{gt}) = \frac{|B_p \cap B_{gt}|}{|B_p \cup B_{gt}|}. \qquad (5)$$

Precision measures the correctness of positive detections:

$$\mathrm{Precision} = \frac{TP}{TP + FP}. \qquad (6)$$

Recall measures the fraction of ground-truth UAVs correctly detected:

$$\mathrm{Recall} = \frac{TP}{TP + FN}. \qquad (7)$$

Average Precision (AP) is computed as the area under the Precision–Recall curve. The **mean Average Precision (mAP)** is the mean AP across classes (here, UAV is treated as the primary class). We report:

• mAP@0.5: AP at a single IoU threshold of 0.50.

• mAP@0.5:0.95: AP averaged over IoU thresholds {0.50, 0.55, . . . , 0.95} (step 0.05), which is more strict and better reflects localization quality [22].

is reported as the fraction of test images for which the detector produces a correct decision under the same matching rule (IoU≥0.5 and confidence ≥ τ). Specifically, an image is counted as correct if (i) at least one UAV is present and at least one TP is produced, or (ii) no UAV is present and no prediction is produced. This definition avoids ambiguous box-level true negatives in dense detection settings.

Figure 4: Real-Time Drones Example Detection using our System

Yolo Model Performance: Table III reports the results of individual YOLO architectures trained under identical pre-processing and training conditions. Across model families, the small variants consistently improved mAP, indicating better localization and classification capacity at the cost of higher computation.

Table 3: Evaluation Results of Yolo Architectures (v8, v11, v12) across Nano and Small Variants Using AdamW and SGD Optimizers with Varying Batch Sizes

Model Version	Size	Optimizer	Batch Size	Learning Rate	Accuracy (%)	Precision (%)	Recall (%)	mAP@0.5 (%)	mAP@0.5:0.95 (%)
YOLOv8	Nano	AdamW	16	1×10^{-3}	87.5	84.8	83.5	83.1	61.2
		AdamW	32	1×10^{-3}	88.2	85.4	84.1	83.9	62.1
		SGD	16	1×10^{-3}	88.6	85.9	84.9	84.4	63.8
		SGD	32	1×10^{-3}	89.1	86.2	85.5	85.0	64.5
	Small	AdamW	16	1×10^{-3}	89.8	87.2	86.5	86.1	67.4
		AdamW	32	1×10^{-3}	90.4	88.1	87.2	86.8	68.3
		SGD	16	1×10^{-3}	90.8	88.5	87.9	87.2	69.5
		SGD	32	1×10^{-3}	91.2	89.0	88.4	87.5	70.1
YOLOv11	Nano	AdamW	16	1×10^{-3}	89.4	86.9	86.0	85.5	65.9
		AdamW	32	1×10^{-3}	90.1	87.5	86.9	86.2	66.8
		SGD	16	1×10^{-3}	90.9	88.2	87.4	87.1	68.1
		SGD	32	1×10^{-3}	91.5	88.9	88.0	87.7	69.4
	Small	AdamW	16	1×10^{-3}	91.9	89.5	88.6	88.4	71.8
		AdamW	32	1×10^{-3}	92.3	90.2	89.1	89.0	72.5
		SGD	16	1×10^{-3}	92.7	90.6	89.7	89.6	73.5
		SGD	32	1×10^{-3}	93.1	91.0	90.2	90.1	74.2
YOLOv12	Nano	AdamW	16	1×10^{-3}	91.2	88.4	87.8	87.5	70.2
		AdamW	32	1×10^{-3}	91.8	89.2	88.5	88.1	71.0
		SGD	16	1×10^{-3}	92.1	89.7	88.9	88.6	72.4
		SGD	32	1×10^{-3}	92.5	90.1	89.4	89.0	73.2
	Small	AdamW	16	1×10^{-3}	92.9	90.8	89.6	89.8	74.1
		AdamW	32	1×10^{-3}	93.4	91.2	90.3	90.2	74.8
		SGD	16	1×10^{-3}	93.7	91.5	90.6	90.5	75.1
		SGD	32	1×10^{-3}	94.0	91.8	90.9	90.8	75.6

Note: Each model variant was trained for 100 epochs using identical datasets and preprocessing. YOLOv12-Small (SGD) achieved the highest overall performance, balancing accuracy, precision, and recall while maintaining real-time inference capability.

Ensemble Learning Performance: Ensemble strategies were evaluated using the same dataset and matching criteria as the single-model experiments. As shown in Table IV, both the *Soft-Voting and Weighted Boxes Fusion (WBF)* ensembles improved detection robustness. In particular, WBF produced the best overall performance, which is consistent with its design goal of improving localization by merging highly overlapping predictions using confidence-weighted averaging [19].

Table 4: Performance of Ensemble Learning Approaches on UAV Detection Dataset

Ensemble Method	Base Models	Accuracy (%)	Precision (%)	Recall (%)	mAP@0.5 (%)
Soft-Voting Ensemble (n)	YOLOv8-n + YOLOv11-n + YOLOv12-n	94.7	92.8	91.9	92.6
Soft-Voting Ensemble (s)	YOLOv8-s + YOLOv11-s + YOLOv12-s	95.3	93.5	92.6	93.1
WBF Ensemble (n)	YOLOv8-n + YOLOv11-n + YOLOv12-n	95.8	94.0	93.1	93.4
WBF Ensemble (s)	YOLOv8-s + YOLOv11-s + YOLOv12-s	96.1	94.2	93.8	93.9

Note: Ensemble configurations combining YOLOv8, YOLOv11, and YOLOv12 achieved the highest overall performance. The WBF approach demonstrated better localization accuracy by leveraging weighted confidence fusion across model predictions.

Discussion

Results clearly indicate a consistent performance improvement across YOLO generations, with YOLOv12 outperforming YOLOv8 and YOLOv11 due to its refined backbone and efficient ELAN++ connections. However, ensemble learning further elevated detection accuracy, achieving a top accuracy of 96.1% and 93.9% mAP@0.5, marking an overall improve-ment of approximately 3% over the best single YOLO model. Figure 4 presents an example of our system results.

Qualitative inspection revealed that the ensemble effectively mitigated false negatives, particularly in complex or low-visibility scenes. Common misclassifications included con-fusion between birds, drones, and distant aerial objects, especially in night and foggy conditions. Nevertheless, the ensemble framework provided robust and consistent detections, maintaining real-time operation at 65 FPS on an RTX 4060 GPU.

These findings confirm that ensemble learning offers a powerful yet efficient solution for UAV detection, achieving high reliability while maintaining edge-device compatibility. Future enhancements may include temporal tracking or fusion with thermal imagery to further reduce visual ambiguity in adverse weather or low-light conditions.

Conclusion and Future Work

This paper has proposed a holistic real-time UAV detection system using the latest YOLO models and ensemble techniques. The paper has comprehensively compared the YOLOv8, YOLOv11, and YOLOv12 models with Nano and Small sizes using various optimizers and hyperparameters. The experimental results have shown an improvement in performance with each new generation of YOLO models, with YOLOv12-Small having the best single-model accuracy of 94.0% and mAP@0.5 of 90.8%.

To further enhance detection reliability, Two ensemble learning methods, namely Soft-Voting and Weighted Boxes Fusion (WBF), were applied by fusing the predictions of YOLOv8, YOLOv11, and YOLOv12 models. The ensemble learning models resulted in a substantial decrease in the number of false negatives and improved localization consistency, reaching a peak accuracy of 96.1% and mAP@0.5 of 93.9%. The above findings validate the effectiveness of multi-model fusion in improving robustness in challenging scenarios like low-light imaging, background clutter, and visually similar objects (birds vs. drones).

In addition to model evaluation, the proposed system com-bines a PyQt5-based GUI with radar-style visualization and automated Telegram notifications, offering a complete end-to-end solution for UAV monitoring.

Future research will focus on the integration of the proposed framework on embedded computing systems such as NVIDIA Jetson and Raspberry Pi to evaluate its efficiency on resource-constrained systems. The integration of multi-object tracking algorithms (such as the Kalman Filter algorithm, SORT algorithm, or DeepSORT will make it easier to calculate trajectories and track drones in videos. Further improvements may include multi-sensor fusion (like the fusion of radar sensors and thermal sensors) to make the framework more robust in adverse weather conditions or at night [24,25].

Moreover, research on the Transformer-based detection model and the lightweight attention mechanism can help enhance the model's understanding of the context and remove the confusion between drones and similar objects. The extension of the dataset to include more diverse environmental factors and adversarial samples will also be beneficial in improving the generalization performance of the model. In summary, the proposed system offers a scalable and efficient platform for the development of intelligent UAV detection and monitoring systems.

References

1. Rahman MH, Sejan MAS, Aziz MA, Tabassum R, Baik JI, et al. (2024) A comprehensive survey of unmanned aerial vehicles detection and classification using machine learning approach: Challenges, solutions, and future directions. Remote Sensing 16: 879.
2. Alif MAR (2024) Yolov11 for vehicle detection: Advancements, performance, and applications in intelligent transportation systems. arXiv preprint.
3. Bochkovskiy A, Wang CY, Liao HYM (2004) Yolov4: Optimal speed and accuracy of object detection. arXiv:2004.10934.
4. Lo´pez-Gonza´lez PJ, Reyes-Gonza´lez D, Moreno-Va´zquez O, Vivar-Ocampo R, Zamora-Castro SA, et al. (2026) Inspection and evaluation of urban pavement deterioration using drones: Review of methods, challenges, and future trends. Future Transportation 6: 10.
5. Chaurasia A, Jocher G (2023) Ultralytics yolov8 documentation.
6. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only

look once: Unified, real-time object detection. in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 779-788.
7. Jocher G (2021) Yolov5: An improved yolo framework for real-time object detection. GitHub repository.
8. Tai NT, DucThang B, Hung NN (2025) Yoloxpress: A lightweight real-time unmanned aerial vehicle detection algorithm. International Journal of Systematic Innovation 9: 85-95.
9. Titu MFS, Pavel MA, Michael GKO, Babar H, Aman U, et al. (2024) Real-time fire detection: Integrating lightweight deep learning models on drones with edge computing. Drones 8: 483.
10. Bakirci M (2025) Performance evaluation of low-power and lightweight object detectors for real-time monitoring in resource-constrained drone systems. Engineering Applications of Artificial Intelligence 159.
11. Ciccone F, Ceruti A (2025) Real-time search and rescue with drones: A deep learning approach for small-object detection based on yolo. Drones 9: 514.
12. Ntousis O, Makris E, Tsanakas P, Pavlatos C (2025) A dual-stage processing architecture for uav object detection and tracking using lightweight onboard and ground-server computations. Technologies 13: 35.
13. Luo J, Chang K, Huang J, Sun X, Ji Y (2026) A uav aerial image small object detection algorithm based on fine-grained feature preservation and multi-scale feature pyramid balancing. Complex & Intelligent Systems 12: 12.
14. Svystun S, Radiuk P, Melnychenko O, Savenko O, Sachenko A, (2025) Yolo ensemble for uav-based multispectral defect detection in wind turbine components. arXiv preprint arXiv:2411.00201.
15. Tian D, Wang C, Zhou D, Yan X, Zeng L (2024) Nfe-yolo: A lightweight and efficient detection network for low, slow, and small drones. IEEE Access.
16. Maaroof MKA, Bouhlel MS (2025) Real-time object detection using yolo-8 model: A drone-based approach. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications 16: 190-204.
17. Li M, Yan N (2025) Ipd-yolo: Person detection in infrared images from uav perspective based on improved yolo11. Digital Signal Processing 105469.
18. Jegham N, Koh CY, Abdelatti M, Hendawi A (2024) Yolo evolution: A comprehensive benchmark and architectural review of yolov12, yolo11, and their previous versions. arXiv preprint arXiv:2411.00201.
19. Solovyev R, Wang W, Gabruseva T (2021) Weighted boxes fusion: Ensembling boxes from different object detection models. Image and Vision Computing 107: 104117.
20. T Dietterich (2000) Ensemble methods in machine learning, in International workshop on multiple classifier systems. Springer 1-15.
21. Everingham M, Van Gool L, Williams CK, Winn J, Zisserman A (2010) The pascal visual object classes (voc) challenge. International journal of computer vision 88: 303-338.
22. Lin TY, Maire M, Belongie S, Hays J, Perona P, et al. (2014) Microsoft coco: Common objects in context, in European conference on computer vision. Springer 740-755.
23. Rezatofighi H, Tsoi N, Gwak J, Sadeghian A, Reid I, and S. et al. (2019) Generalized intersection over union: A metric and a loss for bounding box regression. in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition 658-666.
24. Bewley A, Ge A, Ott L, Ramos F, Upcroft B (2016) Simple online and realtime tracking in 2016 IEEE international conference on image processing (ICIP). Ieee 3464-3468.
25. Wojke N, Bewley A, Paulus D (2017) Simple online and realtime tracking with a deep association metric. in 2017 IEEE international conference on image processing IEEE 3645-3649.