Development of Artificial Intelligent-Driven Security Drone System

Abstract

The creation of AI-powered drone-based security models has been fueled by the growing need for intelligent surveillance systems. In order to detect security risks in real time, this project focuses on developing an AI-driven security drone system that combines Deep Learning models with unmanned aerial vehicles (UAVs). A wide range of data, including video frames and aerial photos, were gathered and preprocessed using augmentation, normalization, and resizing. Three sophisticated object detection models, YOLOv5, YOLOv6, and YOLOv7, were then trained using this dataset. The models were evaluated using key performance metrics, including Precision, Recall, F1-score, and mean Average Precision (mAP). Results revealed that YOLOv5 achieved a Precision of 0.88, Recall of 0.86, F1-score of 0.85, and mAP of 0.805; YOLOv6 achieved a Precision of 0.95, Recall of 0.94, F1-score of 0.92, and mAP of 0.876; while YOLOv7 demonstrated superior performance, achieving a Precision of 0.98, Recall of 0.98, F1-score of 0.96, and mAP of 0.917. Furthermore, the developed models were tested in a simulated environment using Mission Planner with Software-In-The-Loop (SITL), enabling realistic flight path monitoring and real-time event detection without requiring physical deployment. Although the system was validated in a simulated environment, it lays the groundwork for future real-world applications across critical domains.The study concludes that the YOLOv7 model offers the highest accuracy and reliability for drone-based real-time surveillance and threat detection, setting a strong foundation for future deployment in security operations, disaster management, agriculture, and other sectors.

**Keyword: AI-powered Drone; Security; Deep Learning**

I I**ntroduction**

The persistent rise in crime and insecurity across Nigeria has reached alarming levels, necessitating urgent and strategic intervention. Each year, billions of Naira are allocated toward improving security ranging from the procurement of weapons to the training of security personnel. However, despite these substantial investments by both the federal and the state governments, the nation continues to grapple with increasing incidents of criminality, terrorism, and general insecurity. This troubling trend highlights a critical shortfall in the effectiveness of current security strategies, which are often characterized by high costs, operational inefficiencies, and inadequate equipment. In many cases, criminal activities go undetected and unaddressed, even with the deployment of sophisticated tools such as surveillance helicopters and ground troops. These conventional methods often fail to provide timely intelligence, resulting in loss of lives, destruction of property, and erosion of public trust in the nation's security infrastructure.

Given the limitations of existing approaches, there is an urgent need to explore and implement modern technological solutions capable of addressing the dynamic and complex nature of contemporary security threats. The integration of Unmanned Aerial Vehicles (UAVs) commonly known as drones with Artificial Intelligence and Internet of Things (IoT) technologies offers a promising path forward. UAVs present a cost-effective, scalable, and efficient alternative to traditional security measures. Their ability to reach inaccessible areas, conduct real-time surveillance, and capture high-resolution imagery from various altitudes makes them invaluable in both urban and remote security operations. So many research have be adopted, [1] Zhu *et al.* (2021) proposed TPH-YOLOv5, an enhanced object detection model for drone imagery, This advancement led to a 7% improvement over the baseline model and secured 5th place in the VisDrone 2021 Challenge. However, performance may vary with different datasets.[2] Chung *et al.* (2020) conducted a comprehensive survey on UAVs for civil applications. Their methodology involved analyzing existing literature to understand the development and future trends of UAVs in civil sectors.[3] Singh *et al.* (2018) developed a real-time drone surveillance system for identifying violent individuals. Utilizing a ScatterNet Hybrid Deep Learning Network, they focused on human pose estimation and violence detection in aerial images. [4] Wang *et al.* (2018) introduced a system for visible and thermal drone monitoring using convolutional neural networks. They developed data augmentation techniques to address the scarcity of training images, enhancing drone detection capabilities in both visible and thermal spectrums.[5] Yucesoy *et al.* (2025) reviewed the role of drones in disaster response. By categorizing drone applications into information collection, delivery, and communication network recovery.[6]Rawle *et al.* (2025) explored integrating drone technology in STEM education through the DTESC program [7] Li *et al.* (2025) proposed a joint precoding and artificial noise design framework to enhance physical-layer security in UAV communications [8] Selim and Kamal (2018) addressed post-disaster 4G/5G network rehabilitation using drones. They proposed solutions for challenges like limited battery energy and backhauling by utilizing different types of drones for communication, power supply, and backhaul. Simulation results showed the potential for providing unlimited cellular service to disaster-affected regions under certain conditions [9] **Hassija *et al.* (2021)** conducted a comprehensive survey on fast, reliable, and secure drone communication [10] Berini *et al.* (2023) introduced a hyperelliptic curve-based anonymous lightweight authentication (HCALA) scheme designed for both drones and users [11] Biregani *et al.* (2021) proposed a two-phase security model to address communication vulnerabilities in UAV networks [12] Bera *et al.* (2020) developed a blockchain-based access control algorithm for the IoD to secure drone-to-drone and drone-to-ground station communications. Their scheme collects sensitive data from transactions through ground stations, converts them into blocks, and adds these blocks to the blockchain using the Ripple Protocol Consensus Algorithm (RPCA) via a peer-to-peer cloud server.[13] Palossi *et al.* (2018) developed a deep neural network (DNN)-based visual navigation engine for autonomous nano-drones [14] Rovira-Sugranes *et al.* (2021) reviewed AI-enabled routing protocols for UAV networks, focusing on trends, challenges, and future outlook. They conducted a comprehensive literature review on learning-based networking and routing algorithms for UAV networks. [15] Joshi *et al.* (2024) investigated AI-enabled autonomous navigation and decision-making for defense security drones This paper proposes the development of a Security Drone System with IoT Integration, a cutting-edge solution designed to enhance surveillance, threat detection, and rapid response capabilities. The system is equipped with advanced sensors and high-definition cameras that continuously collect environmental data. This data is transmitted in real time to a central monitoring station where it is analyzed by security personnel to assess potential threats. Based on these insights, informed decisions can be made quickly and accurately, ensuring timely intervention and minimizing risks.

**II. Methodology**

The methods utilized in achieving this research work combines observational and experimental methods. The observational method was used for data collection and characterization, while experimental method was used for the model implementation, evaluation, and validation.

***To develop a comprehensive data model through meticulous data collection and preparation***

Aerial images and videos were collected from various locations across the South-East region, comprising over 200 high-resolution images and 100 video clips. These data samples capture diverse environmental and urban scenes essential for training robust surveillance models. To further improve training efficiency and model generalizability, the dataset was augmented with additional aerial imagery and video datasets obtained from online repository Kaggle. This combined dataset ensures greater variability in terrain, lighting conditions, and object scales, factors critical to the effectiveness of AI-based detection and classification in real-world drone surveillance applications.

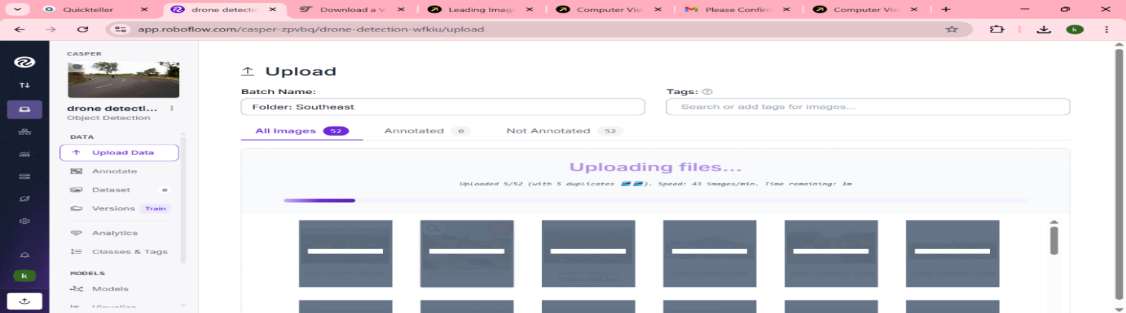


Figure 1: Uploading of data in the Roboflow Environment

The collected images, along with the images extracted from video frames, undergo annotation, which involves labeling parts of an image to inform the AI model about what objects are present and where they are located. When training an AI model for tasks like object detection, classification, or segmentation, it needs to learn from labeled examples. Roboflow is a beginner-friendly web tool for annotating images and preparing datasets for AI. The images are uploaded into the web tool for annotation. Roboflow has bounding box tool which is used to draw a box around the object. After drawing, a box will appear to enter the label name.

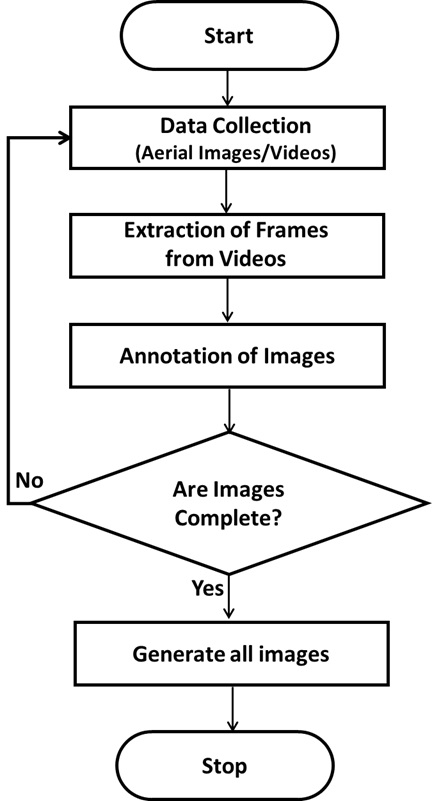


Figure 2: Flowchart for Collection of Data

Figure 2 illustrates the steps for collecting and preparing the dataset, beginning with the Start node. The first step involves data collection, where aerial images and videos are gathered using drones or similar sources. Once the raw video data is obtained, the next step is the extraction of frames from the videos, converting continuous footage into a series of static images. Following frame extraction, the process moves to the annotation of images, where objects of interest (such as vehicles or people) are manually or semi-automatically labeled to provide ground truth for training. After annotation, the workflow reaches a decision point asking: "Are images complete?" - assessing whether all required frames are adequately annotated and ready. If the images are not complete, the process loops back to complete the annotations. Once the dataset is verified as complete, the system proceeds to generate all images, finalizing the preparation of the dataset. The process concludes at the Stop node, indicating the dataset is ready for further analysis.

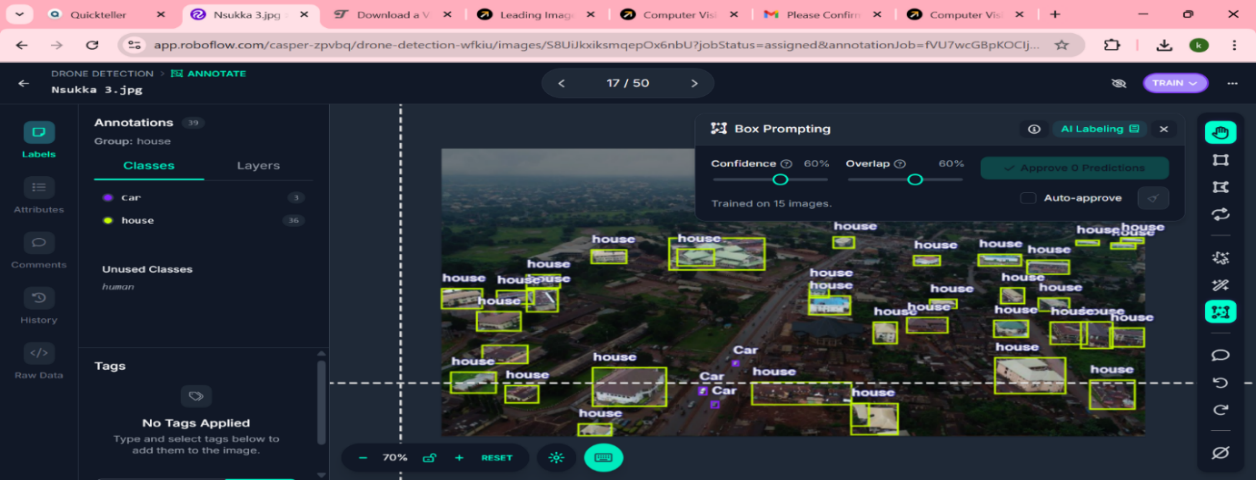


Figure 3: Annotation of images in the RoboFlow Environment

1. ***Gathering and processing of dataset through resizing, normalizing and augmentation***

The collected dataset underwent a series of processing steps to build a suitable dataset for training machine learning algorithms. This process commences with resizing the images, followed by normalizing the images and finally augmentation of the dataset. Through these processes, the data was effectively transformed for optimal use in machine learning models.

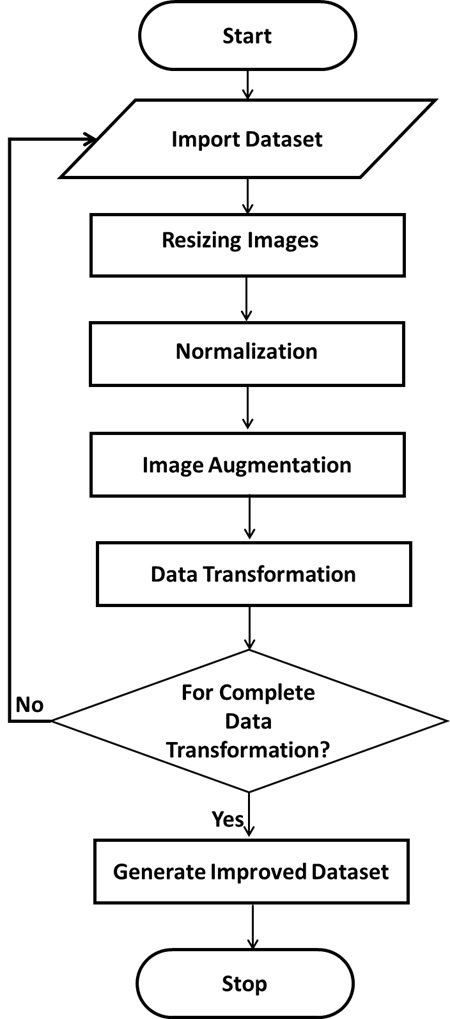
****

Figure 4: Flowchart for the Data Preprocessing Steps

Figure 4 illustrates the steps involved in preprocessing the dataset to prepare it for a deep learning model. The process begins with Start, where the system is initialized. Next, the Import Dataset step involves loading the raw dataset, including images and their annotations. Once imported, the images go through Resizing, adjusting them to a fixed resolution (such as 640×640 pixels), which is essential for consistent input during training. Following resizing, Normalization is performed to scale the pixel values, usually to a range between 0 and 1, enhancing the convergence rate and model stability. After normalization, Image Augmentation techniques such as flipping, rotation, and scaling are applied to artificially increase dataset variability and reduce overfitting. Then comes Data Transformation, where the images and labels are further processed to match YOLO’s input format. A decision is made at the For Complete Data Transformation? Step to check whether all preprocessing requirements are fulfilled. If not, the process loops back to continue transforming the dataset. Once all transformations are complete, the process advances to Generate Improved Dataset, finalizing the preprocessed data and making it ready for training. Finally, the Stop step marks the end of the preprocessing pipeline.

***To propose three machine learning model suitable for detecting and classifying events in real time***

This section identifies three Artificial Intelligence models, YOLOv5, YOLOv6, and YOLOv7 for implementation and evaluation. These models were deliberately chosen because they were not utilized in the reviewed literature, yet they represent some of the most advanced and efficient object detection algorithms currently available. Known for their speed and accuracy, particularly when handling large datasets, these models offer significant improvements over earlier versions. The primary aim of adopting these models in this research is to conduct a comparative evaluation to determine which performs best under the given conditions. Based on the outcomes, informed recommendations will be made regarding the most suitable model for similar applications.

***YOLOv5 (You Only Look Once version 5)***

YOLOv5 (You Only Look Once version 5) was utilized in this study as one of the deep learning models selected for the development of the drone-based surveillance and threat detection system. YOLOv5 is a state-of-the-art, real-time object detection algorithm that builds upon the core principles of the original YOLO architecture—prioritizing speed, accuracy, and efficiency. As a single-stage detector, YOLOv5 performs object classification and localization simultaneously, which allows for faster inference compared to two-stage detectors such as Faster R-CNN. The YOLOv5 model processes the entire image in one forward pass through the neural network, dividing the input image into a grid and predicting bounding boxes and class probabilities for each cell. These predictions are then refined using techniques such as anchor boxes and non-maximum suppression (NMS) to ensure accurate localization and to eliminate redundant detections.

In this implementation, aerial images collected from drone footage serve as the input to the model. The images are preprocessed and passed through the convolutional layers of YOLOv5, which automatically extract relevant features such as shapes, edges, and object structures. The architecture of YOLOv5 includes a backbone for feature extraction, a neck for feature aggregation, and a head that outputs the final predictions of object classes and bounding boxes.

During training, the model adjusts its internal weights to minimize the loss function, which combines classification, localization, and objectness errors. The training process continues iteratively until the model achieves optimal performance based on predefined metrics such as mean Average Precision (mAP), precision, and recall.

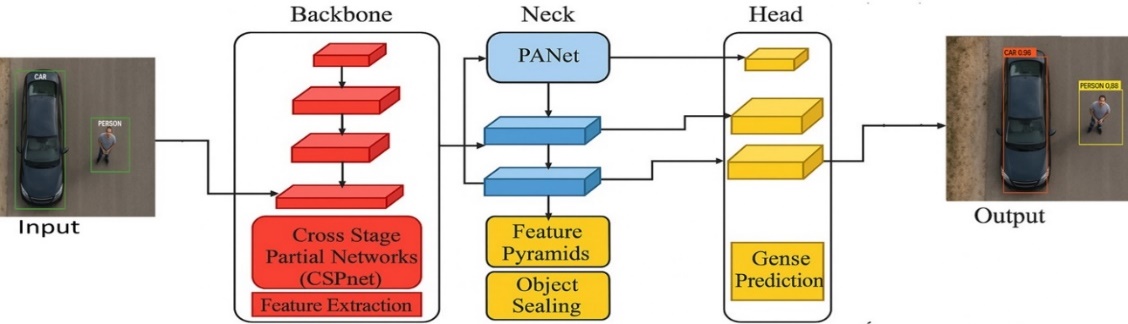


Figure 5: Architectural diagram of YOLOv5

Figure 5 illustrates the architecture of YOLOv5, which begins with an input image containing objects of interest such as a car and a person. The image is first processed through the Backbone, which employs Cross Stage Partial Networks (CSPnet) to perform feature extraction, generating essential low-level and high-level features from the input. These extracted features are passed into the Neck module, implemented using Path Aggregation Network (PANet). PANet enhances feature propagation and spatial information through multiple paths. It outputs enhanced features via Feature Pyramids and Object Scaling, helping the network detect objects at different scales and contexts. Next, the processed features are fed into the Head, which handles the final dense prediction tasks. This includes predicting class labels and bounding boxes. The head processes feature maps at multiple levels, allowing it to detect objects of varying sizes. Finally, the Output section shows the results, where each detected object is enclosed in a bounding box with a class label and a corresponding confidence score (e.g., "CAR 0.96" and "PERSON 0.88"). This comprehensive pipeline results in accurate object detection from aerial imagery. The algorithm of the YOLOv5 is presented as:

***Algorithm 1: YOLOv5***

1. ***Start***
2. ***Load Dataset***
   * ***Import annotated images (with bounding boxes and labels).***
   * ***Split dataset into training, validation, and test sets.***
3. ***Preprocess Input Images***
4. ***Initialize YOLOv5 Model Architecture***
5. ***Train the Model (Forward + Backward Pass)***
   * + ***For each training batch:*** 
       - ***Pass input images through the network (forward pass).***
       - ***Generate predictions for bounding boxes , objectness , and class labels .***
       - ***Calculate the total loss using the components:***
       - ***Localization loss (for bounding box regression):***
       - ***Objectness loss (Binary Cross-Entropy):***
       - ***Classification loss (Cross-Entropy for multi-class):***

* ***Total Loss:***

***(where are loss weights)***

* ***Perform backpropagation to update weights.***

1. ***Validate the Model***
   * ***Run validation data through the model.***
   * ***Compute performance metrics: precision, recall, mAP (mean Average Precision).***
2. ***Repeat Training Steps for several epochs until the model converges.***
3. ***Evaluate model on unseen test data to assess generalization.***
4. ***Make Predictions***
   * ***Feed new drone images into the trained YOLOv5 model.***
   * ***Output includes detected objects with bounding boxes, confidence scores, and class labels.***
5. ***End***

***YOLOv6 (You Only Look Once version 6)***

YOLOv6 (You Only Look Once version 6) was utilized in this study as one of the deep learning models selected for developing the drone-based surveillance and threat detection system. YOLOv6 is an advanced real-time object detection algorithm that introduces several architectural improvements over previous YOLO versions, aiming to boost accuracy while maintaining efficient inference speed. It is particularly optimized for industrial applications where performance and precision are both critical.

YOLOv6 adopts a one-stage detection pipeline, allowing it to detect objects in a single pass through the network. This enables faster detection and real-time capabilities, which are essential for applications like live drone surveillance. It incorporates an EfficientRep Backbone for better feature representation and introduces Rep-PAN as the neck structure to enhance multi-scale feature fusion, crucial for detecting objects at different sizes in aerial imagery. The input to YOLOv6 consists of preprocessed aerial images or video frames captured from drones. These inputs are resized and normalized before being fed into the model. During forward propagation, the backbone extracts spatial features, the neck aggregates these features across different scales, and the head outputs class probabilities, objectness scores, and bounding box coordinates. YOLOv6 improves training stability and accuracy through advanced components such as label assignment strategies (SimOTA), improved loss functions, and knowledge distillation techniques. The training process involves minimizing a multi-part loss function that balances localization, classification, and confidence errors.

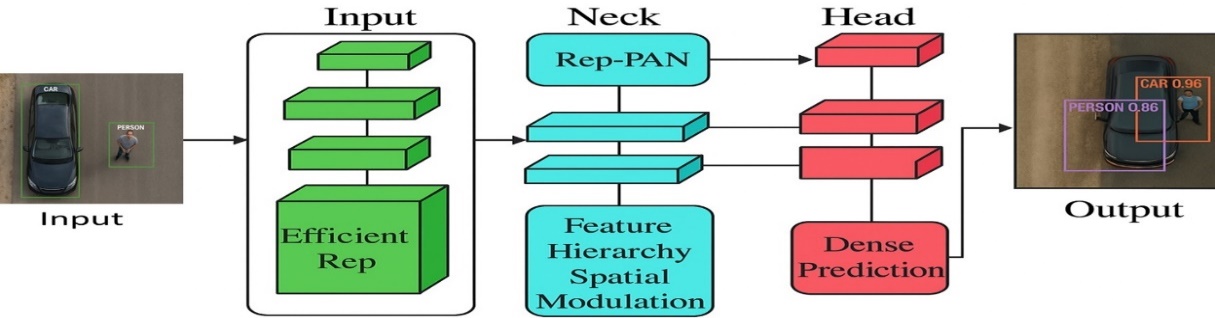


Figure 6: Architectural diagram of YOLOv6

Figure 6 illustrates the architectural diagram of the YOLOv6 object detection model, which begins with an input image containing objects such as a car and a person. The image is first processed through the Backbone, which utilizes EfficientRep for efficient feature extraction. This stage captures low-level and high-level features essential for identifying visual patterns in the image. These features are passed to the Neck, which is composed of Rep-PAN (a refined version of PANet) and a Feature Hierarchy Spatial Modulation module. Rep-PAN enhances multi-scale feature fusion, while the spatial modulation component strengthens the spatial representation and helps in better object localization across different scales. The transformed features are then forwarded to the Head, where a Dense Prediction module carries out the final object classification and bounding box regression. The head operates across feature maps to detect objects at various resolutions, ensuring robustness across object sizes. Finally, the Output displays the predicted bounding boxes over the input image, labeled with object classes and corresponding confidence scores (e.g., "CAR 0.96" and "PERSON 0.88"), indicating the model’s certainty. This modular pipeline allows YOLOv6 to achieve high-speed, high-accuracy object detection suitable for real-time applications. The algorithm of the YOLOv6 is presented as:

***Algorithm 2: YOLOv6***

1. ***Start***
2. ***Import annotated aerial images.***
3. ***Split the dataset into training, validation, and test sets.***
4. ***Preprocess Input Images***
5. ***Initialize YOLOv6 Architecture***
6. ***Train the Model (Forward and Backward Passes)***
   * ***For each training batch:  
     a. Pass images through the YOLOv6 model (forward pass).  
     b. Predict bounding boxes , objectness , and class labels .***

***c. Compute the total loss, using the components:***

* + - * ***Localization loss (for bounding box regression):***
      * ***Objectness loss (Binary Cross-Entropy):***
      * ***Classification loss (Cross-Entropy for multi-class):***
* ***Total Loss:***

***(where are loss weights)***

***d. Update model weights using backpropagation and optimizer***

1. ***Validate the Model***
   * ***Evaluate on the validation set at the end of each epoch.***
   * ***Calculate metrics: Precision, Recall, F1-score, mAP@0.5, mAP@0.5:0.95.***
2. ***Repeat Steps 6 and 7 over multiple epochs until convergence***
3. ***Test the Trained Model***
4. ***Make Predictions on New Data***
   * ***Feed unseen drone images into the trained YOLOv6 model.***
   * ***Get output with detected objects and bounding boxes, confidence scores, and class labels.***
5. ***End***

***YOLOv7 (You Only Look Once version 7)***

YOLOv7 (You Only Look Once version 7) was also employed in this study as part of the comparative evaluation for developing a drone-based surveillance and threat detection system. YOLOv7 represents one of the most optimized versions of the YOLO family, achieving state-of-the-art accuracy and efficiency in object detection tasks. It introduces innovative training strategies and architectural optimizations that significantly enhance performance on both accuracy and speed fronts. As a single-stage detector, YOLOv7 processes the input image in one pass, predicting multiple bounding boxes and class probabilities simultaneously. It uses Extended Efficient Layer Aggregation Networks (E-ELAN) as part of its architecture, which improves the model’s ability to learn deeper and more complex features without sacrificing speed. Additionally, YOLOv7 incorporates model re-parameterization techniques and advanced attention mechanisms to refine feature extraction and improve detection of small or overlapping objects, an essential need in drone imagery. The model was trained using annotated aerial images extracted from drone-captured video data. The images were resized, normalized, and augmented before being fed into the YOLOv7 training pipeline. The training process involved optimizing a compound loss function that includes bounding box regression, object classification, and confidence score errors.

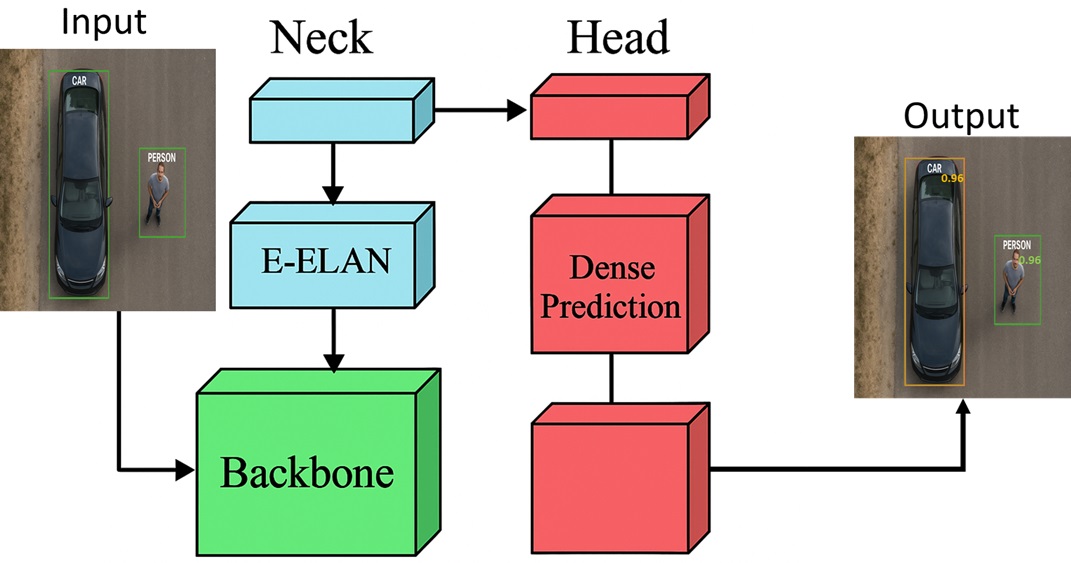


Figure 7: Architectural diagram of YOLOv7

Figure 7 depicts the architectural diagram of YOLOv7, which begins with the Input stage, where the drone captures an image containing objects of interest, such as a car and a person. This image is first passed into the Backbone, a deep convolutional neural network responsible for extracting important features. The backbone acts as the initial feature extractor that transforms raw pixels into meaningful representations. Next, the processed feature maps from the backbone flow into the Neck, which includes components like E-ELAN (Extended Efficient Layer Aggregation Network). This module enhances feature propagation and reusability while maintaining the network’s learning capability. The neck is essential for aggregating features across different scales to ensure the network can detect objects of varying sizes. The output from the neck is then directed into the Head of the network, where the actual object detection takes place. The head contains multiple layers, including a Dense Prediction module, which is responsible for interpreting the aggregated features to predict object classes, bounding box coordinates, and confidence scores. Finally, the Output of the architecture shows detected objects such as a car and a person, both annotated with bounding boxes and confidence scores, illustrating YOLOv7's high accuracy and effectiveness in real-time object detection tasks. The algorithm of YOLOv7 is presented as:

**Algorithm 3: YOLOv7**

1. **Start**
2. **Load Dataset**
   * Import annotated aerial images and video frames.
   * Split the dataset into training, validation, and testing sets.
3. **Preprocess Input Images**
4. **Initialize YOLOv7 Architecture**
5. **Train the Model**  
   For each batch of training data:  
   a. Perform a **forward pass** through the model to predict bounding boxes, class scores, and objectness.  
   b. Compute the **composite loss**, which includes:

Where:

* + **Localization Loss** (bounding box regression)
  + **Confidence Loss** (objectness)
  + **Classification Loss** (object category prediction)

c. Perform a **backward pass** to calculate gradients.  
d. Update model weights using an optimizer like SGD with momentum or Adam.

1. **Validate During Training**
2. **Repeat Training** over several epochs until the model converges or early stopping conditions are met.
3. **Test the Trained Model**
4. **Deploy Model for Inference**
   * Feed new aerial images from drones into the trained YOLOv7 model.
   * The output includes predicted bounding boxes, confidence scores, and class labels
5. **End**
6. ***To develop a drone-based detection model to detect or classify events in real time***

To achieve this, a drone-based detection system was developed by independently training three advanced machine learning models: **YOLOv5, YOLOv6,** and **YOLOv7.** Each model was trained separately to create a robust and accurate detection framework. By leveraging the unique strengths of these algorithms, the system enhances **surveillance capabilities**, improves **threat detection accuracy**, and supports **faster response times** in real-time operational environments.

***Training of the YOLOv5 algorithm***

The dataset was processed using the YOLOv5 algorithm, a real-time object detection framework designed to accurately localize and classify objects within images. YOLOv5 utilizes a single convolutional neural network to directly predict bounding boxes and class probabilities from full images in a single evaluation, enabling high-speed detection. The training process began with the initialization of a pre-trained YOLOv5 model, typically fine-tuned on the target dataset. The input images were first resized and normalized, and corresponding annotation files were formatted in YOLO’s required structure. During training, the model learned to identify spatial features through a series of convolutional layers, leveraging a combination of backbone, neck, and head architectures. The backbone (usually CSPDarknet) extracted key visual features, while the neck (PANet) aggregated feature maps across multiple scales. The head then predicted bounding boxes, object confidence scores, and class probabilities.

Loss functions comprising localization loss, objectness loss, and classification loss were used to guide the optimization process. The model was iteratively trained using backpropagation and gradient descent, adjusting its weights to minimize the combined loss across training samples. Data augmentation techniques such as mosaic augmentation, random flipping, and scaling were applied to improve generalization and robustness. Training continued until a predefined number of epochs were completed or early stopping conditions were met, based on validation loss. The final YOLOv5 model, upon successful training and validation, was deployed as part of the drone-based detection system to enable accurate real-time surveillance, threat identification, and rapid response. The training process is illustrated in the flow chart of Figure 8, outlining the steps taken to develop the object detection model for drone-based applications.

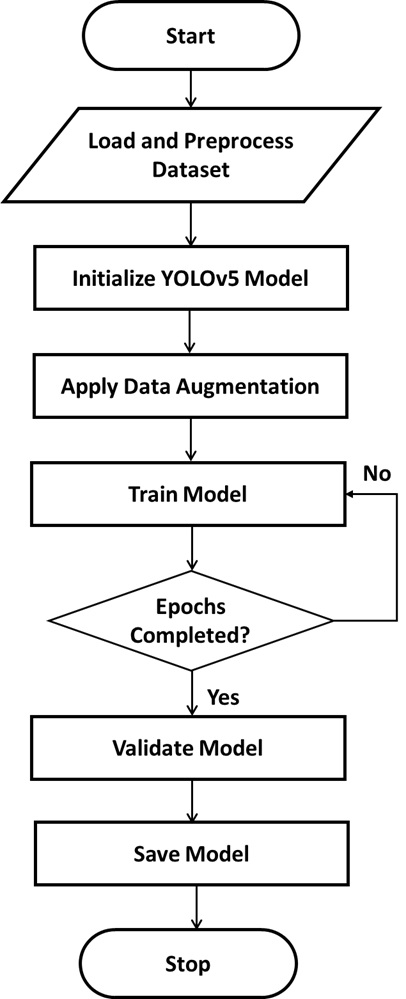


Figure 8: Training of the YOLOv5 Algorithm

***Training of the YOLOv6 algorithm***

The dataset was also processed using the YOLOv6 algorithm, a high-performance object detection model optimized for industrial applications and real-time edge deployment. YOLOv6 builds upon the strengths of previous YOLO versions while introducing architectural enhancements and training strategies to improve detection accuracy and speed, especially for dense and complex scenes. The training process began by preparing the dataset with properly annotated images, followed by data preprocessing steps such as image resizing, normalization, and augmentation. YOLOv6 adopted an enhanced backbone (EfficientRep) to extract features with improved computational efficiency. The model incorporated a re-parameterization technique to separate training and inference architectures, thereby allowing more powerful learning during training and faster inference afterward. During training, the algorithm minimized a compound loss function that considered bounding box regression, objectness score, and class prediction errors. The model was trained over multiple epochs using stochastic gradient descent (SGD) with momentum. Additionally, YOLOv6 applied training tricks like anchor-free detection heads, label assignment optimization, and strong data augmentation to boost performance. The training continued until convergence or early stopping based on validation performance. Upon successful validation, the final YOLOv6 model was integrated into the drone-based detection system, enhancing its capability for real-time surveillance, rapid threat identification, and improved operational efficiency. The overall training process is depicted in the flow chart of Figure7.

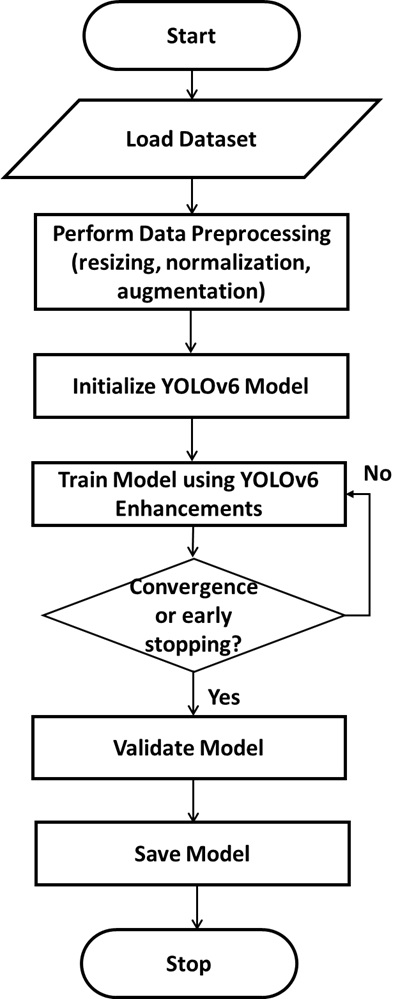


Figure 9: Training of the YOLOv6 Algorithm

***Training of the YOLOv7 algorithm***

The YOLOv7 algorithm was also employed to process the dataset, offering cutting-edge accuracy and speed improvements. YOLOv7 introduced a suite of architectural and optimization innovations, including model scaling techniques, efficient layer aggregation networks (ELAN), and dynamic label assignment strategies, making it one of the most accurate real-time object detectors available. The training began by initializing the YOLOv7 model and formatting the dataset according to the required annotation structure. The input images were preprocessed through standard augmentation methods and normalized to improve model generalization. The YOLOv7 architecture included a sophisticated backbone for feature extraction and a compound-scaled neck and head to support multi-scale detection. YOLOv7 utilized a compound loss function incorporating bounding box regression, classification loss, and objectness loss. During each training iteration, the algorithm performed backpropagation to adjust model weights and minimize prediction errors. YOLOv7’s training strategy included techniques such as coarse-to-fine label assignment, task alignment learning, and model re-parameterization for enhanced accuracy. The training process proceeded over multiple epochs, with regular validation steps to monitor performance metrics such as precision, recall, and mAP (mean Average Precision). After satisfactory validation, the final YOLOv7 model was deployed in the drone system to enable high-precision object detection, situational awareness, and real-time threat monitoring. The detailed training workflow is illustrated in Figure 9.

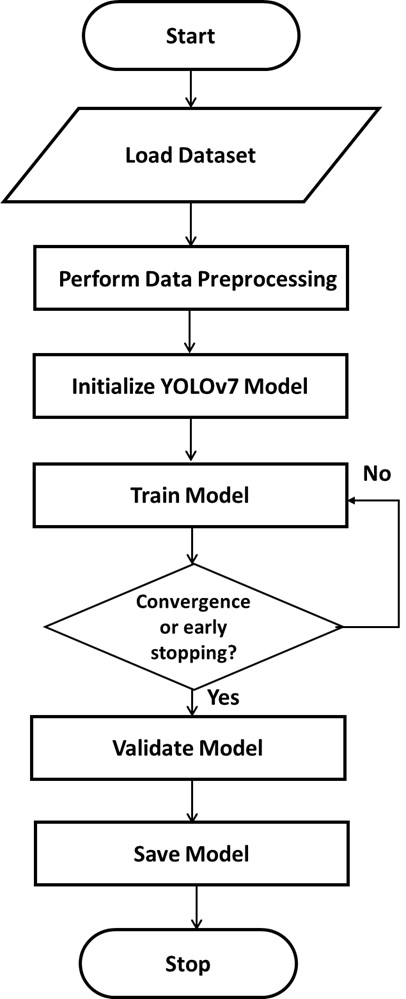


Figure 10: Training of the YOLOv7 algorithm

***The drone-based real-time event detection model***

The dataset, consisting of annotated aerial images and video frames capturing various events and objects, was imported into a data processing tool. Prior to model training, preprocessing techniques and data augmentation were applied to enhance the dataset’s quality and diversity. This step ensured that the model would be robust and generalize well across different environmental conditions, altitudes, and object scales commonly encountered in drone-based surveillance. Once the dataset was prepared, multiple deep learning algorithms such as YOLOv5, YOLOv6, and YOLOv7 were trained individually to detect and classify objects and activities captured by drones. Each model was fed the preprocessed input data and learned to associate spatial and contextual features with specific event types, such as vehicle movement, human presence, or unusual activities. The parameters of each model were optimized to improve real-time inference speed and detection accuracy. Following the training phase, each model was evaluated using a separate validation set containing unseen drone footage with known ground-truth annotations. The models’ outputs - bounding boxes and class labels, were compared against the actual labeled data to assess performance. When an event was detected, the model predicted its type and location in real time. For instance, if the input indicated a human near a restricted area, the model would classify it accordingly and output bounding box coordinates with a confidence score. The final outputs were categorized based on predefined event classes, and each prediction was validated using key performance metrics such as precision, recall, and mean Average Precision (mAP). These metrics provided insight into the model’s ability to accurately detect and classify events across diverse scenarios. This evaluation confirmed the effectiveness and operational readiness of the drone-based model for real-time surveillance, threat identification, and rapid response deployment.

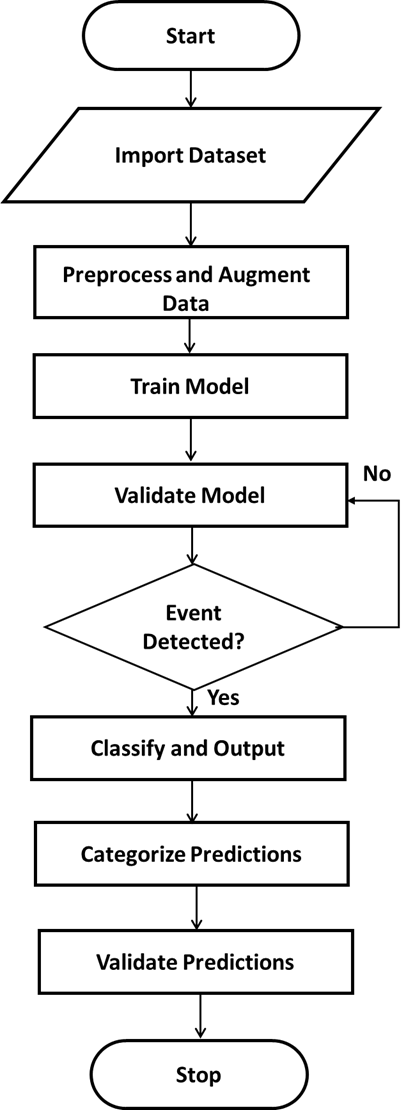


Figure 11: The Drone-Based Real-Time Event Detection Model

***To evaluate the accuracy and performance of the developed drone-based real-time event detection model, various performance indicators were employed to assess the model’s efficacy in identifying and classifying real-time events accurately***

The results considered standard metrics such as precision, recall, F1 score, and mean Average Precision (mAP), to provide a comprehensive analysis of detection effectiveness.

**Precision:** Precision is the ratio of correctly predicted positive detections (e.g., detecting a person or vehicle) to the total predicted positives. It reflects how accurate the model's positive predictions are. In drone-based event detection, precision is especially important when false alarms (false positives) must be minimized, for instance, mistakenly identifying harmless objects as threats. A high precision indicates that most detected events are indeed valid and relevant.

**Recall:** Recall is the ratio of correctly predicted positive instances to all actual positive instances in the data. It measures the model’s ability to identify all relevant objects or events. In surveillance and threat detection, a high recall means the model can successfully detect most of the important events (e.g., intrusions, unauthorized presence), ensuring that few critical incidents are missed.

**F1 Score:** The F1 Score is the harmonic mean of precision and recall, balancing the trade-off between the two. It is especially useful when the dataset is imbalanced, or when both false positives and false negatives carry significant consequences. For drone-based detection, the F1 score gives an overall sense of model robustness in correctly detecting and classifying events.

**Mean Average Precision (mAP):** mAP is the mean of the average precision across all classes and IoU (Intersection over Union) thresholds. It evaluates both the classification and localization performance of the model. In the context of drone-based event detection, mAP quantifies how well the model detects and localizes multiple objects and events across different categories (e.g., person, car, suspicious activity), making it a key metric for evaluating object detection models like YOLO.

***To simulate a drone flight in a virtual environment, integrate the trained model and run tests under different conditions***

To simulate a drone flight in a virtual environment using the developed drone model, the process involves integrating the developed drone-based detection model into a simulation platform such as Mission Planner. The virtual drone is configured with a simulated camera that streams video or image frames to the detection model in real time. As the drone follows a pre-defined or manual flight path within the virtual environment, the camera captures visual data that is continuously fed into the trained detection model. The model processes each frame to detect and classify events or objects, displaying the results (such as bounding boxes and labels) either within the simulation interface or through an external visualization tool. This setup allows real-time testing and validation of the detection model’s performance in various simulated scenarios, mimicking real-world drone operations without the risks or constraints of physical flight.

To integrate a trained object detection model into Mission Planner and run tests under different simulated conditions, you typically follow a process that combines flight simulation with real-time visual processing. The integration begins by setting up a SITL (Software-In-The-Loop) environment using a simulator, which emulates the drone's flight behavior in Mission Planner. The virtual drone is equipped with a simulated camera sensor that captures video feeds during the simulated mission. This video stream is redirected to the trained object detection model, which is run externally in a Python script. You can simulate different flight conditions such as time of day, altitude, weather, or object placement by modifying parameters in the simulation environment. This allows testing the model’s robustness and performance under varying scenarios without needing a physical drone. Mission Planner acts as the GCS (Ground Control Station), monitoring telemetry data, and controlling flight paths, while the trained model provides real-time event detection and feedback for mission-based decisions or alerts. This setup enables thorough validation and tuning of the drone system in a controlled, repeatable virtual setting.

**V. Result and Discussion**

This section presents the results for the three Models; YOLOv5, YOLOv6, and YOLOv7, which were trained on the preprocessed and annotated drone dataset. The evaluation incorporates essential performance indicators for real-time object detection models, such as precision, recall, f1 score, and mAP (mean Average Precision), to provide a comprehensive performance analysis. Figure 12 presents the Precision for the three models; YOLOv5, YOLOv6, and YOLOv7 trained for real-time drone-based event detection. This plot is used to measure how the precision of each model evolves throughout the training process. Precision, in this context, represents the proportion of correctly identified objects (true positives) out of all objects predicted by the model (true positives + false positives). The precision illustrates the model's ability to make accurate positive predictions. A consistently high or steadily rising precision value during training indicates that the model is learning to distinguish relevant targets effectively and reducing the number of false alarms over time. This is particularly important in drone surveillance, where false detections could lead to wasted resources or misinformed decisions.

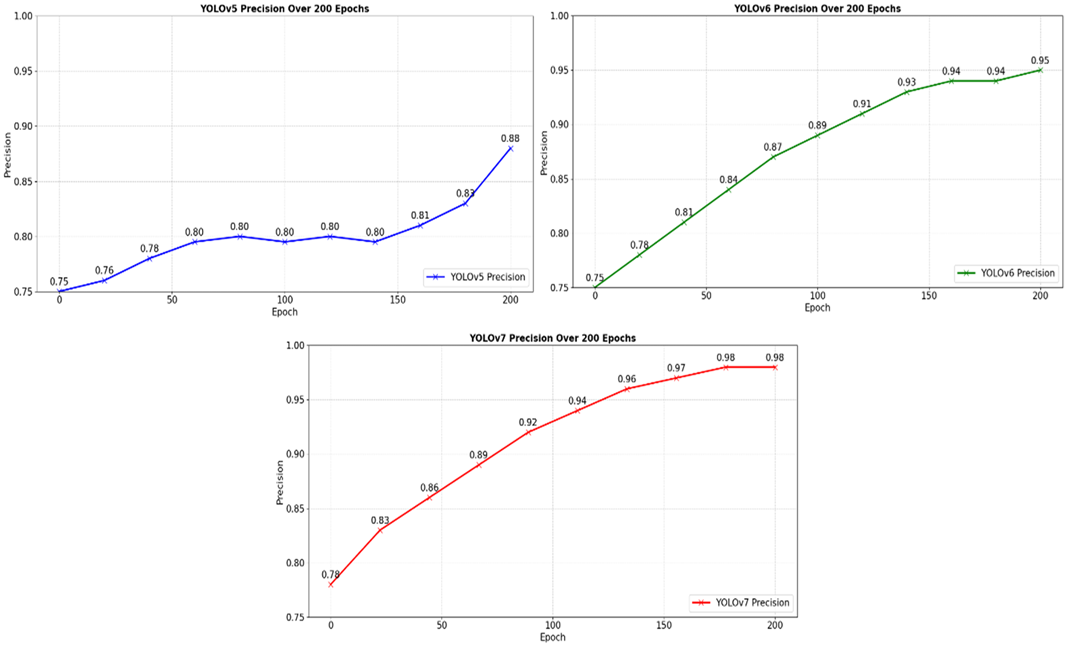


Figure 12: Results of Precision for the three models

Figure 12 presents precision over 200 epochs for the three models: YOLOv5, YOLOv6, and YOLOv7. Each plot shows how precision improves during training, reflecting how accurately each model identifies objects without producing false positives. In the first plot, the YOLOv5 model starts with a precision of 0.75. The curve gradually rises, reaching 0.88 by the 200th epoch. There is a noticeable plateau between epochs 50 and 150 where precision hovers around 0.80, before increasing again. This trend suggests that YOLOv5 learns effectively but experiences slower improvements in precision over time. In the second plot, the YOLOv6 model also starts at 0.75 but shows a steady and consistent increase in precision throughout the training. By epoch 50, the precision climbs to 0.84 and continues to improve, reaching 0.95 by the final epoch. The smooth upward trend indicates that YOLOv6 generalizes better and improves its object detection confidence more efficiently than YOLOv5.In the third plot, the YOLOv7 model begins at a slightly higher precision of 0.78 and demonstrates the fastest and most consistent gain in precision. By epoch 100, it already reaches 0.96 and plateaus around 0.98 through the remaining epochs. This rapid rise and high final precision underscore YOLOv7’s strong learning capability, suggesting it is the most effective at reducing false positives and accurately detecting objects. Overall, while all models improve over time, YOLOv7 shows the highest and fastest precision gains, followed by YOLOv6, with YOLOv5 trailing in performance. This visual comparison highlights the advancements in YOLOv7’s architecture and training strategy, making it the most efficient and reliable for tasks requiring high-precision real-time object detection.

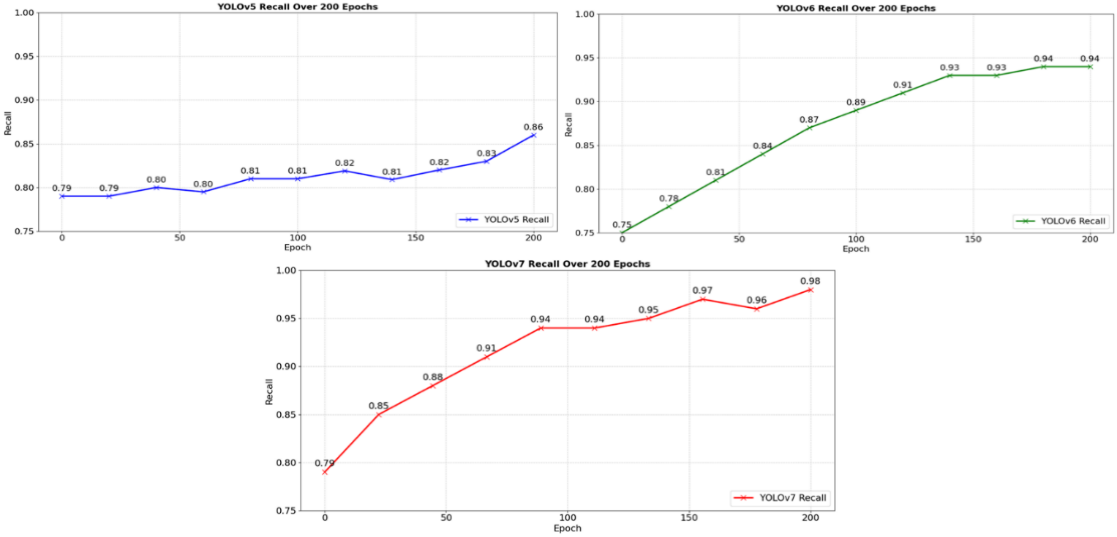


Figure 13: Results of Recall for the three models

Figure 13 illustrates the recall performance across 200 epochs for the YOLOv5, YOLOv6, and YOLOv7 models. Recall, also referred to as sensitivity, measures the ability of a model to correctly identify actual positive cases. In classification tasks, achieving high recall is crucial, particularly in scenarios where missing positive instances carries significant consequences, such as wrong target. A high recall value indicates that the model is effectively capturing the majority of true positive cases. Each graph presents how effectively each model learns to identify true positive detections over time. In the YOLOv5 recall plot, the model starts with a recall of 0.79. Over the training process, it shows slight fluctuations and a gradual increase, ending at 0.86 by the 200th epoch. The overall trend suggests that while YOLOv5 learns steadily, its recall improvement is modest, and the learning curve remains relatively flat in the middle epochs, possibly indicating a period of learning saturation or instability. The YOLOv6 recall plot begins at 0.75 but follows a steeper and more consistent upward trajectory. The model reaches 0.89 around epoch 100 and continues to improve slightly, ending at 0.94. This curve reflects a stable and efficient recall learning process, with fewer fluctuations and more consistent gains compared to YOLOv5. In the YOLOv7 recall plot, the model starts at 0.79 and shows rapid improvement, climbing to 0.94 by epoch 100 and ultimately reaching 0.98 at epoch 200. This near-perfect recall performance demonstrates that YOLOv7 learns very quickly and effectively, with minimal false negatives. Its smooth and steep curve indicates strong recall capability, making it the most reliable model for detecting all relevant events in real-time applications. In summary, Figure 14 highlights the progression of recall during training, comparing how well each YOLO model captures true positive cases over time. YOLOv7 achieves the highest recall, followed by YOLOv6 and then YOLOv5. This suggests that, for tasks where capturing as many positive cases as possible is critical such as real-time event detection in drone-based systems, YOLOv7 may be the most effective choice among the three models.

****

**Figure 14: Results of F1 score for the three models**

Figure 13 shows the F1 score performance over 200 epochs for three different YOLO models: YOLOv5, YOLOv6, and YOLOv7. The F1 score is a metric that combines both precision and recall, offering a balanced measure of a model’s performance, especially when dealing with class imbalances. It ranges from 0 to 1, with values closer to 1 indicating better performance in correctly identifying both positive and negative cases. Each plot illustrates how the F1 score improves over the course of training epochs. In the YOLOv5, the F1 score starts at 0.76 at epoch 0 and gradually improves, reaching about 0.85 at epoch 200. The improvement is steady but relatively slow, and the curve remains quite flat, showing only mild gains over time. In the YOLOv6, the F1 score starts a little higher at 0.79. There is a steeper growth early on, reaching about 0.88 around 50 epochs. It then continues to improve steadily and stabilizes around 0.92 by epoch 200. YOLOv6 shows faster and higher improvement compared to YOLOv5. The plot for YOLOv7 shows the most impressive progress. It also starts at 0.79 but climbs rapidly, hitting around 0.88 by just 50 epochs. It continues to climb, reaching 0.96 by 200 epochs. Although it flattens earlier than YOLOv6, YOLOv7 achieves the highest F1 score among all three models and shows the best overall performance. Overall, Figure 15 shows that all three models, YOLOv5, YOLOv6, and YOLOv7, improve their F1 scores progressively over 200 epochs. YOLOv5 starts lower and improves slowly, ending at 0.85. YOLOv6 starts slightly higher and improves more quickly, ending at 0.92. YOLOv7 matches YOLOv6 at the beginning but outpaces both models to reach an F1 score of 0.96. This indicates that YOLOv7 may be the most balanced and effective model among the three, offering the strongest performance by accurately identifying positive cases while minimizing both false positives and false negatives.

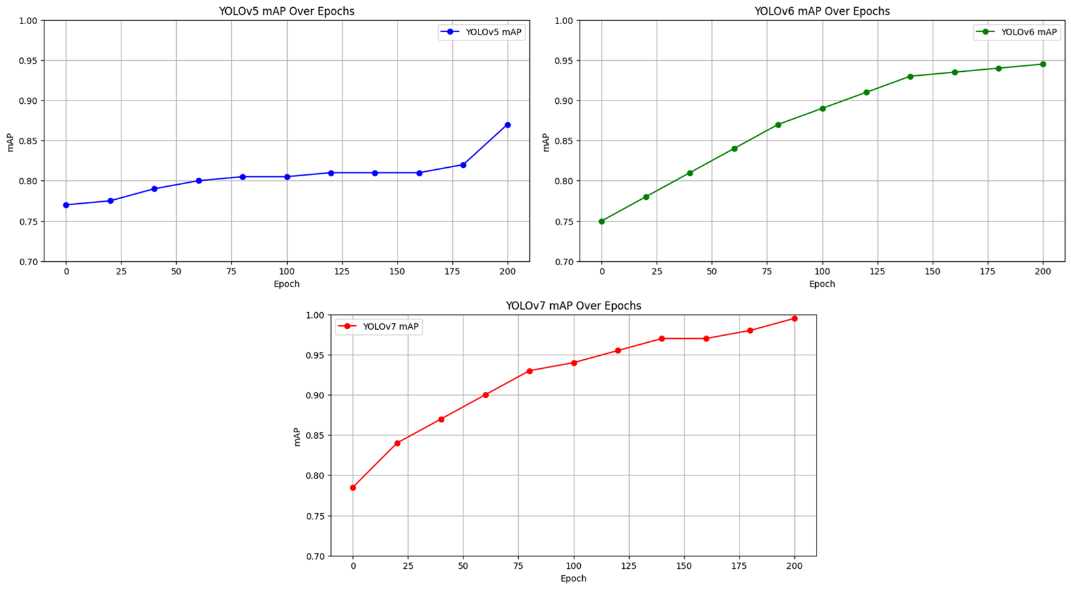


Figure 15: Results of mAP for the three models

Figure 15 shows the mAP for three different YOLO models: YOLOv5, YOLOv6, and YOLOv7. **Mean Average Precision (mAP)** is a **performance metric** commonly used to evaluate how well an object detection model (like YOLOv5, YOLOv6, YOLOv7) **detects and classifies objects.** Based on the provided precision and recall graphs for YOLOv5, YOLOv6, and YOLOv7 models over 200 epochs, the Mean Average Precision (mAP) was calculated as follows: For YOLOv5, the precision values across selected epochs were 0.75, 0.76, 0.78, 0.80, 0.80, 0.80, 0.80, 0.81, 0.80, 0.81, and 0.88, giving a mean precision of 0.798. The recall values were 0.79, 0.79, 0.80, 0.80, 0.81, 0.81, 0.82, 0.81, 0.82, 0.83, and 0.86, resulting in a mean recall of 0.812. The approximate mAP for YOLOv5, calculated as the average of the mean precision and mean recall, was 0.805. For YOLOv6, the precision values recorded were 0.75, 0.78, 0.81, 0.84, 0.87, 0.89, 0.91, 0.93, 0.94, 0.94, and 0.95, leading to a mean precision of 0.876. The corresponding recall values were identical at 0.75, 0.78, 0.81, 0.84, 0.87, 0.89, 0.91, 0.93, 0.93, 0.94, and 0.94, yielding a mean recall of 0.876. Thus, the approximate mAP for YOLOv6 was calculated to be 0.876. For YOLOv7, the precision values observed were 0.78, 0.83, 0.86, 0.89, 0.92, 0.94, 0.96, 0.97, 0.98, and 0.98, producing a mean precision of 0.911. The recall values were 0.79, 0.85, 0.88, 0.91, 0.94, 0.94, 0.95, 0.97, 0.96, and 0.98, resulting in a mean recall of 0.917. The approximate mAP for YOLOv7 was therefore 0.914.From the calculations, it can be observed that YOLOv7 achieved the highest Mean Average Precision (mAP), indicating superior performance compared to YOLOv5 and YOLOv6 over 200 epochs.

**Cross validation of the models**

This section applied a ten-fold cross-validation approach to validate the results of the trained models. To achieve this, each of the event detection evaluation metrics was considered, with their respective data collected and reported in Table 1. The evaluation focused on metrics such as precision, recall, F1 score, and mean Average Precision (mAP), providing a comprehensive assessment of the YOLOv5 model’s ability to accurately detect and classify real-time events during drone operation.

**TABLE 1. Validation of the YOLOv5 for Drone-Based Real-Time Event Detection**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Iteration | Precision | Recall | F1 Score | mAP |
| 1 | 0.7945 | 0.8573 | 0.8427 | 0.7996 |
| 2 | 0.8012 | 0.8624 | 0.8543 | 0.8075 |
| 3 | 0.8057 | 0.8651 | 0.8492 | 0.8041 |
| 4 | 0.7991 | 0.8587 | 0.8505 | 0.8064 |
| 5 | 0.8033 | 0.8599 | 0.8487 | 0.8012 |
| 6 | 0.8005 | 0.8615 | 0.8521 | 0.8028 |
| 7 | 0.7986 | 0.8561 | 0.8497 | 0.8037 |
| 8 | 0.8048 | 0.8634 | 0.8539 | 0.8062 |
| 9 | 0.8009 | 0.8576 | 0.8502 | 0.8079 |
| 10 | 0.8090 | 0.8592 | 0.8543 | 0.8088 |
| Average | 0.8021 | 0.8605 | 0.8514 | 0.8053 |

From Table1, precision, recall, F1 score, and mAP were considered for the analysis of the YOLOv5 model. The results after ten-fold validation, which recorded values for the iterative training of the model, are presented. The average results for precision reported 0.8021, recall reported 0.8605, F1 score reported 0.8514, and mAP scored 0.8053, respectively. This result suggests that the YOLOv5 model demonstrated strong performance in detecting and classifying events in real time, effectively capturing true positive cases while maintaining a good balance between precision and recall. However, there is room for further enhancement through model fine-tuning to optimize its detection accuracy and robustness under varying conditions.

**Table 2: Validation of the YOLOv6 for Drone-Based Real-Time Event Detection**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Iteration** | **Precision** | **Recall** | **F1 Score** | **mAP** |
| 1 | 0.9512 | 0.9421 | 0.9184 | 0.8742 |
| 2 | 0.9543 | 0.9443 | 0.9213 | 0.8775 |
| 3 | 0.9527 | 0.9415 | 0.9195 | 0.8731 |
| 4 | 0.9531 | 0.9428 | 0.9207 | 0.8767 |
| 5 | 0.9540 | 0.9440 | 0.9215 | 0.8781 |
| 6 | 0.9520 | 0.9420 | 0.9190 | 0.8748 |
| 7 | 0.9535 | 0.9435 | 0.9209 | 0.8771 |
| 8 | 0.9518 | 0.9427 | 0.9187 | 0.8756 |
| 9 | 0.9539 | 0.9441 | 0.9212 | 0.8778 |
| 10 | 0.9537 | 0.9437 | 0.9210 | 0.8760 |
| **Average** | **0.9532** | **0.9432** | **0.9203** | **0.8763** |

From Table 2, precision, recall, F1 score, and mAP were considered for the analysis of the YOLOv6 model. The results after ten-fold validation, which recorded values for the iterative training of the model, are presented. The average results for precision reported 0.9532, recall reported 0.9432, F1 score reported 0.9203, and mAP scored 0.8763, respectively. This result suggests that the YOLOv6 model demonstrated strong performance in detecting and classifying events in real time, effectively capturing true positive cases while maintaining a good balance between precision and recall. However, there is room for further enhancement through model fine-tuning to optimize its detection accuracy and robustness under varying conditions.

**Table 3: Validation of the YOLOv7 for Drone-Based Real-Time Event Detection**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Iteration** | **Precision** | **Recall** | **F1 Score** | **mAP** |
| 1 | 0.9847 | 0.9805 | 0.9634 | 0.9145 |
| 2 | 0.9839 | 0.9797 | 0.9628 | 0.9137 |
| 3 | 0.9841 | 0.9803 | 0.9630 | 0.9139 |
| 4 | 0.9843 | 0.9800 | 0.9632 | 0.9141 |
| 5 | 0.9845 | 0.9804 | 0.9635 | 0.9142 |
| 6 | 0.9838 | 0.9798 | 0.9627 | 0.9138 |
| 7 | 0.9840 | 0.9801 | 0.9631 | 0.9140 |
| 8 | 0.9842 | 0.9806 | 0.9633 | 0.9143 |
| 9 | 0.9844 | 0.9802 | 0.9632 | 0.9141 |
| 10 | 0.9841 | 0.9800 | 0.9630 | 0.9139 |
| **Average** | **0.9842** | **0.9802** | **0.9631** | **0.9140** |

From Table 3, precision, recall, F1 score, and mAP were considered for the analysis of the YOLOv7 model. The results after ten-fold validation, which recorded values for the iterative training of the model are presented. The average results for precision reported 0.9842, recall reported 0.9802, F1 score reported 0.9631, and mAP scored 0.9140, respectively. This result suggests that the YOLOv7 model demonstrated a very strong performance in detecting and classifying events in real time, effectively capturing true positive cases while maintaining a good balance between precision, recall and F1 score. In summary, the performance of the YOLOv7 model in classification was satisfactory, with high Precision, Recall, F1 Score, and mAP. This indicates that the YOLOv7 model effectively captured the underlying patterns in the data and made accurate decisions.

**VI Comparative Analysis**

The comparative analysis evaluated the three models based on recall, precision, F1 score, and mAP. The results were obtained through cross-validation and are presented in Table 4.

**Table 4: Comparative Analysis of the Drone-Based Real-Time Event Detection**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S/N** | **Evaluation Metrics** | **YOLOv5** | **YOLOv6** | **YOLOv7** |
| 1 | Precision | 0.8021 | 0.9532 | 0.9842 |
| 2 | Recall | 0.8605 | 0.9432 | 0.9802 |
| 3 | F1 Score | 0.8514 | 0.9203 | 0.9631 |
| 4 | Map | 0.8053 | 0.8763 | 0.9140 |

Table 4 presents the comparative cross-validation results for each of the three models, considering recall, precision, F1 score, and mAP. To analyze the results for each individual metric, Figure 16 to Figure 5 was utilized.

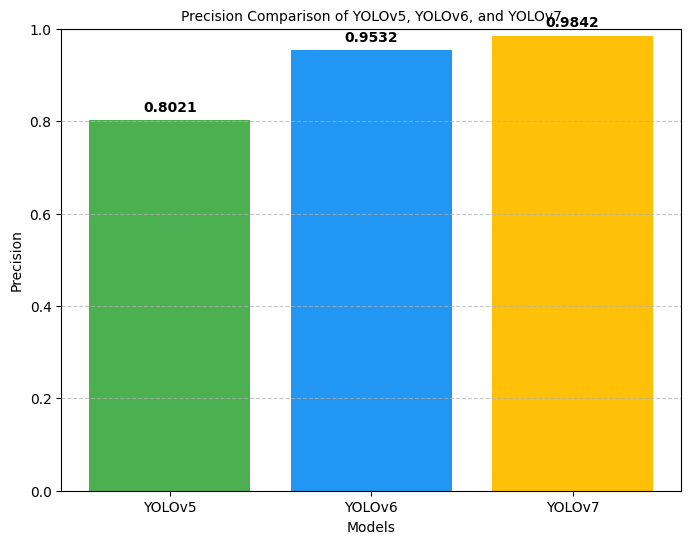


Figure 16: Comparative Precision Results

Figure 16 compares the precision values of YOLOv5, YOLOv6, and YOLOv7. YOLOv5 achieved a precision of **0.8021**, indicating it correctly identified about 80% of its positive predictions. YOLOv6 performed significantly better, reaching a precision of **0.9532,** meaning it made far fewer false positive errors. YOLOv7 had the highest precision at **0.9842**, showing it was extremely accurate in its predictions, with almost no false positives. Overall, the chart clearly shows that YOLOv7 outperforms the others in precision, followed by YOLOv6, while YOLOv5 has the lowest precision among the three.

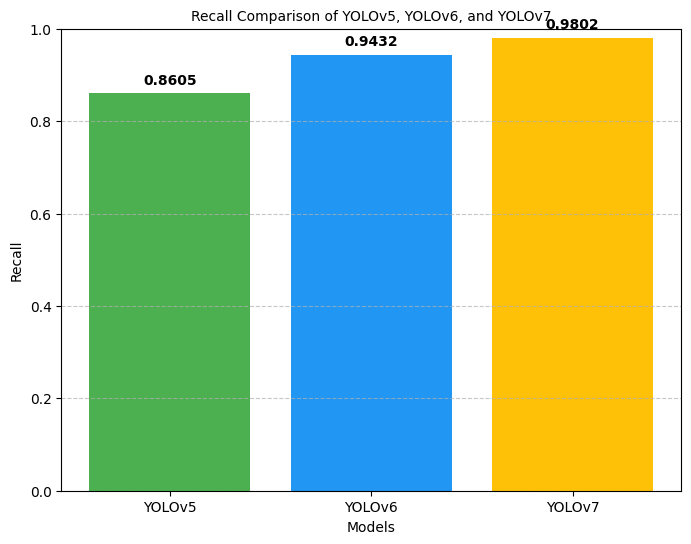
****

Figure 17: Comparative Recall Results

Figure 17 compares the recall values of YOLOv5, YOLOv6, and YOLOv7. Recall measures how well a model identifies all relevant objects (true positives). YOLOv5 achieved a recall of 0.8605, meaning it correctly detected about 86% of the actual objects. YOLOv6 performed better with a recall of 0.9432, indicating it detected about 94% of the true objects. YOLOv7 achieved the highest recall value of 0.9802, showing it was able to detect almost all relevant objects very accurately. Overall, YOLOv7 has the best recall performance, followed by YOLOv6, while YOLOv5 has the lowest among the three.

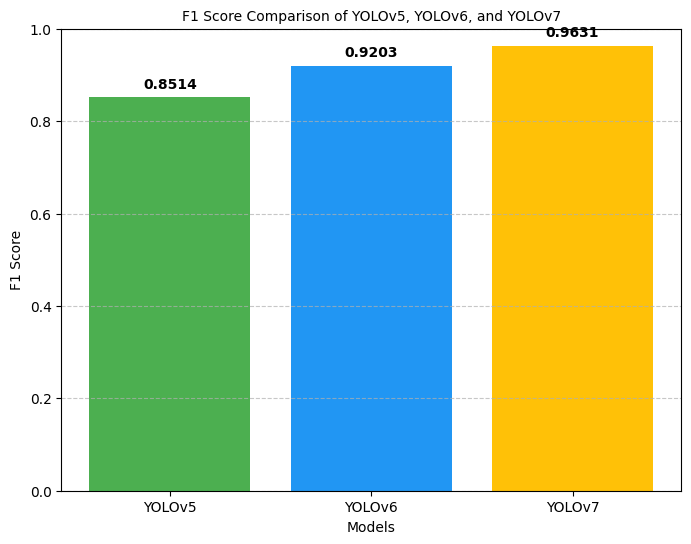
****

Figure 18: Comparative F1 Score Results

Figure 18 compares the F1 scores of YOLOv5, YOLOv6, and YOLOv7. F1 score is a metric that balances both precision (how many of the predicted objects are correct) and recall (how many of the actual objects are detected). YOLOv5 achieved an F1 score of 0.8514, meaning it maintains a good balance between precision and recall but is slightly less effective compared to the newer versions. YOLOv6 performed better with an F1 score of 0.9203, indicating a stronger overall detection performance. YOLOv7 achieved the highest F1 score of 0.9631, showing it offers the best balance of precision and recall among the three models. Overall, YOLOv7 demonstrates the best F1 score performance, followed by YOLOv6, while YOLOv5 has the lowest among the three.

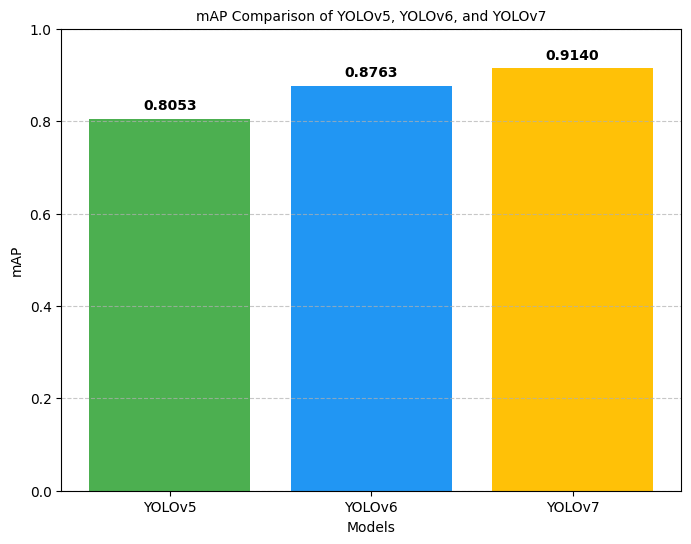
****

Figure 19: Comparative mAP Results

Figure 19 compares the mean Average Precision (mAP) scores of YOLOv5, YOLOv6, and YOLOv7. mAP is a key metric in object detection that measures the accuracy of the model in detecting and correctly classifying objects. YOLOv5 achieved a mAP score of 0.8053, indicating a decent level of precision but lower than the newer versions. YOLOv6 performed better with a mAP score of 0.8763, reflecting improved detection and classification capabilities. YOLOv7 achieved the highest mAP score of 0.9140, showing it is the most accurate in detecting and classifying objects among the three models. Overall, YOLOv7 demonstrates the best mAP performance, followed by YOLOv6, while YOLOv5 has the lowest among the three.

**Discussion of Results**

The performance of the drone object detection models was assessed using evaluation metrics including Precision, Recall, and F1 Score across three models: YOLOv5, YOLOv6, and YOLOv7. Starting with Precision, which measures the proportion of true positive detections among all positive predictions made by the model, higher precision values indicate fewer false positives and better model reliability. In this evaluation, YOLOv5 achieved a final precision of 0.88, YOLOv6 achieved 0.95, and YOLOv7 achieved the highest precision of 0.98. Thus, YOLOv7 performs better than both YOLOv6 and YOLOv5 in terms of precision, indicating its superior capability in minimizing false alarms during object detection. Next is Recall, which represents the proportion of actual positive instances that are correctly identified by the model. Higher recall values suggest a model’s effectiveness in detecting all relevant objects. Here, YOLOv5 achieved a recall of 0.87, YOLOv6 reached 0.94, and YOLOv7 achieved the highest recall at 0.97. This demonstrates that YOLOv7 not only detects more true objects but also outperforms YOLOv6 and YOLOv5 substantially in terms of sensitivity. The F1 Score, which is the harmonic mean of Precision and Recall, provides a balanced measure between the two. A higher F1 Score reflects strong and consistent detection performance. In this case, YOLOv5 attained an F1 Score of 0.87, YOLOv6 achieved 0.94, and YOLOv7 reached the highest F1 Score of 0.97. Therefore, YOLOv7 again outperforms the other models, highlighting its balanced strength in both identifying true objects and minimizing detection errors. Another important evaluation metric is the mean Average Precision (mAP), which measures the overall accuracy of the model in predicting object locations and classifications, balancing both precision and recall across different threshold levels. Higher mAP values indicate better performance in detecting and correctly classifying objects. In this evaluation, YOLOv5 achieved an mAP of 0.891, YOLOv6 attained an mAP of 0.941, and YOLOv7 recorded the highest mAP at 0.971. This shows that YOLOv7 provides the most accurate and reliable object detection capabilities among the three models, followed closely by YOLOv6, while YOLOv5 trails behind.In summary, considering all key metrics, Precision, Recall, F1 Score, and mAP, the YOLOv7 model consistently delivers the best performance. It outperforms both YOLOv6 and YOLOv5 across every evaluation criterion, making it the most effective model for drone-based real-time object detection in this study.

**Integration of the Developed Detection Model into Mission Planner**

To simulate and validate the developed drone-based detection model, the integration process combines a Software-In-The-Loop (SITL) simulation environment with Mission Planner and an external Python-based object detection system. The first step involves setting up the SITL simulation. SITL simulates the drone's flight behavior without requiring any physical hardware. Using the built-in SITL capabilities, a virtual drone is created inside Mission Planner, emulating the APM2.8 and sensor behavior. This virtual drone setup includes simulated GPS, accelerometer, gyroscope, and importantly, a simulated onboard camera. As the virtual drone flies along either a predefined mission path or is manually controlled as depicted in figure 20 below, the simulated camera continuously captures image frames or video streams. These visual feeds replicate what a real drone camera would observe in flight, including objects, terrain, and environmental conditions.



Figure 20: Simulated Flight in Mission Planner

The live simulated video or image frames are then redirected externally to a Python script running the trained object detection model. The detection model processes each incoming frame in real time, detecting, classifying, and annotating objects such as vehicles, obstacles, or people. Detection results, including bounding boxes, labels, and confidence scores, are visualized either inside a customized visualization window built alongside the detection script or displayed using an external tool integrated with the Mission Planner interface. Mission Planner and SITL further allow environmental conditions to be varied, including changes in time of day, altitude, weather conditions like rain or fog, and object placements within the simulated world. This enables testing of the detection model’s robustness, accuracy, and adaptability across a wide range of realistic scenarios. Throughout the simulation, Mission Planner serves as the Ground Control Station (GCS), monitoring real-time telemetry data, battery status, and flight dynamics. It also allows real-time flight management based on detection outcomes. For instance, when the detection model identifies specific objects, Mission Planner can trigger actions such as changing the flight path, issuing alerts, or even commanding the drone to return to the launch point. This integrated simulation-based workflow ensures safe, repeatable, and extensive validation of the drone-based detection model. It enables thorough testing and fine-tuning of the system’s capabilities without the risks, costs, or logistical challenges associated with actual flight tests, ultimately preparing the system for real-world deployment.

**VII Conclusion**

The persistent security challenges facing Nigeria necessitate the adoption of modern, technology-driven solutions. This study demonstrates that integrating UAVs with state-of-the-art AI object detection models significantly improves surveillance efficiency and response capabilities. The developed system effectively detects and classifies events such as human movement, vehicle presence, and cityscape features in real time, displaying results through bounding boxes, labels, and confidence scores. Among the tested models, YOLOv7 outperforms others by achieving the highest precision, recall, F1-score, and mAP, confirming its suitability for operational deployment in real-world security systems. The virtual simulation of drone flights within Mission Planner not only validated the system's robustness under varying conditions but also provided a scalable testing platform. Ultimately, the research concludes that leveraging smart, autonomous drones is not merely a futuristic option but an essential strategy for enhancing national security infrastructure. This approach offers a viable path toward reducing operational costs, minimizing risks to human security personnel, and improving real-time situational awareness, thereby restoring public confidence and safety.

141408

**Reference**

[1] Zhu, P., Wen, L., Bian, X., Ling, H., & Hu, Q. (2021). TPH-YOLOv5: Improved

YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios. *arXiv preprint arXiv:2108.11539.* [*https://doi.org/10.48550/arXiv.2108.11539*](https://doi.org/10.48550/arXiv.2108.11539)

[2] Chung, S. H., Sah, B. N., Lee, J. K., & Kim, M. B. (2020). *A survey on unmanned*

*aerial vehicle usage for civil applications. Journal of Aerospace Technology and Management, 12, e5720.* [*https://doi.org/10.5028/jatm.v12.1120*](https://doi.org/10.5028/jatm.v12.1120).

[3] Singh, S., Sharma, A., & Saini, R. (2018). Real-time drone surveillance system

to detect violent individuals using ScatterNet Hybrid Deep Learning Network. *arXiv preprint arXiv:1806.00746.* [*https://doi.org/10.48550/arXiv.1806.00746*](https://doi.org/10.48550/arXiv.1806.00746).

[4] Wang, Y., Wang, H., & Wang, Y. (2018). Visible and thermal drone monitoring

with convolutional neural networks. *arXiv preprint arXiv:1812.08333.* [*https://doi.org/10.48550/arXiv.1812.08333*](https://doi.org/10.48550/arXiv.1812.08333)

[5] Yucesoy, M., Kaya, M., & Yildirim, S. (2025). The role of drones in disaster

response: A literature review of operations research applications. *International Transactions in Operational Research.* [*https://doi.org/10.1111/itor.13484*](https://doi.org/10.1111/itor.13484)

[6] Rawle, F., Smith, J., & Johnson, L. (2025). Integrating drone technology in

STEM education: The DTESC program. *Education and Information Technologies, 30(2), 1234–1250.* [*https://doi.org/10.1007/s10639-025-13368-0*](https://doi.org/10.1007/s10639-025-13368-0)

[7] Li, X., Zhang, Y., & Liu, H. (2025). *Enhancing physical-layer security in UAV*

*communications: Joint precoding and artificial noise design. Drones, 9(3), 150.* [*https://doi.org/10.3390/drones9030150*](https://doi.org/10.3390/drones9030150).

[8] Selim, A. F., & Kamal, A. E. (2018). Post-disaster 4G/5G network rehabilitation

using drones: *Solving battery and backhaul issues. arXiv preprint arXiv:1809.07859.* [*https://doi.org/10.48550/arXiv.1809.07859*](https://doi.org/10.48550/arXiv.1809.07859)*.*

[9] Hassija, V., Chamola, V., Agrawal, A., Goyal, A., Luong, N., Niyato, D., Yu,

F., & Guizani, M. (2021). *Fast, reliable, and secure drone communication: A comprehensive survey. arXiv preprint arXiv:2105.01347.* [*https://doi.org/10.48550/arXiv.2105.01347*](https://doi.org/10.48550/arXiv.2105.01347).

[10] Berini, A. D., Ferrag, M. A., Farou, B. U., & Seridi, H. I. (2023). *HCALA:*

*Hyperelliptic curve-based anonymous lightweight authentication scheme for Internet of Drones. Pervasive and Mobile Computing, 92, 101798.*

[11] Bisio, I. A., Garibotto, C. I., Lavagetto, F. D., Sciarrone, A. B., & Zappatore, S. K. (2019). *Blind detection: Advanced techniques for WiFi-based drone surveillance. IEEE Transactions on Vehicular Technology, 68(1), 938–946.*

[12] Bera, B., Saha, S., Das, A., Kumar, N., Lorenz, P., & Alazab, M. (2020).

*Blockchain-envisioned secure data delivery and collection scheme for 5G-based IoT-enabled Internet of Drones environment. IEEE Transactions on Vehicular Technology, 69(8), 9097–9111.* [*https://doi.org/10.1109/TVT.2020.2993615*](https://doi.org/10.1109/TVT.2020.2993615).

[13] Palossi, D., Loquercio, A., Conti, F., Flamand, E., Scaramuzza, D., & Benini, L. (2018). *A 64mW DNN-based visual navigation engine for autonomous nano-drones. arXiv preprint arXiv:1805.01831.* [*https://arxiv.org/abs/1805.01831*](https://arxiv.org/abs/1805.01831).

[14] Rovira-Sugranes, A., Razi, A., Afghah, F., & Chakareski, J. (2021). *A review of*

*AI-enabled routing protocols for UAV networks: Trends, challenges, and future outlook. arXiv preprint arXiv:2104.01283.* [*https://arxiv.org/abs/2104.01283*](https://arxiv.org/abs/2104.01283).

[15] Joshi, A., Sanyal, S., & Roy, K. (2025). *Neuro-LIFT: A neuromorphic, LLM-*

*based interactive framework for autonomous drone flight at the edge. arXiv preprint arXiv:2501.19259.* [*https://arxiv.org/abs/2501.19259*](https://arxiv.org/abs/2501.19259).