Democratic and Popular Algerian Republic

Ministry of Higher Education and Scientific Research

Yahia fares university of Médéa



Faculty of Sciences

Department of Mathematics and Computer Science

Final Master Project

Branch: Computer Science

Specialty: Systems Engineering and web Technologies

Theme

Drone detection and classification using deep leaning

Prepared and presented by

BENHAFRI MERIEM

CHARRIF MERIEM

Proposed and directed by

Dr. KHELDOUN AHMED

Defended on June , 2024, In front of the jury composed of:

Mr. Tchoketch Kebir Sofiane : Yahia Fares University - President

Mr. Ismail Hadjir : Yahia Fares University - Examiner

Year: 2023-2024

Acknowledgement

We are deeply grateful for the help and assistance of our supervisor, Dr. Kheldoun Ahmad. His strong management throughout our research journey has been precious. Dr. kheldoun was always there to help us with everything, from providing crucial information and resources to attending to every small detail. His loyalty, patience, and support have been the backbone of our work helping us to overcome each obstacle and reach each goal. We sincerely appreciate all of his hard work.

We are also grateful to the members of the jury for accepting to be with us and sharing their relevant experience to evaluate our work. Their insights and feedback will greatly enrich our research. We are thankful for their time, effort, and constructive criticism, which significantly improve the quality of our work.

We would also like to extend our heartfelt thanks to our co-supervisor, Mr. Mekid Hamza from Icosnet SPA, where we completed our internship. His valuable guidance, indispensable information, and Continuous support have been instrumental in our success. His encouragement and belief in our abilities have been a great source of motivation for us.

We also want to extend our heartfelt thanks to Othmane Bakir Torki, who has been the most encouraging person in the entire university. His constant encouragement and positivity have been a source of inspiration. May Allah bless him with all the best and grant him all that he desires.

We would like to express our sincere gratitude to Miss louz khadidja for her invaluable support and encouragement.

Special thanks go to Mr. Tchoketch Kebir Sofiane for inspiring us with his ideas. Without his presence and direction, we would not be where we are today.

Furthermore, we would like to thank all the researchers worldwide for giving vital information that provided us with inspiration. We are appreciative of everyone who has supported us thus far.

Dedications

To my beloved family, for supporting me during the hard times, especially my grandma for her prayers, without which I would not have been able to overcome many challenges.
To every teacher who has given their best to inspire us with their knowledge since my early school days and over the past five years. Your commitment and passion for teaching have profoundly shaped my learning experience.
To my dear friends, for their unwavering support, understanding, and faith in me throughout this journey, especially ***Fatma Zohra***, ***Aicha***, ***Asma***, ***Hadil***, ***Naziha***, ***Asala*** , ***Innes*** ,***Soulaf*** *and* ***Meriem***. *Your encouragement and belief in my abilities have been a source of strength and motivation. I am deeply grateful for your patience and for standing by me during both the highs and lows of this period. Your friendship has been invaluable, and I could not have accomplished this without you.*
Special thanks to my project partner, ***Benhafri Meriem***, *for her dedication, creativity, and collaborative spirit throughout our work together. Your contributions have been instrumental in achieving our milestones.*
And to everyone who has played a role in shaping my character and helping me reach this significant achievement, I extend my heartfelt thanks.

Charrif Meriem

Dedications

I would start by thanking God for answering my prayers and for providing me with guidance at every turn, down to the slightest detail, throughout my trip ,alhamdoulilah.
I dedicate the work to my parents who have always been there for me and have given me encouragement. Their perseverance, support, and encouragement have been a source of gratitude for me all my life.
and to my dear sisters and brothers, who serve as my inspiration and light in this world, I shall always be grateful to Youcef for his unwavering support and assistance.to Soufiane, for his ability to make sense of and streamline my existence. souad for her wisdom and love, and for the way she has helped me to understand and simplify my things which has allowed me to gain control of numerous situations. Nabila for being a never-ending source of motivation and solid encouragement during my road. I also want to thank all of my friends and supporters for helping me get to where I am now. You have all been instrumental in my success.

Benhafri Meriem

Abstract

This document explores the application of deep learning models, specifically YOLOv5 and YOLOv8, for drone detection, with a focus on addressing the challenge of detecting small drones. Through rigorous experimentation and optimization of training parameters, our study aims to enhance the accuracy and efficiency of drone detection systems. The results demonstrate significant benefits, particularly in real-time performance and precision, crucial for enhancing security and surveillance capabilities in various environments.

Keywords: drone detection, deep learning, YOLOv5, YOLOv8, small drones

Résumé

Ce document explore l'application des modèles d'apprentissage profond, notamment YOLOv5 et YOLOv8, pour la détection des drones, en mettant l'accent sur le défi de détecter les petits drones. À travers des expérimentations rigoureuses et l'optimisation des paramètres d'entraînement, notre étude vise à améliorer la précision et l'efficacité des systèmes de détection de drones. Les résultats montrent des avantages significatifs, notamment en termes de performance en temps réel et de précision, essentiels pour renforcer les capacités de sécurité et de surveillance dans divers environnements.

Mots-clés : détection de drones, apprentissage profond, YOLOv5, YOLOv8, petits drones

ملخص

يستكشف هذا البحث تطبيق نماذج التعلم العميق، وبالأخص YOLOv5 و YOLOv8 ، للكشف عن الطائرات بدون طيار، مع التركيز على معالجة تحدي الكشف عن الطائرات الصغيرة. من خلال التجارب الدقيقة وتحسين معايير التدريب، تهدف دراستنا إلى تحسين دقة وكفاءة أنظمة الكشف عن الطائرات بدون طيار. تظهر النتائج فوائد كبيرة، خاصة في الأداء في الوقت الحقيقي والدقة، وهي أمور حيوية لتعزيز قدرات الأمن والمراقبة في البيئات المختلفة.

الكلمات المفتاحية: الكشف عن الطائرات بدون طيار، التعلم العميق، YOLOv5 ، YOLOv8 ، الطائرات الصغيرة

Contents

I General Introduction **1**
- I.1 Problematic . . . 2
- I.2 Motivation and Objectives . . . 2
- I.3 Thesis Structure . . . 2

II Model Landscape and Deep Learning Foundations **4**
- II.1 Introduction . . . 5
- II.2 Artificial Intelligence . . . 5
  - II.2.1 Machine Learning . . . 5
  - II.2.2 Deep Learning . . . 7
  - II.2.3 Artificial Neural Network . . . 7
  - II.2.4 Convolutional Neural Network . . . 10
  - II.2.5 YOLO . . . 13
- II.3 Evaluation Metrics of Models: . . . 16
  - II.3.1 Conclusion . . . 17

III Drone Detection **18**
- III.1 Introduction . . . 19
- III.2 Introduction to Drones: . . . 19
- III.3 Object Recognition . . . 21
- III.4 Small Object Detection: . . . 23
- III.5 Drone's apllication . . . 27
- III.6 Related Works . . . 27
- III.7 Conclusion . . . 28

IV Conception **29**
- IV.1 Introduction . . . 30
- IV.2 Proposed Methodology . . . 30
- IV.3 Phases . . . 31
  - IV.3.1 Datasets . . . 31
  - IV.3.2 Training Data Selection . . . 32
  - IV.3.3 Data Preprocessing . . . 34
  - IV.3.4 Annotation Filtering . . . 35
  - IV.3.5 Training and Validation Split . . . 35
  - IV.3.6 Training and Detection . . . 36
- IV.4 Chosen Drone Detection Model: YOLOv5 . . . 36
- IV.5 SAHI-powered YOLOV5 . . . 36
  - IV.5.1 Integration and Workflow: . . . 37
  - IV.5.2 Benefits of this Approach: . . . 38

IV.6 Ensemble learning 39
IV.7 Conclusion 39

V Test And Results **40**
V.1 Introduction 41
V.2 Development Environment 41
V.2.1 Hardware specifications 41
V.2.2 Software specifications and libraries 41
V.3 Implementation and Results 42
V.3.1 Model Selection 42
V.3.2 Training parameters 43
V.3.3 Results: 43
V.3.4 Model Training and Performance on Global Dataset 46
V.3.5 Test models on Safeshore dataset 48
V.3.6 Comparison Results with Models Proposed in Literature 49
V.4 The graphical interface 49
V.5 Conclusion 50

General Conclusion **51**

Bibliography **53**

List of Tables

IV.1 Information about the datasets 32

V.1 Training parameters for the best performing model (YOLOv5). 43
V.2 Training and validation metrics of different models on the global dataset. . . . 47
V.3 Faster R CNN model training metrics 47
V.4 Detailed training and validation metrics for VGG 16 and MobileNet_v2 models 47
V.5 Performance metrics of different models on the global dataset. 47
V.6 Performance metrics of different models on the SafeShore dataset. 48
V.7 Performance metrics of different models on the SafeShore dataset. 48
V.8 Comparison results between this work and prior research 49

List of Figures

II.1 Key Approaches in Machine Learning [1] . . . 5
II.2 Supervised Learning . . . 6
II.3 Unsupervised Learning . . . 6
II.4 Backpropagation learning rule-based supervised network [2]. . . . 8
II.5 Artificial Neuron Design . . . 8
II.6 CNN architecture. . . . 10
II.7 Convolution Layer Without Padding[3]. . . . 11
II.8 convolution operation [3]. . . . 11
II.9 Illustration of the pooling techniques[4]. . . . 12
II.10 S × S grid on input[5]. . . . 13
II.11 Area of Intersection . . . 15
II.12 Area of Union . . . 15

III.1 Multicopter models [6] . . . 20
III.2 Four Major Types of UAV (drone)[7]. . . . 20
III.3 Overview of Object Recognition Computer Vision Tasks [8]. . . . 21
III.4 classification example [9]. . . . 21
III.5 Steps needed for object classification. (a) Features-Based Model. (b) Deep Learning Strategy.[10] . . . 22
III.6 Samples of drone localization[11]. . . . 23
III.7 object-detection[11]. . . . 23
III.8 Feature Fusion Module[12]. . . . 24
III.9 Fusion Method Overview[13]. . . . 25
III.10uav-yolo(based on YOLOv5s Model)[14]. . . . 26

IV.1 Schematic Representation of the Drone Detection Model Development Framework. . . . 30
IV.2 Data Histogram [15] . . . 32
IV.3 Frame Distribution for Training and Validation Sets. . . . 33
IV.4 Frame Distribution for Training and Validation Sets. . . . 34
IV.5 Comparison of Frames 536 Frame 537 . . . 34
IV.6 Comparison of Frames 536 and 552 . . . 35
IV.7 Comparison of Frames 40 and 56 . . . 35
IV.8 Overall Workflow for Object Detection with SAHI and YOLOv5 . . . 38

V.1 Precision, recall and map_0.5 over epochs . . . 44
V.2 Train vs validation box loss . . . 45
V.3 Train vs validation object loss . . . 45
V.4 Confusion matrics . . . 46
V.5 Developed system displaying real-time drone detection results. . . . 50

List of Abbreviations

AI Artificial Intelligence. 5

AIS Automatic Identification System. 27

CNN Convolutional Neural Network. 7

DBN Deep Belief Network. 7

FPN Feature Pyramid Network. 25

FPS Frames Per Second. 47

HOG Histogram of Oriented Gradients. 22

IOU Intersection Over Union. 14

LBP Local Binary Patterns. 22

LSTM Long Short-Term Memory. 22

mAP mean Average Precision. 43

ML Machine Learning. 5

MM Magnifier Method. 25

PAN Position-Aware Network. 25

R-CNN Region-based Convolutional Neural Network. 22

RDB Reconstruction Detection Branch. 25

Relu Rectified Linear Unit. 9

ResNet Residual Network. 27

RNN Recurrent Neural Network. 7

SAHI Slicing Aided Hyper Inference. 36

SSD Single Shot Multibox Detector. 22

SURF Speeded Up Robust Features. 22

SVM Support Vector Machine. 27

Tahn Hyperbolic Tangent. 9

UAS Unmanned Aerial Systems. 19

YOLO You Only Look Once. 5

Chapter I

General Introduction

Unmanned aerial vehicles (UAVs), or drones, have gained significant popularity in recent times. Their origins trace back to the military field, specifically designed for operating in hazardous or human-denied areas. Over the past decade, UAVs have become increasingly attractive for a multitude of applications beyond the military domain. These applications span civilian and commercial sectors, including surveillance, search and rescue, traffic and weather monitoring, precision agriculture, aerial imaging, and many more[16].

Their quick ascent has significantly impacted various sectors. As technology continues to advance, drones are expected to become even more prevalent in our society. However, this rapid adoption also presents challenges, particularly regarding privacy, security, and managing airspace for both drones and traditional aircraft. Ensuring the confidentiality of sensitive data, such as personal and military information is vital. Additionally, the rise in privacy concerns caused by unauthorized UAV deployment necessitates regulations to restrict drone usage.

Small drones add another layer of complexity to these challenges. Detecting and classifying these long-range drones is crucial for maintaining security and privacy.

To address these challenges and guarantee safe and secure integration of drones into our airspace, the ability to identify and classify them, including small drones, becomes imperative.

I.1 Problematic

The drone regulatory landscape in our nation is unduly restrictive.It is quite difficult to purchase, import, and use drones for commercial and other useful purposes because of these strict laws. The legal obstacles and intricate bureaucratic procedures provide a significant barrier for people and companies looking to take advantage of the many benefits drone technology has to offer.Furthermore, there are concerns about the security risk because drones could endanger privacy and public safety. Unauthorized drone flights have the potential to cause privacy violations by collecting private data without permission. Drones can also be used maliciously, such carrying illegal goods or spying on people without permission. Although the stringent laws may appear onerous, they serve a dual purpose of reducing security threats and guaranteeing the appropriate application of drone technology. It is still a difficult issue for regulatory organizations to strike a balance between the advantages of drone innovation and the requirements for security and privacy.

I.2 Motivation and Objectives

The objective of this project is to create real-time drone classification and detection technologies in order to reduce the risks posed by unlicensed UAVs, including small drones. Authorities will be stronger able to control airspace security and react quickly when facing threats partly due to these advanced technologies.

I.3 Thesis Structure

This work will contain four chapters

- Chapter 1 :

  The present state of artificial intelligence (AI) techniques used for drone detection and classification is covered in detail in this chapter.

- Chapter 2 : This chapter covered Drone detection techniques to maintain safe airspace and explore obstacles in the form of small item detection, such as drones, and strategies to overcome them.
- Chapter 3 :

  We delve into the inner workings of our drone detection system in this chapter. It illustrates a methodical approach to system development, covering data organisation, preparation, and detection model training.
- Chapter 4 : This section we talk about the technical features of our project, such as the development environment, datasets utilized, and implementation specifics.

Chapter II

Model Landscape and Deep Learning Foundations

II.1 Introduction

In this chapter we will explore an overview about foundational concepts of artificial intelligence (Artificial Intelligence (AI)) and its subset .Machine Learning, and its ability to learn from data for predictions and decisions. It covers various Machine Learning (ML) approaches and their applications. Further topics include Deep Learning, Neural Networks, and the efficient You Only Look Once (YOLO) algorithm. Evaluation metrics for performance analysis are also discussed. An engaging overview of these technologies is provided, offering a robust foundation for further exploration of AI's cutting-edge innovations.

II.2 Artificial Intelligence

Artificial intelligence is a vast area of study. This field of study helps in finding solutions to real world problems. Artificial Intelligence is a way of making a computer, a computer-controlled robot, or a software think intelligently, in the similar manner the intelligent humans think. AI is accomplished by studying how human brain thinks and how humans learn, decide, and work while trying to solve a problem, and then using the outcomes of this study as a basis of developing intelligent software and systems. This is accomplished by applying a variety of deep learning and machine learning techniques, each suited to a particular set of goals and activities.

II.2.1 Machine Learning

Machine learning, a subfield of artificial intelligence, allows computers to learn from data patterns without explicit instructions. It focuses on the development of statistical models and algorithms to make predictions or judgments based on previously learned patterns. The broad terrain of machine learning includes numerous forms, each adapted to certain tasks and aims.

In the following discussion, we will quickly investigate key types of machine learning shown in Figure II.1:

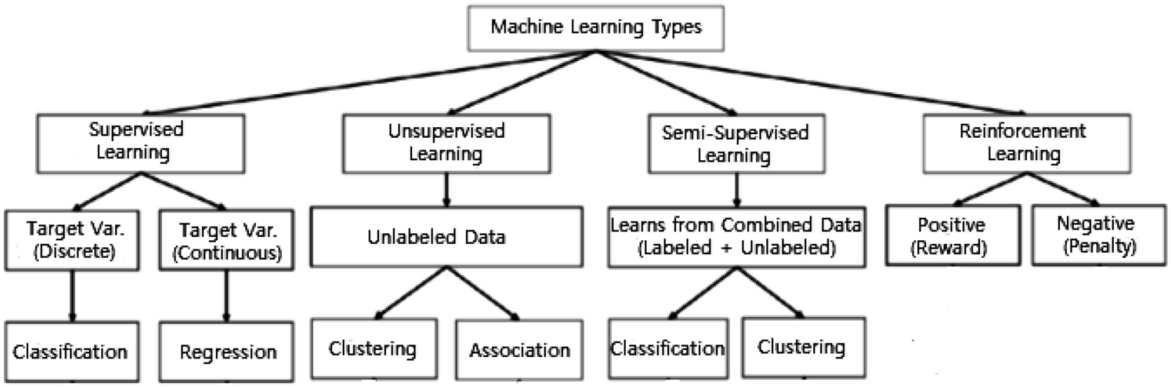


Figure II.1: Key Approaches in Machine Learning [1]

- **Supervised:** Supervised learning is typically the task of machine learning to learn a function that maps an input to an output based on sample input-output pairs. It uses labeled training data and a collection of training examples to infer a function. Supervised learning is carried out when certain goals are identified to be accomplished from a certain set of inputs, i.e., a task-driven approach. The most common supervised

tasks are “classification” that separates the data, and “regression” that fits the data[1], as shown in Figure II.2, where two classes of data points are separated by a decision boundary.

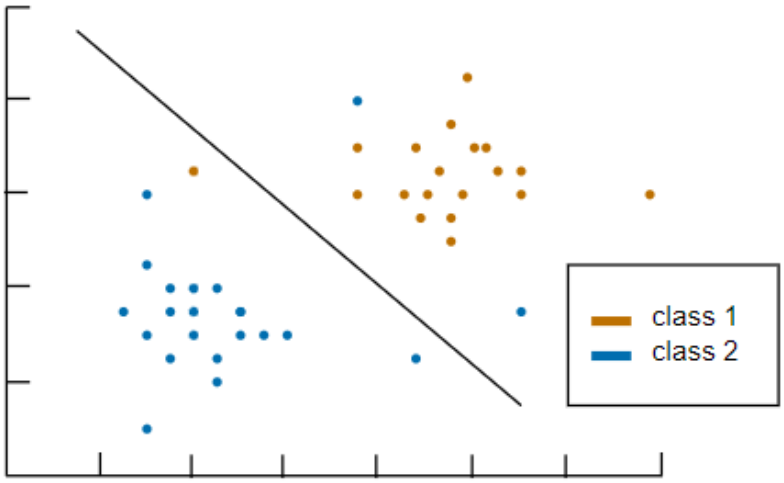


Figure II.2: Supervised Learning

- **Unsupervised:** Unsupervised learning analyzes unlabeled datasets without the need for human interference, i.e., a data-driven process . This is widely used for extracting generative features, identifying meaningful trends and structures, groupings in results, and exploratory purposes. The most common unsupervised learning tasks are clustering, density estimation, feature learning, dimensionality reduction, finding association rules, anomaly detection, etc[1], as illustrated in Figure II.3, where data points of a single color are grouped into clusters.

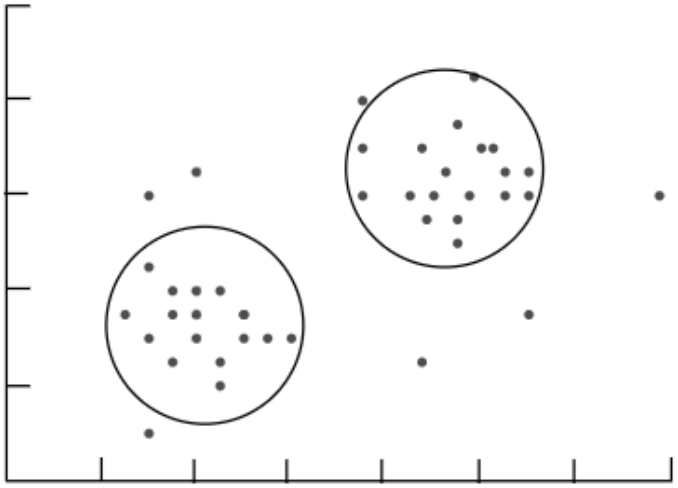


Figure II.3: Unsupervised Learning

- **Semi-supervised:** Semi-supervised learning can be defined as a hybridization of the above-mentioned supervised and unsupervised methods, as it operates on both labeled

and unlabeled data . The ultimate goal of a semi-supervised learning model is to provide a better outcome for prediction than that produced using the labeled data alone from the model. Some application areas where semi-supervised learning is used include machine translation, fraud detection, labeling data and text classification[1].

- **Reinforcement:** Reinforcement learning is a type of machine learning algorithm that enables software agents and machines to automatically evaluate the optimal behavior in a particular context or environment to improve its efficiency , i.e., an environment-driven approach. This type of learning is based on reward or penalty, and its ultimate goal is to use insights obtained from environmental activists to take action to increase the reward or minimize the risk[1].

II.2.2 Deep Learning

Deep learning is one of the fastest growing areas of data science.it refers to a class of algorithms which are based on artificial neural networks optimized to work with unstructured data such as images, voice, videos and text. has become broadly utilized in the fields of classification and pattern recognition. In general, deep learning applies multiple information-processing layers within a hierarchical design. Furthermore, deep learning is a hot research topic, meaning that several deep neural network (DNN) architectures, such as convolutional neural networks (Convolutional Neural Network (CNN)s) [8], recurrent neural networks (Recurrent Neural Network (RNN)s) [9], deep belief nets (Deep Belief Network (DBN)s) [17].

Unlike traditional machine learning, deep learning works by helping computers learn from experience. It does this by organizing knowledge into a hierarchy of concepts.it is like building complicated ideas from simpler ones. This hierarchy, shown as layers in a graph, can go pretty deep, letting the computer understand and learn complex patterns all by itself.

II.2.3 Artificial Neural Network

An Artificial Neural Network (ANN) is a mathematical model that tries to simulate the structure and functionalities of biological neural networks[18]. A neural network, in essence, is a data processing system comprising numerous interconnected processing elements. This architecture draws inspiration from the intricate connections found in the cerebral cortex of the brain. ANNs excel at handling unstructured data, including images, voice, videos, and text, making them a crucial component of deep learning.

When discussing ANNs, it's important to distinguish between ANN models (the network's arrangement) and ANN algorithms (the computations that produce the network outputs). Once structured for a particular application, an ANN can vary in the number of layers it comprises[2]. Figure II.4 illustrates the backpropagation learning rule-based supervised network, showcasing the computational processes involved in training ANN models.

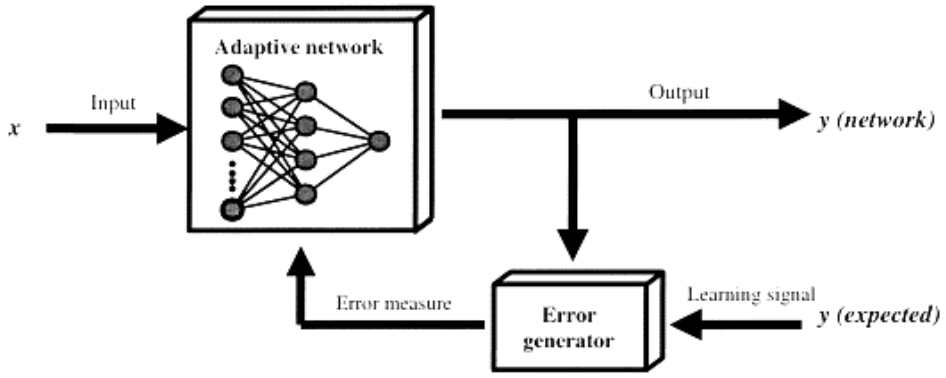


Figure II.4: Backpropagation learning rule-based supervised network [2].

The key components of an ANN

The architecture of artificial neural networks (ANNs) serves as the foundational framework for their ability to learn and extract complex patterns from data. Comprising interconnected layers, neurons, and a variety of components, the design of ANNs plays a pivotal role in their capacity to model intricate relationships and make predictions across various domains.

- **Neurons (Nodes or Units):** Artificial neuron is a basic building block of every artificial neural network. Its design and functionalities are derived from observation of a biological neuron that is basic building block of biological neural networks (systems) which includes the brain, spinal cord and peripheral ganglia[19]. Figure II.5 displays the architecture of an artificial neuron, emphasizing its structural components and operational processes.

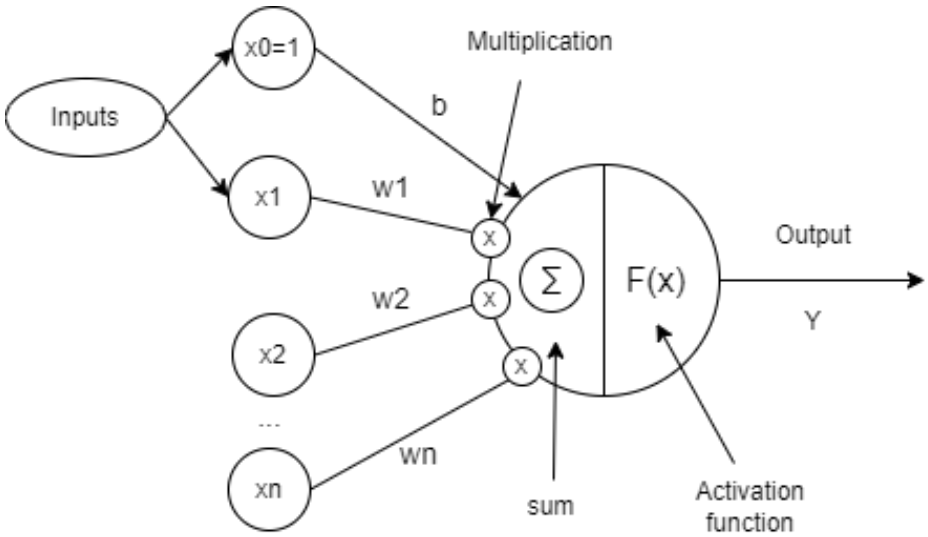


Figure II.5: Artificial Neuron Design

In case of artificial neuron the information comes into the body of an artificial neuron via inputs that are weighted (each input can be individually multiplied with a weight). The body of an artificial neuron then sums the weighted inputs, bias and "processes" the sum with a transfer function. At the end an artificial neuron passes the processed information via output(s)[19]. Benefit of artificial neuron model simplicity can be seen

in its mathematical description below:

$$y(k) = F\left(\sum_{i=0}^{m} w_i(k) \cdot x_i(k) + b\right) \tag{II.1}$$

- **Layers:** Artificial Neural Networks (ANNs) are structured into layers, and the essential components include three types of layers.
  - **Input layer:** The input layer receives the data that is to be processed by the network. It serves as the initial point where external information is fed into the network for further analysis, where each node represents a feature of the data.
  - **Hidden layer:** Intermediate layers between the input and output layers .These layers are where the actual learning takes place .Neurons in the hidden layers evaluate the input and pass on the results through to the next layer .
  - **Output layer:** It takes the combined results from the hidden layers and produces the fimal output or prediction.

  Each of these layers is connected to one another and work together to process the data and produce an output.

- **Activation functions:**
  - **Sigmoid:** A function that transforms input values into probabilities, often used in binary classification for its ability to model confidence levels. Mathematically it can be represented as:
$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{II.2}$$
  - **Hyperbolic Tangent (Tahn) Function:** This function is similar to the sigmoid function but centered around zero ,the output values vary between -1 and 1. It can be used in the hidden layers to introduce non-linearity . Mathematically it can be represented as:
$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{II.3}$$
  - **Rectified Linear Unit (Relu):** A function that outputs the input directly if it is positive; otherwise, it outputs zero.It is widely used in in hidden layers due to simplicity and effectiveness. In mathematical terms, it is defined as:
$$\text{Leaky ReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{otherwise} \end{cases} \tag{II.4}$$
  - **Softmax:** A mathematical function transforming a vector of raw scores into a probability distribution,ensuring that the output values sum to 1. It takes the form:
$$\text{Softmax}(\mathbf{x})_i = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{II.5}$$

II.2.4 Convolutional Neural Network

CNN is a one of deeplearning's algorithm that aims to process image data and any task related to object recognition such as image classification ,detection and segmentation by using Convolutional layers. Each layer acts like a magnifying glass . In the context of image processing, convolution involves a mathematical operation that combines input data with a filter/kernel to produce a feature map.

Convolutional layers are crucial for capturing hierarchical patterns in data. As the network goes deeper, higher-level features are learned by combining lower-level features detected by the filters in previous layers. This hierarchical representation allows CNNs to effectively learn and recognize complex patterns in visual data, making them particularly powerful for tasks like image classification and object detection.Figure II.6 presents the architecture of a typical CNN, let's delve deeper into how these layers work together to process and extract features from input data :

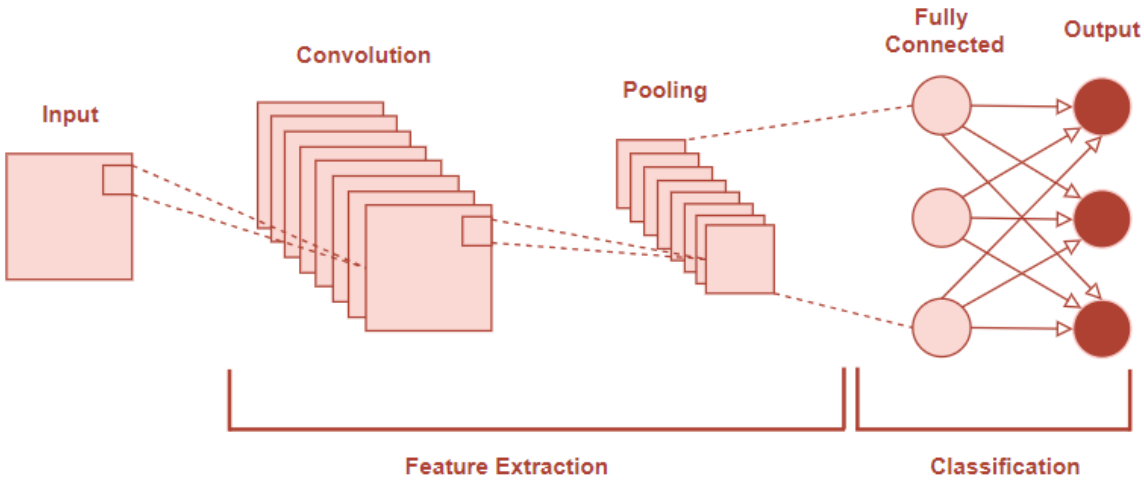


Figure II.6: CNN architecture.

- **Convolutional layer:** It's responsible for extract features from the input data as images or other data type it has a set of learnable filters or kernels. Each filter is a small matrix that slides across the input data and multiplies each element at each position via a convolution operation:

$$x_j = f\left(\sum_{i \in M_j} x_i^{-1} \cdot k_{ij} + b_j\right) \tag{II.6}$$

  - **Padding:** During convolution Three factors impact the size of the output feature map, the stride, the kernel size, and the size of the input feature map. If we simply apply the kernel on the input feature map, then the output feature map will be smaller than the input. This can result in the loss of information at the borders of the input feature map. In Order to maintain the boundary information we use the padding which is adding extra pixels around the border of the input feature map before convolution.
    - **Without padding:** Calculate the output matrix size with this operation :

$$x = n - f + 1 * n - f + 1 \tag{II.7}$$

For a visual representation, Figure II.7 shows a convolutional layer without padding.

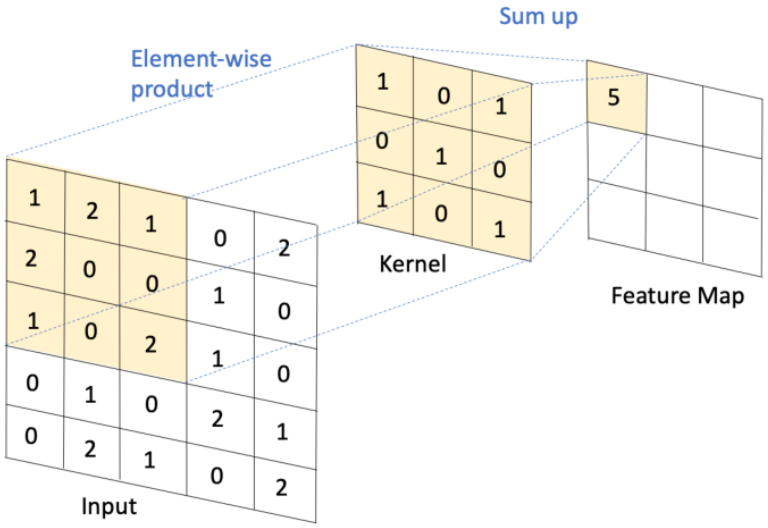


Figure II.7: Convolution Layer Without Padding[3].

∗ **With padding:** In adding padding case the output matrix size calculated with this operation:

$$x = n + 2p - f + 1 * n + 2p - f + 1 \tag{II.8}$$

and padding's value with:

$$p = (f - 1)/2 \tag{II.9}$$

For a visual illustration, see Figure II.8, which demonstrates a convolutional layer with padding.

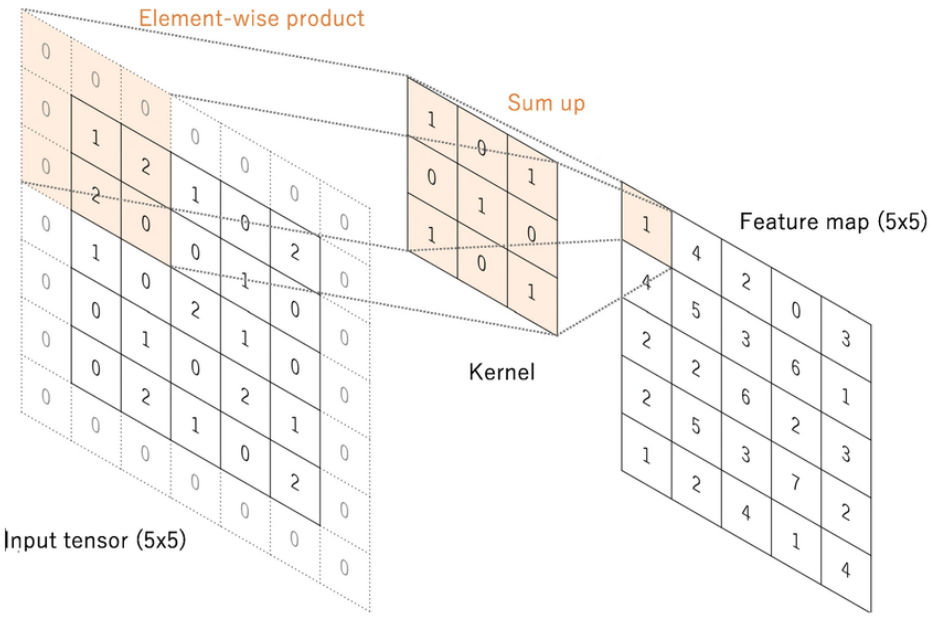


Figure II.8: convolution operation [3].

Where:

- **n:** size of the input n*n
- **f:** size of the filter f*f
- **p:** is the padding value

– **Stride :** It is typically The number of pixels that determines how the filter sweeps across the input data horizontally as well as vertically. The stride controls how the kernel convolves around the input. When the stride is 1, the kernel is shifted over the input one element at a time. The stride is normally set in such a way that the output volume is an integer[20].

- **Pooling layer :** Its main purpose is to reduce the spatial dimensions and aids in lowering the network's computational complexity. Common choices include max pooling, which takes the max within each filter, which uses the average operator[21]. For an illustration of these pooling techniques, refer Figure II.9.

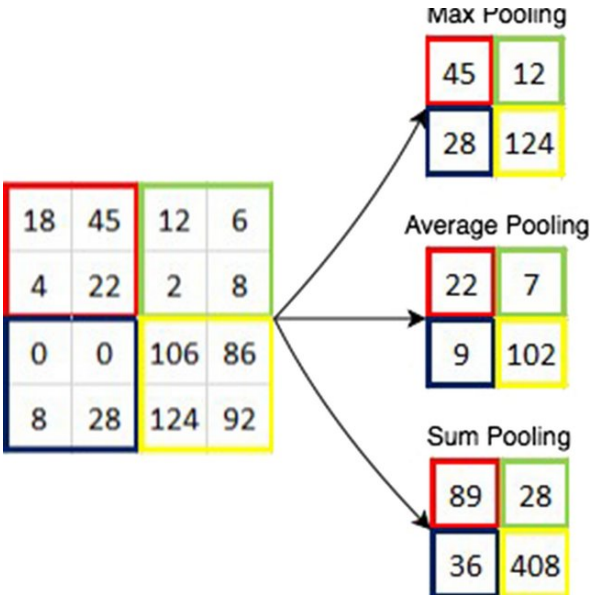


Figure II.9: Illustration of the pooling techniques[4].

- **Fully connected layer:** After extracting features, this layer takes the output and transforms it to 1 dimension array. Each neuron is connected to every element of the array. Consequently, the network will be able to determine patterns among the data and make anticipates

$$y = \sigma(w^T x + b) \tag{II.10}$$

where:

– σ: denotes an activation function

– w : s the weight matrix

– x : represents the flattened input features

– b : is the bias vector

II.2.5 YOLO

YOLO, an acronym for 'You Only Look Once,' is a real-time object detection algorithm in the field of computer vision. This approach employs an end-to-end neural network, capable of handling both object identification and classification in a single pass, defining it as a single-stage detector. The strength of YOLO lies in its ability to quickly analyze entire images or video frames, dividing them into a grid and predicting bounding boxes and class probabilities simultaneously. This efficiency positions YOLO as a prominent solution in applications requiring rapid and accurate object detection.

Key Concepts:

- **Grid-based processing:** YOLO divides the input (image or videos) into grid cells of equal shape, forming an SxS grid ,as shown in Figure II.10. each cell is responsible for predicting the coordinates of the bounding box along with the class of the object that it covers and the associated probability/confidence value for those boxes.



Figure II.10: S × S grid on input[5].

- **Predictions by each grid cell:**

  The algorithm proceeds to identify bounding boxes that include in individual objects within the image. Multiple bounding boxes can be generated, each corresponding to a different object. YOLO uses a single regression module to determine the attributes of these bounding boxes.

  - **Bounding box coordinates :**
    The coordinates typically consist of four values:
    - **Center Coordinates (x, y):** These represent the coordinates of the center of the bounding box within the grid cell. it is related to the size of the gride cell, where (0, 0) is the top-left corner, and (1, 1) is the bottom-right corner of the cell.
    - **Width (w) and Height (h):** These represent the width and height of the bounding box.The values are often normalized relative to the dimensions of the entire image.

- **Object Class Prediction:** Apart from the coordinates of the bounding box, each cell predicts the class of the object it covers

$$\Pr(\text{Class}_i \mid \text{Object})$$

This involves assigning a specific label or category to the detected object. The classes could include various objects like person, car, or any other predefined categories depending on the application.

- **Probability/Confidence Value[5] :** As well as identifying bounding box coordinates and an object's class, each cell also gives a probability or confidence value.This value is a reflection of the model's certainty that an object is present within the predicted bounding box. A higher confidence value indicates a stronger belief in the accuracy of the prediction. It is definded as :

$$\Pr(Object) \times IOU_{\text{pred}}^{\text{truth}} \tag{II.11}$$

In case no objects exists in the cell the probability should be zero. The multiplication of the conditional class probabilities and the individual box confidence predictions gives us class-specific confidence scores for each box[5] .

$$\Pr(\text{Class}_i \mid \text{Object}) * \Pr(\text{object}) * IOU_{\text{pred}}^{\text{truth}} = \Pr(\text{Class}_i) * IOU_{\text{pred}}^{\text{truth}} \tag{II.12}$$

The final vector representation for each bounding box is denoted as Y and is structured in the following format:

$$Y = [\text{pc, bx, by, bh, bw, c1, c2,..,cn}]$$

Where:

- **pc:** Probability/Confidence - Represents the confidence level that the grid cell contains an object.
- **bx:** Bounding Box X-coordinate - Indicates the x-coordinate of the center of the bounding box relative to the grid cell.
- **by:** Bounding Box Y-coordinate - Indicates the y-coordinate of of the center of the bounding box relative to the grid cell.
- **bh:** Bounding Box Height - Represents the height of the bounding box.
- **bw:** Bounding Box Width - Represents the width of the bounding box.
- **c1, c2,..,cn:** Class probabilities - Probabilities associated with each class (object category) in the dataset.

- **Intersection Over Unions:**

Intersection Over Union (IOU) is a metric used to evaluate to evaluate the accuracy of an object detection algorithm by mesuring the overlap between the ground truth bounding box and the predicted bounding box . We calculate it by deviding the area of intersection between the predicted and ground truth bounding boxes union area .

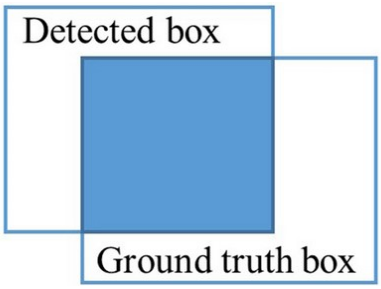


Figure II.11: Area of Intersection

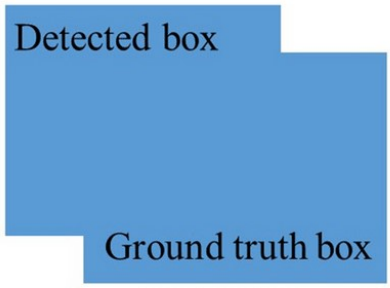


Figure II.12: Area of Union

$$IoU = \frac{\text{Area of Intersection}}{\text{Area of Union}}$$

- **Enhanced loss function** The YOLO loss function is often composed of several terms, including coordinates loss (L_{box}), class probability (L_{cls}) and object confidence loss(L_{conf}) where

  $loss_{i,j} = loss_{i,j}^{xywh} + loss_{i,j}^{p} + loss_{i,j}^{c}$

  - **coordinaates loss**

$$\text{loss}_{i,j}^{xywh} = \frac{\lambda_{\text{coord}}}{N_{ldj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{i,j}^{\text{obj}} \ (\Delta x_{i,j} + \Delta y_{i,j} + \Delta w_{i,j} + \Delta h_{i,j}) \tag{II.13}$$

    with :

$$\Delta x_{i,j} = [(x_{i,j} - \hat{x}_{i,j})^2$$
$$\Delta y_{i,j} = (y_{i,j} - \hat{y}_{i,j})^2$$
$$\Delta w_{i,j} = \left(\sqrt{w}_{i,j} - \sqrt{\hat{w}}_{i,j}\right)^2$$
$$\Delta h_{i,j} = \left(\sqrt{h}_{i,j} - \sqrt{\hat{h}}_{i,j}\right)^2$$

  - **class probability loss**

$$\text{loss}_{i,j}^{p} = -\frac{\lambda_{\text{class}}}{N_{L,obj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{i,j}^{\text{obj}} \sum_{c \in \text{ class}} p_{i,j}^{c} \log\left(\hat{p}_{i,j}^{c}\right) \tag{II.14}$$

– **Object confidence loss**

$$\text{loss}_{i,j}^{c} = \frac{\lambda_{\text{obj}}}{N^{\text{conf}}}\alpha + \frac{\lambda_{\text{nocobj}}}{N^{\text{obj}}}\beta \tag{II.15}$$

with:

$$\alpha = \sum_{i=0}^{S^2}\sum_{j=0}^{B} L_{i,j}^{\text{obj}} \left(IOU_{\text{Prediction }i,j}^{\text{ground truth}} - \widehat{C}_{i,j}\right)^2$$

$$\beta = \sum_{i=0}^{S^2}\sum_{j=0}^{B} L_{i,j}^{\text{noobj}} \left(0 - \widehat{C}_{i,j}\right)$$

where λ_{coord} , $\lambda_{\text{obj}}, \lambda_{\text{noobj}}$ and λ_{class} are the hype parameters, $L_{i,j}^{\text{obj}}$ indicates if an object appears in cell i based on the prediction of the bounding box j.

II.3 Evaluation Metrics of Models:

The performance, accuracy, and efficacy of statistical or machine learning models are evaluated using certain metrics, which are standardised criteria. These metrics provide quantitative insights into how well a model performs on a specific task, such as classification, regression, or clustering. The selection of metrics aids practitioners and researchers in determining the model's dependability and appropriateness for a particular application, depending on the model's nature and the analysis's objectives. Among the common evaluation metrics are:

- **Mean Average Precision (mAP):**

  Mean Average Precision is commonly used in object detection and information retrieval. It calculates the average precision for each class and then averages those values across all classes. Higher mAP indicates better model performance across all classes.

- **F1-score :**

$$F1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{II.16}$$

- **accuracy:**

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Number of Predictions}} \tag{II.17}$$

- **precision:**

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{II.18}$$

- **Recall:**

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{II.19}$$

where:

- True Positives: Number of correctly predicted positive instances.
- True Negatives: Number of correctly predicted negative instances.
- Total Number of Predictions: Total number of instances predicted by the model.

II.3.1 Conclusion

In this chapter, we explored the fundamentals of Artificial Intelligence (AI) and its branch, Machine Learning (ML). We delved into various ML techniques like supervised learning, unsupervised learning, and deep learning. We then focused on Convolutional Neural Networks (CNNs) used for image recognition and YOLO for object detection. Finally, we discussed metrics to evaluate the performance of these models.

Chapter III

Drone Detection

III.1 Introduction

Drone technology presents an exciting and challenging area of research in the field of object detection and recognition. With the increasing prevalence of drones in various industries and applications, it is essential to develop effective detection methods to ensure the safe and efficient operation of these unmanned aerial systems (Unmanned Aerial Systems (UAS)).

In this chapter, we provide a comprehensive guide to drone detection techniques, highlighting their potential applications and advancements. The chapter covers essential detection methods, including radar-based, multi-sensor fusion. These techniques are highly effective for tracking and identifying drones, offering accurate and reliable solutions. Furthermore, we explore small object detection strategies, showcasing feature fusion and enhancement techniques, as well as YOLOv5's innovative approach. These strategies overcome the challenges of detecting small objects, improving accuracy and precision.

III.2 Introduction to Drones:

A drone is an aircraft without a pilot, controlled from the ground, used for taking photographs, dropping bombs, delivering goods, etc[22].they are more formally known as unmanned aerial vehicles (UAVs) or unmanned aircraft systems with embedded systems that integrate onboard sensors and a global positioning system (GPS)drones can be remotely controlled or fly autonomously through the use of software-controlled flight plans.

- **A Quick Guide to Types and Uses:** Drones comes in various types as shown in the Figure III.2, each designed for specific purposes and functionalities. Here are some common types of drones:
  - **Multi-Rotor Drones:** Multi-rotor drones, also known as rotary-wing, These drones have gained popularity for both personal and commercial use due to their affordability and ease of use. Additionally, they are ideal for applications like aerial photography and surveillance because of their compact size and exceptional control over position and framing. The term "multi-rotor" originates from the fact that these drones incorporate more than one motor,The most common type in this category is the quadcopter, but there are also common variants like octocopters, which have eight rotors, and hexacopters, which have six rotors. Figure III.1 shows a sample of multicopter models.



Figure III.1: Multicopter models [6]

- **Fixed-Wing Drones:** Unmanned aerial vehicles with fixed wings, or fixed-wing drones, are built like conventional airplanes and have fixed wings. The aerodynamic form of these drones allows for effective long-range travel over wide-open spaces. They are frequently used in mapping, surveying, and agricultural applications. They offer extended flight durations, making them ideal for tasks requiring coverage of large areas and detailed data collection.
- **Single-Rotor Drones:** Single-rotor drones mimic the design of traditional helicopters with one rotor provide power,plus a tail rotor to control direction and stability. They are strong and durable offering longer flight times and greater payload capacity compared to multi-rotor drones. They are suitable for tasks like surveillance and industrial inspections.
- **Fixed-Wing Hybrid VTOL:** Hybrid drones combine the features of both multi-rotor and fixed-wing designs to offer both the efficiency of fixed-wing flying and the vertical take-off and landing (VTOL) capabilities of multi-rotor drones. This design is useful for applications that need to be flexible in many types of terrain. There are only a few hybrid models available now, but as technology gets better, more people might start using them in the future.

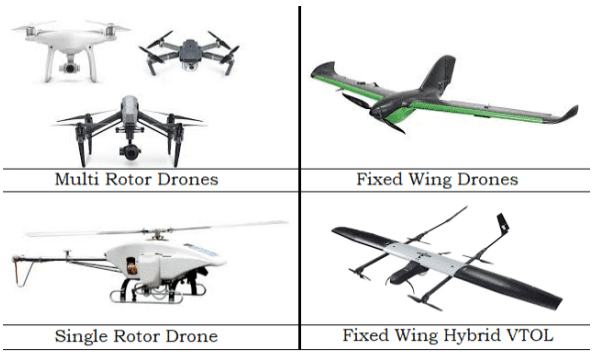


Figure III.2: Four Major Types of UAV (drone)[7].

III.3 Object Recognition

Object recognition is a general term to describe a collection of related computer vision tasks that involve identifying objects in digital photographs. Object recognition is a fundamental process that serves as a gateway from vision to cognitive processes such as classification, language, and reasoning. The visual representations that allow us to recognize items can tell us more than just what We are examining[23]. For a comprehensive view of these tasks, consult Figure III.3.

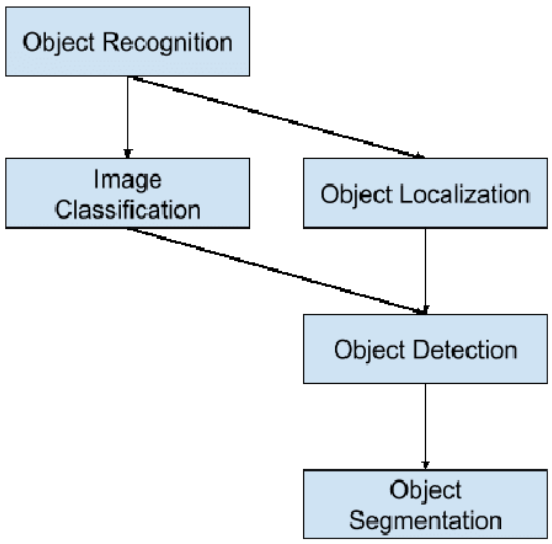


Figure III.3: Overview of Object Recognition Computer Vision Tasks [8].

We can distinguish between these three computer vision tasks:

- Image classification : Involves predicting the class of one object in an image. There are a main processes in the classifying process as shown in Figure III.4:

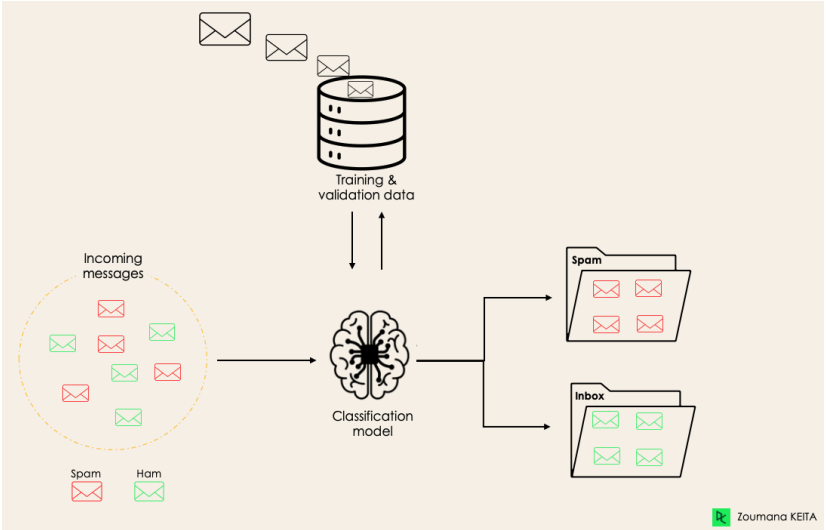


Figure III.4: classification example [9].

- **Feature Extraction in Image Classification:** It is a crucial pre-processing phase, which comes before we start the classification process. It is a technique strategically employed to optimize processing resources without compromising essential data integrity.The main goal of this technique as illustrated in Figure III.5 is to extract unique features from the input data, like corners, edges, and textures. Because these variables capture the distinctive characteristics that distinguish different objects, they are essential to accurate classification. By quantifying and describing these features, methods such as Histogram of Oriented Gradients (Histogram of Oriented Gradients (HOG)), Speeded Up Robust Features (Speeded Up Robust Features (SURF)), and Local Binary Patterns (Local Binary Patterns (LBP)) open the door to more successful and resource-efficient classification[10].

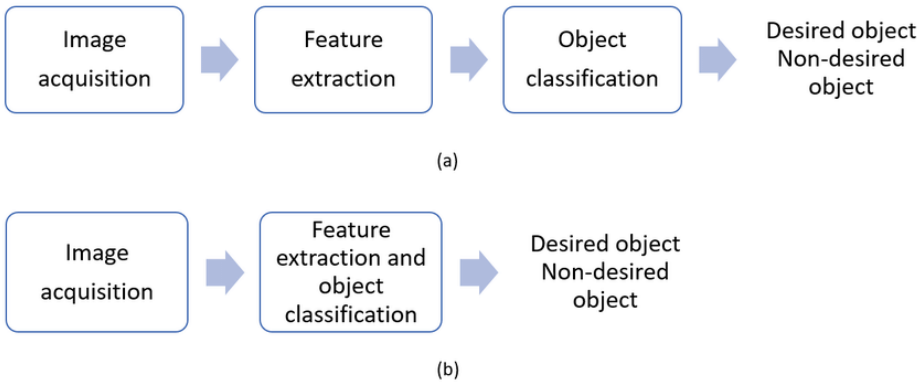


Figure III.5: Steps needed for object classification. (a) Features-Based Model. (b) Deep Learning Strategy.[10]

- **The traning phase:** Training data consists of input-output pairs, the input refers to the data provided to the model, which could include features and labels for supervised learning algorithms such as Support Vector Machines (SVM), or raw data for unsupervised learning algorithms like K-means. For neural network algorithms such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Long Short-Term Memory (Long Short-Term Memory (LSTM)) networks, the input typically consists of the input data itself, which the model processes to make predictions or learn patterns, and the output is the corresponding label or target that the model should predict.
- **Testing Data** After having training the model we should test it also known as validation data essential part in order to determine how effectively a trained model generalizes to new, unseen cases it is used to evaluate the model's performance via various metrics, including the widely used F1-score, accuracy, precision, and recall

- **Object localization:** Refers to identifying the location of one or more objects in an image and drawing abounding box around their extent. Several of the most popular techniques for object localization include: Bounding Box Regression,Single Shot MultiBox,Detector(Single Shot Multibox Detector (SSD)), You Only Look Once (YOLO), Mask Region-based Convolutional Neural Network (R-CNN) and Region-based Convolutional Neural Networks (R-CNN). Figure III.6 provides visual demonstrations of drone localization.



Figure III.6: Samples of drone localization[11].

- **Object detection:** Combines these two tasks and localizes and classifies one or more objects in an image. It goes further simple object localization by not only determining the location of objects but also classifying them into predefined categories such as : Faster R-CNN, MobileNet-SSD, YOLO and RetinaNet. Figure III.7 illustrates the output of an object detection model applied to an image.

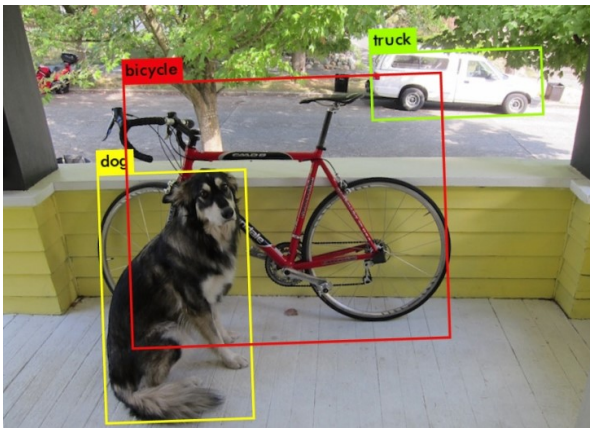


Figure III.7: object-detection[11].

III.4 Small Object Detection:

The field of object detection has witnessed significant advancements with the development of advanced algorithms, showcasing unprecedented results in accurately identifying and classifying objects. However, despite these advancements,there is still a significant gap, one that grows significantly when it comes to the challenging problem of small object detection.

Small object detection(SOD) demands specialized attention due to the limited visual footprint of those objects such as drones. The complexity of precisely locating and classifying these little objects poses special difficulties that are beyond the scope of existing object detection methods. Address the challenge of small object detection by exploring the essential methods employed to overcome this obstacle and narrow the existing gap. Here are some of the used techniques to solve this problem:

- **Feature Fusion and Enhancement for Small Object Detection:** One notable solution that shows up in the literature for addressing the complexities of small object

detection is the approach centered around Feature Fusion and Enhancement. In this regard, two distinct methodologies are highlighted.Firstly, in the context of Single Shot Multibox Detector (SSD), Feature Fusion and Enhancement involve the integration of enhanced features, achieved through methods such as pooling and deconvolution on the VGGNet backbone, into the SSD architecture. This integration enhances the overall capability of SSD, with a specific emphasis on improving performance for small object detection [12]. The process includes the utilization of a Feature Fusion Module III.8

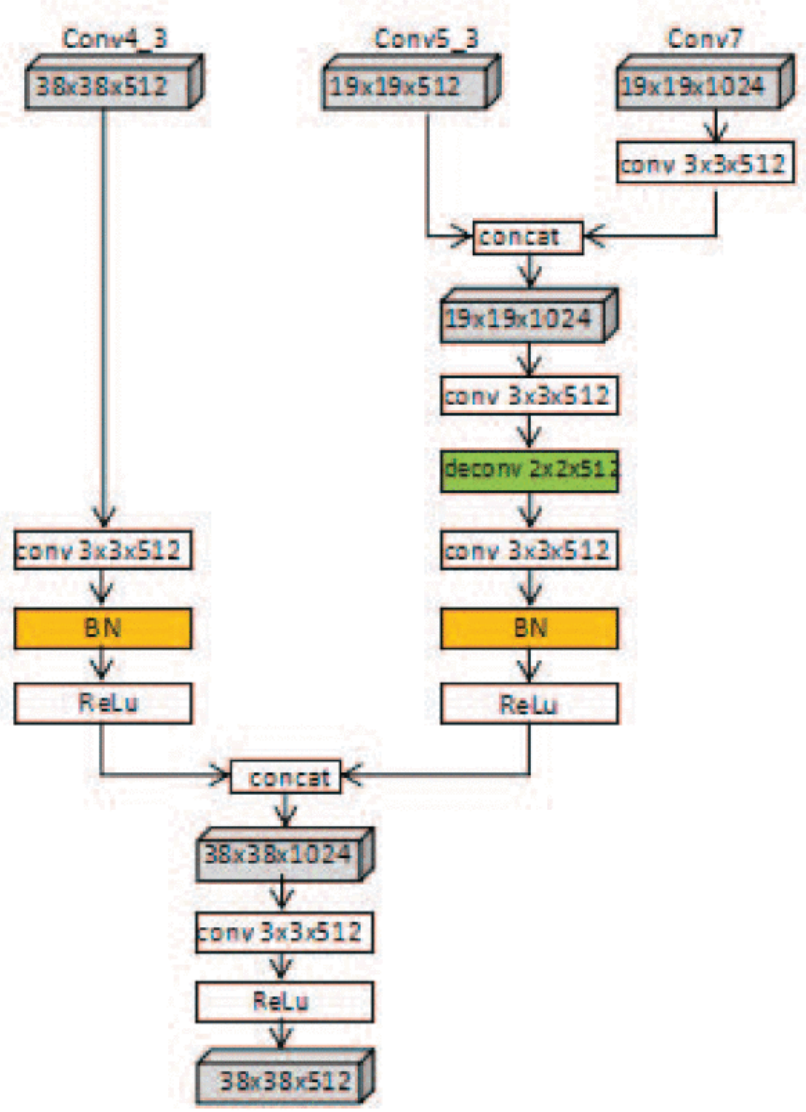


Figure III.8: Feature Fusion Module[12].

Secondly, feature fusion techniques are introduced to the design of the Faster R-CNN framework through Feature Fusion and Enhancement. Enhanced features from channel conversion and VGGNet backbone operations are seamlessly incorporated into the Faster R-CNN architecture. The fused features, denoted as P2 P5 as shown in (III.9), are directed to the Region Proposal Network (RPN) layer for training.This integration significantly improves the detection effectiveness of Faster R-CNN, especially in scenarios involving small objects[13].

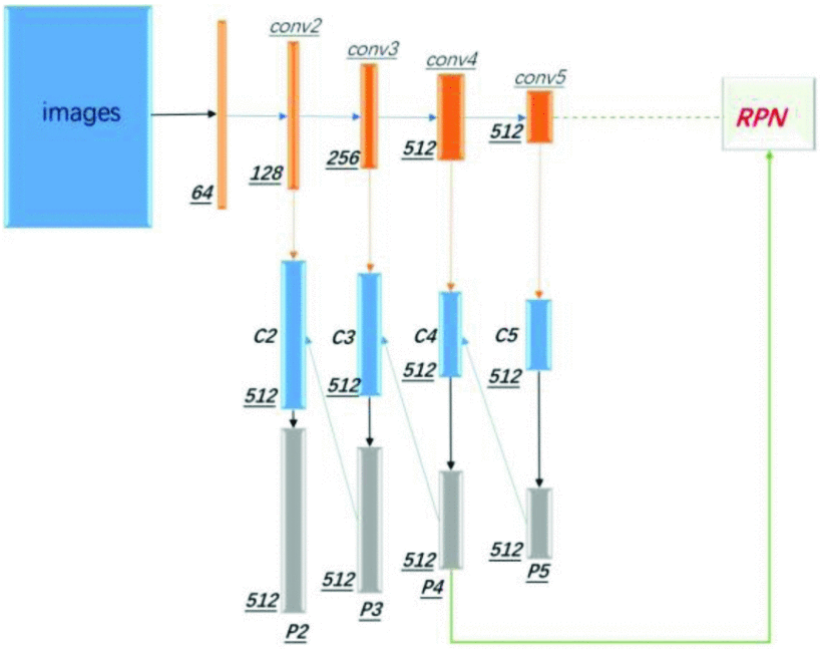


Figure III.9: Fusion Method Overview[13].

- **YOLOv5's strategies[14]:**
  - **Data Augmentation(Magnigier Method):**
    The Magnifier Method (Magnifier Method (MM)) is a data augmentation technique used for improving the detection of small objects. This technique increases the relative size of small objects in an image, by cropping the image, zooming in on the object of interest, and then merging the detection results, YOLOv5 effectively converts the task of detecting small objects into detecting medium-sized objects .
  - **FPN+PAN Module:** YOLOv5 uses a combination of the classic Feature Pyramid Network (FPN) (Feature Pyramid Network) and Position-Aware Network (PAN) (Position-Aware Network) to create a powerful neck network. This network strengthens the positioning of objects, making it an effective strategy for detecting small objects.
  - **Reconstruction Detection Branch (Reconstruction Detection Branch (RDB)):**
    This branch is specifically designed to detect small pixel objects. By focusing on these objects, YOLOv5 can improve its precision in detecting smaller objects.

  The full structure and integration of these components are shown in Figure III.10, which provides a detailed visual representation of the architecture based on YOLOv5s model.

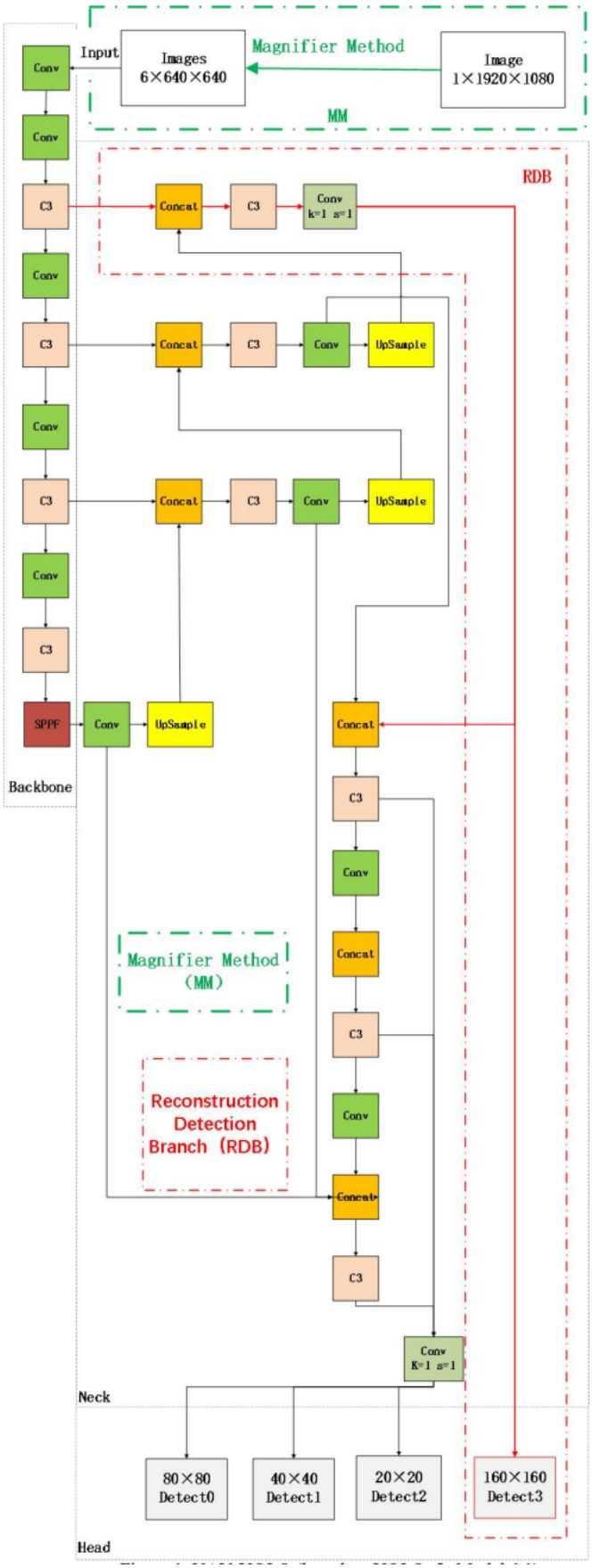


Figure III.10: uav-yolo(based on YOLOv5s Model)[14].

These methodologies allow YOLOv5 to excel in speed and precision, making it a powerful tool for real-time detection and analysis in various applications.

III.5 Drone's apllication

Drones are useful in many contexts for a number of reasons, including cost savings, the elimination of the need to deploy personnel to hazardous or difficult-to-reach areas, and quicker, more productive work. Their biggest potential applications lie in civil and public sectors, encompassing a wide range of areas: transportation, security services, damage management, agriculture, alpinism, municipalities, environmental protection, military and police support, and even disaster protection/management[24].

III.6 Related Works

The author of [25] presented a CNN-based drone detection method that combines Support Vector Machine (SVM) and KNN with CNN. KNN is employed to determine whether or not a particular data point (such as an image) contains a drone, and SVM is cited for information categorization, which may be understood as a stage of the detection process. SVM is utilized to determine whether or not data points (such as photographs) contain drones and uses CNN to examine pictures and extract essential information that helps identify drones in the pictures. According to the outcomes, the CNN, SVM, and KNN classifiers' sensitivity, precision, and accuracy are as follows: CNN—Sensitivity 91%, Precision 96%, Accuracy 93%; SVM—Sensitivity 94%, Precision 86%, Accuracy 88%; KNN—Sensitivity 91%, Precision 74%, Accuracy 88%. The results show that the CNN-based method performs better in terms of sensitivity, precision, and accuracy for drone detection. In [26], a number of state-of-the-art techniques that make use of deep Convolutional Neural Networks (CNNs) are discussed. Faster R-CNN was used in the experiments together with the Caffe deep learning package, and transfer learning from ImageNet was used to fine-tune the models due to the dataset's restricted quantity of images for deep learning. The results of the experiment show that the VGG16 model, trained for 80k iterations, achieved the highest mean average precision (MAP) of 0.66, outperforming the ZF and VGG_CNN_M_1024 models, which achieved MAPs of 0.61 and 0.60, respectively, after 100k and 90k iterations. The work in [27] used CNN with YOLOv3 and background subtraction methods. the work [28]evaluates a variety of deep learning models, including as hybrid models that combine CNN and LSTM, Long Short-Term Memory (LSTM) networks, and Convolutional Neural Networks (CNNs) ,To train and evaluate their algorithms, researchers use real data from the Automatic Identification System (Automatic Identification System (AIS)). Among these models, the hybrid LSTM-CNN model is especially significant since it outperforms the individual LSTM and CNN models in terms of vessel manoeuvring behaviour prediction. This opens the door to the development of precise digital twins, which aid in the optimisation and real-time decision-making of maritime operations Additionally, Wisniewski et al in the work [29]. (2022) used convolutional neural networks (CNNs) trained on synthetic data to study drone model categorization. They classified drone models with an accuracy of 92.40% by using several DeseNet designs (DenseNet121, DenseNet161, DenseNet169, and DenseNet201) and Residual Network (ResNet) architectures (ResNet18, ResNet50, ResNet50-wide, and ResNet101). By utilising the advantages of cutting-edge neural network designs to handle synthetic datasets, this study demonstrates the efficacy of CNN-based techniques for drone detection and classification.

III.7 Conclusion

In this chapter, we explored drone detection, a critical aspect for safe and efficient operation of unmanned aerial vehicles (UAVs). We discussed various drone types and their applications, then delved into object recognition tasks like classification and detection, with a specific focus on overcoming challenges in detecting small objects. We explored methods like feature fusion and YOLOv5's strategies to improve detection accuracy.

Chapter IV

Conception

IV.1 Introduction

In this chapter We present the considerable structure that forms the basis of our drone detection model and the techniques implemented in it . This carefully constructed framework is divided into multiple important stages, each of which is expertly constructed to build upon the previous one. Acquisition of data, pre-processing, model training, and real-world assessment are some of these stages. Every stage is essential to guaranteeing the detection system's overall accuracy, flexibility, and dependability of the overall detection system.

IV.2 Proposed Methodology

Our project on drone detection progresses through a well-planned structure. We are guided by this paradigm through three interrelated phases: dataset, pre-processing, model training and real-world evaluation. Initially, we gathered a large set of annotated datasets, which served as a solid basis for our analysis. After gathering the data, we carefully processed it across a number of pre-processing stages to make sure it was ready for analysis. Subsequently, we proceeded to the training stage, utilising advanced algorithms to precisely train our model for drone recognition. In order to determine the model's efficacy and dependability in practical situations, we lastly carried out extensive assessments of its performance. To enhance comprehension of our development framework, we've crafted a schematic representation of its architecture, as depicted in Figure IV.1. This visual aid elucidates the three fundamental phases that underpin the design and implementation of our framework.

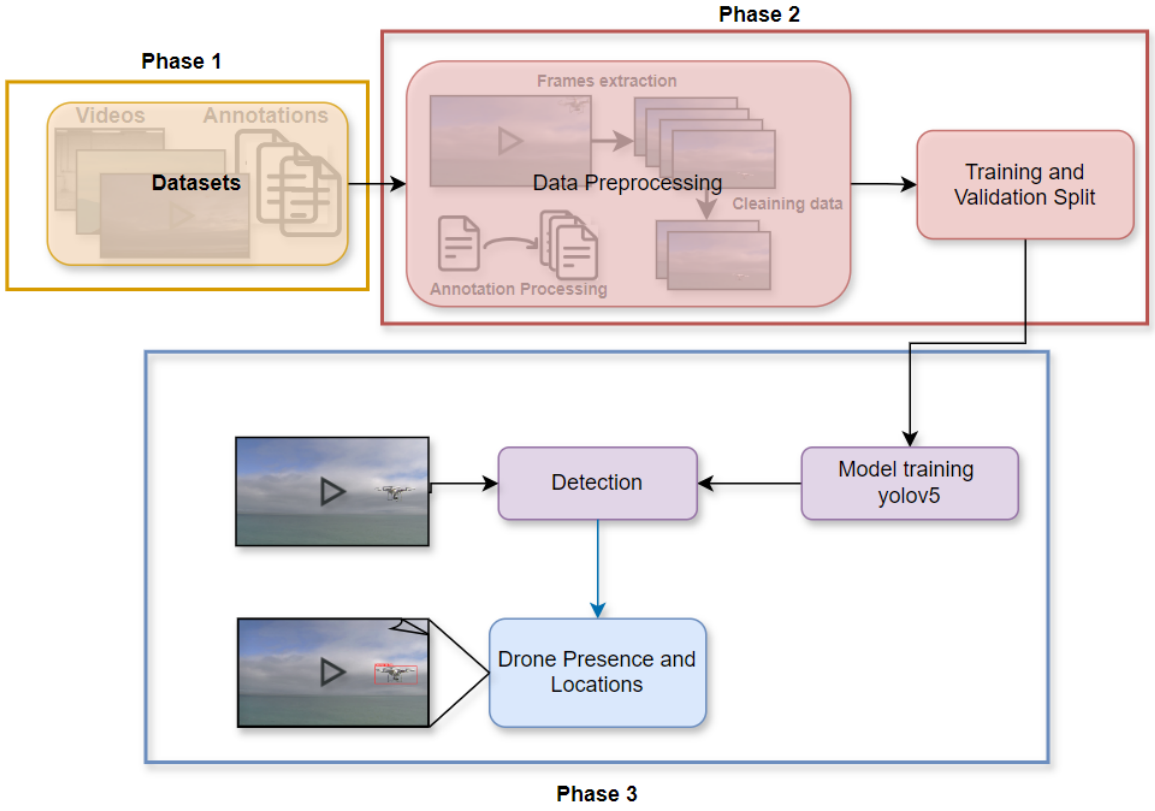


Figure IV.1: Schematic Representation of the Drone Detection Model Development Framework.

IV.3 Phases

IV.3.1 Datasets

To develop an effective small drone detection system, we began by focusing on collecting relevant datasets featuring examples of small drones. Through extensive research, we identified the Safeshore dataset [30] as a comprehensive and valuable resource in this domain. This dataset provides a wide range of samples that depict small drone activities in various scenarios and against complex backgrounds. Additionally, it offers well-annotated videos, detailing the precise locations and movements of drones within each frame.

Choosing the right dataset is a critical part of our process. To ensure the datasets meet the needs of our models and research objectives, we carefully select and evaluate them. The effectiveness and precision of our deep learning models depend fundamentally on this meticulous consideration.

Dataset 1 : SafeShore dataset

the Drone-vs-Bird dataset is from the challenge called Drone-vs-Bird Detection.the dataset for this challenge is dynamic,always expanding and being updated for new challenge editions. It is made available to the community for research purposes, encouraging more developments in this field. this datasetinvolves of video captured on camera that includes birds and drones. Compared to static photos, these videos evidently reflect real-world situations, giving drone recognition systems a more realistic training environment. There are matching annotations for every video frame that are kept in different text files. Annotation files contain vital data for our algorithms that are used to detect drones. An annotation file's lines each represent one frame, or image, in a video sequence. The frame number is specified in the first element, which helps the algorithm determine the information's temporal sequence. To help the model determine how many drones or birds are present, the second element shows the number of objects observed in that frame. Every object in the frame might be given a distinct identity by an optional third element. The items' location and dimensions are specified by the following four elements: the bounding box's top-left corner's X and Y coordinates,then the bounding box's pixels for height and width. Lastly, the class label is the most important component for classification. The label "drone" indicates the type of object found inside the bounding box, allowing the computer to identify the visual characteristics that set drones apart from birds[30].

Dataset 2 : USC-GRAD-STDdb

The USC-GRAD-STDdb dataset offers a valuable resource for researchers working on small object detection. It features 115 HD 720p video segments (roughly 1280x720 resolution) containing over 25,000 annotated frames. Notably, these frames focus on small objects of interest, ranging from a mere 4x4 pixels to 16x16 pixels in size. Video lengths vary between 150 and 500 frames, further adding complexity. The dataset boasts high-quality annotations achieved through meticulous bounding box labeling of each object. This ensures reliable evaluation metrics, especially considering the inherent challenge of annotating such tiny objects. The creators employed the ViTBAT tool and made significant efforts to precisely align bounding boxes with objects in each frame, resulting in over 56,000 ground truth labels. This detailed annotation process makes the USC-GRAD-STDdb a valuable tool for researchers developing and evaluating small object detection algorithms Given the presence of numerous potential small object candidates, the annotation is limited to targets that have the potential to move, even if they remain stationary in a particular frame or set of frames. Most of the

videos in the USC-GRAD-STDdb are recorded by drones or from a bird's eye view, covering three main landscapes and five object categories[15], specifically:

- **Air:** drone, bird (57 video segments, 12,139 frames)
- **Sea:**boat (28 video segments, 7,099 frames)
- **Land:**vehicle, person (30 video segments, 6,619 frames)

Figure IV.2 is a visual representation of the of the dataset's object ditribution ,which illustrates the histogram of object occurrences across different categories.

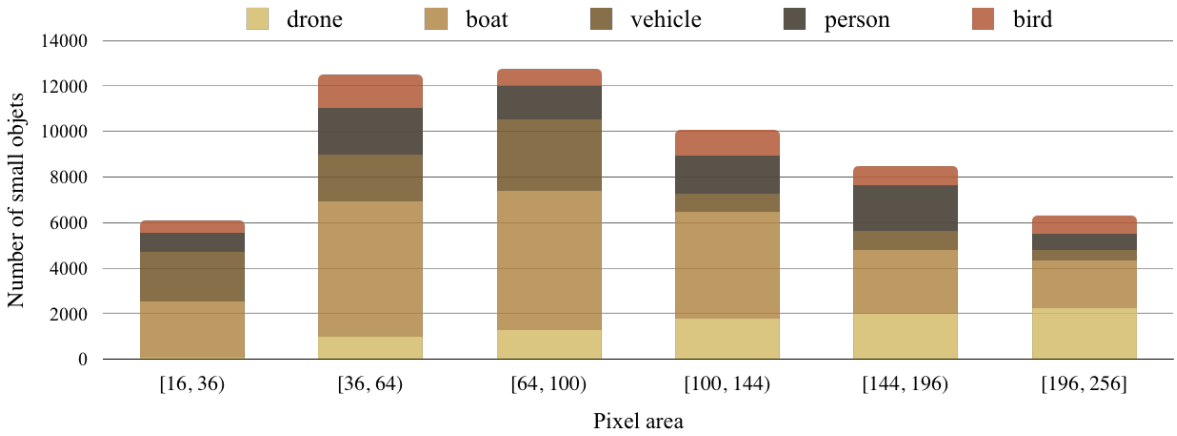


Figure IV.2: Data Histogram [15]

Dataset 3 : Unnamed Drone Detection and Tracking Dataset

This dataset comprises 14 video sequences, each containing approximately 4000 frames with a resolution of 752×480 pixels. The videos were captured by a camera mounted on a drone, filming similar drones while flying outdoors. Each frame may contain up to two objects of the same model. Due to various factors such as changing orientations, lighting conditions, and the small size of the drones, their shapes are often not perfectly visible, leading to highly variable appearances. The dataset includes videos recorded in a variety of indoor and outdoor environments under different lighting and weather conditions, providing a wide range of scenarios for drone detection and tracking tasks[31][32].

For detailed insights into the datasets we utilize, refer to Table IV.1, which provides comprehensive information about the datasets.

Table IV.1: Information about the datasets

Dataset	Nb Videos	Width/Height	FPS	Small drones
Dataset 1	77	1920x1080 3840x2160 720x576	14.00 - 50.00	Yes
Dataset 2	67	1280x720	23.98 - 30.00	Yes
Dataset 3	14	640x480	25.00 - 40.00	No

IV.3.2 Training Data Selection

For our training, we used a subset of the three datasets. This dataset was selected for several reasons. From each dataset, we carefully selected frames from each video, focusing on

those that contain drones. Specifically, we used a total of 7112 frames for training. Among these, 6693 frames have at least one drone present. For validation, we used 1706 frames, with 1683 of them containing at least one drone. This selection ensures that our model is trained and validated on relevant data, improving its ability to detect drones effectively in various scenarios. Figure IV.3 illustrates this distribution with a histogram, emphasizing the balanced representation crucial for effective drone detection model training and validation.

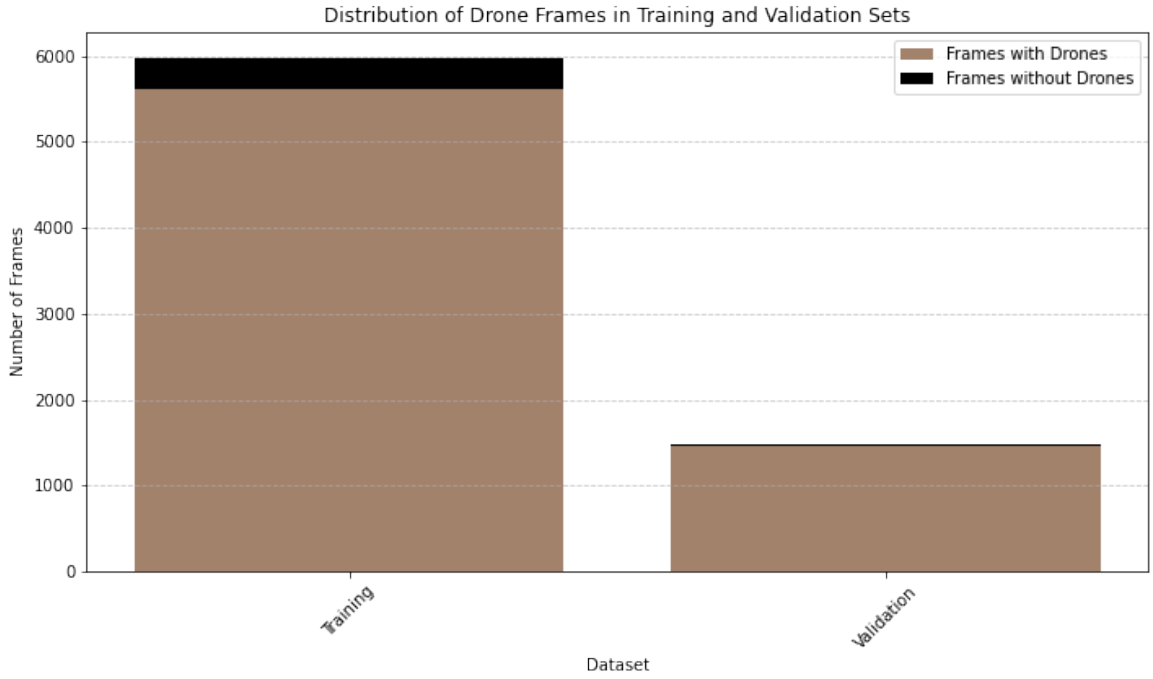


Figure IV.3: Frame Distribution for Training and Validation Sets.

The breakdown of frames used from each dataset is as follows:

- **Dataset1:** 5982 frames for training, 1474 frames for validation.
- **Dataset2:** 473 frames for training, 97 frames for validation
- **Dataset3:** 657 frames for training, 135 frames for validation

As can be seen in the histogram of Figure IV.4, Dataset1 contributes the most frames for training, followed by Dataset3 and Dataset2.

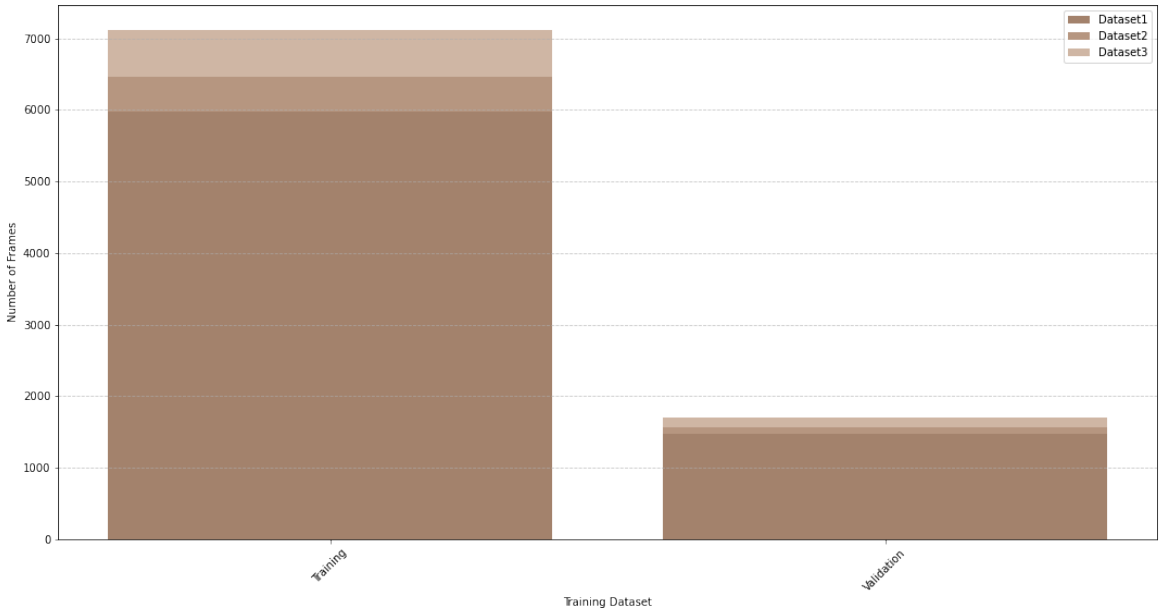


Figure IV.4: Frame Distribution for Training and Validation Sets.

IV.3.3 Data Preprocessing

After obtaining the dataset, our next step involved preprocessing the data to ensure its suitability for training our detection model. Preprocessing is an important step in our workflow as it involves transforming raw video data into a format that our model can effectively analyze. This process includes tasks such as splitting the videos into individual frames, resizing images for consistency, and organizing annotations into a standardized format. Furthermore, we employed a preprocessing technique designed to increase the model's sensitivity to minute variations in drone position and size. To be more precise, we adjusted the frame selection process to account for the varying distances of small drones, ensuring that our model doesn't waste time learning from nearly identical images. For instance, consider frames 536 and 537 (Figure IV.5), where a small drone is depicted. The differences between these successive frames are frequently unnoticeable due to the drone's small size and minimal movement. However, by jumping 16 frames ahead to the frame 552 (Figure IV.6), we can observe slight changes in the drone's position.



Figure IV.5: Comparison of Frames 536 Frame 537



Figure IV.6: Comparison of Frames 536 and 552

On the other hand,in scenarios involving larger drones, such as the one depicted in frame 40 (FigureIV.7), jumping 16 frames ahead results in a significant difference in the drone's position due to its larger size and potentially faster movement.



Figure IV.7: Comparison of Frames 40 and 56

By capturing frames at larger intervals when small drones are detected, we prevent the model from achieving high accuracy in validation but failing in real-life scenarios. This approach provides our model with a more diverse and representative dataset, enhancing its ability to detect small drones accurately in various situations.

IV.3.4 Annotation Filtering

During the preprocessing phase for training the YOLO model, it was observed that some frames indicated the absence of objects according to the annotations, despite the presence of small drones. To address this discrepancy, frames were manually inspected, and those with overlooked small drones were removed from the dataset. This meticulous manual review ensures that the YOLO model is trained on accurate and reliable data, free from misleading or incomplete annotations. By eliminating frames with potentially overlooked small drones, the dataset undergoes refinement, enhancing the model's ability to detect objects with precision and reliability. This proactive data cleaning step contributes to the overall effectiveness of the YOLO model in real-world object detection scenarios.

IV.3.5 Training and Validation Split

After preprocessing the dataset, we split the data into training and validation sets. For each video, we used 80% of the frames for training and 20% for validation. This split allowed our model to learn from diverse data while also testing its performance on unseen examples.

Our goal was to prevent overfitting and ensure the detection model could generalize well to identify new instances of small drones in real-world scenarios.

IV.3.6 Training and Detection

Having preprocessed and split the dataset, the next phase of our project involved training our detection model.For this purpose, we opted to utilize YOLOv5, a state-of-the-art object detection algorithm renowned for its accuracy and efficiency.

In our implementation of YOLOv5, we employed the model to conduct real-time object detection in drone imagery. The preparation of a specialized dataset comprising drone images is pivotal for the training phase. Configuration entails establishing parameters like batch size, image size, learning rate schedule, and the number of training epochs. These parameters play a crucial role in shaping the training process and ultimately influencing the model's performance.

After the training phase,during which the YOLOv5 model uses annotated training data to learn how to recognize objects in photos. In this stage, objects of interest are located and identified by applying the trained model to fresh, unviewed photos.

IV.4 Chosen Drone Detection Model: YOLOv5

Object detection is a very fundamental and well-studied task in the community of computer vision. The purpose of the object detection task is to classify and localize target objects in an image. With the recent advancements in deep learning technologies over the years, several state-of-the-art methods for object detection have emerged . Object detection has been widely applied to many real-world applications, including autonomous driving, robot vision, intelligent transportation, remote sensing, military operations, and surveillance Several object detectors usually perform well on large objects but poorly on small objects[33]. For our task we proposed yolov5.

We have chosen You Only Look Once v5 (YOLOv5) as our deep learning architecture due to its ability to achieve real-time performance while maintaining good accuracy for object detection. This balance is crucial for our project, as we aim to develop a system for real-time drone detection on embedded systems with limited computational resources. Additionally, YOLOv5's user-friendly framework and active community support will facilitate the development process.

IV.5 SAHI-powered YOLOV5

In the conception phase, we considered various approaches for object detection within our project. We opted for a combination of two powerful tools: Slicing Aided Hyper Inference (SAHI) (Slicing Aided Hyper Inference) [34] [35] and YOLOv5 (You Only Look Once version 5). While YOLOv5 excels at object detection, it can face challenges when dealing with small objects within high-resolution images. This is where SAHI comes into play. SAHI addresses this challenge by dividing the high-resolution image into smaller, more manageable slices. Each slice is then processed by the object detection model (in our case, YOLOv5). This allows the model to focus on a more detailed view of the smaller objects within each slice, potentially leading to higher detection accuracy for these objects.

IV.5.1 Integration and Workflow:

The figure IV.8 represent the integration of SAHI and YOLOv5 that involve the following steps:

- **Image Preprocessing:** The high-resolution image would be divided into smaller slices using SAHI.
- **Object Detection on Slices:** Each image slice would be fed into the YOLOv5 model for object detection. This leverages YOLOv5's strengths for identifying objects within each slice.
- **Detection Aggregation:** The individual detections from each slice would be stitched back together to create a comprehensive set of detections for the entire original image.

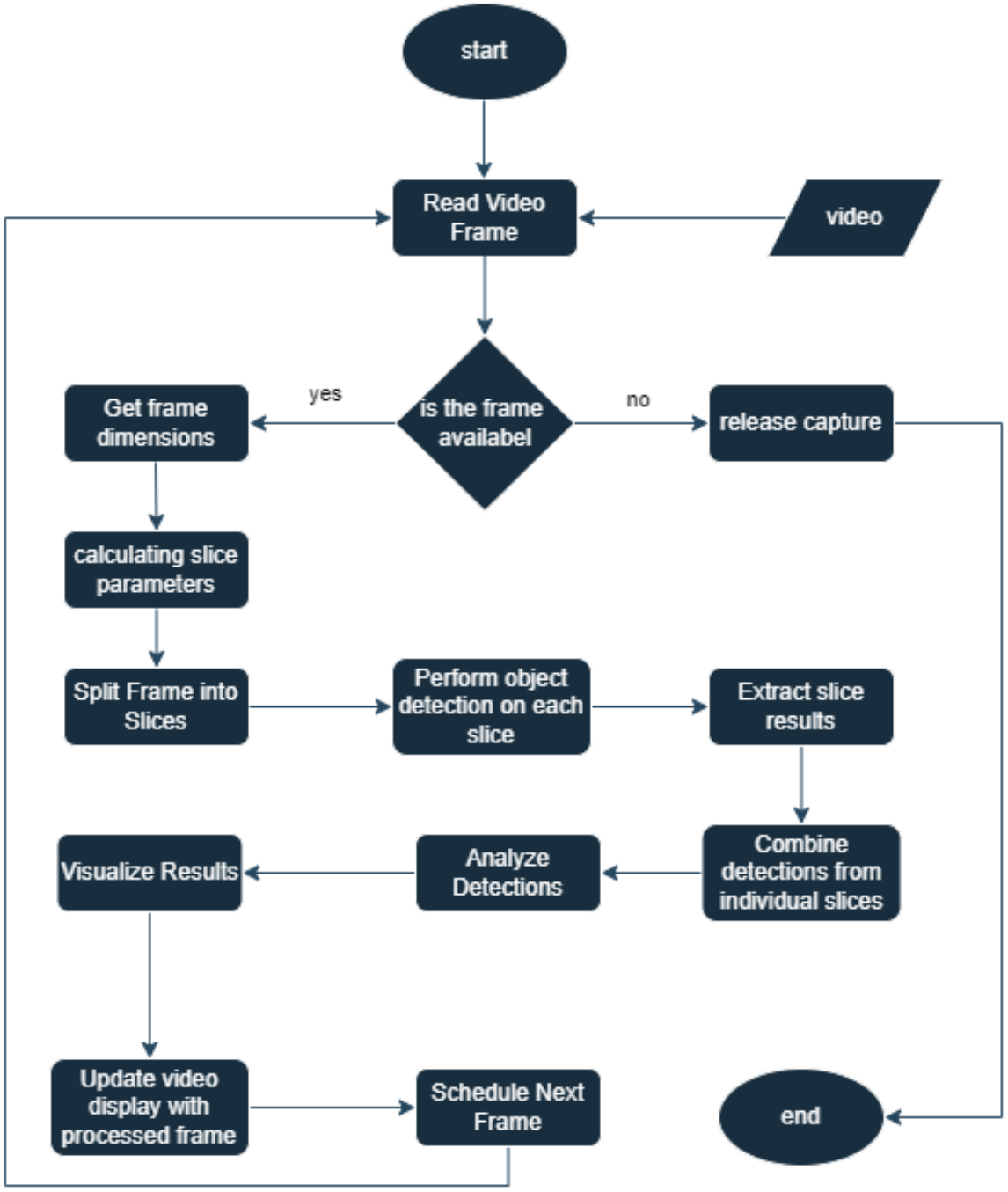


Figure IV.8: Overall Workflow for Object Detection with SAHI and YOLOv5

IV.5.2 Benefits of this Approach:

- **Improved Accuracy for Small Objects:** By combining SAHI and YOLOv5, we aimed to achieve superior detection accuracy for small objects compared to using YOLOv5 alone on the full-resolution image.
- **Maintaining Real-time Performance:** While SAHI introduces some processing overhead, YOLOv5's efficiency helps maintain real-time or near real-time performance, crucial for our application's functionality. Considerations:

The optimal slice size needs to be carefully chosen to balance accuracy and processing efficiency. We planned to evaluate different slice sizes during the development phase to find the optimal configuration for our specific needs.

IV.6 Ensemble learning

In our project, we leveraged ensemble learning to enhance the performance of our drone detection system. Ensemble learning is a technique that combines multiple models to produce a single, more accurate prediction, leveraging the strengths of each individual model.While YOLOv5 performs well in detecting drones and small drones, YOLOv5-P2 excels specifically in detecting smaller drones but struggles when these drones are closer and appear larger in the frame. By combining these models through ensemble learning, we aimed to leverage the strengths of both models to achieve more accurate and robust detections. This approach mitigates individual model weaknesses and improves overall detection performance, ensuring our system remains effective across various drone sizes and distances in real-world scenarios.

IV.7 Conclusion

In this chapter, we outlined the structure of our drone detection model, covering data collection, preprocessing, model training, and real-world evaluation. Each stage is essential for ensuring the system's accuracy and reliability. We chose YOLOv5 for its real-time performance and integrated SAHI to enhance detection of small objects. Additionally, we used ensemble learning to combine different models, improving overall detection accuracy. This structured approach provides a solid foundation for developing an effective drone detection system.

Chapter V

Test And Results

V.1 Introduction

The foundation of any successful project lies in the tools used to bring it to life.This section explores the development environment that supports our solution and offers a road map for others who want to build on or duplicate our efforts or expand upon our work.

This chapter is divided into four main sections: development environment ,dataset results and classification.

V.2 Development Environment

In order to bring our solution to life, we rely on a specific set of hardware and software tools. This section offers insight into the tools we've employed for our project.

V.2.1 Hardware specifications

In order to implement our solution, we utilized the following hardware components:

- A laptop powered by an Intel Core i5-6300HQ processor clocked at 2.30GHz, with 16.0 GB of installed RAM, and equipped with a 512 GB SSD hard drive, running the Windows 10 Pro operating system, was utilized specifically for pre-processing tasks in our project.
- And a laptop equipped with an Intel Core i5-1135G7 microprocessor, capable of reaching speeds up to 4.2 GHz with Intel Turbo Boost Technology. The processor features an 8 MB L3 cache and has 4 cores. The laptop includes an Intel® Integrated SoC chipset, 12 GB DDR4-2666 SDRAM (1 x 4 GB + 1 x 8 GB), a 1 TB 5400 rpm SATA HDD, and a 250 GB SSD nvme.
- We utilized Kaggle's GPU resources, including T4 x2 and P100, for training and validating our model. Additionally.

V.2.2 Software specifications and libraries

Software:

- **Jupiter:** JupyterLab is the latest web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality[36].
- **Python:** Python (versions 3.10.13 on Kaggle and 3.9.12 locally) is an interpreted, high-level, object-oriented programming language. It emphasizes readability with its simple syntax, making it easy to learn and maintain. Python's dynamic features and built-in data structures make it suitable for rapid application development, scripting tasks, and data science. It's free and open-source, available for all major platforms[37].

Libraries

- **Numpy:** NumPy, the workhorse of scientific Python, empowers numerical computations with efficient multidimensional arrays. Unlike standard Python lists, these arrays boast optimized performance, a uniform data type for memory efficiency, and a vast

library of mathematical, statistical, and array manipulation functions. NumPy's foundation is essential for many scientific Python packages, making it a must-learn for anyone delving into scientific computing with Python[38].

- **OpenCV (Open Source Computer Vision Library):** OpenCV is a free and powerful library that provides a comprehensive toolkit for computer vision tasks. With over 2500 algorithms, it empowers developers to tackle a wide range of challenges, from object detection and facial recognition to 3D reconstruction and real-time motion tracking. Its extensive user base, cross-platform compatibility, and support for multiple programming languages make it a versatile choice for developers of all backgrounds. Furthermore, OpenCV's focus on real-time applications and its ability to leverage hardware acceleration like CUDA and OpenCL ensure smooth performance for even the most demanding projects [39].

- **tkinter (tk):** Tkinter is a standard library included with most Python installations. It provides a user-friendly and readily available toolkit for creating graphical user interfaces (GUIs) for desktop applications. Tkinter offers various widgets like buttons, labels, and windows, allowing developers to build interactive applications without needing additional libraries[40].

- **Pillow (PIL):** The Python Imaging Library (PIL), now effectively known as Pillow, is a free and powerful tool for image processing in Python. While PIL itself is no longer actively supported, Pillow forked the project and ensures compatibility with modern Python versions (3.x). It provides user-friendly functionalities for editing, creating, and saving images in various formats like BMP, PNG, JPEG, and TIFF. The library even encourages extending its capabilities by creating decoders for new image formats[41].

- **Weights & Biases (W&B):** An AI development platform that streamlines machine learning project development and optimization. W&B focuses on experiment tracking and visualization, allowing you to monitor training metrics, log results (like object detection data), and visualize performance trends over time. This facilitates insightful analysis and efficient optimization of your machine learning models[42].

- **Matplotlib:** Matplotlib is a graphics library for data visualization package in Python which encompasses as an integral aspect in the python data science stack and it is easily supported with NumPy, Pandas and other relevant libraries[43].

- **Pytorch**: PyTorch is a Python package that provides two high-level features: tensor computation and deep neural networks. it is an open source machine learning and deep learning library developed by Facebook, Inc. It is Python-based,and a very rich library that provides numerous functions that enable building blocks for deep learning that provides for creating tensors and performing data munging operations, linear algebra, and mathematical operations[44]

V.3 Implementation and Results

V.3.1 Model Selection

In our experiments, we selected a diverse set of models to evaluate their performance on the drone detection task, focusing on both state-of-the-art object detection architectures and models specifically tuned for small object detection. The models chosen were YOLOv5, YOLOv5-p2, YOLOv8.

V.3.2 Training parameters

In the process of training our YOLOv5 model for drone detection, we carefully selected our training parameters through extensive experimentation. Initially, we conducted multiple trials with a large number of epochs to understand the model's learning dynamics. Through these trials, we observed the impact of different parameters on the model's performance and chose the best settings documented in Table V.1 to ensure optimal accuracy and generalization.

Table V.1: Training parameters for the best performing model (YOLOv5).

Parameter	**Value**
weights	yolov5s
Batch Size	10
Epochs	300
Input Resolution	1920x1920
Hardware	GPU P100
Software	PyTorch
Patience	10
Hyperparameters	hyp.scratch-med

V.3.3 Results:

After completing the training phase, we evaluated the performance of our YOLOv5 model using various metrics. In this section, we present the results, including the training and validation loss curves, which provide insights into the model's learning process and convergence. Additionally, Figure V.1 illustrates the training progress with curves for recall, precision, and mean Average Precision (mean Average Precision (mAP)) at IoU threshold 0.5 over epochs. The model concluded training at epoch 123, as an early stopping criterion was implemented due to no significant improvement in performance over the last 20 epochs. This decision ensured that our model was trained to its optimal state, balancing performance with computational efficiency.

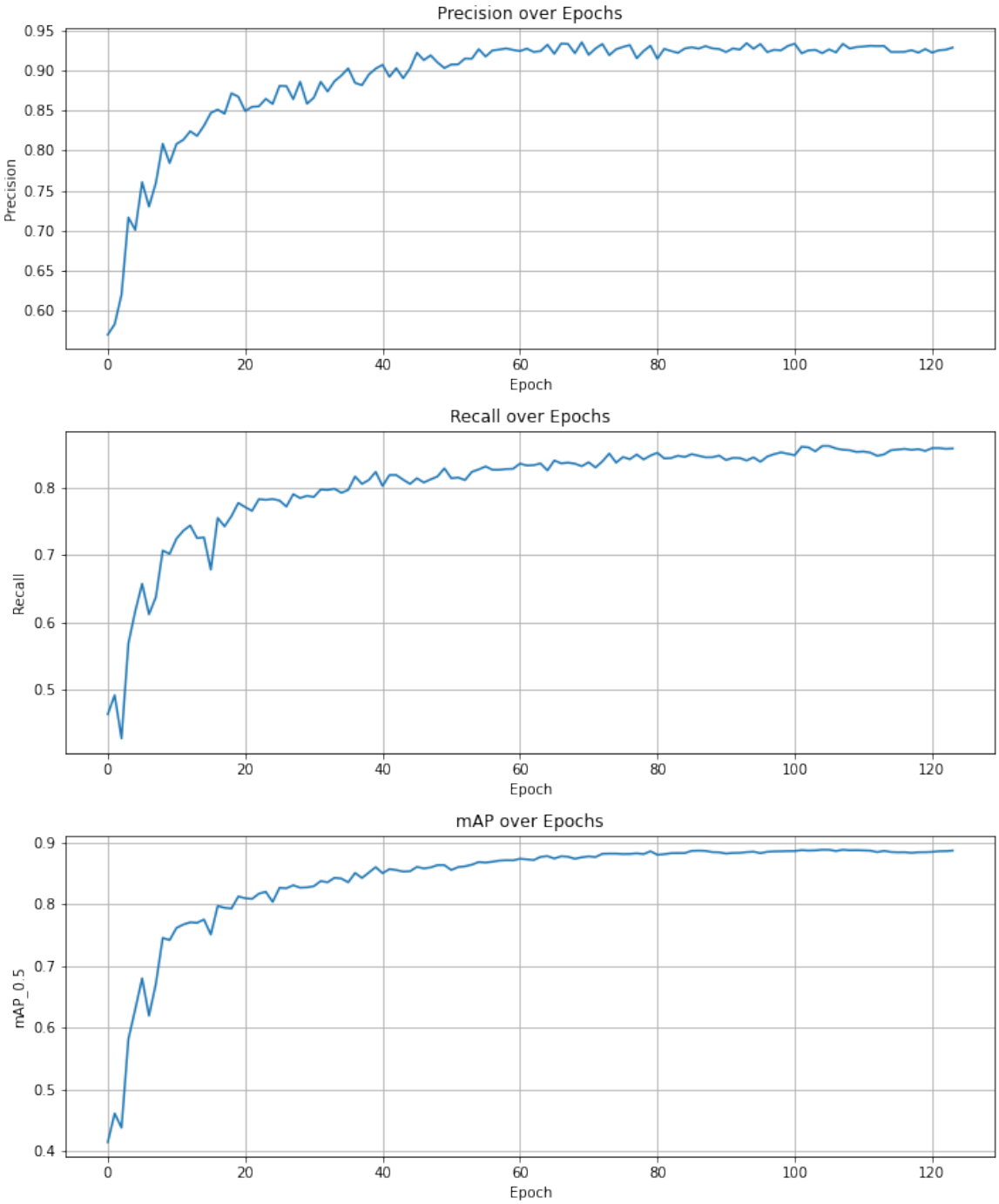


Figure V.1: Precision, recall and map_0.5 over epochs

Following this, we observed that our YOLOv5 model achieved competitive metrics: a mean Average Precision (mAP) of 0.885 at an IoU threshold of 0.5, a recall of 0.857, and a precision of 0.923. These results indicate that our model excels in detecting drones with high accuracy, achieving strong overall performance in object detection tasks. The mAP score reflects the model's ability to accurately localize and classify drones, while high recall underscores its sensitivity in identifying drones across the dataset. Additionally, high precision demonstrates the model's capability to minimize false detections, ensuring reliable and precise drone detection in diverse scenarios.

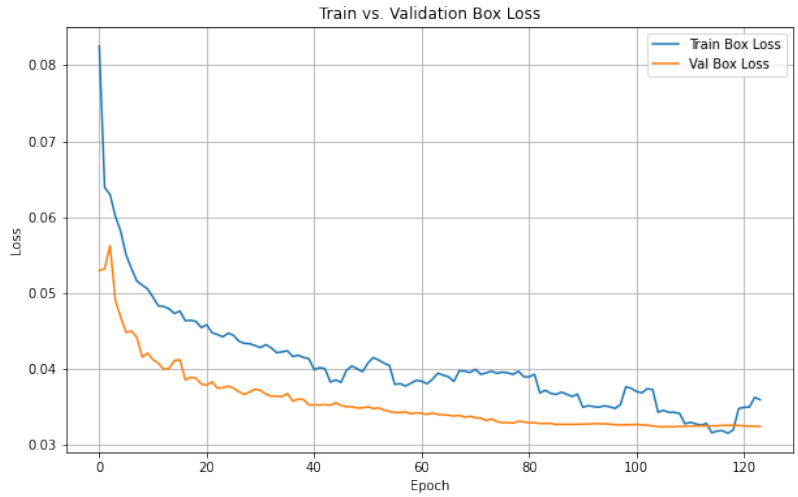


Figure V.2: Train vs validation box loss

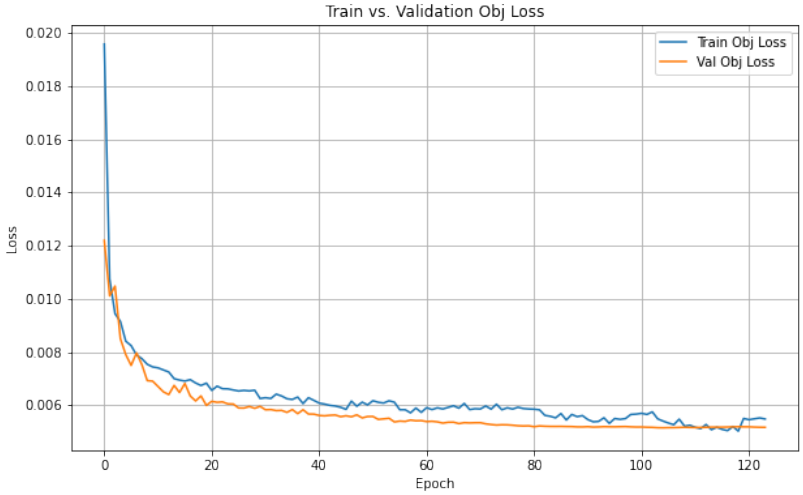


Figure V.3: Train vs validation object loss

Figures V.2 and V.3 illustrate the training versus validation object and box losses, respectively, with final values of 0.005 and 0.0358. These curves illustrate the convergence of our model during training, showing minimal discrepancy between training and validation losses. This consistency indicates that our model generalizes well to unseen data, validating its robustness and effectiveness in real-world drone detection applications.

- **Confusion matrix**

  Next, we explore the model's performance metrics using a confusion matrix to further evaluate its classification accuracy. The confusion matrix, shown below (FigureV.4), summarizes the model's predictions compared to ground truth labels:

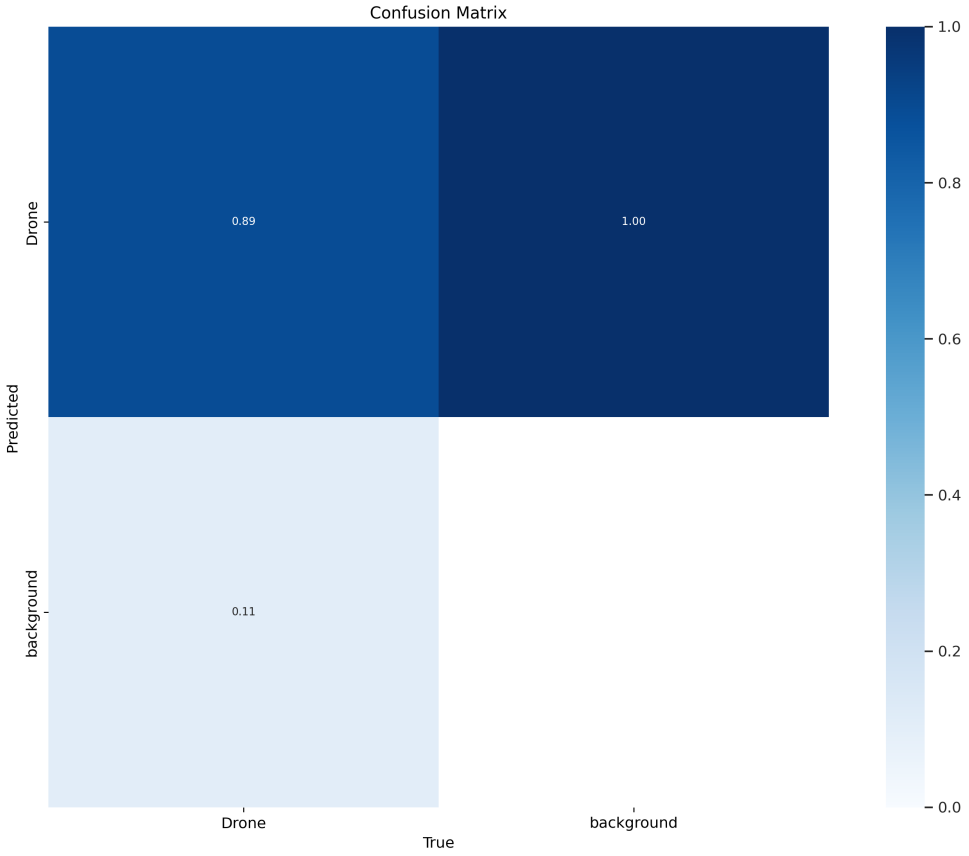


Figure V.4: Confusion matrics

This matrix reveals that 89% of the drones were classified correctly while 11% were misclassified for background, The misclassification can be attributed to the small size of the drones, making them harder to detect accurately.

V.3.4 Model Training and Performance on Global Dataset

The models were trained on a comprehensive global dataset that encompassed a variety of environments and object types. This training phase aimed to ensure the models could generalize well across different scenarios by optimizing parameters for the best performance.

Training and Validation Details

The following tables (Tables V.2,V.8,V.4) provide an overview of the training and validation performance of each model, including the number of iterations (epochs), total training time in hours, training and validation box loss, as well as training and validation object loss.

Table V.2: Training and validation metrics of different models on the global dataset.

Model	**Epochs**	**Total Time (hrs)**	**Train box_loss**	**Val box_loss**	**train obj_loss**	**val obj_loss**
YOLOv5	123	51	0.0355	0.0325	0.0055	0.0052
YOLOV5-p2	150	62	0.0367	0.0316	0.0018	0.0014
YOLOv8	100	25	1.2923	1.4627	1.1874	1.2504

Table V.3: Faster R CNN model training metrics

Model	**Epochs**	**Train box_loss**	**Train obj_loss**	**Train loss_class**	**Train loss_rpn_bbox**
Fasterrcnn_Resnet50_fpn_v2	85	0.0198	0.0012	0.0076	0.0057
Fasterrcnn_efficientnet_b0	40	0.0162	0.0068	0.0118	0.0027
Fasterrcnn_mini_squeezenet1_1_small_head	35	0.912	0.0013	–	–

Table V.4: Detailed training and validation metrics for VGG 16 and MobileNet_v2 models

Model	**Epochs**	**Train Loss**	**Validation Accuracy**	**Validation Loss**
Mobilenet_v2	35	0.0043	0.9145	0.0028
VGG_16	35	0.0180	0.8361	0.0173

Performance on the Global Dataset

Following the training phase, the models were evaluated on the global dataset to assess their performance. The evaluation metrics included precision, recall, mAP@0.5, mAP@0.5:0.95, and frames per second (Frames Per Second (FPS)), providing a comprehensive view of the models' effectiveness and efficiency. The following table(TableV.5) summarizes the performance of each model on the global dataset:

Table V.5: Performance metrics of different models on the global dataset.

Model	**Precision**	**Recall**	**mAP@0.5**	**mAP@0.5:0.95**	**FPS**
YOLOv5	**0.9263**	**0.8571**	**0.888**	**0.507**	**48.39**
YOLOV5-p2	0.926	0.8322	0.873	0.496	37.45
YOLOv5+YOLOV5-p2	0.929	0.854	0.885	0.508	22.47
YOLOv8	0.9313	0.8463	0.8899	0.5155	35.58
sahi+YOLOv5	0.406	0.480	0.829	0.416	1.61

The performance metrics in Table V.5 demonstrate that YOLOv8 achieves the highest mean Average Precision (mAP) at the 0.5:0.95 IoU threshold (0.5155), indicating it is slightly

more accurate in detecting objects in the global dataset. However, the difference in mAP between YOLOv8 and YOLOv5 is minimal, with YOLOv5 achieving 0.507.

The critical factor in choosing YOLOv5 over YOLOv8 is the frames per second (FPS) rate. YOLOv5 achieves a significantly higher FPS rate of 48.39 compared to YOLOv8's 35.58. This higher FPS rate is essential for real-time detection applications, ensuring faster processing and responsiveness. Therefore, despite the marginally lower mAP, YOLOv5 was chosen for its superior real-time performance on the global dataset, making it the optimal choice for our drone detection system.

V.3.5 Test models on Safeshore dataset

To evaluate the performance and robustness of our models, we tested them on the SafeShore dataset, which is a part of our comprehensive global dataset. Each model underwent training using the same dataset but was individually optimized with specific parameters.Tables V.6 and V.7 highlight the performance metrics obtained from this evaluation, demonstrating the effectiveness of each model in detecting drones under various conditions.

Table V.6: Performance metrics of different models on the SafeShore dataset.

Model	**Precision**	**Recall**	**mAP@0.5**	**mAP@0.5:0.95**	**FPS**
YOLOv5	**0.937**	**0.893**	**0.923**	**0.53**	**48.78**
YOLOV5-p2	0.944	0.874	0.911	0.529	37.87
YOLOv5+YOLOV5-p2	0.941	0.893	0.921	0.54	22.32
YOLOv8	0.938	0.868	0.923	0.555	25.18
sahi+YOLOv5	0.431	0.505	0.870	0.442	1.55
Fasterrcnn_Resnet50_fpn_v2	0.272	0.345	0.807	0.380	8.6
Fasterrcnn_efficientnet_b0	0.042	0.159	0.187	0.418	15.8
Fassterrcnn mini squeezenet1_1_small_head	0.197	0.282	0.0.627	0.0.254	56.8

Table V.7: Performance metrics of different models on the SafeShore dataset.

Model	**Accuracy**
Moblinet_v2	0.90
VGG16	0.84

Discussion

The performance metrics in the table indicate that while YOLOv8 achieves the highest mean Average Precision (mAP) at the 0.5:0.95 IoU threshold, suggesting it is the most accurate model for detecting objects in this dataset, the differences among the models are relatively small. YOLOv8 achieves a mAP@0.5:0.95 of 0.555, compared to YOLOv5's 0.53, which is a minor difference in accuracy.

However, when considering frames per second (FPS), which is crucial for real-time detection applications, YOLOv5 excels with an FPS rate of 48.78, significantly higher than YOLOv8's 25.18. This suggests that while YOLOv8 may offer a slight edge in detection accuracy, YOLOv5 is more suitable for scenarios where processing speed is essential.

Additionally, YOLOv5-p2 and the combined YOLOv5+YOLOV5-p2 models also demonstrate competitive performance, with slightly lower mAP scores but varying FPS rates.

In summary, while YOLOv8 leads in detection accuracy, YOLOv5 offers a better balance of accuracy and speed, making it the preferred choice for real-time applications. The choice of model ultimately depends on the specific requirements of accuracy versus processing speed in the given use case.

In other hand, achieving the best possible balance between real-time processing and model performance is our main goal. In order to determine which model would be best for our project, we assessed a number of models and provided a thorough comparison to demonstrate the effectiveness and efficiency of the selected model. This gave us the opportunity to draw attention to the advantages and disadvantages of each model and offer insightful commentary on the choosing procedure.

Since Faster R-CNN is known for its great accuracy and performance even though it doesn't work in real-time, we experimented with it using different backbones. For this reason, we chosed Faster R-CNN with ResNet50-FPN v2 for the reason that produced consistent results, averaging 0.99 accuracy, although it could only handle 8.6 frames per second (FPS), not nearly as fast as real-time processing.Conversely, EfficientNet-B0 yielded reasonable results (15.6 FPS), which offered a higher speed-to-performance ratio but was still less efficient than YOLOv5. Although it had the poorest performance, the Faster R-CNN with MiniSqueezeNet1_1_Small_Head managed to analyze data in real time. It does, however, receive training in the same settings as the other models.whereas even if VGG_16 and Bilenet_v2 are reliable models, their performance metrics are worse to those of other models.

V.3.6 Comparison Results with Models Proposed in Literature

Table V.8 presents a comparison results between our model (YOLOv5) and prior research evaluated on the Safeshore dataset. It is noted that Yolov5 achieved the highest AP score of 53.0, indicating a strong ability to accurately detect small drones. The second work, SqueezeNet, achieved an AP score of 77.0, however, this score represents AP0.5 of the medium area. In contrast, its AP0.5_0.95 is lower compared to our work, highlighting the superior overall performance of YOLOv5.

Table V.8: Comparison results between this work and prior research

Model	AP	Year	Reference
Cascade R-CNN	52.2	2021	[45]
SqueezeNet	77.0	2023	[46]
YOLOv5	53.0	2024	Our work

V.4 The graphical interface

Our drone detection system includes a straightforward desktop application designed to display real-time detection results. This graphical interface is user-friendly As illustrated in Figure V.5, allowing users to see detected drones as they are identified by the system.



Figure V.5: Developed system displaying real-time drone detection results.

V.5 Conclusion

In this chapter we have detailed the development environment and tools essential for our project's success. We provided comprehensive hardware and Software specifications, highlighting the key tools along with essential libraries .We then delve into the effectiveness of different preprocessing techniques and carefully tuned hyperparameters to optimize model performance.

We evaluated the selected model's adaptability across many datasets. Metrics such as accuracy, recall, and mean Average accuracy (mAP) at an IoU threshold of 0.5 are presented together with training and validation loss curves that show the model's learning process and convergence.Our YOLOv5 model achieved a competitive mAP of 0.885, recall of 0.857, and precision of 0.923, indicating high accuracy in drone detection.

A comprehensive analysis of training and validation metrics for several models on the SafeShore dataset as well as a global dataset is also included in this chapter. While YOLOv5 distinguished itself in real-time performance with better frames per second (FPS), making it the recommended option for our application, YOLOv8 earned the greatest mean Average Precision (mAP). Additionally, the chapter highlights the accuracy and processing speeds of several models, such as Faster R-CNN with varied backbones, by analyzing their performance. The last portion presents our PC platform's intuitive graphical user interface, which shows the drone detection system's real-time detection results.

General Conclusion

The advancements in drone detection and classification have witnessed significant progress with the integration of state-of-the-art deep learning techniques. The effectiveness of several state-of-the-art models, such as YOLOv5, Faster R-CNN, was carefully investigated in this work. Our thorough analysis has shown how crucial these models are for precisely identifying small objects—a capacity that is necessary for drone application.

We used a variety of different, well-annotated datasets for our research, including the USC-GRAD-STDdb and the SafeShore dataset. These datasets were essential for training and testing the models and guaranteeing their strong cross-scenario generalization. According to the comparative analysis, YOLOv8 demonstrated greater accuracy in object detection tasks by achieving the greatest mean Average Precision (mAP). On the other hand, because of its greater frames per second (FPS), YOLOv5 proved to be the best option for real-time applications, exhibiting an impressive trade-off between speed and accuracy.

Furthermore, the addition of sophisticated modules to YOLOv5, significantly enhanced the system's capacity to identify small objects. These improvements highlight the model's flexibility and durability in addressing a range of detecting difficulties. We also examined the accuracy and processing speed of Faster R-CNN models with different backbones in our study. This thorough evaluation helped choose the best model to meet particular operational needs by offering insightful information about the decisions between various model designs. To translate these technical advancements into practical applications, we developed a user-friendly graphical interface tailored for real-time drone detection. This interface, designed for seamless deployment on PC platforms, showcases the real-time detection results, providing users with an intuitive and interactive experience. The interface's capability to display live detection results underscores the practical applicability of our system, making it accessible and easy to use for end-users.

To sum up, the integration of powerful deep learning models, extensive and varied datasets, and a user-friendly interface establishes a new standard for dependable and effective drone detection systems. This study establishes the foundation for further research and development in addition to addressing the domain's present problems.The integration of advanced detection techniques and user-centric design principles paves the way for innovative solutions, enhancing the operational capabilities and safety of drone technologies. The approaches and insights suggested in this work will serve as a valuable resource for researchers and practitioners striving to push the boundaries of drone detection and classification.

In the future, we will concentrate on creating an advanced hybrid architecture that combines the accuracy of Faster R-CNN with the real-time processing capabilities of YOLOv5.this innovative architecture seeks to achieve superior performance. We also intend to curate and annotate a large dataset that includes different kinds of drones in order to improve the generalization and classification accuracy of the model. With the use of this dataset, a multi-class drone classifier that can distinguish between various drone types based on their visual and functional characteristics will be developed.Further advancements will involve integrating ad-

vanced sensor technologies, such as infrared and radar systems, to augment visual detection capabilities. In addition, we want to create a desktop application that can detect and classify drones in real time using an easy-to-use graphical interface. This application will allow customers to easily deploy our cutting-edge technology in real-world situations while showcasing live detection results.

Bibliography

[1] Iqbal H Sarker. Machine learning: Algorithms, real-world applications and research directions. *SN computer science*, 2(3):160, 2021.

[2] S Agatonovic-Kustrin and Rosemary Beresford. Basic concepts of artificial neural network (ann) modeling and its application in pharmaceutical research. *Journal of pharmaceutical and biomedical analysis*, 22(5):717–727, 2000.

[3] Md Motiur Rahman Sagar and Martin Dyrba. Learning shape features and abstractions in 3d convolutional neural networks for detecting alzheimer's disease, 09 2020.

[4] Gülseren Seven, Gokhan Silahtaroğlu, Koray Kochan, Ali İnce, Dilek Arici, and Hakan Senturk. Use of artificial intelligence in the prediction of malignant potential of gastric gastrointestinal stromal tumors. *Digestive Diseases and Sciences*, 67:1–9, 01 2022.

[5] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[6] Airsight. Rotorcraft. *Types of Drones*, n.d. Accessed on 2024-02-20.

[7] CFD Flow Engineering. Classification and application of drones. *CFD Flow Engineering*, 2023. Accessed on 2024-02-20.

[8] Naman Jain, Shreesha Yerragolla, Tanuja Guha, and Mohana . Performance analysis of object detection and tracking algorithms for traffic surveillance applications using neural networks. In *Proceedings of the International Conference on Smart Technologies for Mechanical Engineering, Automation and Control*, pages 690–696, 12 2019.

[9] DataCamp. Classification in machine learning: A guide for beginners. *DataCamp Blog*, 2022. Published on September 21, 2022. Accessed on February 21, 2024.

[10] Felipe Viana, Gabriel Araujo, Milena Faria Pinto, Jefferson de Paula, and Diego Haddad. Aerial image instance segmentation through synthetic data using deep learning. *Learning and Nonlinear Models*, 18:35–46, 09 2020.

[11] Hamid R Alsanad, Amin Z Sadik, Osman N Ucan, Muhammad Ilyas, and Oguz Bayat. Yolo-v3 based real-time drone detection algorithm. *Multimedia tools and applications*, 81(18):26185–26198, 2022.

[12] Jiao Yang and Liqi Wang. Feature fusion and enhancement for single shot multibox detector. In *2019 Chinese Automation Congress (CAC)*, pages 2766–2770, 2019.

[13] Zhijun Xue, Wenjie Chen, and Jing Li. Enhancement and fusion of multi-scale feature maps for small object detection. In *2020 39th Chinese Control Conference (CCC)*, pages 7212–7217, 2020.

[14] Xu Yan, Beibei Shen, and Hui Li. Small objects detection method for uavs aerial image based on yolov5s. In *2023 IEEE 6th International Conference on Electronic Information and Communication Technology (ICEICT)*, pages 61–66. IEEE, 2023.

[15] B. Bosquet, M. Mucientes, and V. Brea. STDnet: A ConvNet for small target detection. In *Proceedings of the 29th British Machine Vision Conference*, Newcastle (UK), 2018.

[16] Angelo Coluccia, Gianluca Parisi, and Alessio Fascista. Detection and classification of multirotor drones in radar sensor networks: A review. *Sensors*, 20(15):4172, 2020.

[17] Leila Mohammadpour, Teck Chaw Ling, Chee Sun Liew, and Alihossein Aryanfar. A survey of cnn-based network intrusion detection. *Applied Sciences*, 12(16):8162, 2022.

[18] Mohamad H Hassoun. *Fundamentals of artificial neural networks*. MIT press, 1995.

[19] Kenji Suzuki. *Artificial neural networks: methodological advances and biomedical applications*. BoD–Books on Demand, 2011.

[20] Juan Yepez and Seok-Bum Ko. Stride 2 1-d, 2-d, and 3-d winograd for convolutional neural networks. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 28(4):853–863, 2020.

[21] Manli Sun, Zhanjie Song, Xiaoheng Jiang, Jing Pan, and Yanwei Pang. Learning pooling for convolutional neural network. *Neurocomputing*, 224:96–104, 2017.

[22] Oxford University Press. Drone_1 noun - definition, pictures, pronunciation and usage notes. *Oxford Learner's Dictionaries*, n.d. Accessed on 2024-02-19.

[23] John E Hummel. Object recognition. *Oxford handbook of cognitive psychology*, 810:32–46, 2013.

[24] Drónok fejlődése, csoportosítása, felhasználása és ellenőrzése megtekintése. https://biztonsagtudomanyi.szemle.uni-obuda.hu/index.php/home/article/view/222/202?fbclid=IwAR0LIwC4OiuCmN9XmU2XT7JxmyVqzx54j_gH_zeih1f5SPuJoq6C8sbIlBU. Accessed: February 26, 2024.

[25] Fatemeh Mahdavi and Roozbeh Rajabi. Drone detection using convolutional neural networks. In *2020 6th Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS)*, pages 1–5, 2020.

[26] Muhammad Saqib, Sultan Daud Khan, Nabin Sharma, and Michael Blumenstein. A study on detecting drones using deep convolutional neural networks. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–5, 2017.

[27] Ulzhalgas Seidaliyeva, Daryn Akhmetov, Lyazzat Ilipbayeva, and Eric T. Matson. Real-time and accurate drone detection in a video with a static background. *Sensors*, 20(14), 2020.

[28] Jun-Hwa Kim, Namho Kim, Yong Woon Park, and Chee Sun Won. Object detection and classification based on yolo-v5 with improved maritime dataset. *Journal of Marine Science and Engineering*, 10(3), 2022.

[29] Mariusz Wisniewski, Zeeshan A. Rana, and Ivan Petrunin. Drone model classification using convolutional neural network trained on synthetic data. *Journal of Imaging*, 8(8), 2022.

[30] Angelo Coluccia, Alessio Fascista, Arne Schumann, Lars Sommer, Anastasios Dimou, Dimitrios Zarpalas, Fatih Cagatay Akyon, Ogulcan Eryuksel, Kamil Anil Ozfuttu, Sinan Onur Altinuc, Fardad Dadboud, Vaibhav Patel, Varun Mehta, Miodrag Bolic, and Iraj Mantegh. Drone-vs-bird detection challenge at ieee avss2021. In *2021 17th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–8, 2021.

[31] Artem Rozantsev, Vincent Lepetit, and Pascal Fua. Detecting flying objects using a single moving camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(5):879–892, 2017.

[32] Switchdrive. Switchdrive.

[33] Bharat Mahaur and K.K. Mishra. Small-object detection based on yolov5 in autonomous driving systems. *Pattern Recognition Letters*, 168:115–122, 2023.

[34] Fatih Cagatay Akyon, Cemil Cengiz, Sinan Onur Altinuc, Devrim Cavusoglu, Kadir Sahin, and Ogulcan Eryuksel. SAHI: A lightweight vision library for performing large scale object detection and instance segmentation, November 2021. Software.

[35] Fatih Cagatay Akyon, Sinan Onur Altinuc, and Alptekin Temizel. Slicing aided hyper inference and fine-tuning for small object detection. *2022 IEEE International Conference on Image Processing (ICIP)*, pages 966–970, 2022.

[36] Project Jupyter. Project jupyter.

[37] Python Software Foundation. What is python? executive summary.

[38] NumPy. What is numpy? https://numpy.org/doc/stable/user/whatisnumpy.html, 2024. Accessed: 2024-05-20.

[39] OpenCV. About. https://opencv.org/about/, November 4 2020. Accessed: 2024-05-20.

[40] Python Software Foundation. Tkinter - python interface to tcl/tk. https://docs.python.org/3/library/tkinter.html, 2024. Accessed: 2024-05-20.

[41] *Pillow (PIL Fork)*. Accessed: 2024-05-21.

[42] Weights & Biases. Weights & biases documentation. https://docs.wandb.ai/, 2024. Accessed: 2024-05-20.

[43] Ali Hassan Sial, Syed Yahya Shah Rashdi, and Abdul Hafeez Khan. Comparative analysis of data visualization libraries matplotlib and seaborn in python. *International Journal*, 10(1):45, 2021.

[44] Nikhil Ketkar, Jojo Moolayil, Nikhil Ketkar, and Jojo Moolayil. Introduction to pytorch. *Deep learning with python: learn best practices of deep learning models with PyTorch*, pages 27–91, 2021.

[45] Angelo Coluccia, Alessio Fascista, Arne Schumann, Lars Sommer, Anastasios Dimou, Dimitrios Zarpalas, Miguel Méndez, David de la Iglesia, Iago González, Jean-Philippe Mercier, Guillaume Gagné, Arka Mitra, and Shobha Rajashekar. Drone vs. bird detection: Deep learning algorithms and results from a grand challenge. *Sensors*, 21, 2021.

[46] Hassan J. Al Dawasari, Muhammad Bilal, Muhammad Moinuddin, Kamran Arshad, and Khaled Assaleh. Deepvision: Enhanced drone detection and recognition in visible imagery through deep learning networks. *Sensors*, 23, 2023.