A Project report

On

DEEP LEARNING BASED OBJECT DETECTION FOR AUTONOMOUS DRONES

Submitted in partial fulfillment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

By

U. VARUN	**(204G1A05C0)**
M. UMA	**(204G1A05B5)**
P. SRINATH	**(214G5A0511)**
A. SRINIVASA SREE SHARAN	**(204G1A05A3)**

Under the Guidance of

Mrs. G. Naga Leela M. Tech

Assistant Professor

Department of Computer Science & Engineering

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(AUTONOMOUS)

ROTARYPURAM VILLAGE, B K SAMUDRAM MANDAL, ANANTHAPURAMU - 515701

2023-2024

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(AUTONOMOUS)

(Affiliated to JNTUA, Accredited by NAAC with 'A' Grade, Approved by AICTE, New Delhi & Accredited by NBA (EEE, ECE & CSE)
Rotarypuram Village, BK Samudram Mandal, Ananthapuramu-515701

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Certificate

This is to certify that the Project report entitled **DEEP LEARNING BASED OBJECT DETECTION FOR AUTONOMOUS DRONES** is the bonafide work carried out by **U.Varun, M. Uma, P. Srinath, A. Srinivasa Sree Sharan** bearing Roll Number **204G1A05C0, 204G1A05B5, 214G5A0511, 204G1A05A3** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering** during the academic year 2023 - 2024.

Project Guide

Mrs. G. Naga Leela M. Tech.

Assistant Professor

Head of the Department

Mr. P. Veera Prakash M. Tech, (Ph.D.)

Assistant Professor & HOD

Date:

EXTERNAL EXAMINER

Place: Rotarypuram

DECLARATION

We, Mr. U. Varun with reg no: 204G1A05C0, Ms. M. Uma with reg no: 204G1A05B5, Mr. P. Srinath with reg no: 214G5A0511, Mr. A. Srinivasa Sree Sharan with reg no: 204G1A05A3 students of SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY , Rotarypuram , hereby declare that the dissertation entitled **"DEEP LEARNING BASED OBJECT DETECTION FOR AUTONOMOUS DRONES"** embodies the report of our project work carried out by us during IV year Bachelor of Technology under the guidance of **Mrs. G. Naga Leela** **M. Tech**. Department of CSE, **SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY**, and this work has beensubmitted for the partial fulfillment of the requirements for the award of the Bachelor of Technology degree.

The results embodied in this project have not been submitted to any other University of Institute for the award of any Degree or Diploma.

U. VARUN	Reg no: 204G1A05C0
M. UMA	Reg no: 204G1A05B5
P. SRINATH	Reg no: 214G5A0511
A. SRINIVASA SREE SHARAN	Reg no: 204G1A05A3

VISION & MISSION OF THE INSTITUTION

Vision:

To become a premier Educational Institution in India offering the best teaching and learning environment for our students that will enable them to become complete individuals with professional competency, human touch, ethical values, service motto, and a strong sense of responsibility towards environment and society at large.

Mission:

- Continually enhance the quality of physical infrastructure and human resources to evolve in to a center of excellence in engineering education.

- Provide comprehensive learning experiences that are conducive for the students to acquire professional competences, ethical values, life-long learning abilities and understanding of the technology, environment and society.

- Strengthen industry institute interactions to enable the students work on realistic problems and acquire the ability to face the ever changing requirements of the industry.

- Continually enhance the quality of the relationship between students and faculty which is a key to the development of an exciting and rewarding learning environment in the college.

VISION & MISSION OF THE DEPARTMENT OF CSE

Vision:

To evolve as a leading department by offering best comprehensive teaching and learning practices for students to be self-competent technocrats with professional ethics and social responsibilities.

Mission:

DM 1: Continuous enhancement of the teaching-learning practices to gain profound knowledge in theoretical & practical aspects of computer science applications.

DM 2: Administer training on emerging technologies and motivate the students to inculcate self-learning abilities, ethical values and social consciousness to become competent professionals.

DM 3: Perpetual elevation of Industry-Institute interactions to facilitate the students to work on real-time problems to serve the needs of the society.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that we have now the opportunity to express our gratitude for all of them.

It is with immense pleasure that we would like to express our indebted gratitude to our Guide **Mrs. G. Naga Leela, M. Tech., Assistant Professor, Computer Science & Engineering**, who has guided us a lot and encouraged us in every step of the project work. We thank her for the stimulating guidance, constant encouragement and constructive criticism which have made possible to bring out this project work.

We express our deep-felt gratitude to **Mr. C. Lakshminatha Reddy, Assistant Professor and Mr. M. Narasimhulu, Assistant Professor, Project Coordinators** for their valuable guidance and unstinting encouragement enabled us to accomplish our project successfully in time.

We are very much thankful to **Mr. P. Veera Prakash, M. Tech., (Ph.D.), Assistant Professor & Head of the Department, Computer Science & Engineering,** for his kind support and for providing necessary facilities to carry out the work.

We wish to convey our special thanks to **Dr. G. Bala Krishna, M. Tech., Ph.D., Principal** of **Srinivasa Ramanujan Institute of Technology** for giving the required information in doing our project work. Not to forget, We thank all other faculty and non- teaching staff, and our friends who had directly or indirectly helped and supported us in completing our project in time.

We also express our sincere thanks to the Management for providing excellent facilities**.**

Finally, we wish to convey our gratitude to our families who fostered all the requirements and facilities that we need.

Project Associates

204G1A05C0	VARUN U
204G1A05B5	UMA M
214G5A0511	SRINATH P
204G1A05A3	SRINIVASA SREE SHARAN A

ABSTRACT

Object detection plays a vital role in enabling drones to perceive and interact intelligently with their surroundings. The process involves data collection, selecting appropriate deep learning architectures for accurate detection. In our proposed system, we explore the integration of advanced deep learning techniques into autonomous drones for object detection. The trained models combined with onboard software integration efficiently predict object presence in real-time imagery. The project aims to significantly improve the accuracy and reliability of object recognition, enabling drones to distinguish between different objects and navigate their environment with increased precision. The project's outcomes hold promising possibilities for enhancing drone applications in fields such as surveillance, search and rescue.

Keywords

Object Detection, Autonomous Drones, Computer Vision, Real-time Processing.

CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

DL	Deep Learning
OD	Object Detection
HOG	Histogram Oriented Gradients
CNN	Convolution Neural Network
YOLO	You Only Look Once
SRS	System Requirements Specification
RCNN	Region-based Convolutional Neural Network
UML	Unified Modelling Language
SSD	Single Shot Multibox Detector
GPU	Graphical Processing Unit
UAV	Unmanned Aerial Vehicle
DCNN	Deep Convolutional Neural Networks
OpenCV	Open Source Computer Vision Library

CHAPTER 1
INTRODUCTION

Object detection plays a pivotal role in various applications, ranging from surveillance and security to environmental monitoring and search-and-rescue operations. By harnessing the power of deep learning and the YOLOv3 model, we endeavor to equip drones with the capability to autonomously detect and identify objects in their field of view, enabling swift and informed decision-making during droneoperations. With the increasing integration of drones into various industries, the ability to detect and respond to objects in real- time becomes crucial for enhancing the efficiency effectiveness of drone-based applications.

With the increasing integration of drones into various industries, the ability to detect and respond to objects in real-time becomes crucial for enhancing the efficiency and effectiveness of drone-based applications. The YOLOv3 algorithm, known for its speed and accuracy, is well-suited for deployment on drones, enabling rapid and precise object detection without compromising computational performance.

Throughout the implementation process, we will navigate the complexities of collecting and annotating drone-captured datasets, training the YOLOv3 model, optimizing it for deployment on drone platforms, and integrating the object detection results seamlessly with the drone's control system. The project places a strong emphasis on adaptability to environmental factors, scalability, and real-time processing, ensuring that the deployed system is robust, versatile, and capable of addressing a wide range of scenarios.

As drones continue to revolutionize industries with their versatility and mobility, the integration of advanced deep learning techniques for object detection adds a layer of intelligence that can significantly enhance the capabilities of these unmanned aerial vehicles. This project seeks to contribute to the evolution of drone technology, bringing forth a solution that holds immense potential for applications in security, agriculture, disaster response, and beyond. Through the successful implementation of deep learning-based object detection on drones, we aim to propel the boundaries of what is achievable in autonomous and intelligent drone systems.

A computer vision is a machine learning software library called OpenCV is available for free use. A standard infrastructure for computer vision applications was created with OpenCV in order to speed up the incorporation of artificial intelligence

into products. More than 2500 optimized algorithms are available in the collection, including a wide range of both traditional and cutting-edge computer vision and machine learning techniques. These algorithms can be used to find similar images from an image database, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch together images to produce high-resolution images of entire scenes, and remove red eyes from photographs taken.The project proposes the implementation of object detection using YOLOv3 for its real-time capabilities and accuracy. Customization with specific datasets enables precise detection of user-defined objects. By leveraging YOLOv3's efficiency in resource usage, the project aims to address diverse applications across industries. This initiative aligns with industry trends, offering learning opportunities and potential impact in computer vision technologies.

1.1 Problem Statement

The challenge of deep learning-based object detection for autonomous drones centers on developing lightweight, real-time models to identify objects in diverse environments. Efficient algorithms are needed to process streaming video data onboard, considering the limited computational resources of drones. These systems must be robust to handle varying lighting conditions, weather, and terrain, ensuring reliable performance. Integration with drone control system is crucial for informed navigation and decision-making based on detected objects. Ultimately, this technology enhances drone capabilities for tasks like surveillance, search and rescue, and infrastructure inspection.

1.2 Objectives

To accomplish the project's purpose, the following particular objectives have been established.

i. Develop a deep learning model that can accurately identify and classify various objects in real-time images captured by autonomous drones, enabling safer navigation.

ii. Ensure that the object detection model runs efficiently on the limited computational resources of drones, enabling rapid decision-making and responsive actions in dynamic environments.

1.3 Scope of the Project

The following are the boundaries that have established in the proposed system which defines scope.

i. Design and develop deep learning models tailored for object detection tasks on drones. This includes selecting appropriate architectures such as YOLO.

ii. Gather and curate a diverse dataset of images and videos representing the objects and environments the drones will encounter. Annotate the dataset with bounding boxes or segmentation masks for training the object detection models.

iii. Integrate the object detection system with the overall drone control and navigation system. Develop communication protocols and interfaces for receiving and acting upon detected objects, enabling autonomous decision-making and navigation.

iv. Conduct thorough testing and evaluation of the object detection system in simulated and real-world scenarios.

CHAPTER 2

LITERATURE SURVEY

[1] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam, This paper presents a study on designing efficient Convolutional Neural Networks (CNNs) specifically tailored for mobile vision applications. The authors focus on reducing the computational complexity and memory requirements of CNN models, aiming to make them suitable for deployment on mobile and embedded devices. They propose novel architectural designs and optimizations to achieve significant improvements in speed and efficiency without compromising the accuracy of object detection and image classification tasks. The work is motivated by the increasing demand for real-time vision applications on mobile platforms and provides valuable insights into the development of efficient deep learning models for resource-constrained environments.

[2] Yongshin Kim, In this paper, the author investigates the application of Deep Convolutional Neural Networks (DCNNs) for human detection and activity classification based on micro-Doppler radar signatures. The study focuses on utilizing radar data to detect human presence and classify their activities by extracting micro-Doppler features. DCNNs are trained to learn the representations of these features and perform the tasks of detection and classification. The work demonstrates the effectiveness of deep learning techniques in processing radar signals for applications in remote sensing and surveillance, showing promising results in accurately identifying human activities in various environments.

[3] T. R. Latharani, M. Z. Kurian, M. Y. Chidananda, presents a comprehensive study on different techniques and methods for object recognition in the field of computer vision. The authors explore a range of approaches including appearance-based, template-based, part-based, region-based, and contour-based object recognition techniques. Each method is analyzed in terms of its strengths, weaknesses, and suitability for various applications. The study provides insights into the diverse strategies used to recognize objects in images and videos, offering a comparative analysis of their performance metrics and computational requirements. The work aims to contribute to the understanding of object recognition methods and their practical implications in computer vision applications.

[4] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, Y. F, The paper presents the Single Shot MultiBox Detector (SSD), a novel object detection framework designed for real-time processing in computer vision applications. SSD is a unified, single-shot detection model that simultaneously predicts object bounding boxes and class probabilities. It achieves high accuracy in object detection tasks while maintaining efficiency, making it suitable for real-time applications. The authors describe the architecture of SSD, which combines multiple feature maps of different scales to detect objects at various sizes. Experimental results demonstrate the effectiveness and speed of SSD compared to traditional object detection methods, establishing it as a competitive solution for object detection in computer vision tasks.

[5] C. Papageorgiou, T. Poggio, This paper introduces a trainable system for object detection, aiming to develop a robust and adaptable model for detecting objects in images. The authors propose a method that involves training a detection system using a combination of features and classifiers. The system is designed to generalize across different object classes and adapt to varying environmental conditions, making it versatile and capable of detecting objects in diverse settings. The study includes detailed discussions on feature extraction techniques, such as Haar-like features and cascade classifiers, as well as the training process using positive and negative examples to create an effective object detection model. Experimental results demonstrate the system's effectiveness in detecting objects of interest in images with varying backgrounds, lighting conditions, and object orientations. The paper also discusses the potential applications of the trainable system in areas such as surveillance, object tracking, and image analysis, highlighting its significance in advancing the field of object detection in computer vision.

[6] Q. Zhu, S. Avidan, M. Yeh, K. T. Cheng, This paper presents a fast and efficient approach for human detection in images, employing a cascade of Histogram of Oriented Gradients (HOG) features. The authors propose a method that leverages the discriminative power of HOG features to detect humans with high accuracy while maintaining computational efficiency. The cascade architecture is designed to quickly reject negative samples in the early stages of detection, allowing for rapid processing of large image datasets. The study includes a detailed description of the HOG feature extraction process and the cascade classifier framework. Experimental results demonstrate the effectiveness of the proposed approach in achieving real-time human

detection performance on various image datasets. The paper contributes to the field of computer vision by providing a robust and efficient solution for human detection tasks, with potential applications in surveillance, security, and video analysis.

CHAPTER 3
METHODOLOGY

As you can see, each image are presented in the matrix formats, which are made up of rows and columns. The pixel is an image's fundamental building block. A group of pixels make up an image. These are all little squares. We may build the entire image by arranging them side by side. The smallest amount of information that can be present in an image is a single pixel. Every image has pixels with values ranging from 0 to 255.

Each pixel is composed of Three values are R, G, and B, which are the basic colours red, green, and blue. The combination of these three basic colours will create all these colours here in the image so we conclude that a single pixel has three channels, one channel for each one of the basic colours.

3.1 Deep Learning

The deep learning process begins with a well-defined problem statement, where the nature of the task and the target variable are explicitly outlined. Subsequently, the collection and preparation of a relevant dataset take place, emphasizing diversity and proper labeling. Data preprocessing follows, involving cleaning, scaling, and encoding to render the dataset suitable for model training. The selection of an appropriate deep learning architecture, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs), is a crucial step. Constructing the model involves defining its architecture, layers, and connections, aligning its complexity with the problem's intricacy. Model compilation, where optimization parameters are specified, precedes the actual training phase. During training, the model's weights are adjusted iteratively to minimize the defined loss function.

Basic Terminology:

- Dataset: A set of data examples, which contain features important to solving the problem.
- Features: Important pieces of data that help us understand a problem. These are fed into a Deep Learning algorithm to help it learn.
- Model: The representation (internal model) of a phenomenon that a Deep Learning algorithm has learnt. It learns this from the data it is shown during

training. The model is the output you get after training an algorithm. For example, a decision tree algorithm would be trained and produce a decision tree model.

3.2 Algorithm Used

You Only Look Once

YOLO (You Only Look Once) is a popular object detection algorithm that falls under the category of real-time object detection algorithms. YOLO is known for its speed and accuracy, making it suitable for various applications, including surveillance, autonomous vehicles, and robotics. YOLO version 3, or YOLOv3, is the third iteration of the YOLO series, introducing several improvements over its predecessors.

Utilizing YOLOv3 for object detection with drones involves implementing the YOLOv3 algorithm to analyze images or video frames captured by drones in real-time. The process begins with the collection of a diverse dataset of images or video frames captured by drones, encompassing various scenarios and object types relevant to the intended application. Following dataset collection, the data must be annotated, specifying bounding box coordinates and class labels for each object of interest. Subsequent data preprocessing steps involve resizing images, normalizing pixel values, and applying augmentation techniques to enhance the model's generalization capabilities.

The YOLOv3 model, chosen for its real-time performance and accuracy, is then trained on the annotated dataset. Fine-tuning may be applied to adapt the model to drone-specific scenarios, and training metrics such as loss and accuracy are monitored. Validation on a separate dataset ensures the model generalizes well to unseen data, with adjustments made to hyperparameters or model architecture as necessary.

Once trained, the YOLOv3 model is deployed to the drone's onboard computing system. Real-time inference is then implemented, with the model identifying and localizing objects of interest in the drone's field of view. Integration with the drone's control system facilitates applications such as navigation or surveillance.

CHAPTER 4

SYSTEM REQUIREMENTS SPECIFICATIONS

A software requirements specification (SRS) is a description of a software system to be developed. It lays out functional and nonfunctional requirements, and may include a set of use cases that describe user interactions that the software must provide. It is very important in a SRS to list out the requirements and how to meet them. It helps the team to save upon their time as they are able to comprehend how are going to go about the project. Doing this also enables the team to find out about the limitations and risks early on.

A SRS can also be defined as a detailed description of a software system to be developed with its functional and non-functional requirements. It may include the use cases of how the user is going to interact with the software system. The software requirement specification document is consistent with all necessary requirements required for project development. To develop the software system we should have a clear understanding of Software system. To achieve this we need continuous communication with customers to gather all requirements.

A good SRS defines how the Software System will interact with all internal modules, hardware, and communication with other programs and human user interactions with a wide range of real life scenarios. It is very important that testers must be cleared with every detail specified in this document in order to avoid faults in test cases and its expected results.

Qualities of SRS

- Correct
- Unambiguous
- Complete
- Consistent
- Ranked for importance and/or stability
- Verifiable
- Modifiable
- Traceable

Fig 4.1: Types of Requirements in SRS

Some of the goals an SRS should achieve are to:

- Provide feedback to the customer, ensuring that the IT Company understands the issues the software system should solve and how to address those issues.
- Help to break a problem down into smaller components just by writing down the requirements.
- Speed up the testing and validation processes.
- Facilitate reviews.

4.1 Functional Requirements

A Functional Requirement is a description of the service that the software must offer. It describes a software system or its component. A function is nothing but inputs to the software system, its behaviour, and outputs. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform. In software engineering and systems engineering, a Functional Requirement can range from the high-level abstract statement of the sender's necessity to detailed mathematical functional requirement specifications. Functional software requirements help you to capture the intended behaviour of the system.

Benefits of functional requirements:

- Helps you to check whether the application is providing all the functionalities that were mentioned in the functional requirement of that application

- A functional requirement document helps you to define the functionality of a system or one of its subsystems.
- Functional requirements along with requirement analysis help identify missing requirements. They help clearly define the expected system service and behavior.
- Errors caught in the Functional requirement gathering stage are the cheapest to fix.
- Support user goals, tasks, or activities.

4.2 Basic Requirements

1. Data collection: The dataset can be collected with different objects in this project. It consists of First name, middle name (optional), last name, generated pin number, Roll number, Email address, phone number, age, States and districts.

2. Data Preprocessing: The purpose of preprocessing is to convert raw data into a form that fits deep learning. Structured and clean data allows a data scientist to get more precise results from an applied deep learning model. The technique includes data formatting, cleaning, and sampling. Here, data pre-processing focuses on finding the attributes with null values or invalid values and finding the relationships between various attributes as well. Data Pre-processing also helps in finding out the impact of each parameter on the target parameter. To preprocess our datasets we used EDA methodology. All the invalid and null values were handled by removing that record or giving the default value of that particular attribute based on its importance.

3. Model training: After a data scientist has preprocessed the collected data and split it into train and test can proceed with a model training. This process entails ―feeding‖ the algorithm with training data. An algorithm will process data and output a model that is able to find a target value (attribute) in new data an answer you want to get a predictive analysis. The purpose of model training is to develop a model. We trained our model using the random forest algorithm. On training the model it predicts the yield on giving the other attributes of the dataset as input.

4.Model evaluation and testing: The goal of this step is to develop the simplest model able to formulate a target value fast and well enough. A data scientist can achieve this goal through model tuning. That‘s the optimization of model parameters to achieve an algorithm‘s best performance.

Application Requirements

1. The system should perform real-time object detection to identify and track objects in the drone's field of view.
2. And ground station, as well as secure storage of collected data for analysis and future training.
3. Rigorous testing and validation procedures should be in place to ensure the reliability, accuracy, and safety of the object detection system before deployment.

4.3 Non-Functional Requirements

Non-Functional Requirement (NFR) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system. Failing to meet non-functional requirements can result in systems that fail to satisfy user needs. Non-functional Requirements allows you to impose constraints or restrictions on the design of the system across the various agile backlogs. Example, the site should load in 3 seconds when the number of simultaneous users are > 10000. They specify the criteria that can be used to judge the operation of a system rather than specific behaviour. They may relate to emergent system properties such as reliability, response time and store occupancy. Non-functional requirements arise through the user needs, because of budget constraints, organizational policies, the need for interoperability with other software and hardware systems or because of external factors such as:- Product Requirements, Organizational Requirements, User Requirements, Basic Operational Requirement, etc.

Benefits of Non-Functional Requirements:

- The nonfunctional requirements ensure the software system follows legal and compliance rules.
- They ensure the reliability, availability, and performance of the software system.
- They ensure good user experience and ease of operating the software.
- They help in formulating security policy of the software system.

Requirements

1. The system must achieve high performance with minimal processing time for real-time object detection and tracking.

2. Object detection algorithms should deliver high accuracy rates, minimizing false positives and negatives.
3. The system must demonstrate high reliability and robustness, ensuring consistent operation in diverse environmental conditions.
4. Measures should be in place to protect the privacy of individuals captured in drone footage, adhering to data protection regulations.
5. The system should be adaptable to various environmental conditions, such as different terrains, lighting conditions, and weather patterns.
6. Thorough testing and validation processes should be conducted to ensure the system's performance meets predefined benchmarks and safety standards.
7. The solution should be adaptable to various drone models and configurations, promoting flexibility in deployment.
8. The system should be robust to changes in drone altitude, maintaining reliable object detection across different heights.

4.4 Python Libraries:

Normally, a library is a collection of books or is a room or place where many books are stored to be used later. Similarly, in the programming world, a library is a collection of precompiled codes that can be used later on in a program for some specific well-defined operations. Other than pre-compiled codes, a library may contain documentation, configuration data, message templates, classes, and values, etc.

A Python library is a collection of related modules. It contains bundles of code that can be used repeatedly in different programs. It makes Python Programming simpler and convenient for the programmer. As we don't need to write the same code again and again for different programs. Python libraries play a very vital role in fields of Machine Learning, Data Science, Data Visualization, etc.

Working of Python Library

As is stated above, a Python library is simply a collection of codes or modules of codes that we can use in a program for specific operations. We use libraries so that we don't need to write the code again in our program that is already available. But how it works. Actually, in the MS Windows environment, the library files have a DLL extension (Dynamic Load Libraries). When we link a library with our program and run that program, the linker automatically searches for that library. It extracts the functionalities

of that library and interprets the program accordingly. That's how we use the methods of a library in our program. We will see further, how we bring in the libraries in our Python programs.

Python standard library

The Python Standard Library contains the exact syntax, semantics, and tokens of Python. It contains built-in modules that provide access to basic system functionality like I/O and some other core modules. Most of the Python Libraries are written in the C programming language. The Python standard library consists of more than 200 core modules. All these work together to make Python a high-level programming language. Python Standard Library plays a very important role. Without it, the programmers can't have access to the functionalities of Python. But other than this, there are several other libraries in Python that make a programmer's life easier. Let's have a look at some of the commonly used libraries:

1.Pandas: Pandas are an important library for data scientists. It is an open-source machine learning library that provides flexible high-level data structures and a variety of analysis tools. It eases data analysis, data manipulation, and cleaning of data. Pandas support operations like Sorting, Re-indexing, Iteration, Concatenation, Conversion of data, Visualizations, Aggregations, etc.

2. Numpy: The name "Numpy" stands for "Numerical Python". It is the commonly used library. It is a popular machine learning library that supports large matrices and multi-dimensional data. It consists of in-built mathematical functions for easy computations. Even libraries like TensorFlow use Numpy internally to perform several operations on tensors. Array Interface is one of the key features of this library.

3. Flask: Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries.[2] It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

4. OpenCV: OpenCV is an open-source software library for computer vision and machine learning. The OpenCV full form is Open Source Computer Vision Library. It was created to provide a shared infrastructure for applications for computer vision and

to speed up the use of machine perception in consumer products. OpenCV, as a BSD-licensed software, makes it simple for companies to use and change the code. There are some predefined packages and libraries that make our life simple and OpenCV is one of them.

Use of Libraries in Python Program

As we write large-size programs in Python, we want to maintain the code's modularity. For the easy maintenance of the code, we split the code into different parts and we can use that code later ever we need it. In Python, modules play that part. Instead of using the same code in different programs and making the code complex, we define mostly used functions in modules and we can just simply import them in a program wherever there is a requirement. We don't need to write that code but still, we can use its functionality by importing its module. Multiple interrelated modules are stored in a library. And whenever we need to use a module, we import it from its library. In Python, it's a very simple job to do due to its easy syntax. We just need to use import.

4.5 Hardware Requirements

The hardware requirements include the requirements specification of the physical computer resources for a system to work efficiently. The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. The Hardware Requirements are listed below:

System Processor	:	Intel I7
Hard Disk	:	500 GB
Ram	:	8 GB

1. **Processor:** A processor is an integrated electronic circuit that performs the calculations that run a computer. A processor performs arithmetical, logical, input/output (I/O) and other basic instructions that are passed from an operating system (OS). Most other processes are dependent on the operations of a processor. A minimum 1 GHz processor should be used, although we would recommend S2GHz or more. A processor includes an arithmetical logic and control unit (CU), which measures capability in terms of the following:

- Ability to process instructions at a given time
- Maximum number of bits/instructions
- Relative clock speed

Fig 4.2: Processor

The proposed system requires a 2.4 GHz processor or higher.

2. Ethernet connection (LAN) or a wireless adapter (Wi-Fi): Wi-Fi is a family of radio technologies that is commonly used for the wireless local area networking (WLAN) of devices which is based around the IEEE 802.11 family of standards. Devices that can use Wi-Fi technologies include desktops and laptops, smartphones and tablets, TV‘s and printers, digital audio players, digital cameras, cars and drones. Compatible devices can connect to each other over object detection with the help of drones Using Deep Learning Wi- Fi through a wireless access point as well as to connected Ethernet devices and may use it to access the Internet. Such an access point (or hotspot) has a range of about 20 meters (66 feet) indoors and a greater range outdoors. Hotspot coverage can be as small as a single room with walls that block radio waves, or as large as many square 16ilometers achieved by using multiple overlapping access points.

Fig 4.3: Ethernet Connection

3. Hard Drive: A hard drive is an electro-mechanical data storage device that uses magnetic storage to store and retrieve digital information using one or more rigid

rapidly rotating disks, commonly known as platters, coated with magnetic material. The platters are paired with magnetic heads, usually arranged on a moving actuator arm, which reads and writes data to the platter surfaces. Data is accessed in a random-access manner, meaning that individual blocks of data can be stored or retrieved in any order and not only sequentially. HDDs are a type often on volatile storage, retaining stored data even when powered off. 32 GB or higher is recommended for the proposed system.

Fig 4.4: Hard Disk

4. Memory (RAM): Random-access memory (RAM) is a form of computer data storage that stores data and machine code currently being used. A random- access memory device allows data items to be read or written in almost the same amount of time irrespective of the physical location of data inside the memory. In today's technology, random-access memory takes the form of integrated chips. RAM is normally associated with volatile types of memory (such as DRAM modules), where stored information is lost if power is removed, although non- volatile RAM has also been developed. A minimum of 4 GB RAM is recommended for the proposed system.

Fig 4.5: RAM

5. ESP32 Camera: The ESP32 camera module is a versatile tool for creating projects that involve visual data. Featuring the OV2640 camera sensor, it offers a resolution of up to 2 megapixels for capturing clear images. The module supports various modes, including single capture, burst capture, and time-lapse capture, allowing for flexibility in capturing still images. Additionally, it provides the ability to stream video in formats such as MJPEG, making it suitable for applications requiring real-time video transmission.With its integration into the ESP32 microcontroller, the camera module benefits from the powerful processing capabilities of the ESP32, enabling on-board image processing and analysis. This allows for tasks such as face detection, object tracking, and image filtering directly on the ESP32 without the need for external processing units. Developers can take advantage of the ESP32's dual-core architecture and ample memory to implement complex algorithms for image recognition and machine vision applications.

Fig 4.6: ESP32 Camera

6. Drone: Drones, also known as Unmanned Aerial Vehicles (UAVs), are pivotal components in modern object detection systems driven by deep learning technologies. These agile and versatile aerial platforms are equipped with sophisticated cameras and sensors, enabling them to capture high-resolution images and videos of vast landscapes and remote areas. In the domain of surveillance and monitoring, drones excel in swiftly detecting and identifying objects of interest, ranging from vehicles and individuals to potential hazards or anomalies in the environment. Their role extends to emergency response scenarios, where drones aid in rapid search and rescue operations by autonomously scanning disaster-stricken areas and pinpointing survivors or hazards. Moreover, in applications such as environmental monitoring,

precision agriculture, and infrastructure inspection, drones equipped with object detection capabilities contribute significantly to data-driven decision-making and operational efficiency. Harnessing the synergy of drones and deep learning algorithms, organizations can unlock unprecedented insights, efficiency, and safety in various domains.

Fig 4.7 Drone

4.6 Software Requirements

The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product. The requirements can be obvious or hidden, known or unknown, expected or unexpected from client‘s point of view.

Operating system	:	Windows OS 8.1
Coding Language	:	Python
IDE	:	VS code and Arduino IDE
GUI	:	IDLE Prompt

1. VS Code: VS Code is the most popular IDE for Python, and includes great features such as excellent code completion and inspection with advanced debugger and support for web programming and various frameworks. The intelligent code editor provided by VS Code enables programmers to write high quality Python code. The editor enables programmers to read code easily through colour schemes, insert indents on new lines automatically, pick the appropriate coding style, and avail context-aware code completion suggestions.

At the same time, the programmers can also use the editor to expand a code block to an expression or logical block, avail code snippets, format the code base, identify

errors and misspellings, detect duplicate code, and auto-generate code. PyCharm offers some of the best features to its users and developers in the following aspects

- Code completion and inspection
- Advanced debugging
- Support for web programming and frameworks such as Django and Flask

2. Python: It is an object-oriented, high-level programming language with integrated dynamic semantics primarily for web and app development. It is extremely attractive in the field of Rapid Application Development because it offers dynamic typing and dynamic binding options. Python is relatively simple, so it's easy to learn since it requires a unique syntax that focuses on readability. Developers can read and translate Python code much easier than other languages. In turn, this reduces the cost of program maintenance and development because it allows teams to work collaboratively without significant language and experience barriers. Additionally, Python supports the use of modules and a package, which means that programs can be designed in a modular style and code can be reused across a variety of projects.

Fig 4.8: Python Icon

3. Arduino IDE: The Arduino Integrated Development Environment (IDE) is a crucial tool for anyone working with Arduino boards and microcontrollers. Its importance lies in its user-friendly interface, extensive library support, and simplified workflow for writing, compiling, and uploading code to Arduino boards.One of the key aspects of the Arduino IDE is its simplicity, making it accessible to beginners in the world of electronics and programming. With a clear and intuitive interface, users can quickly start writing and testing code without getting bogged down by the complexities of traditional programming environments.Moreover, the Arduino IDE offers a vast collection of libraries, which are pre-written code snippets that simplify the process of

interfacing with sensors, actuators, displays, and other components. This library support greatly reduces the amount of code a user needs to write from scratch, enabling faster prototyping and development of projects. The IDE also includes a built-in serial monitor, a valuable tool for debugging and interacting with the Arduino board. This feature allows users to send and receive data between the board and the computer, making it easier to troubleshoot code and observe sensor readings in real-time.

Fig 4.9: Arduino IDE

CHAPTER 5

SYSTEM ANALYSIS AND DESIGN

Systems development is a systematic process which includes phases such as planning, analysis, design, deployment, and maintenance. System Analysis is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components. System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose. Analysis specifies what the system should do.

System Design is a process of planning a new business system or replacing an existing system by defining its components or modules to satisfy the specific requirements. Before planning, you need to understand the old system thoroughly and determine how computers can best be used in order to operate efficiently. System Design focuses on how to accomplish the objective of the system.

5.1 UML DIAGRAMS:

UML represents Unified Modelling Language. UML is an institutionalized universally useful showing dialect in the subject of article situated programming designing. The fashionable is overseen, and become made by way of, the Object management Group.

The goal is for UML to become a regular dialect for making fashions of item arranged PC programming. In its gift frame UML is contained two noteworthy components: a Meta-show and documentation. Later on, a few type of method or system can also likewise be brought to; or related with, UML. The Unified Modeling Language is a popular dialect for indicating, Visualization, Constructing and archiving the curios of programming framework, and for business demonstrating and different non-programming frameworks. The UML speaks to an accumulation of first-rate building practices which have verified fruitful in the showing of full-size and complicated frameworks. The UML is a essential piece of creating gadgets located programming and the product development method. The UML makes use of commonly graphical documentations to specific the plan of programming ventures.

GOALS:

The Primary goals inside the plan of the UML are as in step with the subsequent:

1. Provide clients a prepared to utilize, expressive visual showing Language on the

way to create and change massive models.

2. Provide extendibility and specialization units to make bigger the middle ideas.

3. be free of specific programming dialects and advancement manner.

4. Provide a proper cause for understanding the displaying dialect.

5. Encourage the improvement of OO gadgets exhibit.

6. Support large amount advancement thoughts, for example, joint efforts, systems, examples and components.

7. Integrate widespread procedures.

USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

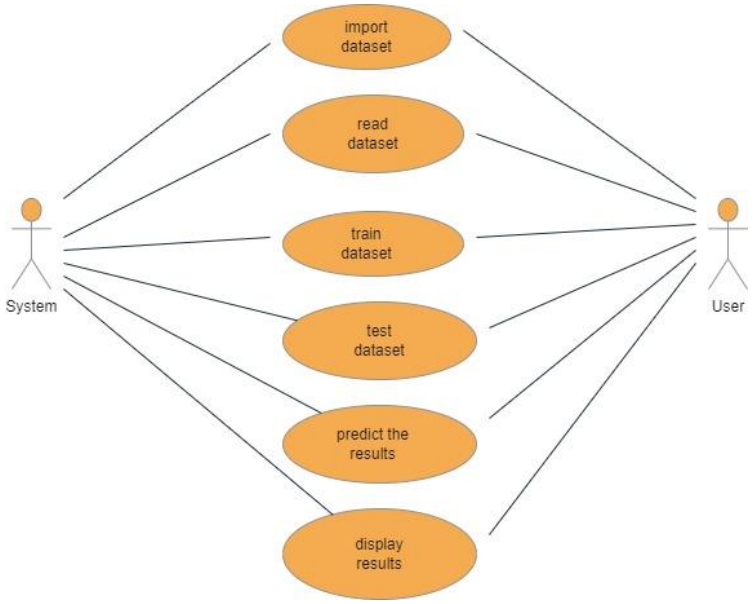

Fig 5.1: Usecase Diagram

CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the

relationships among the classes. It explains which class contains information.

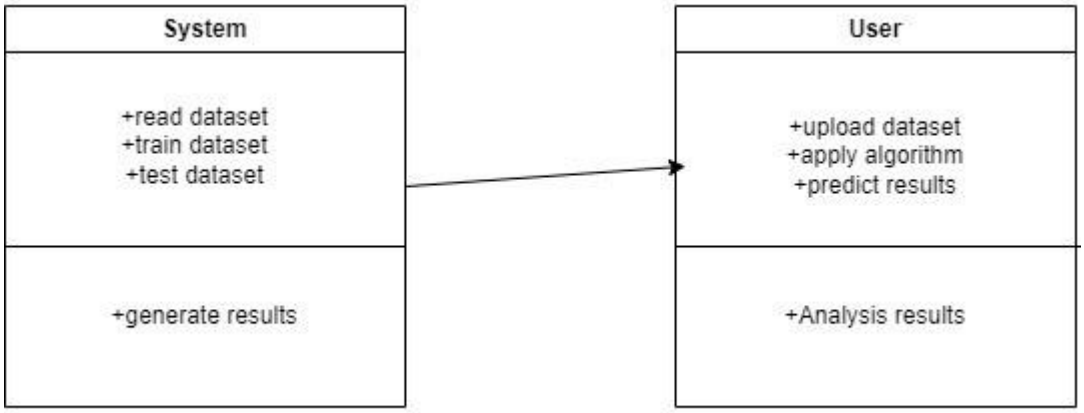

Fig 5.2: Class Diagram

SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

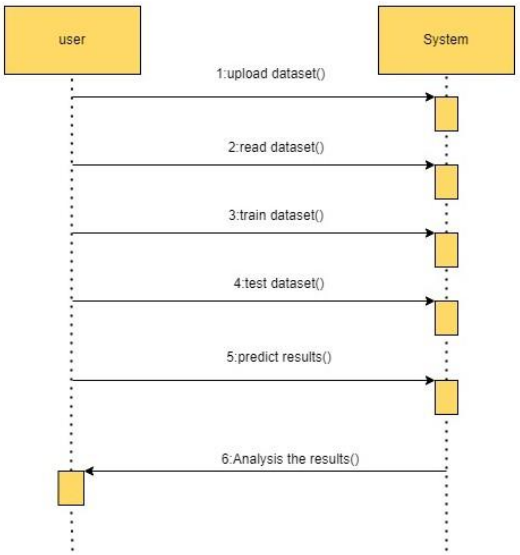

Fig 5.3: Sequence Diagram

COLLABORATION DIAGRAM:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to

describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization where as the collaboration diagram shows the object organization.

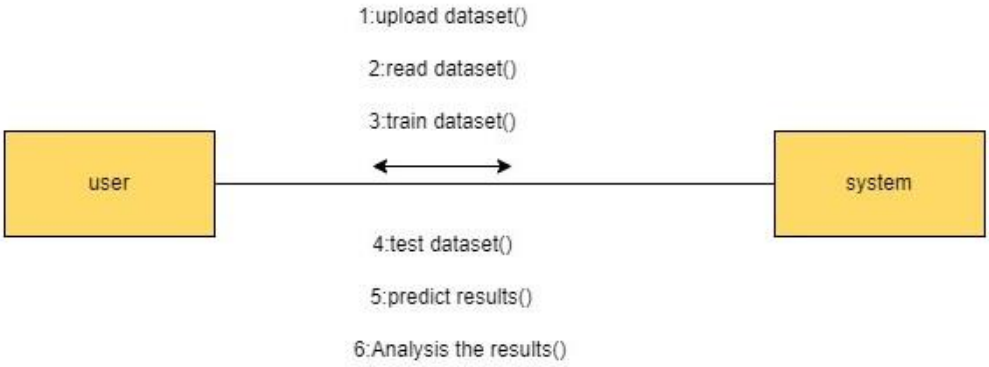

Fig 5.4: Collaboration Diagram

ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

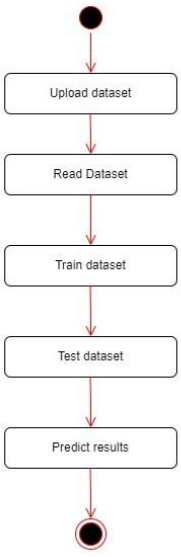

Fig 5.5: Activity Diagram

5.1.1 Usage of UML in Project

As the strategic value of software increases for many companies, the industry looks for techniques to automate the production of software and to improve quality

and reduce cost and time to the market. These techniques include component technology, visual programming, patterns and frameworks. Additionally, the development for the World Wide Web, while making somethings simpler, has exacerbated these architectural problems. The UML was designed to respond to these needs. Simply, systems design refers to the process of defining the architecture, components, modules, interfaces and data for a system to satisfy specified requirements which can bed one easily through UML diagrams.

5.2 Existing System

This model emphasizes an existing method that which was used for survey using the some of the algorithms of deep learning. Here the process is implemented accurately with the algorithms that were used and predicted the accurate detection.

5.3 Disadvantages

- High Computational Demands
- Large Storage Requirements
- Complexity of Implementation

5.4 Proposed System

The proposed system for deep learning-based object detection in autonomous drones harnesses the power of YOLOv3 (You Only Look Once version 3) algorithm, renowned for its real-time detection capabilities and high accuracy. Utilizing pre-trained weights, the system processes live video streams from onboard cameras, enabling swift and precise identification of multiple objects concurrently. Through the efficient YOLOv3 model, objects are annotated with bounding boxes and class labels in real-time, facilitating robust navigation and object avoidance for autonomous drone operations. This system not only ensures efficient real-time processing for dynamic environments but also provides adaptability to custom datasets for specific object detection tasks, promising enhanced situational awareness and operational autonomy for autonomous drone systems.

5.5 Advantages

- Accurate detection
- Robustness to Variations
- Real-Time Processing

5.6 System Architecture

Architecture diagrams can help system designers and developers visualize the high-level, overall structure of their system or application for the purpose of ensuring the system meets their users' needs. They can also be used to describe patterns that are used throughout the design. It's somewhat like a blueprint that can be used as a guide for the convenience of discussing, improving, and following among a team.

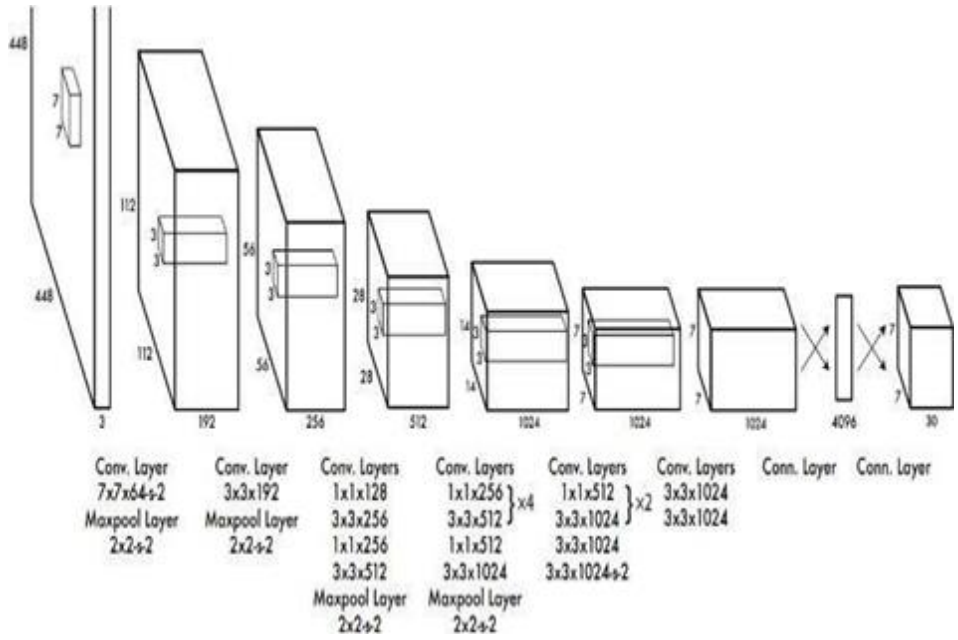

Fig 5.6: System Architecture

5.7 Flowchart

A flowchart is simply a graphical representation of steps. It shows steps in sequential order and is widely used in presenting the flow of algorithms, workflow or processes. Typically, a flowchart shows the steps as boxes of various kinds, and their order by connecting them with arrows. It originated from computer science as a tool for representing algorithms and programming logic but had extended to use in all other kinds of processes. Nowadays, flowcharts play an extremely important role in displaying information and assisting reasoning. They help us visualize complex processes, or make explicit the structure of problems and tasks. A flowchart can also be used to define a process or project to be implemented.

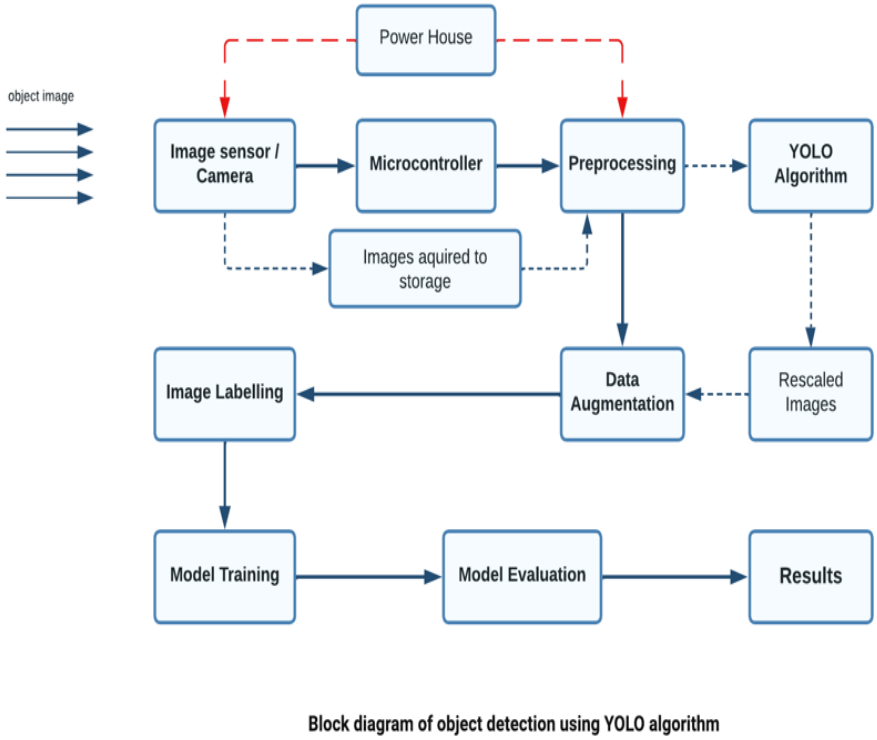

Fig 5.7: Flowchart of the system

CHAPTER 6
IMPLEMENTATION

Implementing a deep learning-based object detection project using drones involves several key steps. Here is a comprehensive overview of the entire process:

1. Problem Definition and Dataset Collection:

Clearly define the object detection problem for your drone application. Collect a dataset of images or video frames captured by drones, ensuring it covers a variety of scenarios and objects relevant to your use case.

2. Data Annotation:

Annotate the dataset by labeling objects of interest in the images. Specify bounding box coordinates and class labels for each annotated object. Tools like LabelImg or RectLabel can assist in this process.

3. Data Preprocessing:

Preprocess the annotated data to make it suitable for training. Resize images to a consistent size, normalize pixel values, and apply data augmentation techniques such as rotation or flipping to increase dataset diversity.

4. Model Selection:

Choose YOLOv3 as the object detection model for your implementation. YOLOv3 is known for its real-time processing capabilities and high accuracy in detecting objects.

5. Model Configuration:

Configure the YOLOv3 model for your specific dataset. Adjust parameters such as the number of classes, anchor boxes, and input dimensions according to the characteristics of your annotated data.

6. Training:

Train the YOLOv3 model on the annotated dataset. Use a machine with GPU support for faster training. Monitor training metrics, and perform validation to ensure the model generalizes well to unseen data.

7. Model Evaluation:

Evaluate the trained YOLOv3 model on a separate test dataset to assess its performance. Analyze metrics like precision, recall, and F1 score. Make adjustments to the model if needed.

8. Model Optimization:

Optimize the YOLOv3 model for deployment on a drone platform. Consider model quantization or other techniques to reduce its size while maintaining performance.

9. Deployment to Drone:

Deploy the trained and optimized YOLOv3 model to the drone's onboard computing system. Ensure compatibility with the drone's hardware and software environment.

10. Real-time Inference:

Implement real-time inference on the drone by processing live video frames through the deployed YOLOv3 model. The model should identify and localize objects in the drone's field of view.

11. Integration with Drone Control:

Integrate the object detection results with the drone's control system. This integration enables the drone to respond to detected objects, such as adjusting its flight path or sending alerts.

12. Testing and Validation:

Conduct thorough testing and validation of the entire system. Test the drone in various scenarios to ensure the YOLOv3 model performs reliably and accurately.

Working Flow of the System

Testing YOLOv3 involves evaluating its performance on a separate dataset, often referred to as the test set, to assess its accuracy, precision, recall, and other relevant metrics. Here is a step-by-step testing process for YOLOv3:

1. Test Dataset Preparation:

Prepare a dedicated test dataset separate from the training dataset. The test set should

include images or video frames that the model has not seen during training. Ensure that the test set is representative of the real-world scenarios the YOLOv3 model will encounter.

2. Model Loading:

Load the pre-trained YOLOv3 model onto the testing environment. You can use the weights obtained after training on the training dataset.

3. Inference on Test Dataset:

Run the YOLOv3 model on the test dataset to perform inference. This involves passing each image through the network to obtain predictions, including bounding box coordinates, class probabilities, and confidence scores.

4. Post-processing:

Apply post-processing steps to filter and refine the model's predictions. Common post-processing steps for YOLOv3 include non-maximum suppression (NMS) to eliminate redundant bounding boxes and thresholding based on confidence scores to filter out low-confidence detections.

5. Visual Inspection:

Visualize the model's predictions on the test dataset. This can involve overlaying bounding boxes on the images or generating visualizations that show the model's detections. Visual inspection helps identify areas of success and potential improvement.

6. Threshold Tuning:

Experiment with different confidence thresholds to find the optimal balance between precision and recall. Adjusting the confidence threshold can impact the number of detected objects and the model's overall performance.

7. Performance Monitoring:

Continuously monitor the model's performance on the test dataset. This may involve periodic retesting with new data or as the model is deployed in different environments.

6.1 Datasets

Deep Learning depends heavily on data. It‘s the most crucial aspect that makes algorithm training possible. It uses historical data and information to gain experiences. The better the collection of the dataset, the better will be the accuracy.

At the time of object detection ESP32 camera will capture the images through webcam and that is going to detect object in the images. This dataset consists of images containing markers of various sizes, orientations, and distances. It provides a diverse set of scenarios for marker detection and localization tasks. This dataset is used to train the model.

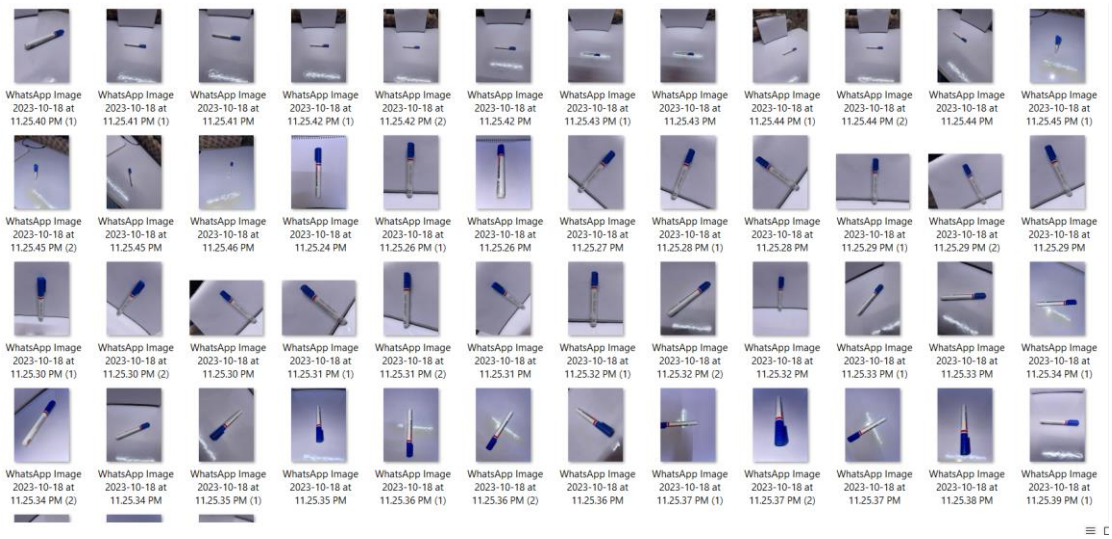

Fig 6.1: Dataset Collection

6.2 Data Pre-Processing

Data Pre-Processing is a Data Mining method that entails converting raw data into a format that can be understood. Real-world data is frequently inadequate, inconsistent, and/or lacking in specific activities or trends, as well as including numerous inaccuracies. This might result in low-quality data collection and, as a result, low-quality models based on that data. Preprocessing data is a method of resolving such problems. Machines do not comprehend free text, image, or video data; instead, they comprehend 1s and 0s. So putting on a slideshow of all our photographs and expecting our machine learning model to learn from it is probably not going to be adequate. Data Pre-processing is the step in any Deep Learning process in which the data is changed, or encoded, to make it easier for the machine to parse it. In other words, the algorithm can now easily interpret the data’s features. Data Pre-processing can be done in four

different ways. Data cleaning/cleaning, data integration, data transformation, and data reduction are the four categories.

6.2.1 Data Cleaning

Data in the real world is frequently incomplete, noisy, and inconsistent. Many bits of the data may be irrelevant or missing. Data cleaning is carried out to handle this aspect. Data cleaning methods aim to fill in missing values, smooth out noise while identifying outliers, and fix data discrepancies. Unclean data can confuse data and the model. Therefore, running the data through various Data Cleaning/Cleansing methods is an important Data Pre-processing step.

6.2.2 Data Integration

It is involved in a data analysis task that combines data from multiple sources into a coherent data store. These sources may include multiple databases. Do you think how data can be matched up?? For a data analyst in one database, he finds Customer ID and in another he finds cust_id, How can he sure about them and say these two belong to the same entity.

6.2.3 Data Reduction

Because data mining is a methodology for dealing with large amounts of data. When dealing with large amounts of data, analysis becomes more difficult. We employ a data reduction technique to get rid of this. Its goal is to improve storage efficiency while lowering data storage and analysis expenses.

Dimensionality Reduction:

A huge number of features may be found in most real-world datasets. Consider an image processing problem: there could be hundreds of features, also known as dimensions, to deal with. As the name suggests, dimensionality reduction seeks to minimize the number of features but not just by selecting a sample of features from the feature set, which is something else entirely Feature Subset Selection or feature selection.

CHAPTER 7
TESTING

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:

- Meets the requirements that guided its design and development,
- Responds correctly to all kinds of inputs,
- Performs its functions within an acceptable time,
- It is sufficiently usable,
- Can be installed and run in its intended environments, and
- Achieves the general result its stakeholder‘s desire.

7.1 Functionality Testing

- Verify accuracy of object detection algorithms by comparing detected objects with ground truth data.
- Test real-time object detection and updates as the drone navigates its environment.
- Validate integration of object detection data with drone's navigation system for precise path planning.
- Ensure system adapts to varying lighting, weather, and environmental conditions during object detection.
- Verify integration with drone APIs for controlling movements based on detection

7.2 Usability Testing

- Verify smooth navigation and intuitive user experience in the application interface.
- Validate accurate handling of user inputs such as mission parameters or target locations.
- Test effective error handling with clear and informative messages for incorrect inputs.
- Verify the reliability of information displayed, ensuring it is sourced from trusted sources.
- Test application responsiveness and performance under varying load conditions.

7.3 Interface Testing

- Verify the user interface navigation flow between YOLOv3 model setup, input configuration, and result display screens.
- Test the input validation for image upload functionality, ensuring correct file formats and sizes are accepted.
- Check the responsiveness of the interface on different devices to ensure YOLOv3 object detection results are displayed optimally.
- Validate the consistency of design elements such as buttons, labels, and image displays across the YOLOv3 model setup interface.
- Ensure user interaction elements like sliders for confidence threshold adjustment and checkboxes for object classes selection function accurately for YOLOv3 object detection settings.

7.4 Performance Testing

- Measure the frames per second (FPS) processed by YOLOv3 to ensure real-time object detection capability.
- Evaluate the precision and recall of YOLOv3 by comparing detected objects with ground truth annotations.
- Test the YOLOv3 model's loading time to ensure quick initialization before detection.

- Assess YOLOv3's ability to detect multiple objects within a single frame for complex scenes

7.5 Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform Object Detection Using Deep Learning basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

7.6 Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

7.7 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

7.8 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

CHAPTER 8

RESULTS

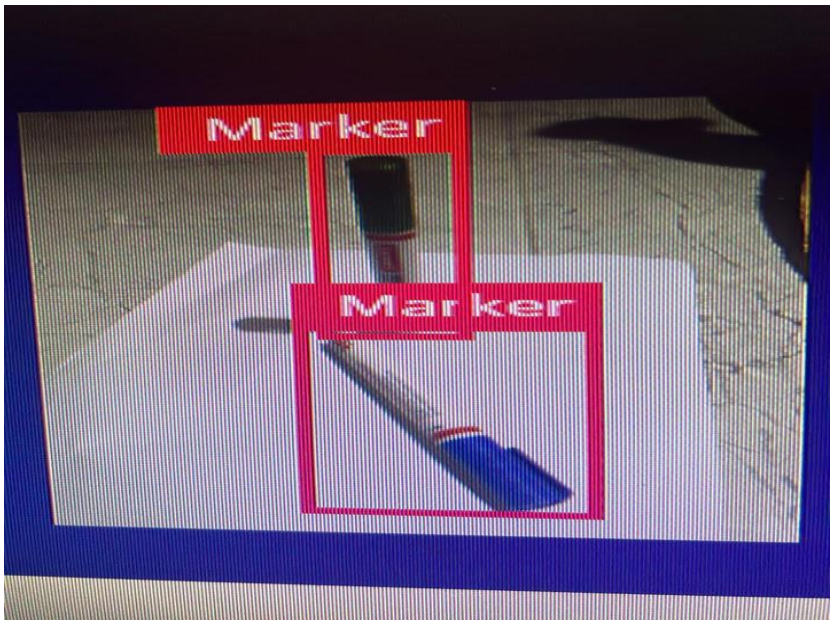

Fig 8.1: Predicted output-Marker

The above image is the output predicted by the system. First image is captured by the camera and output (Marker) is predicted by the system.

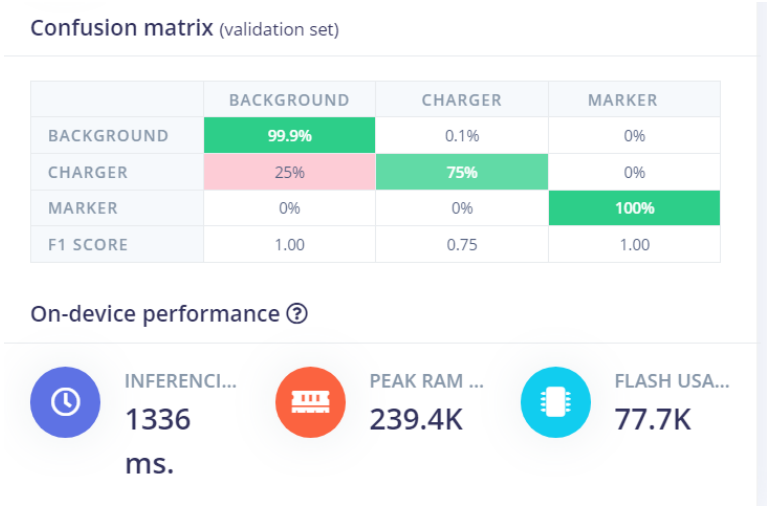

Fig 8.2: Confusion Matrix

The above image is the results generated after testing the trained data.Confusion matrix serves as a tool to evaluate the performance of the object detection model. This matrix

provides a summary of the model's predictions compared to the ground truth labels of the objects in the dataset.

```
0 0.421962 0.586957 0.713489 0.404682
0 0.541249 0.551839 0.831661 0.264214
```

Fig 8.3: Output of trained data

The above image indicates class label and bounding box coordinates of object detection model after testing. Each line includes the class label and normalized bounding box coordinates (x_min, y_min, x_max, y_max) for detected objects. This format facilitates the training of object detection models, aiding in accurate localization and classification of objects within images.

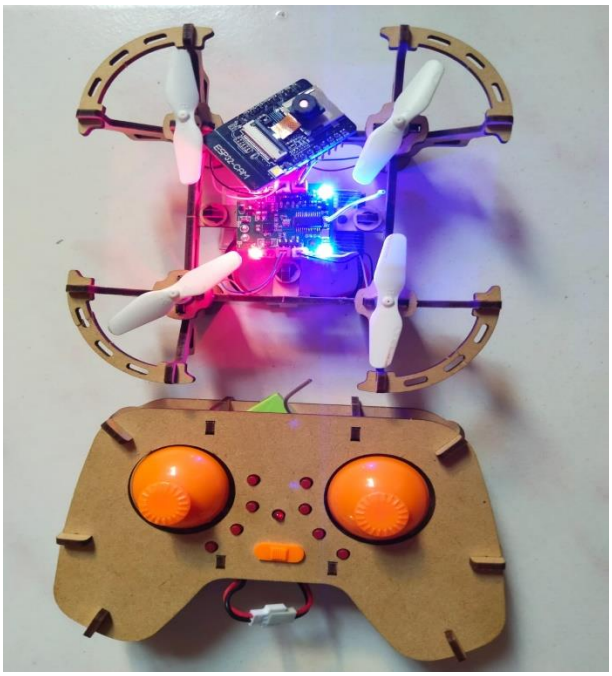

Fig 8.4: Drone Implementation

This hardware setup showing a drone equipped with an ESP32 camera module, illustrating the integration of real-time object detection capabilities into autonomous flight operations.

CONCLUSION AND FUTURE WORK

In conclusion, the integration of the YOLOv3 algorithm into drone operations for real-time object detection has proven to be a transformative advancement. This project successfully implemented a robust system capable of accurately identifying and classifying objects in dynamic environments. The adaptability to environmental factors, scalability, and integration with drone control systems highlight the project's potential across various industries. This achievement sets the stage for ongoing innovation at the intersection of deep learning and autonomous drone technology, paving the way for enhanced capabilities and real-world applications.

Optimizing the YOLOv3 algorithm for real-time performance, possibly through model quantization and hardware acceleration, is crucial. Enhancing object tracking algorithms like SORT or Deep SORT, and training on diverse datasets for better generalization are key steps. Integration of multi-sensor fusion techniques, semantic segmentation, and edge computing can improve scene understanding. Developing user-friendly interfaces for operators and exploring applications in agriculture, infrastructure, and search operations are promising avenues for expansion.

REFERENCES

[1]Howard, A., G., Zhu,M., Chen, B., Kalenichenko., D., Wang, W.,Weyand, T., Andreetto, M. & Adam,H. "Efficient Convolutional Neural Networks for mobile vision applications," Computer Vision and Pattern Recognition, arXiv:1704.04861, 2017.

[2]Kim, Y. "Human detection and activity classification based on microdoppler signatures using Deep Convolutional Neural Networks," IEEE Geoscience and Remote Sensing Letters, 13(1), 2012.

[3]Latharani, T. R., Kurian, M. Z., & Chidananda, M.M. Y. "Various object recognition techniques for computer vision," Journal of Analysis and Computation, 7(1), pp. 39-47, 2011.

[4]Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S. & Fu, C., Y. "SSD: Single Shot MultiBox Detector," In European Conference on Computer Vision, pp. 21-37, 2012.

[5]Papageorgiou, C. & Poggio, T. "A trainable system for object detection," International Journal of Computer Vision, 38(1), pp. 15-33, 2000

[6]Zhu, Q., Avidan, S., Yeh, M., C. & Cheng, K., T. "Fast human detection using a cascade of Histogram of Oriented Gradients," IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006.

Deep Learning Based Object tracking and Detection for Autonomous Drones using YOLOv3

G. Naga Leela
Assistant Professor
Dept of CSE
Srinivasa Ramanujan Institute of Technology
Anantapur
Nagaleelag.cse@srit.ac.in

U. Varun
UG Scholar
Dept of CSE
Srinivasa Ramanujan Institute of Technology
Anantapur
varunu5c0@gmail.com

M. Uma
UG Scholar
Dept of CSE
Srinivasa Ramanujan Institute of Technology
Anantapur
204g1a05b5@srit.ac.in

P. Srinath
UG Scholar
Dept of CSE
Srinivasa Ramanujan Institute of Technology
Anantapur
214g5a0511@srit.ac.in

A. Srinivasa Sree Sharan
UG Scholar
Dept of CSE
Srinivasa Ramanujan Institute of Technology
Anantapur
204g1a05a3@srit.ac.in

Abstract-Object detection plays a vital role in enabling drones to perceive and interact intelligently with their surroundings. The process involves data collection, selecting appropriate deep learning architectures for accurate detection. In our proposed system, we explore the integration of advanced deep learning techniques into autonomous drones for object detection. The trained models combined with onboard software integration efficiently predict object presence in real-time imagery. The project aims to significantly improve the accuracy andreliability of object recognition, enabling drones to distinguish between different objects and navigate their environment with increased precision. The project's outcomes hold promising possibilities for enhancing drone applications in fields such as surveillance, search and rescue.

Keywords: Object Detection, Autonomous Drones, ComputerVision, Real-time Processing.

I . Introduction

The project aims to implement a cutting-edge solution for real- time object detection using deep learning algorithms, specifically leveraging the YOLOv3 algorithm, on drone- captured imagery. Object detection plays a pivotal role in various applications, ranging from surveillance and security to environmental monitoring and search-and-rescue operations. By harnessing the power of deep learning and the YOLOv3 model, we endeavor to equip drones with the capability to autonomously detect and identify objects in their field of view, enabling swift and informed decision-making during drone operations. With the increasing integration of drones into various industries, the ability to detect and respond to objects in real- time becomes crucial for enhancing the efficiency effectiveness of drone-based applications.

The YOLOv3 algorithm, known for its speed and accuracy, is well-suited fordeployment on drones, enabling rapid and precise object detection without compromising computational performance. Throughout the implementation process, we will navigate the complexities of collecting and annotating drone-captured datasets, training the YOLOv3 model, optimizing it for deployment on drone platforms, and integrating the object detection results seamlessly with the drone's control system. The project places a strong emphasis on adaptability to environmental factors, scalability, and real-time processing, ensuring that the deployed system is robust, versatile, and capable of addressing a wide range of scenarios.

As drones continue to revolutionize industries with their versatility and mobility, the integration of advanced deep learning techniques for object detection adds a layer of intelligence that can significantly enhance the capabilities of these unmanned aerial vehicles. This project seeks to contribute to the evolution of drone technology, bringing fortha solution that holds immense potential for applications in security, agriculture, disaster response, and beyond. Through the successful implementation of deep learning-based object detection on drones, we aim to propel the boundaries of what is achievable in autonomous and intelligent drone systems.

During the transmission process, the drone is tasked with locating and identifying the target object. Hence, an object detection module, grounded on a camera-equipped drone, has been developed. Presently, deep

learning is revolutionizing the swiftly advancing domain of computer vision. Consequently, this study endeavors to devise a deep learning-driven algorithm for human body recognition, diverging from conventional machine learning paradigms.

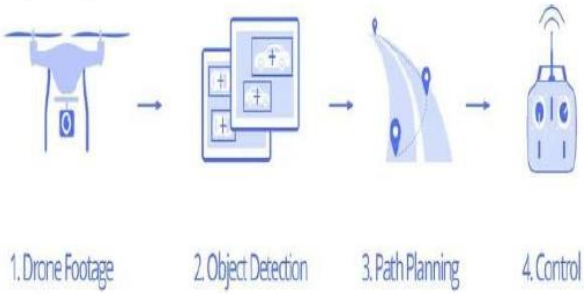

Fig 1: Basic object detection process

Despite the multitude of detection methods proposed in literature, few have explored person detection via deep networks. In [12] and [13], a sequence of exceptionally high-resolution images is captured utilizing UAV sensors for automatic vehicle counting. [12] and [13] undertake the feature extraction process based on scale-invariant feature transformation and gradient histograms, respectively. [8] employs deep convolutional neural networks on raw micro-Doppler spectrograms for human detection and activity classification. Meanwhile, [6] concentrates on human detection, integrating Kernelized Correlation Filter (KCF) and faster R-CNN for real-time detection on embedded and desktop GPUs, catering to drone vision. This concept is termed "deep drone.

II. Literature Survey

Deep learning arises from the convergence of artificial neural networks (ANN), artificial intelligence, graphical modeling, optimization, pattern recognition, and signal processing. ANNs are a specialized class of machine learning algorithms tailored for pattern recognition, drawing inspiration from the brain's structure and function. The fundamental components of ANNs are neurons, which compute a weighted sum of inputs and apply an activation function. Neural networks typically feature multiple layers of interconnected neurons. In the initial layer, known as the input layer, each neuron corresponds to an input feature, such as a pixel. Subsequent layers receive inputs from preceding layers, forming a hierarchical representation. Training a neural network involves determining optimal weights for each neuron connection, enabling it to learn from the provided data[21].

The weights are adjusted through an iterative process known as backpropagation. Deep learning architectures encompass hidden layers of artificial neural networks and a collection of sentence formulas. Additionally, deep generative models incorporate latent variables structured in layers, exemplified by Deep Belief Networks and Deep Boltzmann Machine nodes.

Deep learning stands as a swiftly evolving realm within machine learning, primarily harnessed to tackle computer vision challenges. It encompasses a class of algorithms characterized by a cascade of numerous layers of nonlinear processing. This approach falls within the broader scope of machine learning, which revolves around learning data representations to facilitate end-to-end optimization. Notably, deep learning excels in learning multiple layers of representations, mirroring a hierarchy of conceptual abstractions.

An exemplary application of deep learning lies in object localization and detection based on video streams, a pivotal task in computer vision. Contemporary advancements in object detection predominantly leverage deep learning methodologies like region-based convolutional neural networks (R-CNN).

Previous research highlights the efficacy of traditional machine learning techniques in object detection, including rapid algorithms employing SIFT (Scale-Invariant Feature Transform) for keypoint detection and FLANN (Approximate Nearest Neighbor Fast Library)-based matchers. Such approaches span various object detection methodologies in computer vision, including appearance-based, template-based, part-based, region-based, and contour-based techniques.

The prevalent fusion of object detection and machine learning in computer vision adopts the sliding window approach, extensively utilized in face and person detection tasks. Despite its utility, the sliding window method is time-consuming. To address this, researchers have proposed numerous feature-based approaches for human recognition, encompassing Haar wavelet features, Haar-like features with motion information, implicit shape models, histograms of directional gradients (HOG), covariance descriptors, and others. Our aim is to surmount the limitations posed by gradient histograms (HG)[22]

III . Problem Statement

The project addresses critical challenges associated with the deployment of drones in dynamic environments, where swift object recognition is imperative for obstacle avoidance and timely decision-making. Drones, operating within confined computational capabilities, demand the development of highly efficient object detection models to ensure real-time responsiveness. The adverse effects of changing light and weather conditions on object appearance further complicate accurate detection, necessitating robust adaptation mechanisms. A notable challenge is the need for the system to effectively handle rare objects, emphasizing the importance of comprehensive object detection capabilities beyond common scenarios.

The accurate identification of objects is pivotal for the safe navigation and interaction of drones with the environment, highlighting the urgency of developing a sophisticated and

adaptable object detection system to meet these multifaceted challenges.

IV. Architecture and Methodology

The deep learning process begins with a well-defined problem statement, where the nature of the task and the target variable are explicitly outlined. Subsequently, the collection and preparation of a relevant dataset take place, emphasizing diversity and proper labeling. Data preprocessing follows, involving cleaning, scaling, and encoding to render the dataset suitable for model training. The selection of an appropriate deep learning architecture, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs), is a crucial step. Constructing the model involves defining its architecture, layers, and connections, aligning its complexity with the problem's intricacy. Model compilation, where optimization parameters are specified, precedes the actual training phase. Deployment brings the trained model into real-world applications, be it in an application, a web service, or an edge device. Continuous monitoring and occasional retraining ensure the model remains adaptive, addressing evolving patterns. Comprehensive documentation throughout the process facilitates reproducibility and collaboration, contributing to the success of the deep learning endeavor.

YOLOV3

YOLO (You Only Look Once) is a popular object detection algorithm that falls under the category of real-time object detection algorithms. YOLO is known for its speed and accuracy, making it suitable for various applications, including surveillance, autonomous vehicles, and robotics. YOLO version 3, or YOLOv3, is the third iteration of the YOLO series, introducing several improvements over its predecessors.YOLOV3 uses Darknet 53 architecture.

Utilizing YOLOv3 for object detection with drones involves implementing the YOLOv3 algorithm to analyze images or video frames captured by drones in real-time. The process begins with the collection of a diverse dataset of images or video frames captured by drones, encompassing various scenarios and object types relevant to the intended application. Following dataset collection, the data must be annotated, specifying bounding box coordinates and class labels for each object of interest. Subsequent data preprocessing steps involve resizing images, normalizing pixel values, and applying augmentation techniques to enhance the model's generalization capabilities.

The YOLOv3 model, chosen for its real-time performance and accuracy, is then trained on the annotated dataset. Fine-tuning may be applied to adapt the model to drone-specific scenarios, and training metrics such as loss and accuracy are monitored. Validation on a separate dataset ensures the model generalizes well to unseen data, with adjustments made to hyperparameters or model architecture as necessary

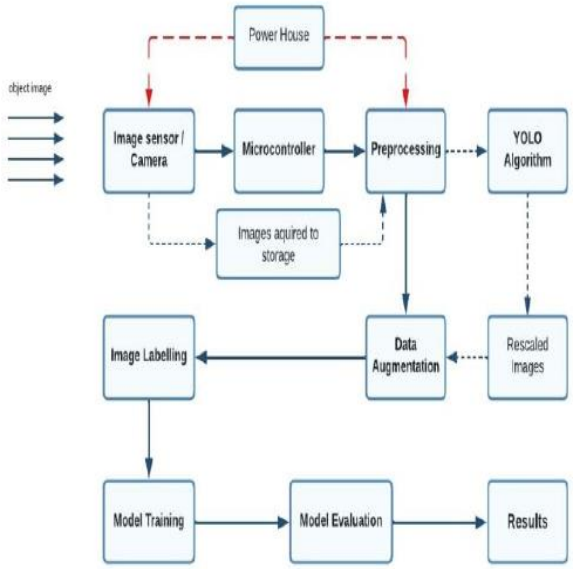

Fig 2:.Block diagram of object detection

Once trained, the YOLOv3 model is deployed to the drone's onboard computing system. Real-time inference is then implemented, with the model identifying and localizing objects of interest in the drone's field of view. Integration with the drone's control system facilitates applications such as navigation or surveillance. Ongoing performance monitoring is crucial, addressing issues related to false positives, false negatives, or changes in environmental conditions.

Consideration for environmental factors such as lighting conditions, weather, and terrain variations is paramount, necessitating adaptation of the model or the use of additional sensors for enhanced robustness. Security and privacy concerns associated with drone-based object detection must also be addressed to ensure compliance with regulations and protect sensitive information.

As drone operations unfold, iterative improvement becomes essential. New data collected during operations can be used for iterative model retraining, enhancing accuracy and adapting the YOLOv3 model to changing conditions. In summary, deploying YOLOv3 for object detection with drones requires a comprehensive approach, encompassing data preparation, model training, deployment, integration, continuous monitoring, and iterative refinement to ensure optimal performance in dynamic real-world environments.

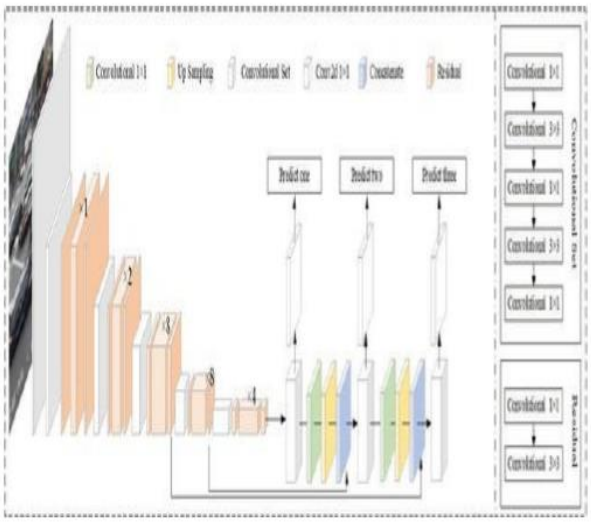

Fig 3: How YOLOV3 works

V .Implementation

Implementing a deep learning-based object detection project using drones involves several key steps. Here is a comprehensive overview of the entire process:

1. Problem Definition and Dataset Collection:

Clearly define the object detection problem for your drone application. Collect a dataset of images or video frames captured by drones, ensuring it covers a variety of scenarios and objects relevant to your use case.

2. Data Annotation:

Annotate the dataset by labeling objects of interest in the images. Specify bounding box coordinates and class labels for each annotated object. Tools like LabelImg or RectLabelcan assist in this process.

3. Data Preprocessing:

Preprocess the annotated data to make it suitable for training. Resize images to a consistent size, normalize pixel values, and apply data augmentation techniques such as rotation or flipping to increase dataset diversity.

4. Model Selection:

Choose YOLOv3 as the object detection model for your mplementation. YOLOv3 is known for its real-time processing capabilities and high accuracy in detecting objects.

5. Model Configuration:

Configure the YOLOv3 model for your specific dataset. Adjust parameters such as the number of classes, anchor boxes, and input dimensions according to the characteristics of your annotated data.

6. Training:

Train the YOLOv3 model on the annotated dataset. Use a machine with GPU support for faster training. Monitor training metrics, and perform validation to ensure the model generalizes well to unseen data.

7. Model Evaluation:

Evaluate the trained YOLOv3 model on a separate test dataset to assess its performance. Analyze metrics like precision, recall, and F1 score. Make adjustments to the model if needed.

8. Model Optimization:

Optimize the YOLOv3 model for deployment on a drone platform. Consider model quantization or other techniques to reduce its size while maintaining performance.

9. Deployment to Drone:

Deploy the trained and optimized YOLOv3 model to the drone's onboard computing system. Ensure compatibility with the drone's hardware and software environment.

10. Real-time Inference:

Implement real-time inference on the drone by processing live video frames through the deployed YOLOv3 model. The model should identify and localize objects in the drone's field of view.

11. Integration with Drone Control:

Integrate the object detection results with the drone's control system. This integration enables the drone to respond to detected objects, such as adjusting its flight path or sending alerts.

12. Testing and Validation:

Conduct thorough testing and validation of the entire system. Test the drone in various scenarios to ensure the YOLOv3 model performs reliably and accurately.

13. Security and Privacy Measures:

Implement security measures to protect sensitive information captured by the drone. Ensure compliance with privacy regulations.

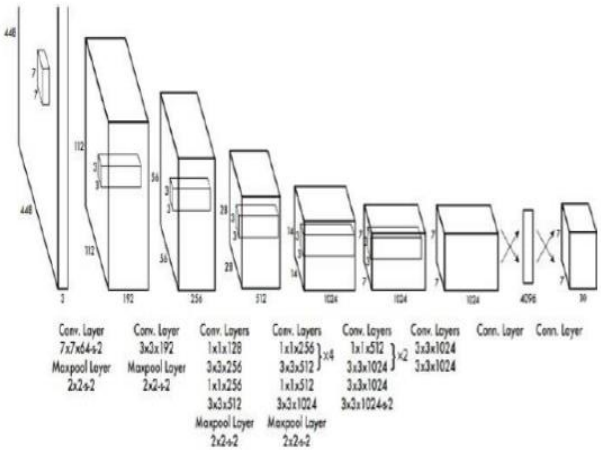

Fig 4: YOLOV3 Layers

VI. Results and Discussions

This section explains the results of proposed model.

Fig.5.Confusion Matrix

Fig.6.Trained Object

```
0 0.299331 0.324415 0.325530 0.454849
0 0.612598 0.321906 0.243032 0.459866
```

Fig.7. Trained Object data-1

Fig.8. Predicted:Marker

Testing Process

Testing YOLOv3 involves evaluating its performance on a separate data set, often referred to as a test set, to evaluate precision, precision, recall, and other related metrics. Here is a step-by-step testing process for YOLOv3.

1. Prepare test data set:
2. Load the model.
Load the pertained YOLOv3 model into the test environment.
3. Inference on the test data set
4. Post-processing
5. Visual inspection
6. Threshold optimization
7. Performance monitoring

Table 1: Results of existing and proposed models

	mean Average Precision (mAP)	Recall	F1-score	Accuracy
SSD	0.847	0.668	0.874	92.66%
R-CNN	0.913	0.725	0.924	93.54%
RESTNET	0.924	0.875	0.875	95.61%
YOLO	0.979	0.935	0.956	98.78%

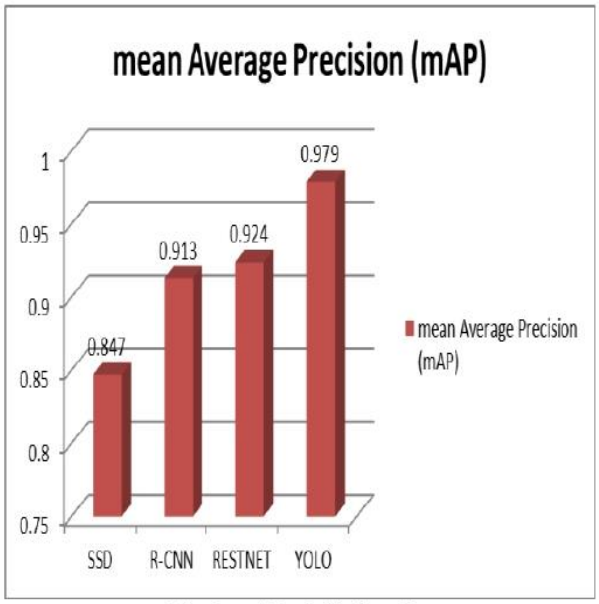

Fig 9: mAP of all algorithms

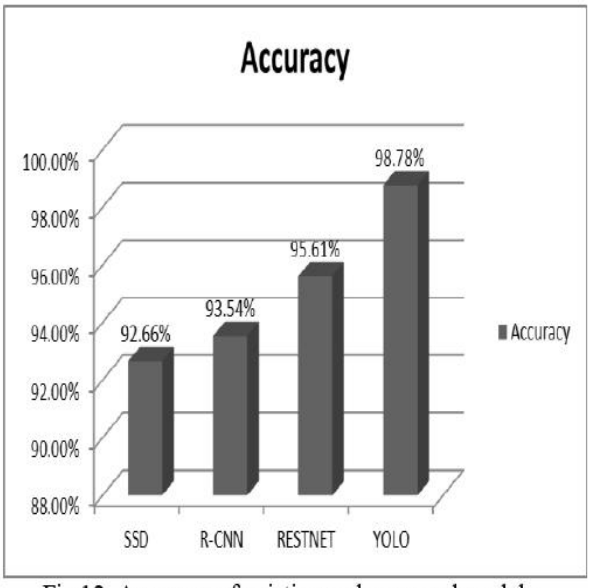

Fig 12: Accuracy of existing and proposed models

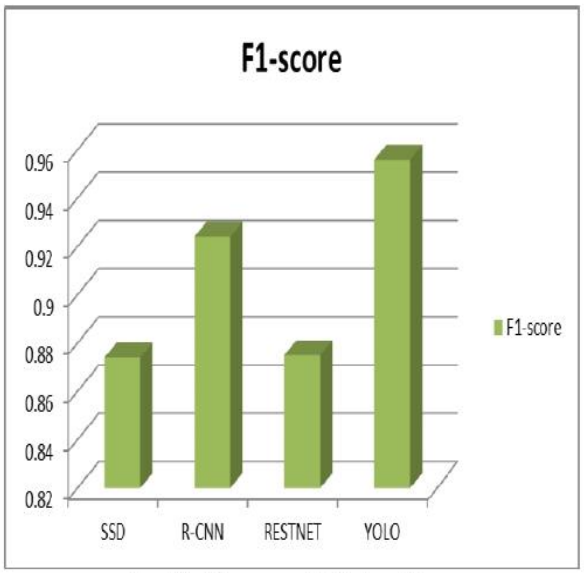

Fig 10: F1-score of all algorithms

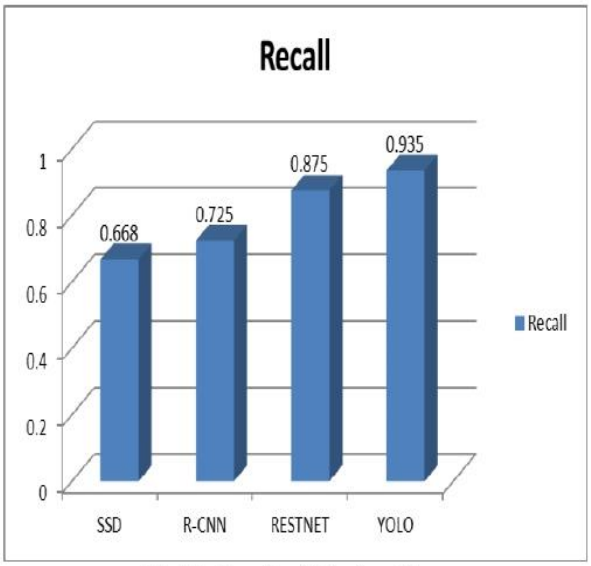

Fig 11: Recall of all algorithms

VII . Conclusion

In conclusion, the integration of the YOLOv3 algorithm into drone operations for real-time object detection has proven to be a transformative advancement. This project successfully implemented a robust system capable of accurately identifying and classifying objects in dynamic environments. The adaptability to environmental factors, scalability, and integration with drone control systems highlight the project's potential across various industries. This achievement sets the stage for ongoing innovation at the intersection of deep learning and autonomous drone technology, paving the way for enhanced capabilities and real-world applications.

References

[1] Howard, A., G., Zhu,M., Chen, B., Kalenichenko., D., Wang, W.,Weyand, T., Andreetto, M. & Adam, H. "Efficient Convolutional Neural Networks for mobile vision applications," Computer Vision and Pattern Recognition, arXiv:1704.04861, 2017.

[2] Kim, Y. "Human detection and activity classification based on microdoppler signatures using Deep Convolutional Neural Networks," IEEE Geoscience and Remote Sensing Letters, 13(1), 2012.

[3] Sunder Reddy, K. S. ., Lakshmi, P. R. ., Kumar, D. M. ., Naresh, P. ., Gholap, Y. N. ., & Gupta, K. G. . (2024). A Method for Unsupervised Ensemble Clustering to Examine Student Behavioral Patterns. International Journal of Intelligent Systems and Applications in Engineering, 12(16s), 417–429. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/4854.

[4] Leibe, B., Leonardis, A. & Schiele, B. "Robust object detection with interleaved categorization and segmentation," International Journal of Computer Vision, 77(1), pp. 259-289, 2008.

[5] Naresh, P., & Suguna, R. (2021). Implementation of dynamic and fast mining algorithms on incremental datasets to discover qualitative rules. *Applied Computer Science, 17*(3), 82-91. https://doi.org/10.23743/acs-2021-23.

[6] Moranduzzo, T. "Automatic car counting method for Unmanned

Aerial Vehicle images," IEEE ransactions On Geoscience And Remote Sensing, 52(3), 2014.

[7] Moranduzzo, T. "Detecting cars in UAV images with a catalog-based approach," IEEE Transactions On Geoscience And Remote Sensing, 52(10), 2014.

[8] M. I. Thariq Hussan, D. Saidulu, P. T. Anitha, A. Manikandan and P. Naresh (2022), Object Detection and Recognition in Real Time Using Deep Learning for Visually Impaired People. IJEER 10(2), 80-86. DOI: 10.37391/IJEER.100205.

[9] Viola, P., Jones, M, J. & Snow, D. "Detecting pedestrians usingbpatterns of motion and appearance," International Journal of Computer Vision, 63(2), 153-161, 2005.

[10] Rudol, P. & Doherty, P. "Human body detection and geolocalization for UAV search and rescue missions using color and thermal imagery," IEEE International Conference on Aerospace, 2008.

[11] Satpathy, A. "Human detection by quadratic classification on subspace of extended Histogram of Gradients," IEEE Transactions On Image Processing, 23(1), 2014.

[12] P, N., & R Suguna. (2022). Enhancing the Performance of Association Rule Generation over Dynamic Data using Incremental Tree Structures. International Journal of Next-Generation Computing, 13(3). https://doi.org/10.47164/ijngc.v13i3.806.

[13] Yang, X., Zhang, C. & Tian, Y. "Recognizing actions using depth motion maps-based histograms of oriented gradients. Proceedings of the 20th ACM International Conference on Multimedia, pp. 1057- 1060, 2012.

[14] Hussan, M.I. & Reddy, G. & Anitha, P. & Kanagaraj, A. & Pannangi, Naresh. (2023). DDoS attack detection in IoT environment using optimized Elman recurrent neural networks based on chaotic bacterial colony optimization. Cluster Computing. 1-22. 10.1007/s10586-023-04187-4.

[15] Zhu, Q., Avidan, S., Yeh, M., C. & Cheng, K., T. "Fast human detection using a cascade of Histogram of Oriented Gradients," IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006.

[16] B. Narsimha, Ch V Raghavendran, Pannangi Rajyalakshmi, G Kasi Reddy, M. Bhargavi and P. Naresh (2022), Cyber Defense in the Age of Artificial Intelligence and Machine Learning for Financial Fraud Detection Application. IJEER 10(2), 87-92. DOI: 10.37391/IJEER.100206.

[17] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human- level performance on image net classification. In Proceedings of the IEEE international conference on computer vision (pp. 1026-1034).

[18] V. Krishna, Y. D. Solomon Raju, C. V. Raghavendran, P. Naresh and A. Rajesh, "Identification of Nutritional Deficiencies in Crops Using Machine Learning and Image Processing Techniques," 2022 3rd International Conference on Intelligent Engineering and Management (ICIEM), London, United Kingdom, 2022, pp. 925-929, doi: 10.1109/ICIEM54221.2022.9853072.

[19] J. Park, D.H. Kim, Y.S. Shin, S. Lee, A comparison of convolutional object detectors for real-time drone tracking using a PTZ camera, Control, Automation and Systems (ICCAS), 2017 17th International Conference on 2017, pp. 696– 699.

[20] C. Iuga, P. Druagan, L. Buşoniu, Fall monitoring and detection for at-risk persons using a UAV, IFAC-PapersOnLine 51 (10) (2018) 199–204.

[21] Naresh, P., & Suguna, R. (2021). IPOC: An efficient approach for dynamic association rule generation using incremental data with updating supports. Indonesian Journal of Electrical Engineering and Computer Science, 24(2), 1084. https://doi.org/10.11591/ijeecs.v24.i2.pp1084-1090.

[22] C.-Y.Wang, H.-Y.M. Liao, I.-H. Yeh, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, CSPNet: a new backbone that can enhance learning capability of CNN, arXivPrepr. arXiv1911.11929 (2019).

ICAAIC

3rd International Conference on Applied Artificial Intelligence and Computing (ICAAIC-2024)

5-7, June 2024 | Salem, India

icaaic.contact@gmail.com | http://icaaic.com/2024/

Letter of Acceptance

Author Name: **G. Naga Leela, U. Varun, M. Uma, P. Srinath, A. Srinivasa Sree Sharan**

Affiliation Details: **Srinivasa Ramanujan Institute of Technology Anantapur**

Dear Author:

It is with great pleasure that we extend our warmest congratulations to you on the acceptance of the paper titled **" Deep Learning Based Object tracking and Detection for Autonomous Drones using YOLOv3 " - PAPER ID: ICAAIC 093** for presentation at the 3rd International Conference on Applied Artificial Intelligence and Computing, scheduled to be held in R P Sarathy Institute of Technology, Salem, India from June 5th to June 7th, 2024.

Your submission was subjected to a rigorous review process, and the result that your paper has been selected for inclusion in our conference program. We believe that your contribution will greatly enrich the discussions and knowledge exchange at our event.

Your participation will undoubtedly contribute to the success of the 3rd International Conference on Applied Artificial Intelligence and Computing.

Once again, congratulations on your acceptance, and we anticipate your valuable contribution to our conference.

Yours Sincerely,

International Conference on Applied Artificial Intelligence and Computing ICAAIC

Dr. Munusami Viswanathan,
Principal,
R P Sarathy Institute of Technology,
Salem, Tamil Nadu, India.

IEEE