A

Project Report

REAL TIME OBJECT DETECTION AND TRACKING

Submitted to

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE AND TECHNOLOGIES RKVALLEY

in partial fulfillment of the requirement for the award of the Degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

Submitted by

BOYA NAGAMANI (R180373)

CHINTHAPANDU VENKATA SUPRAJA (R180380)

Under the Guidance of

Ms. S.RAJESWARI, Guest Faculty

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES
(catering the Educational Needs of Gifted Rural Youth of AP)

R.K Valley, Vempalli(M), Kadapa(Dist) – 516330

2020 - 2024

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES
(A.P.Government Act 18 of 2008) RGUKT-RKValley

Vempalli, Kadapa, Andhrapradesh - 516330.

CERTIFICATE OF PROJECT COMPLETION

This is to certify that I have examined the thesis entitled as "**REAL TIME OBJECT DETECTION AND TRACKING**" submitted by **BOYA NAGAMANI (R180373), CHINTHAPANDU VENKATA SUPRAJA (R180380)** under our guidance and supervision for the partial fulfilment for the degree of Bachelor of Technology in computer Science and Engineering during the academic session February 2023 – July 2023 at RGUKT-RKVALLEY.

Project Guide

Ms. S.Rajeswari ,
Guest Faculty in Dept of CSE,
RGUKT-RK Valley.

Head of the Department

Mr. N.Satyanandaram,
Lecturer in Dept of CSE,
RGUKT-RK Valley.

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE AND TECHNOLOGIES

(A.P.Government Act 18 of 2008) RGUKT-

RK Valley

Vempalli, Kadapa, Andhrapradesh-516330.

DECLARATION

We, **BOYA NAGAMANI (R180373), CHINTHAPANDU VENKATA SUPRAJA (R180380)** hereby declare that the project report entitled "**REAL TIME OBJECT DETECTION AND TRACKING**" done under guidance of **Ms. S. Rajeswari** is submitted in partial fulfillment for the degree of Bachelor of Technology in Computer Science and Engineering during the academic session February 2023 – July 2023 at RGUKT-RK Valley. I also declare that this project is a result of our own effort and has not been copied or imitated from any source. Citations from any websites are mentioned in the references. To the best of my knowledge, the results embodied in this dissertation work have not been submitted to any university or institute for the award of any degree or diploma.

Date :

Boya Nagamani-R180373

Chinthapandu Venkata Supraja-R180380

Place : RK Valley

ACKNOWLEDGEMENT

I would like to express my deep sense of gratitude & respect to all those people behind the screen who guided, inspired and helped me crown all my efforts with success. I wish to express my gratitude to **S.Rajeswari** for her valuable guidance at all stages of study, advice, constructive suggestions, supportive attitude and continuous encouragement, without which it would not be possible to complete this project.

I would also like to extend our deepest gratitude & reverence to the Director of RGUKT, RK Valley **Prof. K. Sandyarani** and HOD of Computer Science and Engineering **Mr. N. Satyanandaram** for their constant support and encouragement.

Last but not least I express my gratitude to my parents for their constant source of encouragement and inspiration for me to keep my morals high.

With Sincere Regards,

Boya Nagamani-R180373,

Chinthapandu Venkata Supraja-R180380,

ABSTRACT

Efficient and accurate object detection has been an important topic in the advancement of computer vision systems. With the advent of deep learning techniques, the accuracy for object detection has increased drastically. The project aims to

Object recognition is the process of recognizing objects based on their characteristics like color, shape, and with the particular occurrence of the object in digital videos and as well as normal daily life. This object detection is the already known and introduced technology that performs the program using computer technology as the platform and using some image processing units for the detection of the objects using some class of data sets given by the user. Dual priorities, an imbalance in the classes, a lack of data, etc. are a few significant challenges. As this object detection is working as a real-time application and is mostly used for tracking objects, counting crowds, and self-driving cars, and even more useful in traffic control and also for security surveillance purposes which is the major purpose of usage. Some techniques like deep learning-based object detection a relatively new approach that has blurred the lines between object localization and detection. Region-Based Convolutional Neural Network was used to deal with it. Another was a fast Region-Based Convolutional Neural Network, which was proposed in Python and C++ and raises the problem of Region-Based Convolutional Neural Network and Spatial Pyramid Pooling-net when dealing with speed and accuracy. It was resolved using a deep VGG16 network. This application eventually came up with some basic concepts that are likely OpenCV, an open-source library with some functions that can be used for object detection, and the combination of python 2.7, improving the accuracy and efficiency of object detection. Even though many applications are available initially, by using the simple functions in OpenCV and NumPy, the proposed work achieves accurate results in finding the objects easily.

Index

CHAPTER 1

INTRODUCTION

1.1 Motivation

Object detection is a fundamental task in computer vision that involves identifying and localizing objects of interest within an image or video. It has numerous applications across various domains, including surveillance, autonomous driving, robotics, augmented reality, and image retrieval. OpenCV (Open Source Computer Vision Library) is a widely used open-source library that provides a comprehensive set of tools and algorithms for computer vision tasks, including object detection.

There are several key motivations for using OpenCV for object detection:

1. **Efficiency**: OpenCV is designed to be highly efficient and optimized for real-time applications. It provides a wide range of algorithms and techniques that are specifically tailored for object detection tasks. These algorithms are implemented using optimized C/C++ code, making them highly efficient and suitable for real-time processing on resource-constrained devices such as embedded systems or mobile devices.

2. **Versatility**: OpenCV offers a wide range of object detection algorithms, allowing developers to choose the most appropriate technique based on their specific requirements. It includes both traditional computer vision techniques, such as Haar cascades and HOG (Histogram of Oriented Gradients), as well as more advanced deep learning-based approaches like Faster R-CNN (Region-based Convolutional Neural Networks) and YOLO (You Only Look Once). This versatility enables developers to tackle different types of object detection problems, from simple object recognition to complex scene understanding.

3. **Ease of Use**: OpenCV provides a high-level API that simplifies the

process of developing object detection applications. It abstracts away many low-level details, such as memory management and algorithm implementation, allowing developers to focus on the core logic of their applications.Additionally, OpenCV supports multiple programming languages, including C++, Python, Java, and MATLAB, making it accessible to a wide range of developers with different programming backgrounds.

4. **Integration with other Computer Vision Tasks**: OpenCV seamlessly integrates with other computer vision tasks, such as image preprocessing, feature extraction, and image segmentation. This integration allows developers to build end-to-end object detection pipelines by combining multiple algorithms and techniques. For example, object detection can be combined with object tracking or semantic segmentation to achieve more robust and accurate results.

5. **Large Community and Support**: OpenCV has a large and active community of developers and researchers who contribute to its development and maintenance. This community provides extensive documentation, tutorials, and code examples that help newcomers get started with object detection using OpenCV. Additionally, there are numerous online forums and communities where developers can seek help or share their experiences with OpenCV-based object detection projects.

In conclusion, the motivation for using OpenCV for object detection lies in its efficiency, versatility, ease of use, integration with other computer vision tasks, and the support provided by its large community.

1.2 Objective of Project :

This project aims to do real-time object detection through a laptop camera or webcam using OpenCV and MobileNetSSD. The idea is to loop over each frame of the video stream, detect objects like person, chair, dog, etc. and bound each detection in a box.

1.3 Features:

There are several key features of object detection algorithms that contribute to their effectiveness and efficiency:

1. **Localization**: Object detection algorithms are designed to not only identify objects but also precisely localize them within the image or video frame. Localization involves determining the spatial extent of the detected objects by drawing bounding boxes around them. This information is crucial for understanding the position, size, and orientation of objects in the scene.

2. **Classification**: In addition to localization, object detection algorithms perform classification to determine the category or class label of each detected object. This involves assigning a specific label to each bounding box, indicating the type of object present (e.g., person, car, dog). Classification is typically achieved using machine learning techniques such as deep neural networks, which have shown remarkable performance in recent years.

3. **Multiple Object Detection**: Object detection algorithms are designed to handle scenarios where multiple objects of different classes may be present in an image or video frame. These algorithms can detect and classify multiple objects simultaneously, providing a comprehensive understanding of the visual scene. This capability is particularly important in applications such as pedestrian detection in autonomous driving or crowd monitoring in surveillance systems.

4. **Real-time Performance**: Many object detection algorithms strive to achieve real-time performance, meaning they can process images or video frames at a speed that matches or exceeds the frame rate of the input source.

5. **Scale and Rotation Invariance**: Object detection algorithms should be able to handle objects of various sizes and orientations. They should be robust to scale changes (e.g., detecting both small and large objects) and rotation variations (e.g., detecting objects at different angles).

CHAPTER- 2

REQUIREMENT ANALYSIS

2.1. Software requirement:

Front end	Python libraries: 1.open cv 2.object recogintion modules 4.dlib 5.yolo 6.numpy
Back end	Python
Webcam	any compatible device camera
Operating System	Ubuntu,windows
Software	Ubuntu,windows

2.1.1 FUNCTIONAL REQUIREMENTS

1. Adaptability: Object detection systems should be adaptable to different hardware platforms and computational resources. They should be able to run efficiently on different devices, such as embedded systems, GPUs, or cloud-based servers. This requirement allows for the deployment of object detection systems in diverse computing environments.

2. Support for multiple object classes: Object detection systems should have the capability to detect and classify objects belonging to multiple classes or categories. This requirement enables the system to handle complex scenes with diverse objects present simultaneously.

3. Integration with other systems: Object detection systems often need to be integrated with other components or systems, such as tracking algorithms, decision-making modules, or user interfaces. Therefore, the system should have the ability to communicate and exchange data with other software components

4.Support for different input sources: Object detection systems should be able to handle various input sources, including images, videos, and live camera feeds. This requirement allows for flexibility in using different types of datasources depending on the application's needs.

2.1.2 NON-FUNCTIONAL REQUIREMENTS

- **Usability Requirement**

The system shall allow the users to access the system from the system using any web browsers. The system uses a web browsers as an interface. Since all users are familiar with the general usage of a website, no special training is required. The system is user friendly which makes the system easy.

- **Availability Requirement**

The system is available 100% for the user and is used 24 hrs a day and 365 days a year. The system shall be operational 24 hours a day and 7 days a week.

- **Efficiency Requirement**

Mean Time to Repair (MTTR) - Even if the system fails, the system will be recovered back up within an hour or less.

- **Accuracy**

The system should accurately provide real time information taking into consideration various concurrency issues. The system shall provide 100% access reliability.

- **Reliability Requirement**

The system has to be 100% reliable due to the importance of data and the damages that can be caused by incorrect or incomplete data.

2.2 Technologies Used

2.2.1 NUMPY

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It is open-source software. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

2.2.2 Python:

Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a generalpurpose language, meaning it can be used to create a variety of different programs and isn't specialized for any specific problems.

Features:

Easy To Learn and Readable Language. Python is extremely easy to learn

- Interpreted Language.
- Dynamically Typed Language
- Open Source and Free
- Large Standard Library
- High-Level Language
- Object Oriented Programming Language
- Large Community Support

2.2.3 Dlib

- Dlib is one of the most powerful and easy-to-go open-source library consisting of machine learning library/algorithms and various tools for creating software.
- DLib: Library for Machine Learning.
- Dlib along with OpenCV can handle bad and inconsistent lighting and various facial positions such as tilted or rotated faces.
- Dlib is ahead of the Haar cascade classifier over implementation, speed, and accuracy.

2.2.4 Opencv:

- OpenCV is a great tool for image processing and performing computer vision tasks. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more.
- OpenCV is a huge open-source library for computer vision, machine learning, and image processing.
- OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human.

2.2.5 YOLO :

YOLO (You Only Look Once) is a method / way to do object detection. It is the algorithm /strategy behind how the code is going to detect ob jects in the image.

YOLO takes entirely different approach. It looks at the entire image only once and goes through the network once and detects objects. Hence the name. It is very fast. That's the reason it has got so popular.

There are other popular object detection frameworks like **Faster R-CNN** and **SSD** that are also widely used.

In this post, we are going to look at how to use a pre-trained YOLO model with OpenCV and start detecting objects right away.

CHAPTER-3

SOFTWARE ENVIRONMENT

3.1 OpenCv

OpenCV is a Python open-source library, which is used for computer vision in Artificial intelligence, Machine Learning, face recognition, etc.

In OpenCV, the CV is an abbreviation form of a computer vision, which is defined as a field of study that helps computers to understand the content of the digital images such as photographs and videos.

The purpose of computer vision is to understand the content of the images. It extracts the description from the pictures, which may be an object, a text description, and three-dimension model, and so on. For example, cars can be facilitated with computer vision, which will be able to identify and different objects around the road, such as traffic lights, pedestrians, traffic signs, and so on, and acts accordingly.

Computer vision allows the computer to perform the same kind of tasks as humans with the same efficiency. There are a two main task which are defined below:

- **Object Classification -** In the object classification, we train a model on a dataset of particular objects, and the model classifies new objects as belonging to one or more of your training categories.
- **Object Identification -** In the object identification, our model will identify a particular instance of an object - for example, parsing two faces in an image and tagging one as Virat Kohli and other one as Rohit Sharma.

3.2 History

OpenCV stands for Open Source Computer Vision Library, which is widely used for image recognition or identification. It was officially launched in 1999 by Intel. It was written in C/C++ in the early stage, but now it is commonly used in Python for the computer vision as well.

The first alpha version of OpenCV was released for the common use at the IEEE Conference on Computer Vision and Pattern Recognition in 2000, and between 2001 and 2005, five betas were released. The first 1.0 version was released in 2006.The second version of the OpenCV was released in October 2009 with the significant changes. The second version contains a major change to the C++ interface, aiming at easier, more type-safe, pattern, and

better implementations.Currently, the development is done by an independent Russian team and releases its newer version in every six months.

3.3 Features of OpenCV Library

Using OpenCV library, you can −

- Read and write images
- Capture and save videos
- Process images (filter, transform)
- Perform feature detection
- Detect specific objects such as faces, eyes, cars, in the videos or images.
- Analyze the video, i.e., estimate the motion in it, subtract the background, and track objects in it.

OpenCV was originally developed in C++. In addition to it, Python and Java bindings were provided. OpenCV runs on various Operating Systems such as windows, Linux, OSx, FreeBSD, Net BSD, Open BSD, etc.

3.4 OpenCV Library Modules

Following are the main library modules of the OpenCV library.

Core Functionality

This module covers the basic data structures such as Scalar, Point, Range, etc., that are used to build OpenCV applications. In addition to these, it also includes the multidimensional array **Mat**, which is used to store the images. In the Java library of OpenCV, this module is included as a package with the name **org.opencv.core**.

Image Processing

This module covers various image processing operations such as image filtering, geometrical image transformations, color space conversion, histograms, etc. In the Java library of OpenCV, this module is included as a package with the name **org.opencv.imgproc**.

Video

This module covers the video analysis concepts such as motion estimation, background subtraction, and object tracking. In the Java library of OpenCV, this module is included as a package with the name **org.opencv.video**.

Video I/O

This module explains the video capturing and video codecs using OpenCV library. In the Java library of OpenCV, this module is included as a package with the name **org.opencv.videoio**.

calib3d

This module includes algorithms regarding basic multiple-view geometry algorithms, single and stereo camera calibration, object pose estimation, stereo correspondence and elements of 3D reconstruction. In the Java library of OpenCV, this module is included as a package with the name **org.opencv.calib3d**.

Features2d

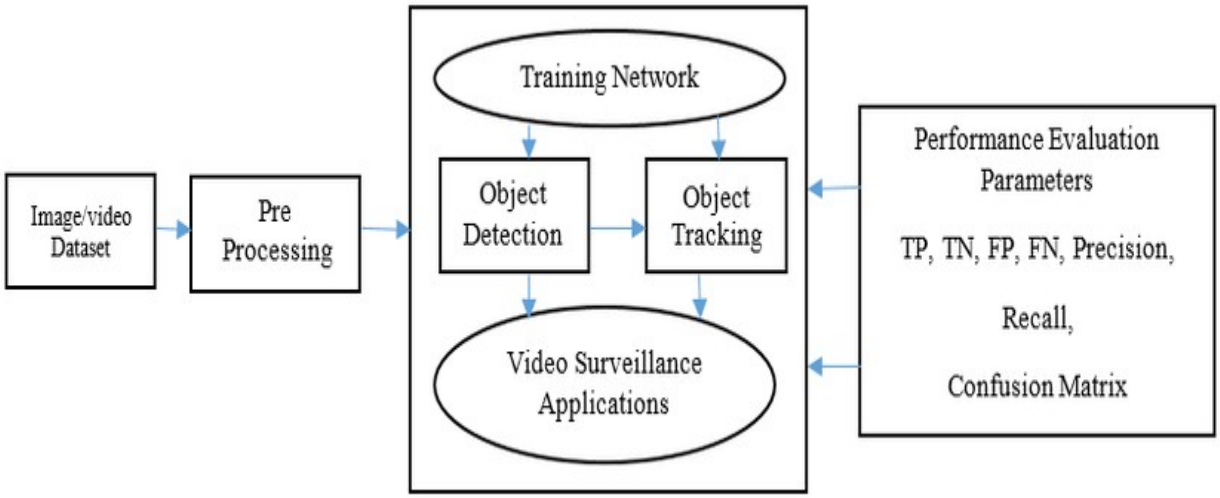
This module includes the concepts of feature detection and description. In the Java library of OpenCV, this module is included as a package with the name org.opencv.features2d.

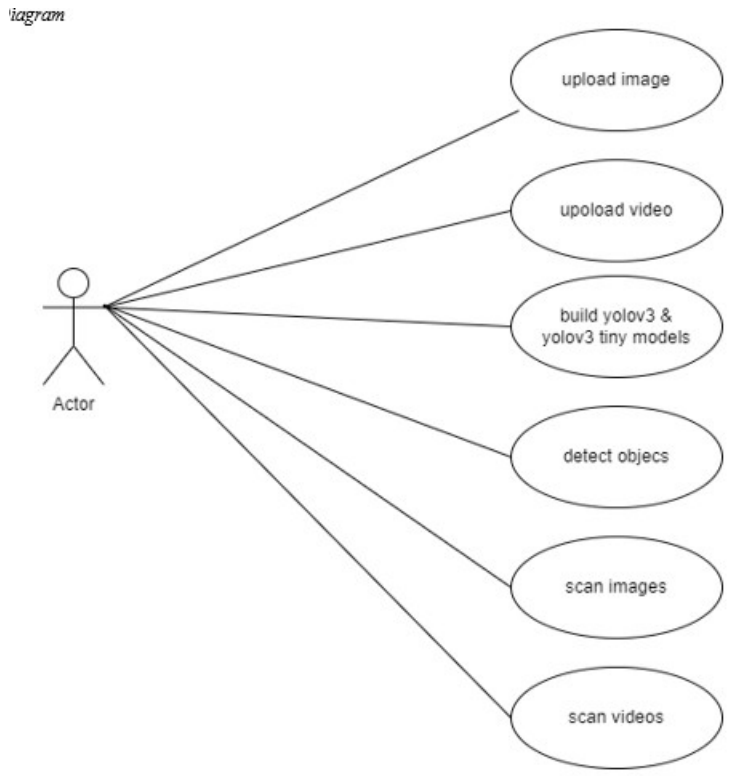
Objdetect

This module includes the detection of objects and instances of the predefined classes such as faces, eyes, mugs, people, cars, etc. In the Java library of OpenCV, this module is included as a package with the name **org.opencv.objdetect**.

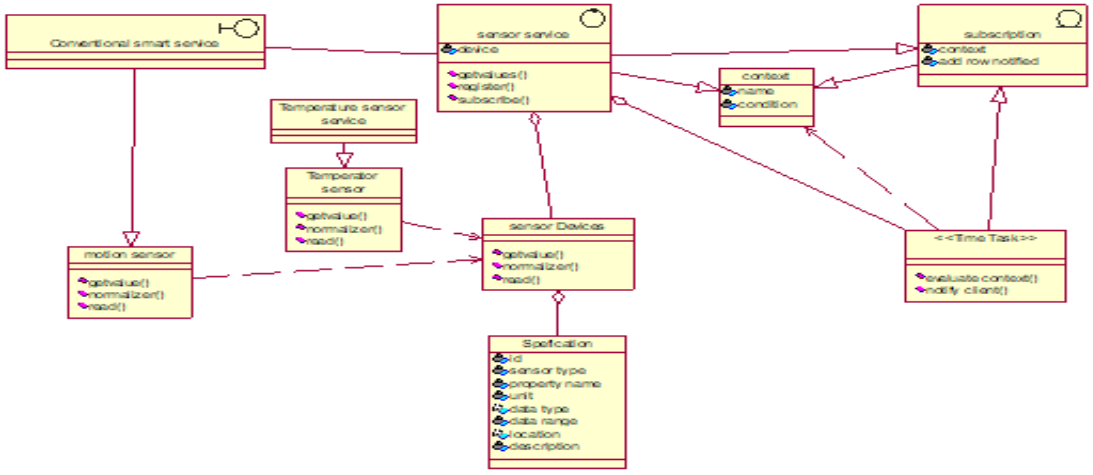
3.5 System architecture:

Context diagram:





usecase diagram



Class diagram

CHAPTER -4

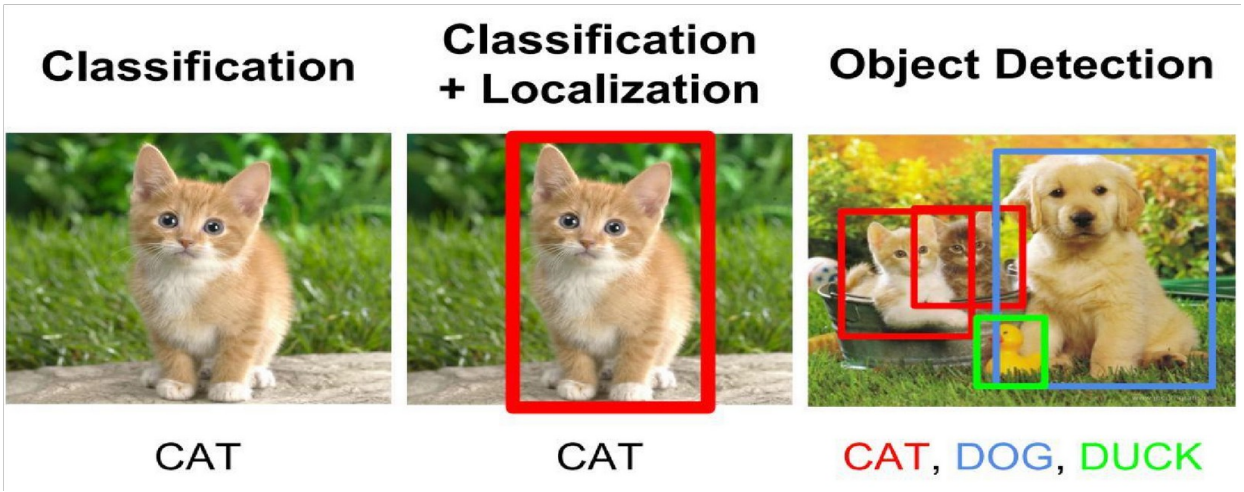
4.IMPLEMENTATION

Object detection is a well-known computer technology connected with computer vision and image processing.With the advent of deep learning techniques, the accuracy for object detection has increased drastically.

It focuses on detecting objects or its instances of a certain class (such as humans, flowers, animals) in digital images and videos. There are various applications including face detection, character recognition, and vehicle calculator.

4.1 Problem Statement

Many problems in computer vision were saturating on their accuracy before a decade. However, with the rise of deep learning techniques, the accuracy of these problems drastically improved. One of the major problem was that of image classi cation, which is de ned as predicting the class of the image. A slightly complicated problem is that of image localiza-tion, where the image contains a single object and the system should



predict the class of the location of the object in the image (a

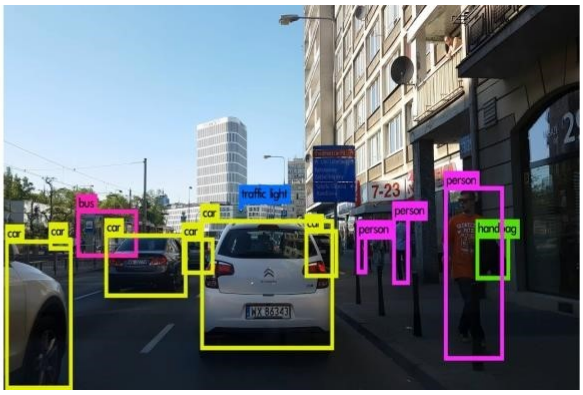
bounding box around the object). The more complicated problem (this project), of object detection involves both classi cation and localization. In this case, the input to the system will be a image, and the output will be a bounding box corresponding to all the objects in the image, along with the class of object in each box. An overview of all these problems is depicted in Fig. 1.

4.2 Applications

A well known application of object detection is face detection, that is used in almost all the mobile cameras. A more generalized (multi-class) application can be used in autonomous driving where a variety of objects need to be detected. Also it has a important role to play in surveillance systems. These systems can be integrated with other tasks such as pose estimation where the rst stage in the pipeline is to detect the object, and then the second stage will be to estimate pose in the detected region. It can be used for tracking objects and thus can be used in robotics and medical applications. Thus this problem serves a multitude of application

(a) Surveillance

(b) Autonomous vehicles



4.3 Challenges

The major challenge in this problem is that of the variable dimension of the output which is caused due to the variable number of objects that can be present in any given input image. Any general machine learning task requires a xed dimension of input and output for the model to be trained. Another important obstacle for widespread adoption of object detection systems is the requirement of real-time (>30fps) while being accurate in detection. The more complex the model is, the more time it requires for inference; and the less complex the model is, the less is the accuracy. This trade-o between accuracy and performance needs to be chosen as per the application. The problem involves classi cation as well as regression, leading the model to be learnt simultaneously. This adds to the complexity of the problem.

4.4 PROPOSED SYSTEM

One of the important fields of Artificial Intelligence is Computer Vision the science of computers and software systems that can recognize and understand images and scenes. Computer Vision is also composed of various aspects such as image recognition, object detection, image generation, image super-resolution and more. Object detection is probably the most profound aspect of computer vision due the number of practical use cases. Object detection refers to the capability of software systems to locate objects in an image/scene and identify each object. It has been widely used for face detection, vehicle detection, pedestrian counting, web images, security systems and driverless cars. There are many ways object detection can be used as well in many fields of practice.

Like every other computer technology, a wide range of creative and amazing uses of object detection will definitely come from the efforts of computer programmers and software developers.

Getting to use modern object detection methods in applications and systems, as well as building new applications based on these methods is not a straight forward task. Early implementations of object detection involved the use of classical algorithms, like the ones supported in OpenCV, the popular computer vision library. However, these classical algorithms could not achieve enough performance to work under different conditions.

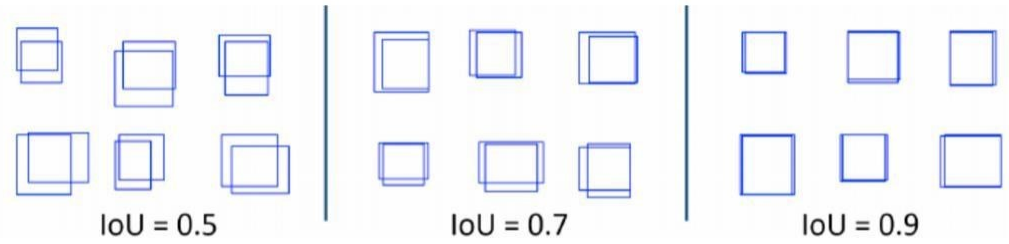
This project took use of several software libraries, packages and programs to utilize machine learning. Python was the choice of programming language, and TensorFlow was used for the deep learning computations, which in turn has a list of dependencies. TensorFlow offers a version for CPU usage and another for GPU, this project used the GPU version. Said version requires extra programs from the GPU designer NVIDIA, such as CUDA Toolkit, cuDNN and their GPU drivers. So far NVIDIA is the leading GPU designer for deep learning (also crypto mining and other similar high complex tasks) since they also write programs that are compatible with their cards that enable much of this capacity. The card used for this project was a GeForce GTX 990 TI.

4.5 EXISTING SYSTEM

There has been a lot of work in object detection using traditional computer vision techniques (sliding windows, deformable part models). However, they lack the accuracy of deep learning based techniques. Among the deep learning based techniques, two broad class of methods are prevalent: two stage detection (RCNN [1], Fast RCNN [2], Faster RCNN [3]) and uni ed detection (Yolo [4], SSD [5]). The major concepts involved in these techniques have been explained below.

4.5.1 Bounding Box

The bounding box is a rectangle drawn on the image which tightly ts the object in the image. A bounding box exists for every instance of every object in the image. For the box, 4 numbers (center x, center y, width, height) are predicted. This can be trained using a distance measure between predicted and ground truth bounding box. The distance measure is a jaccard distance which computes intersection over union between the predicted and ground truth boxes as shown in Fig.



4.5.2 Classification + Regression

The bounding box is predicted using regression and the class within the bounding box is predicted using classi cation. The overview of the architecture is shown in Fig. 4

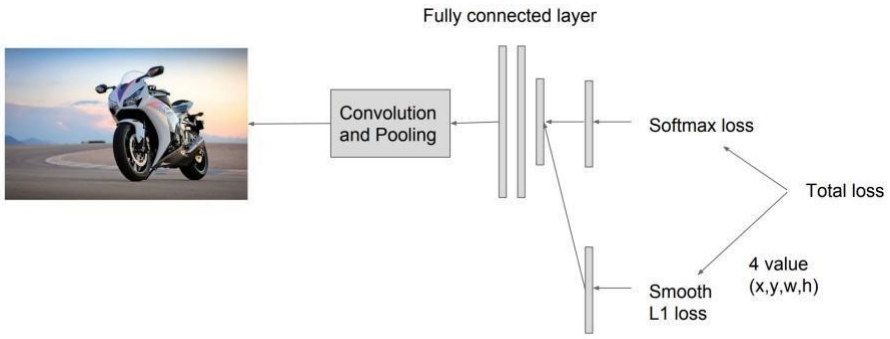
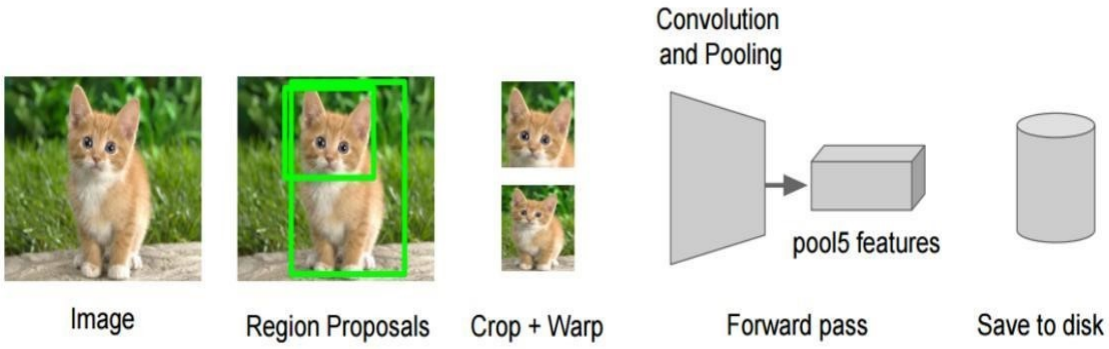


Figure : Architecture overview

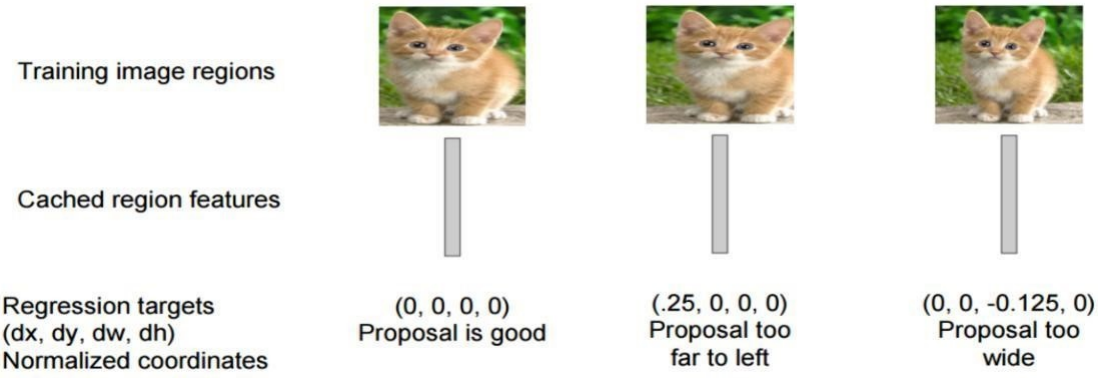
4.5.3 Two-stage Method

In this case, the proposals are extracted using some other computer vision technique and then resized to xed input for the classi cation network, which acts as a feature extractor. Then an SVM is trained to classify between object and background (one SVM for each class). Also a bounding box regressor is trained that outputs some some correction (o sets) for proposal boxes. The overall idea is shown in Fig. 5 These methods are very accurate but are computationally intensive (low fps). sta

(a) Stage 1.



(b) Stage 2



4.5.4 Unified Method

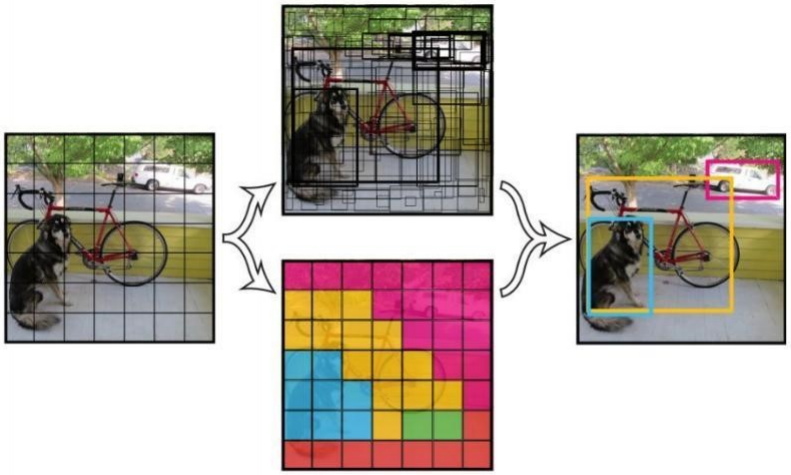
The difference here is that instead of producing proposals, pre-de ne a set of boxes to look for objects. Using convolutional feature maps from later layers of the network, run another network over these feature maps to predict class scores and bounding box o sets. The broad idea is depicted in Fig. 6. The steps are mentioned below:

1. Train a CNN with regression and classi cation objective.

2. Gather activation from later layers to infer classi cation and location with a fully connected or convolutional layers.

3. During training, use jaccard distance to relate predictions with the ground truth.

4. During inference, use non-maxima suppression to lter multiple boxes around the same object.



The major techniques that follow this strategy are: SSD (uses di erent activation maps (multiplescales) for prediction of classes and bounding boxes) and Yolo (uses a single ac-tivation map for prediction of classes and bounding boxes). Using multiple scales helps to achieve a higher mAP(mean average precision) by being able to detect objects with di erent sizes on the image better. Thus the technique used in this project is SSD.

4.6 IMPLEMENTATION AND RESULTS

The project is implemented in python 3. TensorFlow was used for training the deep network and OpenCV was used for image pre-processing.

The system specifications on which the model is trained and evaluated are mentioned as follows: CPU - Intel Core i7-7700 3.60 GHz, RAM - 32 Gb, GPU - Nvidia Titan Xp.

INPUT **GROUND EARTH**

PREDICTION

CHAPTER-5

CONCLUSION

An accurate and efficient object detection system has been developed which achieves comparable metrics with the existing state-of-the-art system. This project uses recent techniques in the eld of computer vision and deep learning. Custom dataset was created using labelImg and the evaluation was consistent. This can be used in real-time applications which require object detection for pre-processing in their pipeline.

An important scope would be to train the system on a video sequence for usage in tracking applications. Addition of a temporally consistent network would enable smooth detection and more optimal than per-frame detection.

5.1 FUTURE WORKS

Computer vision is still a developing discipline, it has not been matured to that level where it can be applied directly to real life problems.

After few years computer vision and particularly the object detection would not be any more futuristic and will be ubiquitous. For now, we can consider object detection as a sub-branch of machine learning.

ImageAI provides many more features useful for customization and production capable deployments for object detection tasks. Some of the features supported are:

Adjusting Minimum Probability: By default, objects detected with a probability percentage of less than 50 will

not be shown or reported. You can increase this value for high certainty cases or reduce the value for cases where all possible objects are needed to be detected.

Custom Objects Detection: Using a provided CustomObject class, you can tell the detection class to report detections on one or a few number of unique objects.

Detection Speeds: You can reduce the time it takes to detect an image by setting the speed of detection speed to “fast”, “faster” and “fastest”.

Input Types: You can specify and parse in file path to an image, Numpy array or file stream of an image as the input image.

Output Types: You can specify that the detect Objects From Image function should return the image in the form of a file or Numpy array

CHAPTER-6

REFERENCES

- https://github.com/Surya-Murali/Real-Time-Object-Detection-With-OpenCV.github
- https://docs.python.org/3.8/library/index.html
- https://www.geeksforgeeks.org/introduction-to-multi-task-learningmtl-for-deep-learning/
- https://www.geeksforgeeks.org/introduction-machine-learning-using-python/
- https://www.geeksforgeeks.org/learning-model-building-scikit-learn-python-machinelearning-library/
- https://github.com/tirthajyoti/Machine-Learning-with-Python/blob/master/Pandas%20and%20Numpy/Numpy_Pandas_Quick.ipynb
- https://github.com/tirthajyoti/Machine-Learning-with-Python/blob/master/OOP_in_ML/Class_MyLinearRegression.ipynb

THANKING YOU