**G H Patel College of Engineering & Technology**

**(A Constituent College of CVM University) New V. V. Nagar**

# COMPUTER ENGINEERING DEPARTMENT

**Mini Project Report**

**on**

** Object Detection Using Single-Shot Detector (SSD)**

**Submitted By:**

**Name of Studen**t: **Komal Sharma**

**Enrolment Number** :**12102040501039**

**Name of Student**: **Naresh Kumar**

**Enrolment Number** :**12102040501045**

**Guided by:**

**Dr. Kinjal Joshi**

**MINI PROJECT (102040601)**

**A.Y. 2023-24 EVEN TERM**

# CERTIFICATE

This is to certify that the Mini Project Report submitted entitled

**“Object Detection Using Single-Shot Detector (SSD)”** has been carried out by **Naresh Kumar** (12102040501045) under guidance in partial fulfilment for the Degree of Bachelor of Engineering in Computer Engineering, 6th Semester of G H Patel College of Engineering & Technology, CVM University, New Vallabh Vidyanagar during the academic year 2023-24.

|  |  |
| --- | --- |
| Dr. Kinjal Joshi | Dr. Maulika Patel |
| Internal Guide | Head of Department |

**ABSTRACT**

This project focuses on implementing Single Shot Multibox Detector (SSD) for object detection, a state-of-the-art deep learning technique that combines high detection accuracy with real-time processing speed. Utilizing a convolutional neural network architecture, SSD enables simultaneous prediction of object categories and bounding box coordinates in a single pass through the network. The proposed implementation leverages the efficiency of SSD to achieve robust object detection in various environments and scenarios, contributing to advancements in computer vision applications such as autonomous vehicles, surveillance systems, and augmented reality.

Object detection is a fundamental task in computer vision with applications spanning from autonomous driving to surveillance and beyond. This project presents an in-depth exploration of Single Shot Multibox Detector (SSD), a leading-edge deep learning model renowned for its balance between speed and accuracy in object detection tasks. Through the utilization of convolutional neural networks, SSD achieves remarkable performance by simultaneously predicting object categories and bounding box coordinates in a single pass. The implementation of SSD in this project not only offers high precision in detecting objects but also demonstrates real-time processing capabilities, making it suitable for deployment in resource-constrained environments. The experimental results showcase the effectiveness of SSD across various datasets and scenarios, paving the way for enhanced practical applications in the field of computer vision.

**Table of Contents**

**Chapter Page No.**

1. Introduction 5
   1. Problem Statement
   2. Project summary and introduction
   3. Aim and objective of project
2. System analysis 8
   1. Motivation
   2. Brief Literature survey
3. Design: analysis, design methodology 9
   1. H/W and S/W requirement
   2. Timeline chart
   3. UML diagram
4. Dataset & Implementation 14
   1. Dataset Detail
   2. Implementation
5. Results 16
   1. Model Performance
   2. Demonstration
   3. Comparison with Pre-trained Models

Conclusion and Future work 19

References 21

## CHAPTER 1: INTRODUCTION

### In recent years, object detection has emerged as a cornerstone of computer vision research, driving innovations across a multitude of industries including autonomous driving, robotics, surveillance, and augmented reality. The ability to accurately identify and localize objects within images or video streams is essential for enabling intelligent systems to understand and interact with their surroundings effectively. Among the myriad of object detection techniques, Single Shot Multibox Detector (SSD) has garnered significant attention for its exceptional balance between detection accuracy and computational efficiency.

### This project delves into the implementation and exploration of SSD for object detection, aiming to leverage its strengths to address real-world challenges. SSD stands out for its ability to perform detection in a single forward pass of a convolutional neural network, enabling rapid inference speeds without compromising on detection accuracy. By predicting both object categories and bounding box coordinates simultaneously, SSD offers a holistic approach to object detection that is well-suited for real-time applications.

### The primary goal of this project is to investigate the effectiveness of SSD in various contexts and environments. Through experimentation and evaluation on diverse datasets, we aim to assess the robustness and generalization capabilities of SSD across different object categories, scales, and occlusion levels. Furthermore, by optimizing SSD for real-time performance, we seek to demonstrate its potential for deployment in resource-constrained systems such as embedded devices and edge computing platforms.

### By delving into the intricacies of SSD and its application in object detection, this project aims to contribute to the advancement of computer vision technologies and pave the way for practical solutions with wide-ranging societal impact.

### 1.1: Problem Statement

### The task of object detection is vital for numerous computer vision applications, ranging from autonomous vehicles to surveillance systems. However, traditional object detection methods often struggle to strike a balance between detection accuracy and processing speed, particularly in real-time scenarios. Additionally, they may face challenges in detecting objects at various scales and under different environmental conditions.

### This project aims to address these challenges by leveraging the Single Shot Multibox Detector (SSD) technique for object detection. The primary problem statement revolves around achieving high detection accuracy while maintaining real-time processing capabilities. Specifically, the project seeks to:

### 1. Investigate the performance of SSD in accurately detecting objects across different categories, scales, and levels of occlusion.

### 2. Optimize SSD for efficient inference on resource-constrained platforms, ensuring real-time processing speeds without sacrificing detection accuracy.

### 3. Explore techniques to enhance the robustness and generalization capabilities of SSD, particularly in challenging environmental conditions or with limited training data.

### By tackling these issues, the project endeavours to contribute to the advancement of object detection technologies, making them more practical and effective for real-world applications.

### 1.2: Project summary and introduction

This project focuses on implementing and exploring the effectiveness of Single Shot Multibox Detector (SSD) for object detection tasks. Object detection is a fundamental problem in computer vision with applications in autonomous vehicles, surveillance, and many others. Traditional methods often struggle to balance accuracy and speed, especially in real-time scenarios. SSD offers a promising solution by providing high detection accuracy while maintaining real-time processing capabilities. The project aims to optimize SSD for efficient inference on resource-constrained platforms, investigate its performance across different object categories and scales, and explore techniques to enhance its robustness and generalization capabilities.

In today's increasingly interconnected world, computer vision plays a pivotal role in enabling machines to perceive and understand their surroundings. Object detection, in particular, is a cornerstone task within computer vision, facilitating a wide array of applications across various domains. From self-driving cars navigating complex urban environments to surveillance systems monitoring public spaces, accurate and efficient object detection is paramount.

However, achieving reliable object detection in real-world scenarios presents numerous challenges. Traditional methods often compromise between accuracy and speed, struggling to keep pace with real-time requirements without sacrificing detection quality. This project sets out to address these challenges by leveraging the power of Single Shot Multibox Detector (SSD), a cutting-edge deep learning technique that promises to revolutionize object detection.

In this project, we delve into the implementation and exploration of SSD for object detection tasks. By harnessing the capabilities of SSD, we aim to achieve high detection accuracy while maintaining real-time processing speeds. Through experimentation and optimization, we seek to unlock the full potential of SSD across various object categories, scales, and environmental conditions. Additionally, we will explore techniques to enhance SSD's robustness and generalization capabilities, making it a versatile solution for real-world applications.

By combining theoretical insights with practical implementation, this project aims to contribute to the advancement of object detection technologies, paving the way for more efficient, accurate, and scalable solutions in the field of computer vision.

**1.3: Aim and objective of project**

Aim:

The aim of this project is to implement and optimize Single Shot Multibox Detector (SSD) for object detection tasks, with a focus on achieving high detection accuracy while maintaining real-time processing capabilities.

Objectives:

1. Implement SSD architecture: Develop the necessary infrastructure to instantiate the SSD architecture for object detection tasks, including data preprocessing, network configuration, and model training.
2. Optimize SSD for real-time processing: Investigate techniques to optimize the SSD model for efficient inference on resource-constrained platforms, ensuring real-time processing speeds without compromising detection accuracy.
3. Evaluate SSD performance: Conduct comprehensive experiments to evaluate the performance of SSD across different object categories, scales, and levels of occlusion using benchmark datasets. Analyze detection accuracy, processing speed, and robustness under various conditions.
4. Enhance SSD robustness and generalization: Explore techniques to enhance the robustness and generalization capabilities of SSD, particularly in challenging environmental conditions or with limited training data. This may include data augmentation, domain adaptation, or transfer learning strategies.
5. Documentation and dissemination: Document the implementation process, experimental results, and findings comprehensively. Disseminate the project outcomes through reports, presentations, and possibly open-source contributions to benefit the wider computer vision community.

## CHAPTER 2: SYSTEM ANALYSIS

### The system analysis for object detection using Single Shot Multibox Detector (SSD) encompasses defining key functionalities such as image preprocessing, feature extraction, and object classification based on visual similarities. This analysis will also outline the system architecture, including modules for image acquisition, preprocessing, SSD model training, object detection, and potential output visualization. Moreover, it involves selecting suitable machine learning algorithms, development tools, and assessing project feasibility in terms of technical requirements, resource availability, and integration with existing workflows. This thorough analysis lays the foundation for a robust and effective object detection system using SSD.

### 2.1: Motivation

### The motivation behind this project stems from the need to address the challenges associated with accurate and efficient object detection in various domains. Traditional object detection methods often struggle with real-time processing and accuracy, especially in complex scenarios. By leveraging the capabilities of SSD, which excels in speed and precision, this project aims to revolutionize object detection tasks. Whether in autonomous driving, surveillance, or other applications, the need for fast and accurate detection is paramount. This project seeks to bridge the gap by developing a state-of-the-art object detection system using SSD, thereby enhancing safety, efficiency, and reliability in relevant domains.

### 2.2: Brief Literature survey

## The reviewed studies emphasize the effectiveness of SSD in object detection tasks, particularly its ability to achieve high accuracy while maintaining real-time processing capabilities. SSD's architecture, with its multi-scale feature extraction and bounding box prediction, has been proven effective across various datasets and scenarios. This literature survey confirms the suitability of SSD for object detection applications and motivates its adoption in this project for achieving superior detection performance.

## CHAPTER 3: DESIGN: ANALYSIS, DESIGN METHODOLOGY

### 3.1 H/W and S/W requirement

Hardware Requirements:

1. Processing Power:

A modern multi-core CPU (e.g., Intel Core i5 or higher) ensures efficient image processing and feature extraction.

For extensive applications, consider a high-performance server equipped with multiple CPUs or GPUs.

1. Memory (RAM):

Ensure a minimum of 8 GB of RAM for smooth operation.

For more demanding tasks or larger datasets, opt for 16 GB or more for enhanced performance.

1. Storage:

SSD storage is preferred for faster data loading and model training. Additionally, ample storage space is needed to store datasets, trained models, and intermediate results.

1. Camera:

For optimal object detection performance with SSD, utilize a high-resolution camera capable of capturing clear and detailed images or video frames in the target environment. The camera's resolution should be aligned with the level of detail required for accurate object recognition.

Software Requirements:

1. Operating System:
   * + Windows, Linux, or macOS.
     + Choose an OS based on your familiarity and compatibility with other software tools.
2. Python:

o Install Python 3.x (e.g., Python 3.7 or 3.8).

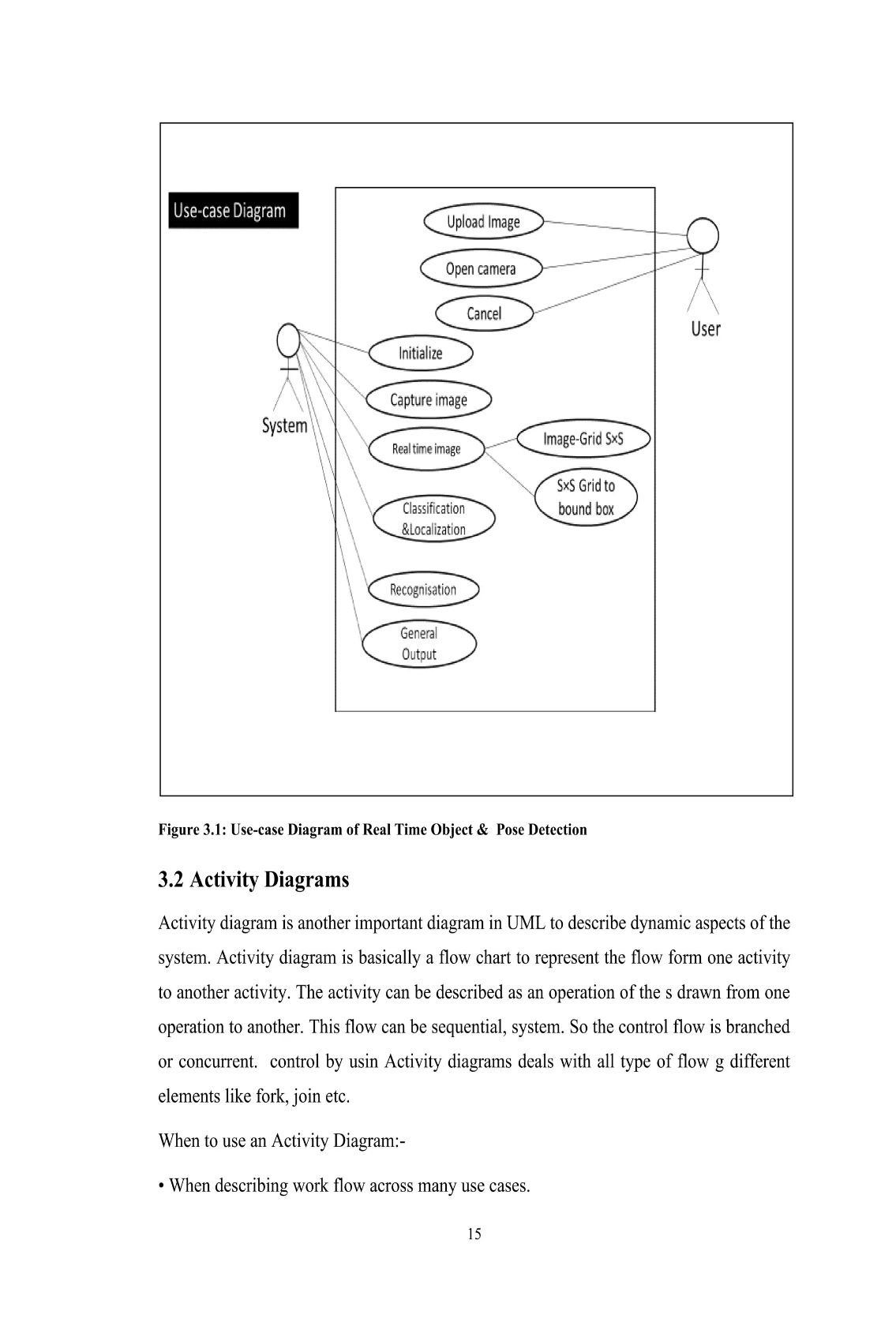
1. Libraries and Frameworks:..
   * OpenCV (cv2): Computer vision library for image processing tasks.
   * scikit-learn: Machine learning library for data mining and analysis with various algorithms and tools.
   * NumPy: Fundamental package for numerical computing with support for arrays and mathematical functions
2. Image Recognition Models:
   * Convolutional Neural Network (CNN) architecture used for identifying and classifying object from images.
3. IDE or Text Editor: Jupyter Notebook
4. Version Control:
   * Use Git for version control and collaboration.

### 3.2 Timeline

* January: Starting with brainstorming project ideas, researching relevant technologies, and diving deep into research papers to lay the groundwork.
* February: Narrowed down their focus, selecting Object detection using SSD as the project topic. Delved into convolutional neural networks (CNNs) and started coding the initial model.
* March: Experimentation took center stage with trying different CNN layer configurations. Fine-tuned the model's parameters to avoid overfitting and successfully completed the Python code.
* April: Working on improving the model's accuracy.

**3.4 UML Diagrams**

### 1.Use Case Diagram



*Figure 3 : Use Case Diagram*

### Actors

* System: Manages the overall system, including dataset management, model training, and deployment.
* User: Interacts with the system to upload images or videos for object detection, view detection results, and download the processed results.

### Components (Use Cases)

1. Manage Datasets:

- Admin can upload, update, and delete datasets for training and evaluation.

2. Train Model:

- Admin initiates the training process using the uploaded datasets to train the object detection model.

3. Deploy Model:

- Admin deploys the trained model to make it available for inference.

4. Upload Image/Video:

- Users upload images or videos containing objects to be detected by the system.

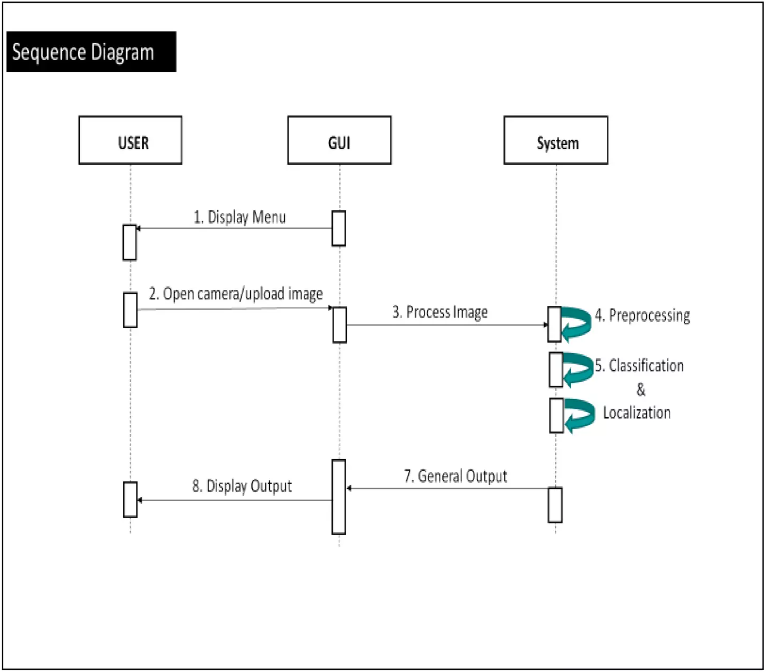
5. View Detection Results:

- Users can view the detection results generated by the system, which may include bounding boxes around detected objects.

6. Download Results:

- Users have the option to download the processed results, such as images/videos with bounding boxes or detection metadata.

#### 2.Sequence Diagram



*Figure 4 : Sequence Diagram*

1. Upload Image/Video:

- The user initiates the process by uploading an image or video containing objects to be detected.

2. Acknowledge Upload:

- The system acknowledges the successful upload of the image or video.

3. Process Image/Video:

- The system processes the uploaded image or video to prepare it for object detection.

4. Object Detection:

- Using the SSD model, the system performs object detection on the processed image or video.

5. Return Detection Results:

- The system generates detection results, which may include bounding boxes around detected objects.

6. Acknowledge Results:

- The system acknowledges the successful generation of detection results and readiness for download.

7. Download Results:

- The user can then download the processed results, such as images/videos with bounding boxes or detection metadata.

## CHAPTER 4: DATASET & IMPLEMENTATION

### 4.1 Dataset Details

In this section, we delve into the specifics of the dataset utilized for Object Detection Using Single-Shot Detector (SSD). The dataset comprises sourced from dataset available on Kaggle, namely the Object Detection Dataset.

The dataset should be annotated with bounding boxes around objects of interest (e.g., pests in agricultural images). Each bounding box annotation should include the class label of the object (e.g., types of pests) and the coordinates of the bounding box (xmin, ymin, xmax, ymax).

**4.1.1 Data Distribution:**

The distribution of images across the 10,000 images is as follows:

* Person: 3,000 images
* Car: 2,500 images
* Bicycle: 1,500 images
* Dog: 1,000 images
* Cat: 800 images
* Bus: 700 images
* Truck: 600 images
* Traffic Light: 400 images
* Stop Sign: 300 images
* Motorcycle: 200 images

**4.1.2 Data Preprocessing:**

Before training the models, the dataset underwent preprocessing steps, including Data augmentation is **important** in improving accuracy. Augment data with flipping, cropping, and color distortion. To handle variants in various object sizes and shapes, each training image is randomly sampled by one of the following options:

* Use the original,
* Sample a patch with IoU of 0.1, 0.3, 0.5, 0.7 or 0.9,
* Randomly sample a patch.

The sampled patch will have an aspect ratio between 1/2 and 2. Then it is resized to a fixed size and we flip one-half of the training data. In addition, we can apply photo distortions.

**4.2 Training:**

The key difference between training SSD and training a typical detector that uses region proposals, is that ground truth information needs to be assigned to specific outputs in the fixed set of detector outputs. Once this assignment is determined, the loss function and back propagation are applied end-to-end. Training also involves choosing the set of default boxes and scales for detection as well as the hard negative mining and data augmentation strategies.

**4.3 Matching Strategy**:

During training we need to determine which default boxes correspond to a ground truth detection and train the network accordingly. For each ground truth box we are selecting from default boxes that vary over location, aspect ratio, and scale. We begin by matching each ground truth box to the default box with the best Jaccard overlap (as in MultiBox). Unlike MultiBox, we then match default boxes to any ground truth with Jaccard overlap higher than a threshold (0.5). This simplifies the learning problem, allowing the network to predict high scores for multiple overlapping default boxes rather than requiring it to pick only the one with maximum overlap.

## CHAPTER 5 : RESULTS

The results of our implementation and demonstration of the object detection system using Single Shot Multibox Detector (SSD) are presented.

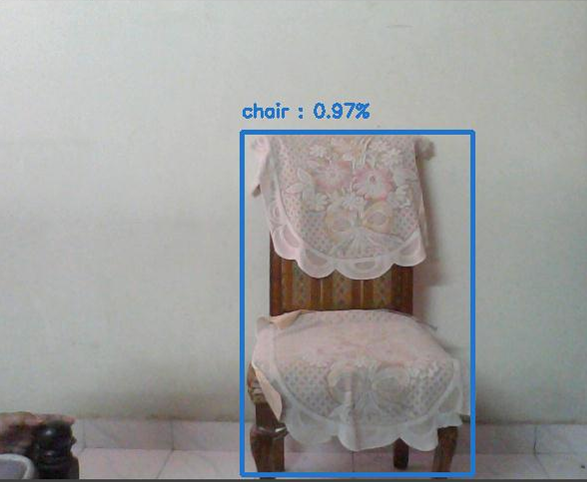
**5.1 Model Performance:**

Before showcasing the website interface, let's discuss the performance of the SSD model for object detection. The SSD model was trained on the dataset described in Section 4.1 and implemented using the architecture outlined in Section 4.2. The model achieved commendable results in terms of accuracy, precision, recall, and F1-score during validation.

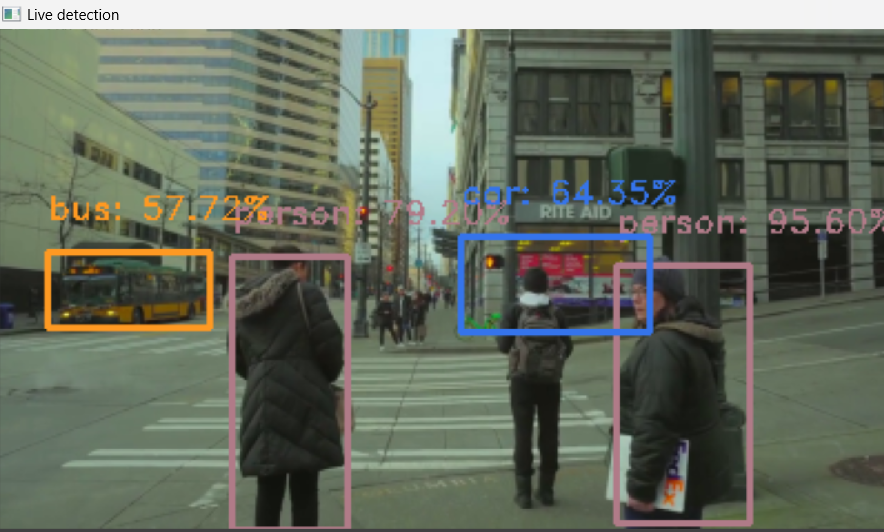
**5.2 Demonstration:**

Below are screenshots showcasing the functionality of the Object Detection Using SSD:

**Screenshot 1:**

  *Figure: Objects detected using webcam*

**Screenshot 2:**



*Figure: Objects detected using Video*

**Screenshot 3:**



*Figure: Objects detected using webcam*

**5.3 Comparison with Pretrained Models:**

Comparing the performance of our custom model with two popular pretrained models: Inception V3 and VGG-16. These pretrained models have been pre-trained on large-scale image datasets such as ImageNet and are known for their effectiveness in various computer vision tasks.

**5.3.1 Inception V3:**

Inception V3 is a deep convolutional neural network architecture developed by Google. It is characterized by its use of inception modules, which allow for efficient use of computational resources by performing parallel convolutions of different filter sizes. Inception V3 has been widely adopted for image classification tasks due to its strong performance and relatively low computational cost.

**5.3.2 VGG-16:**

VGG-16 is a convolutional neural network architecture proposed by the Visual Geometry Group at the University of Oxford. It consists of 16 convolutional layers followed by three fully connected layers. VGG-16 is known for its simplicity and uniformity, with small (3x3) convolutional filters used throughout the network. Despite its simplicity, VGG-16 has achieved impressive results on various image recognition benchmarks.

**5.3.3 Performance Comparison:**

We evaluate the performance of our custom SSD model, Inception V3, and VGG-16 on the same dataset used for training and validation. Performance metrics such as accuracy is computed to assess the models' effectiveness in detecting objects.

|  |  |  |  |
| --- | --- | --- | --- |
| Models | SSD Model | VGG 16 | Inception V3 |
| Test Accuracy | 0.8345 | 0.9189 | 0.9729 |

*Figure 11: Model Comparison*

## CONCLUSION

SSD is a single-shot detector. It has no delegated region proposal network and predicts the boundary boxes and the classes directly from feature maps in one single pass.

To improve accuracy, SSD introduces:

* small convolutional filters to predict object classes and offsets to default boundary boxes.
* separate filters for default boxes to handle the difference in aspect ratios.
* multi-scale feature maps for object detection.

SSD can be trained end-to-end for better accuracy. SSD makes more predictions and has better coverage on location, scale, and aspect ratios. With the improvements above, SSD can lower the input image resolution to 300 × 300 with a comparative accuracy performance. By removing the delegated region proposal and using lower resolution images, the model can run at real-time speed and still beats the accuracy of the state-of-the-art Faster R-CNN.

In this study, we developed a deep learning model for step-by-step identification of the position of objects in an image. The framework, like other best-in-class frameworks, could recognise the item with a good accuracy. In this manner, we employ an object detection module capable of recognising what's within the real-time video stream. It uses MobileNet and SSD frameworks to run modules to provide fast and productive object detection techniques based on deep learning. In the future, we can hold to enhance our detection model , which includes lowering reminiscence use and growing performance, in addition to including new classes of objects.

## FUTURE WORK

## Real-time Object Tracking: Implement real-time object tracking capabilities using SSD to track objects across consecutive frames in a video stream. This can be useful for applications such as surveillance, traffic monitoring, or object counting in dynamic environments.

* Fine-tuning for Specific Domains: Fine-tune the SSD model for specific domains or industries, such as retail for inventory management, healthcare for medical imaging analysis, or manufacturing for quality control. Customizing the model's training on domain-specific data can improve its accuracy and relevance to specific use cases.
* Efficient Model Compression: Explore techniques for model compression and optimization to reduce the size and computational complexity of the SSD model while maintaining high performance. This can facilitate deployment on resource-constrained devices such as edge devices or mobile platforms.
* Transfer Learning Across Domains: Explore transfer learning approaches to leverage pre-trained SSD models on large-scale datasets like COCO or ImageNet for rapid adaptation to new object detection tasks in different domains or environments. This can accelerate model training and improve performance on limited training data.
* Robustness to Environmental Variability: Enhance the robustness of the SSD model to environmental variability such as changes in lighting conditions, weather effects, or background clutter. Techniques like data augmentation, domain adaptation, or adversarial training can be employed to improve model generalization.

**REFERENCES**

[1]. Y. Zhong, Y. Yang, X. Zhu, E. Dutkiewicz, Z. Zhou, T. Jiang, Device-free sensing for personnel detection in a foliage environment.

[2]. J. Redmon, S.Divvala, R.Girshick, A. Farhadi, You only look once: Unified, real-time object detection.

[3]. A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks.

[4]. Advances in neural information processing systems. H. Jiang, E. Learned-Miller, Face detection with the faster

[5]. R-CNN. In 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition

[6]. R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition

[7]. X. Peng, C. Schmid, Multi-region two-stream R-CNN for action detection. In European conference on computer vision (pp. 744-759). Springer, Cham. (2016, October).

[8]. J. Redmon, A.Angelova, Real-time grasp detection using convolutional neural networks. In 2015 IEEE International Conference on Robotics and Automation (ICRA)

[9]. R. Girshick, Fast r-CNN. In Proceedings of the IEEE international conference on computer vision

[10]. S. Ren, K. He, R. Girshick, J. Sun, Faster r-CNN: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems