SOFTWARE REQUIREMENTS SPECIFICATION

For

Object Detection App

Version 1.0

Group 13

- Gourav Ahlawat
- Anjani Kumar
- Yash Vashistha
- Mullapudi Surya Karthik
- G Aakash

Submitted to:

Dr. Puneet Gupta

Assistant Professor

15 March 2023

Contents

1 Introduction

The purpose of this document is to provide a detailed description of the system requirements for the development of an application (1) and a website (both referred as application (1) from here on) that can take inputs from a live stream or stored media, detect objects in the input video or image, and create bounding boxes around the detected objects along with a label of the object detected.

1.1 Purpose

This system aims to provide an application (1) that can detect objects in images and video, create bounding boxes around those objects, and label them accordingly. The system should be able to handle both live streams and stored media as input. Object detection is breaking into a wide range of industries, with use cases ranging from personal security to productivity in the workplace. Object detection and recognition are applied in many areas of computer vision, including image retrieval, security, surveillance, automated vehicle systems, and machine inspection. Significant challenges stay in the field of object recognition. The possibilities are endless regarding future use cases for object detection. So, an object detection application (1) is a solution where we can get information about our surroundings.

1.2 Intended Audience

The intended audience for this system includes:

- Developers who are interested in building computer vision applications (1) that involve object detection.
- Engineers who are involved in designing and implementing real-time video processing systems.
- Researchers who are working on machine learning and computer vision algorithms for object detection.
- Businesses need to automate object detection tasks for their operations, such as surveillance, quality control, or inventory management.
- Individuals interested in exploring computer vision and machine learning technology and its applications in various fields.

1.3 Project Scope

The system will consist of a software application (1) that can run on a mobile device. The application (1) will be responsible for processing input images or video frames, running object detection algorithms on those frames, and generating bounding boxes and labels for the detected objects.

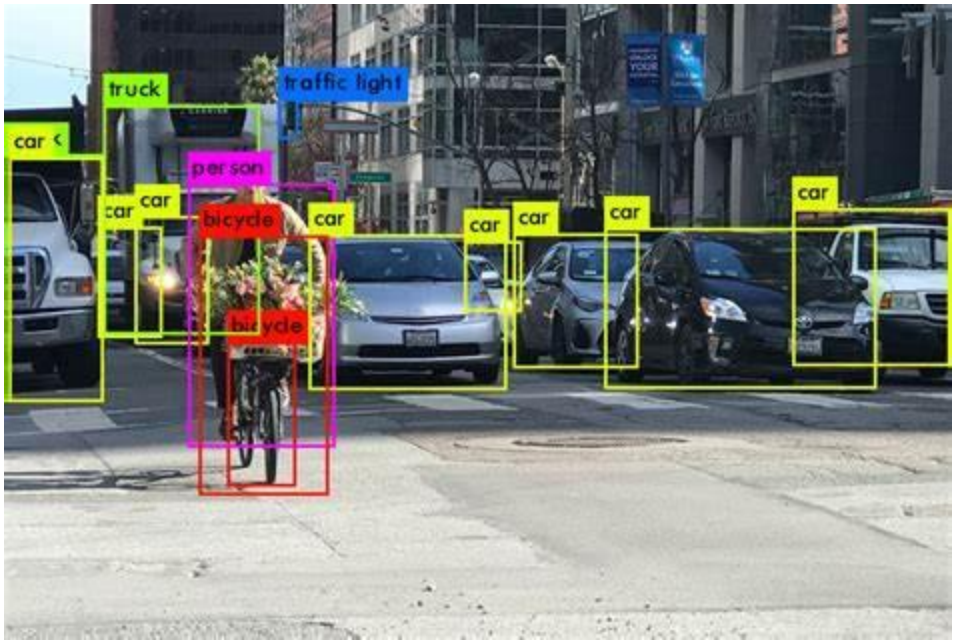

Figure 1.1

Figure 1.1 is an example of object detection in an autonomous car.

1.4 User Requirements

- The application (1) shall be able to take input from stored media.
- The application (1) shall be able to detect objects in the input video or image.
- The application (1) shall be able to create bounding boxes around the detected objects.
- The application (1) shall be able to label the detected objects.
- The application (1) shall be able to process input in real-time for live stream input.
- The application (1) shall provide an option to save the output video or image with bounding boxes and labels.

2 System Features

2.1 System Architecture

Modules: React Native, TensorFlow js, expo.

React Native: React Native is an open-source UI software framework Meta Platforms, Inc created. It is used to develop applications (1) for Android, Android TV, iOS, macOS, tvOS, Web, Windows, and UWP by enabling developers to use the React framework along with native platform capabilities.

Expo: Expo is an open-source framework for apps that run natively on Android, iOS, and the web. The expo npm package enables a suite of incredible features for React Native apps.

TensorFlow: TensorFlow is a free, open-source software library for machine learning and artificial intelligence. It can be used across various tasks but focuses on the training and inference of deep neural networks. It is used to build/train and get inferences from machine learning models.

Android Studio will act as the local server for React Native.

React Native will take the input from the user and show the output to the user.

TensorFlow takes the input from React Native, passes it through the model frame-wise, and produces the results, which are then processed and given to React Native for displaying to the end user.

2.2 System Requirements Specification

2.2.1 Functional Requirements:

Input

The application (1) should be able to take inputs from Livestream or stored media in the following formats:

- Livestream video
- Stored video
- Image file

Object Detection

The application (1) should be able to detect objects within the input video or image using object detection techniques, such as YOLO (Redmon, 2016), Faster R-CNN (R. Gavrilescu, 2018), or SSD (Wei Liu, 2016) we used COCO SSD : @license

* Copyright 2019 Google LLC. All Rights Reserved.

* Licensed under the Apache License, Version 2.0 (the "License");

* you may not use this file except in compliance with the License.

* You may obtain a copy of the License at

*

* http://www.apache.org/licenses/LICENSE-2.0

*

* Unless required by applicable law or agreed to in writing, software

* distributed under the License is distributed on an "AS IS" BASIS,

* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

* See the License for the specific language governing permissions and

* limitations under the License..

Bounding Box Generation

The application (1) should generate bounding boxes around the detected objects. These bounding boxes should be accurate and should cover the object entirely.

Labeling

The application (1) should be able to label the objects detected within the input video or image. The labels should be clear and easy to read.

Output

The application (1) should be able to output the detected objects along with their bounding boxes and labels. The output should be in the following formats:

- Video with bounding boxes and labels overlaid on the original video
- Image with bounding boxes and labels overlaid on the original image

2.2.2 Non-Functional Requirements

- The system shall have a user-friendly interface that is easy to use and understand.
- The system shall accurately detect objects and generate bounding boxes and labels.
- The system shall have low live stream latency to minimize the input and output lag.
- The system shall be able to handle multiple types of objects and labels.
- The system shall be able to run on a variety of mobile devices.
- The system shall have adequate security measures to protect the data and user privacy.

2.3 System Evolution

As technology continues to evolve, so will the system described. Here are some possible ways the system could grow over time:

- Improved Object Detection Models: As machine learning algorithms and neural networks continue to evolve, there may be newer and more efficient object detection models that can be used to improve the accuracy and speed of the system.
- Integration with Cloud Services: The system could be integrated with cloud services like AWS or Azure to use their machine learning services and improve performance.
- Real-time Object Tracking: The system could evolve to include real-time object tracking capabilities, where the system can track the movement of detected objects over time and provide additional insights to the user.
- Enhanced User Interface: The system's user interface could be improved to provide more intuitive controls and better visualization of the object detection results.
- Integration with other systems: The system could be integrated with other systems, such as security systems or robotics platforms, to provide object detection capabilities to these systems.
- Support for More Input Types: The system could be expanded to support more input types, such as audio or 3D models, to provide more comprehensive object detection capabilities.

Overall, the system has the potential to evolve and improve over time as technology advances and new object detection applications are discovered.

3 Appendices

3.1 Hardware Requirements:

- Mobile device: The application (1) will require a mobile device (smartphone or tablet) with sufficient processing power and memory to handle the video and image processing algorithms.
- Camera: The application (1) will require access to the mobile device's camera to capture live video feeds for object detection.
- Storage: The application (1) may require additional storage space on the mobile device to store recorded video or image data for object detection.

3.2 Software Requirements

- Mobile Device: The user will need a mobile device compatible with the operating system required by the application (1). For example, the user may need an Android or iOS device.
- Application Store: The user will need access to an application store, such as Google Play or Apple App Store, to download and install the application (1).
- Internet Connection: An active internet connection will be required for the user to use the application (1) to detect objects in a live video stream.

- Permissions: The user may need to grant the application (1) permission to access the device's camera, microphone, or storage, depending on the specific implementation of the system.

Overall, the software requirements for the end user of the mobile application (1) are relatively simple and familiar to most mobile applications (1). The main requirement is that the user has a compatible mobile device and internet access to the system's live-streaming features.

3.3 Server Hardware Requirements

- CPU: Object detection algorithms typically require significant computational resources, so you should choose a server with a powerful CPU. Ideally, you should choose a server with multiple CPU cores or a multi-processor system. The exact CPU requirements will depend on the complexity of your algorithm and the size of the videos being processed.
- GPU: If your object detection algorithm supports GPU acceleration, you should consider using a server with a powerful GPU. GPUs can significantly speed up object detection algorithms by performing many computations in parallel. Again, the exact GPU requirements will depend on the complexity of your algorithm and the size of the videos being processed.
- RAM: Object detection algorithms can be memory-intensive, so you should choose a server with enough RAM to handle the processing of large videos. The exact amount of RAM required will depend on the size of the videos being processed and the complexity of the algorithm.
- Storage: Your server should have enough storage to store the videos being processed and any intermediate results generated during the object detection process.
- Network: The network connection of your server should be fast enough to receive and process video data in real-time. The exact network requirements will depend on the size of the videos being processed and the number of concurrent users.
- Power Supply: Make sure the server has an adequate power supply to handle the hardware components and avoid system crashes or data loss.

4 Glossary

- Appendices: Additional sections at the end of a document that provide supplementary information or details.
- Application (1): A software program designed to perform a specific task or set of tasks.
- Applications: How a technology or system can be utilized in real-world situations or contexts.
- Application Store: An online marketplace where users can download and install applications (1).
- Audio: Sound data that can be used as input to the system.
- Bounding box: A rectangle or square surrounding an object detected in an image or video, used to indicate the object's location.
- Computer vision: The field of study focused on enabling computers to interpret and understand visual data from the world, such as images and videos.
- Cloud Services: Online services that provide access to computing resources and data storage over the internet.
- Deep neural network: A type of neural network designed to handle complex tasks by simulating the structure and function of the human brain.
- End User: The individual or group who uses the mobile application (1).
- Enhanced User Interface: An improved interface with better visualization and more intuitive controls.
- Faster R-CNN: Faster Region-based Convolutional Neural Network, a deep learning object detection algorithm.
- Hardware Requirements: The hardware components needed for the system to function properly, such as mobile devices, cameras, and storage.
- Input: Data that is provided to a system for processing.
- Integration: The process of combining two or more systems or technologies by working together.
- Label: A text annotation associated with an object detected in an image or video used to describe the object.
- Latency: The delay between an input signal and the corresponding output signal.

- Livestream: A real-time video feed that is broadcast over the internet.
- Machine learning: A type of artificial intelligence that allows systems to learn and improve from experience without being explicitly programmed.
- Memory: The storage space available on a device for data and application (1) files.
- Neural Networks: A machine learning algorithm inspired by the structure and function of the human brain.
- Object detection: Identifying and locating objects within an image or video.
- Processing Power: The capacity of a device to perform calculations and execute instructions quickly and efficiently.
- Real-time Object Tracking: The process of tracking the movement of detected objects over time in real time.
- Security Measures: Methods used to protect the data and user privacy of the system.
- Software Requirements: The specific software components needed for the system to function properly, such as mobile operating systems, application stores, and internet connection.
- SSD: Single Shot MultiBox Detector, a real-time object detection algorithm.
- Stored media: Video or image files saved on a device or server.
- System: A collection of interconnected components that work together to achieve a common goal.
- User-friendly Interface: An interface that is intuitive and easy for the end-user.
- YOLO: You Only Look Once, a real-time object detection system.