VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgaum-590018

A PROJECT REPORT **(15CSP85)** ON

"AID FOR THE VISUALLY IMPAIRED"

Submitted in Partial fulfillment of the Requirements for the Degree of

Bachelor of Engineering in Computer Science & Engineering

By

MOHAMMED FARDEEN (1CR16CS092)

RIZWAN A.M. (1CR16CS135)

SAHANA B. (1CR16CS143)

SALMAN MOHAMMED (1CR16CS145)

Under the Guidance of,

Dr. Jhansi Rani P

Professor, Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled **"AID FOR THE VISUALLY IMPAIRED"** carried out by **Mr. MOHAMMED FARDEEN**, USN **1CR16CS092**, **Mr. RIZWAN A.M.**, USN **1CR16CS135**, **Ms. SAHANA B.**, USN **1CR16CS143, Mr. SALMAN MOHAMMED**, USN **1CR16CS145,** bonafide students of CMR Institute of Technology, in partial fulfillment for the award of **Bachelor of Engineering** in Computer Science and Engineering of the Visveswaraiah Technological University, Belgaum during the year 2019-2020. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

Dr. Jhansi Rani P	**Dr. Prem Kumar Ramesh**	**Dr. Sanjay Jain**
Professor	**Professor & Head**	**Principal**
Dept. of CSE, CMRIT	**Dept. of CSE, CMRIT**	**CMRIT**

External Viva

Name of the examiners — Signature with date

1. _____

2. _____

DECLARATION

We, the students of Computer Science and Engineering, CMR Institute of Technology, Bangalore declare that the work entitled "**AID FOR THE VISUALLY IMPAIRED**" has been successfully completed under the guidance of Prof. Dr. Jhansi Rani P, Computer Science and Engineering Department, CMR Institute of technology, Bangalore. This dissertation work is submitted in partial fulfillment of the requirements for the award of Degree of Bachelor of Engineering in Computer Science and Engineering during the academic year 2019 - 2020. Further the matter embodied in the project report has not been submitted previously by anybody for the award of any degree or diploma to any university.

Place:

Date:

Team members:

MOHAMMED FARDEEN (1CR16CS092) _____

RIZWAN A.M. (1CR16CS135) _____

SAHANA B. (1CR16CS143) _____

SALMAN MOHAMMED (1CR16CS145) _____

ABSTRACT

The computer vision based assistive technology for the blind and visually impaired is a developing area. The assistive technology helps the visually impaired by providing them with a greater independence. By enabling them with their day-to-day activities like indoor and outdoor navigation, object detection.

Blind and visually impaired people can make use of the assistive device as proposed by our Project. The proposed system incorporates several assistance features like getting to know about the object in the line of vision of a person and the distance between the device and the object. All these details will be provided to the user as a voice feedback in a device which will be an asset for them according to their needs.

ACKNOWLEDGEMENT

We take this opportunity to express my sincere gratitude and respect to **CMR Institute of Technology, Bengaluru** for providing me a platform to pursue my studies and carry out my final year project

We have a great pleasure in expressing my deep sense of gratitude to **Dr. Sanjay Jain,** Principal, CMRIT, Bangalore, for his constant encouragement.

We would like to thank **Dr. Prem Kumar Ramesh,** HOD, Department of Computer Science and Engineering, CMRIT, Bangalore, who has been a constant support and encouragement throughout the course of this project.

We consider it a privilege and honor to express my sincere gratitude to my guide **Dr. Jhansi Rani P, Professor,** Department of Computer Science and Engineering, for the valuable guidance throughout the tenure of this review.

We also extend our thanks to all the faculty of Computer Science and Engineering who directly or indirectly encouraged me.

Finally, we would like to thank our parents and friends for all their moral support they have given me during the completion of this work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

This Project aims to help and aid the Visually Impaired, enabling them to be aware of their surroundings, to walk independently with minimum human intervention/support and thereby ensuring their safety.

This Project falls under the following domains:

- Computer Vision for Real Time Object Detection/Recognition
- Internet of Things for Sensors and Alerts

This would be implemented by the usage of Proximity Sensors, Real Time Object Detection/Recognition and Text-to-Speech Algorithms.

The proposed Solution is a mobile app which performs real time Object Detection/Recognition which interacts with a Proximity Sensor and notifies the user about the Object and it's proximity as speech.

Existing Systems make use of Raspberry Pi. However, we decided to implement a Mobile App instead, because the current generation Smartphones have much better processing power than a Raspberry Pi.

1.1 Relevance of the Project

Computer vision is a field that deals with acquiring, processing, examining and understanding the images. Output is in form of description or an interpretation or some quantitative measurements to obtain an understanding high dimensional data from the real world in order to produce numerical or symbolic information to make a decision.

The World Health Organization estimates that there are around 39 million blind people around the globe. Thus, there is a need for assistive and rehabilitative devices.

Generally, the distance can be measured using pulse echo and phase measurement method. Here, the distance is measured using pulse echo method. The ultrasonic module transmits a signal to the object, then receives echo signal from the object and produces output signal whose time period is proportional to the distance of the object.

An IoT based Obstacle Detection and Alerting System in Vehicles using Ultrasonic Sensor will be used as a reference for implementing proximity sensor module.

Table 1.1 Summary of the Approaches

	Size	**Range**	**Ultrasonic Frequency**
Using 8051 Microcontroller	Bulky and consumes more space	Up to 2.5 meters	40 kHz
NodeMCU	Small and compact	Up to 4 meters	

	Latency	Speed	Accuracy
MobileNet/ TensorFlow	Less Latency and Optimized	High-speed Outputs With A Higher Computation Time	Higher Accuracies
YOLO	More Latency when compared to MobileNet	Suitable for high-speed outputs	Not that High

1.1.1 Limitations of existing systems

1. YOLO imposes strong spatial constraints on bounding box predictions since each grid cell only predicts two boxes and can only have one class. This

spatial constraint limits the number of nearby objects that our model can predict.

2. Raspberry Pi has lesser computation power than mobile phones.
3. YOLO doesn't use softmax and instead uses logistic which reduces performance.
4. Raspberry Pi does not replace your computer, since the Ethernet is only a 10/100 and the processor is not as fast, it is time consuming to download and install software and is unable to do any complex multitasking.
5. Already proposed systems assume that the person is familiar with his surroundings to have a proper reaction.
6. Using the vibration patterns as feedback instead of audio format is not an excellent solution as the person can lose the sense of discrimination of such technique over the time; especially because there are multiple vibrations that need to be known by the user.

1.1.2 Advantages of proposed system

1. Older systems used raspberry pi, but using a mobile app makes the computation faster.
2. Advantages of MobileNets are: Reduced network size - 17MB, Reduced number of parameters - 4.2 million, Faster in performance and are useful for mobile applications and Small, low-latency convolutional neural networks.
3. SSD : Accuracy increases with the number of default boundary boxes. Multi-scale feature maps improve the detection of objects at different scale. Designing better default boundary boxes will help accuracy.
4. SSD attains a better balance between swiftness and precision. SSD also uses anchor boxes at a variety of aspect ratio comparable to Faster-RCNN and learns the off-set to a certain extent than learning the box. In order to hold the scale, SSD predicts bounding boxes after multiple convolutional layers.
5. Advantages of NodeMCU: Low cost, Integrated support for WIFI network, Reduced size of the board, Low energy consumption
6. The biggest advantage of using the 8051 is it's simplified architecture and instruction set. Newer processors run at around 20 mips, so computing power

isn't too much an issue, and the basic underlying architecture means that you can program it in assembly rather than a high level language.

1.2 Problem Statement

1.2.1 Goal

To Implement and deploy an Object Recognizer and Proximity Detector for the Visually Impaired using Computer Vision and IoT

1.2.2 Comparison with Existing Systems

Existing Systems make use of Raspberry Pi. However we decided to implement a Mobile App instead, because the current generation Smartphones have much better processing power than a Raspberry Pi.

1.3 Objective

- Real Time Object Recognition/Detection
- Retrieving Information about an Object's Proximity from the user as measured by the Microcontroller with the help of the Proximity Sensor
- Conversion of this Proximity and Object Information from Text to Speech and deliver it to the user, all in Real Time

1.4 Scope of the Project

To be usable in both an indoor and outdoor environment with proper lighting. The objects are detected in the users line of sight and at a specified minimum distance from the user.

The proposed Solution is a mobile app which performs real time Object Detection/Recognition which interacts with a Proximity Sensor and notifies the user about the Object and it's proximity as speech.

This project would help the Visually Impaired to navigate & understand their Surroundings better, and as a result, increasing their safety.

1.4.1 Constraints

- It won't function as efficiently in a poorly lit or extremely lit environment
- The algorithm draws a bounding box around the entire object so it can only recognize a complete object and not just a part of the object

1.5 Agile Methodology

Table 1.1 Agile Methodology

Story ID	Requirement description	User stories/Task	Description
1	Real Time Object Detection/Recognition	The user wants to identify the objects present around him/her in the line of vision for better understanding of the surrounding environment.	This requirement is fulfilled using computer vision algorithms. The algorithm being used is YOLO.
2	Determining the proximity of detected objects	The user wants to know the distance of the detected objects within the line of vision.	This requirement is implemented using a proximity sensor which works on the concept of echo-location.

Story ID	Requirement description	User stories/Task	Description
3	Conversion of Object and Proximity Information to speech	Since the user is visually impaired, the information about the detected objects and their proximity is delivered to the user as audio.	This requirement is fulfilled with the help of a microcontroller. The microcontroller obtains the readings from the sensor as a unit of distance and sends this information to an interface which will be developed which in turn performs text to speech conversion and delivers this to the user as audio.
4	Development of an interface	The user wants an easy handle to gain access to the algorithm and functionalities mentioned above via an interface.	This interface can be provided via a mobile application using an Android IDE.

CHAPTER 2

LITERATURE SURVEY

Literature survey includes the survey of place people and publications is context of Research. It is a phase where the researcher tries to know of what all the literature are related to one area of interest.

2.1 Overview

Two microcontrollers have been found suitable to be used in this project – 8051 and NodeMCU. The 8051 microcontroller is able to detect the proximity of an object within the range of 2.5 metres and NodeMCU which has a range of upto 4 metres.

The connection between the microcontroller and the android device is established via Bluetooth. MobileNET is now preferred over YOLO for Object Detection as YOLO has more latency.

2.2 Research Papers

We looked at multiple research papers that have been published related to the proposed project in order to gain information and make observations on the requirements, challenges and previously implemented methods. The research papers considered were in relation to Computer Vision, IoT based Ultrasonic Detection, Object Detection in Mobile Phones.

2.2.1 Computer Vision based Assistive Technology For Blind and Visually Impaired People

The Approach in [1] taken is a sensory substitution system for the totally blind which gives a visual experience through live camera views by image-to-sound renderings.

The proposed system incorporates text recognition, object identification, door detection and security system into a single device. The output is provided to the user through a speech system. The object detection module in the system will help the

blind by locating the lost object or while searching for a object. The system consists of a video camera capturing the scene.

During experimentation it was discovered that there are certain limitations to the system. While capturing the images sometimes problem arises with an unusable image, due to blurring caused by focus and improper holding of the camera and depth of the field. To a limit this can be solved by using a deblurring and an auto focus system. But few of the images were not good enough and thus had to be discarded.

System performance was also compromised with the processor speed. As the technology is developing exponentially this could be solved in the future, with a much better performing compact and cheaper processor. The architecture for computer vision based assistive technology is given in Fig. 2.1.

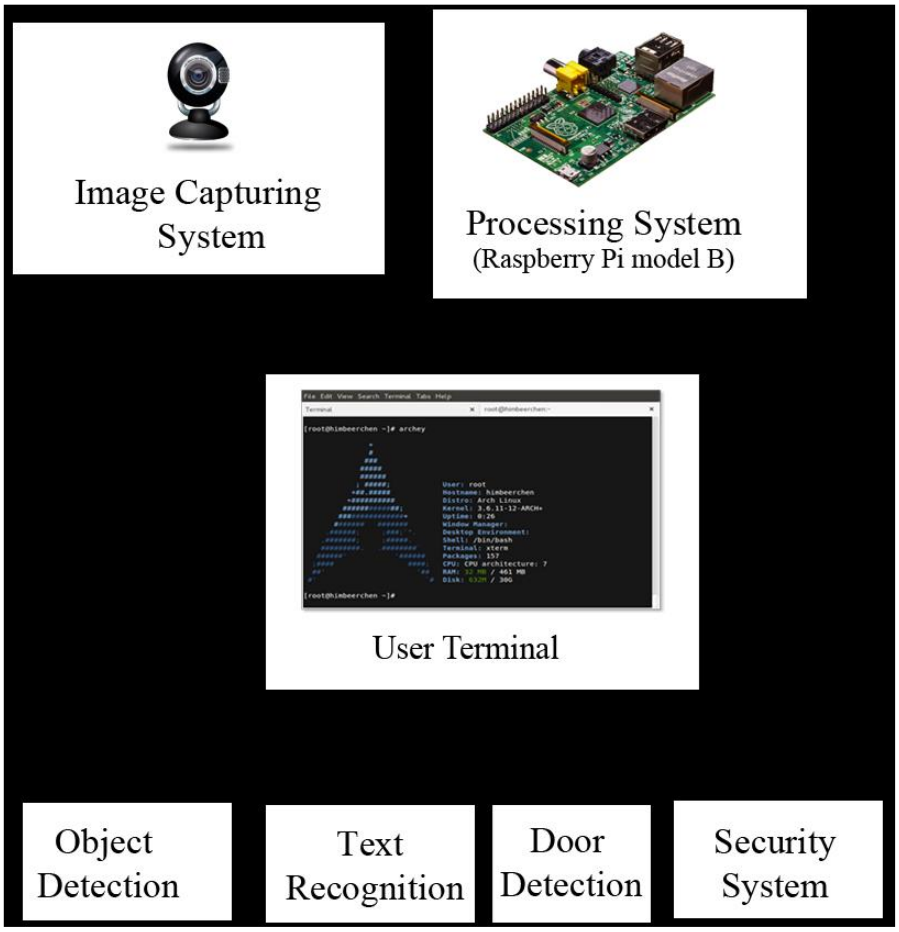

Fig 2.1: Architecture for computer vision based assistive technology

The paper proposed a cheaper and efficient assistive device for the blind and visually impaired. The device can be used for in real-time thus it can be used as a mobile assistive device for the blind and visually impaired. Further researches will be done to improve and also to include additional features to the system.

2.2.2 An IoT based Obstacle Detection and Alerting System in Vehicle using Ultrasonic Sensor

In today's world as the population increases day by day the numbers of vehicles also increase on the roads and highways. This results in more accidents that interns lead to the traffic jams and public should get help instantaneously.

Recent technological developments have made a lot of improvements in avoiding the accidents and ease human life. IoT has made promising improvements in developing some methods to avoid the accidents. The obstacles which are the main reasons for accidents can be detected and accidents can be avoided.

Distance measurement of an obstacle in front or by the side of a moving vehicle is required in the current traffic scenario. The devices used to measure the distance may be small or large and simple or complex. There are various different kinds of sensors available to perform this. In most of the applications, low cost, accuracy of the devices and speed to be considered. To measure the distance from obstacle, ultrasonic sensors play a vital role. Ultrasonic sensors are very flexible in distance measurement. They provide inexpensive solutions.

A simple approach for obstacle detection and alerting a user in using ultrasonic sensor and IOT is discussed here. This approach uses an ultrasonic sensor which is mounted on a vehicle with a servo motor which can rotate 0 to 180 degrees and can alert the person about the distance between the person and obstacle

This approach shown in [2] uses ultrasound sensors with simple data fusion technique. HC-SR04 is used as ultrasonic sensor for distance measurement for distances from 2cm to 400cm.

The ultrasonic sensor transmits sound waves and receives sound reflected from an object. When ultrasonic waves are incident on an object, reflection of the energy takes place up to 180 degrees. If the obstacle is near the incident energy is reflected back very soon. If the object is far then the reflected signal will take some small amount of time to reach the receiver. The Methodology for this approach is shown in Fig 2.2.

In Conclusion, An IoT based approach for detecting an obstacle and alerting system is discussed in this paper. The methodology employs a distance measurement sensor namely ultrasonic sensor which detects the range of the obstacle and Arduino processes it to alert the vehicle users to avoid the possible accidents.

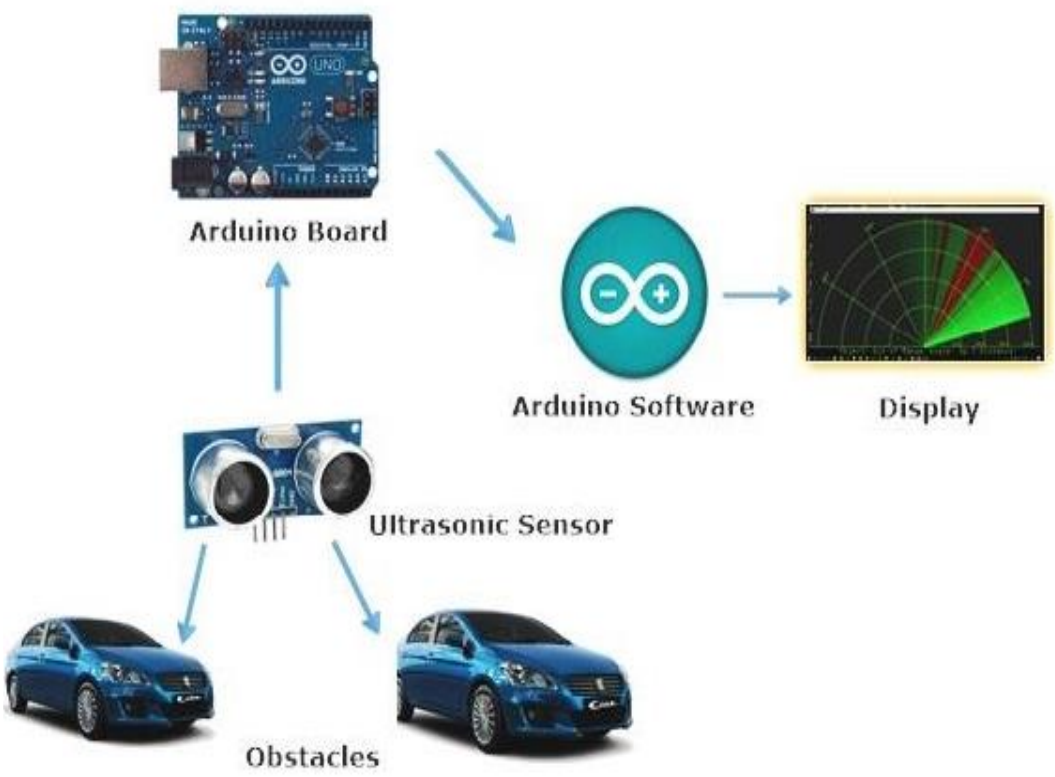

Fig 2.2 Methodology

2.2.3 Object Detection and Its Implementation on Android Devices, Stanford University

Deep learning based object detection has been very successful in recent years. Especially the CNN (convolutional neural network) model has significantly improved the recognition accuracy on large data-sets.

As mobile computing devices are very popular and comparatively powerful, people want to embrace the benefits of CNN with their mobile devices. However, to enable their mobile application, new CNN architectures need to be developed to overcome the above issues. Also, most deep learning frameworks have provided interface for mobile platforms, including iOS and Android. Object Detection in Android devices is elaborated in [3].

The input to the app is a camera stream, then an inference interface is used to call the model pretrained and installed on the android device to produce the same type information (bounding box coordinates, confidence score and class scores).

For the implementation of CNN model in Android device, we used the interface provided by ”Tensorflow Camera Demo”. First, the CNN model parameters need to be trained and saved into a proto buffer file. Android app can load tensor with values, run the graph and read tensor output values. The app is designed with a streaming video from the camera, and each image frame is passed to the CNN model for object detection. And then the detected results are marked with boxes in real time.

A difficult task would be to recognize partially blocked objects, especially for objects with most of their surfaces blocked

The interface between Tensorflow and Android is still not perfect as the latency caused by interface is longer than the actual computation time in the graph. In addition, there is no documentation for the interface. Tensorflow Lite for mobile platform can be used to significantly improve the model. Android App Architecture is shown in Fig 2.3.

R-CNN can be regarded as a cornerstone for the development of CNN for object detection. A large amount of work is based on this architecture and achieves great accuracy. However, a recent work shows that CNN based object detection can be even faster. YOLO (You Only Look Once) is such an architecture integrating region proposition and object classification into one single stage, which significantly contributes to simplification of the pipeline of object detection, as well as reduction of the total computation time.

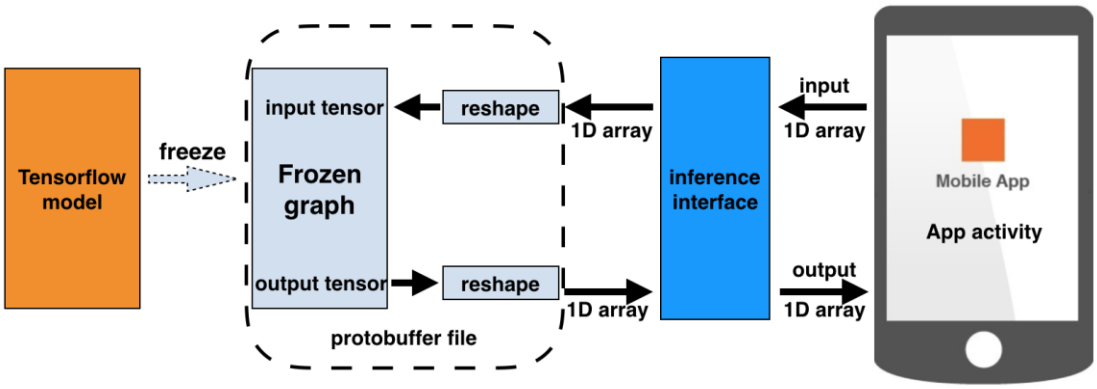

Fig 2.3 Android App Architecture

2.2.4 YOLOv3: An Incremental Improvement

You only look once (YOLO) is an object detection system targeted for real-time processing. [4]

YOLO divides the input image into an **S×S** grid. Each grid cell predicts only **one** object. Each grid cell predicts a fixed number of boundary boxes. In this example, the yellow grid cell makes two boundary box predictions (blue boxes) to locate where an object is. YOLO's network design is shown in Fig 2.4.

However, the one-object rule limits how close detected objects can be. For that, YOLO does have some limitations on how close objects can be.

For each grid cell,

- it predicts **B** boundary boxes and each box has one **box confidence score**,
- it detects **one** object only regardless of the number of boxes B,
- it predicts **C conditional class probabilities** (one per class for the likeliness of the object class).

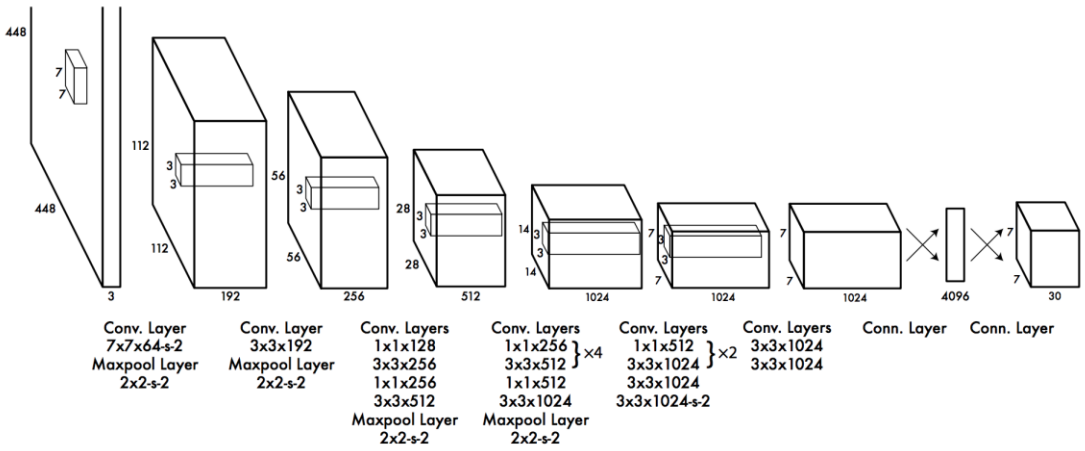

Fig 2.4 Network Design

YOLO has 24 convolutional layers followed by 2 fully connected layers (FC). Some convolution layers use 1 × 1 reduction layers alternatively to reduce the depth of the features maps. For the last convolution layer, it outputs a tensor with shape (7, 7, 1024). The tensor is then flattened. Using 2 fully connected layers as a form of linear regression, it outputs 7×7×30 parameters and then reshapes to (7, 7, 30), i.e. 2 boundary box predictions per location.

Benefits of YOLO

- Fast. Good for real-time processing.
- Predictions (object locations and classes) are made from one single network. Can be trained end-to-end to improve accuracy.

- YOLO is more generalized. It outperforms other methods when generalizing from natural images to other domains like artwork.
- Region proposal methods limit the classifier to the specific region. YOLO accesses to the whole image in predicting boundaries. With the additional context, YOLO demonstrates fewer false positives in background areas.
- YOLO detects one object per grid cell. It enforces spatial diversity in making predictions.

YOLOv2

SSD is a strong competitor for YOLO which at one point demonstrates higher accuracy for real-time processing. Comparing with region based detectors, YOLO has higher localization errors and the recall (measure how good to locate all objects) is lower. YOLOv2 is the second version of the YOLO with the objective of improving the accuracy significantly while making it faster

Classification

Datasets for object detection have far fewer class categories than those for classification. To expand the classes that YOLO can detect, YOLO proposes a method to mix images from both detection and classification datasets during training. It trains the end-to-end network with the object detection samples while backpropagates the classification loss from the classification samples to train the classifier path. This approach encounters a few challenges:

- How do we merge class labels from different datasets? In particular, object detection datasets and different classification datasets uses different labels.
- Any merged labels may not be mutually exclusive, for example, *Norfolk terrier* in ImageNet and *dog* in COCO. Since it is not mutually exclusive, we can not use softmax to compute the probability.

Hierarchical classification

Without going into details, YOLO combines labels in different datasets to form a tree-like structure **WordTree**. The children form an is-a relationship with its parent like biplane is a plane. But the merged labels are now not mutually exclusive. The hierarchical WordTree is shown in Fig 2.5.

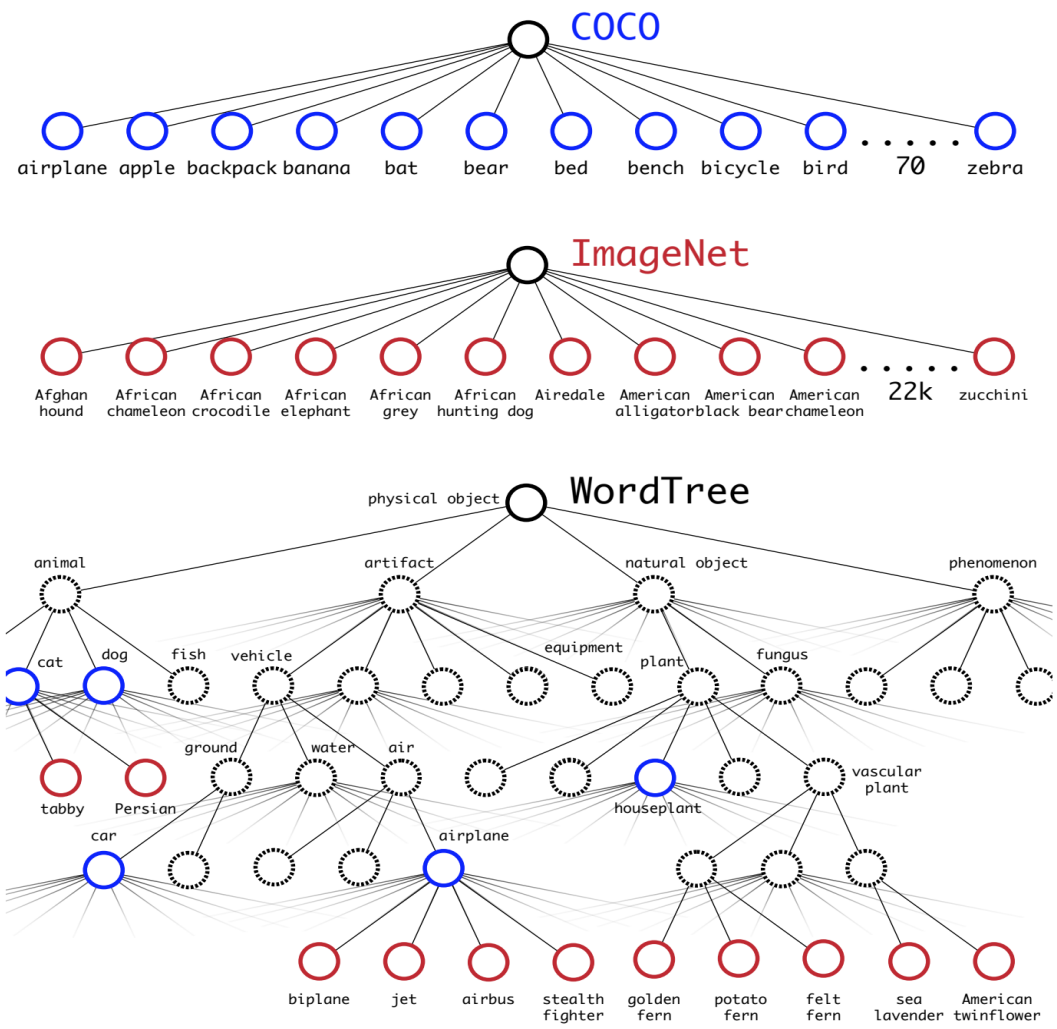

Fig 2.5 Combining COCO and ImageNet labels to a hierarchical WordTree

Most classifiers assume output labels are mutually exclusive. It is true if the output are mutually exclusive object classes. Therefore, YOLO applies a softmax function to convert scores into probabilities that sum up to one. YOLOv3 uses multi-label classification. For example, the output labels may be “pedestrian” and “child” which

are not non-exclusive. (the sum of output can be greater than 1 now.) YOLOv3 replaces the softmax function with independent logistic classifiers to calculate the likeliness of the input belongs to a specific label. Instead of using mean square error in calculating the classification loss, YOLOv3 uses binary cross-entropy loss for each label. This also reduces the computation complexity by avoiding the softmax function.

YOLOv3 predicts an objectness score for each bounding box using logistic regression. YOLOv3 changes the way in calculating the cost function. If the bounding box prior (anchor) overlaps a ground truth object more than others, the corresponding objectness score should be 1. For other priors with overlap greater than a predefined threshold (default 0.5), they incur no cost. Each ground truth object is associated with one boundary box prior only. If a bounding box prior is not assigned, it incurs no classification and localization lost, just confidence loss on objectness.

2.2.5 MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, Google Inc.

MobileNets are based on a streamlined architecture that uses depth-wise separable convolutions to build light weight deep neural networks.

MobileNets primarily focus on optimizing for latency but also yield small networks.

The MobileNet model elaborated in [5] is based on depthwise separable convolutions which is a form of factorized convolutions which factorize a standard convolution into a depthwise
convolution and a 1×1 convolution called a pointwise convolution. For MobileNets the depthwise convolution applies a single filter to each input channel. The pointwise convolution then applies a 1×1 convolution to combine the outputs the depthwise convolution. A standard convolution both filters and combines inputs into a new set of outputs in one step. The depthwise separable convolution splits this into two layers, a separate layer for filtering and a separate layer for combining. This factorization has the effect of drastically reducing computation and model size.

A standard convolutional layer takes as input a DF ×DF × M feature map F and produces a DF × DF × N feature map G where DF is the spatial width and height of a square input feature map1, M is the number of input channels (input depth), DG is the spatial width and height of a square output feature map and N is the number of output channel (output depth).

The standard convolutional layer is parameterized by convolution kernel K of size DK ×DK ×M×N where DK is the spatial dimension of the kernel assumed to be square and M is number of input channels and N is the number of output channels as defined previously.

The output feature map for standard convolution assuming stride one and padding is computed as:

$G_{k,l,n} = X_{i,j,m} K_{i,j,m,n} \cdot F_{k+i-1,l+j-1,m}$

Standard convolutions have the computational cost of:

$D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F$

where the computational cost depends multiplicatively on the number of input channels M, the number of output channels N the kernel size Dk × Dk and the feature map size DF × DF . MobileNet models address each of these terms and their interactions. First it uses depthwise separable convolutions to break the interaction between the number of output channels and the size of the kernel.

The standard convolution operation has the effect of filtering features based on the convolutional kernels and combining features in order to produce a new representation.

The filtering and combination steps can be split into two steps via the use of factorized convolutions called depthwise separable convolutions for substantial reduction in computational cost.

Depthwise separable convolution are made up of two layers: depthwise convolutions and pointwise convolutions.

We use depthwise convolutions to apply a single filter per each input channel (input depth). Pointwise convolution, a simple 1×1 convolution, is then used to create a

linear combination of the output of the depthwise layer. MobileNets use both batchnorm and ReLU nonlinearities for both layers.

Depthwise convolution with one filter per input channel
(input depth) can be written as:
G,k,l,m =Xi,jKˆ
i,j,m · Fk+i−1,l+j−1,m (3)
where Kˆ is the depthwise convolutional kernel of size DK × DK × M where the mth filter in Kˆ is applied to the mth channel in F to produce the mth channel of the filtered output feature map Gˆ .
Depthwise convolution has a computational cost of:
DK · DK · M · DF · DF
Depthwise convolution is extremely efficient relative to
standard convolution. However it only filters input channels, it does not combine them to create new features. So an additional layer that computes a linear combination of the output of depthwise convolution via 1 × 1 convolution is needed in order to generate these new features.
The combination of depthwise convolution and 1 × 1 (pointwise) convolution is called depthwise separable convolution.
Depthwise separable convolutions cost:
DK · DK · M · DF · DF + M · N · DF · DF
which is the sum of the depthwise and 1 × 1 pointwise convolutions.
By expressing convolution as a two step process of filtering and combining we get a reduction in computation of:
DK · DK · M · DF · DF + M · N · DF · DF
DK · DK · M · N · DF · DF=1/N+1/D^2K

MobileNet can also be deployed as an effective base net-work in modern object detection systems. SSD Architecture is shown in Fig 2.6.

MobileNet achieves comparable results to other networks with only a fraction of computational complexity and model size. Models are implemented using TensorFlow. Sample outputs are shown in Fig 2.7.

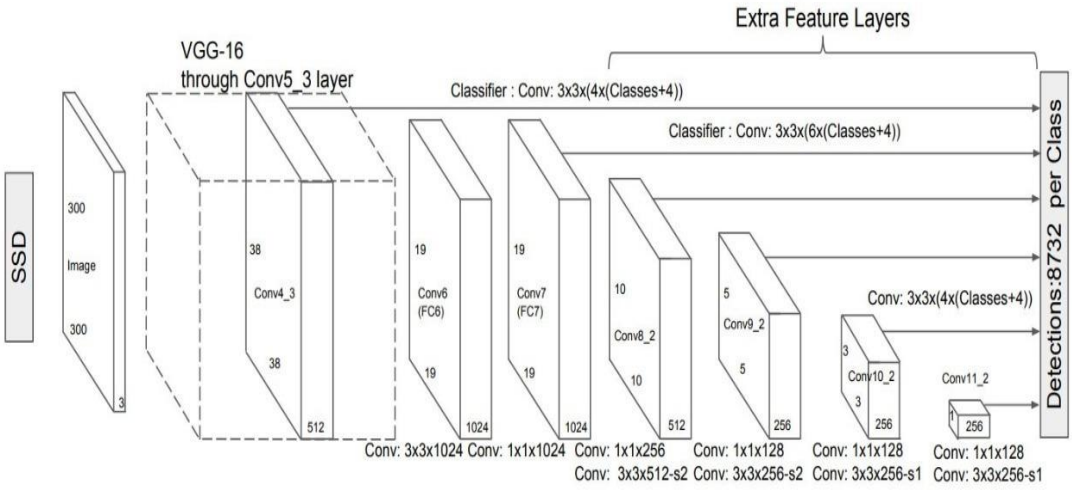

Fig 2.6 SSD Architecture

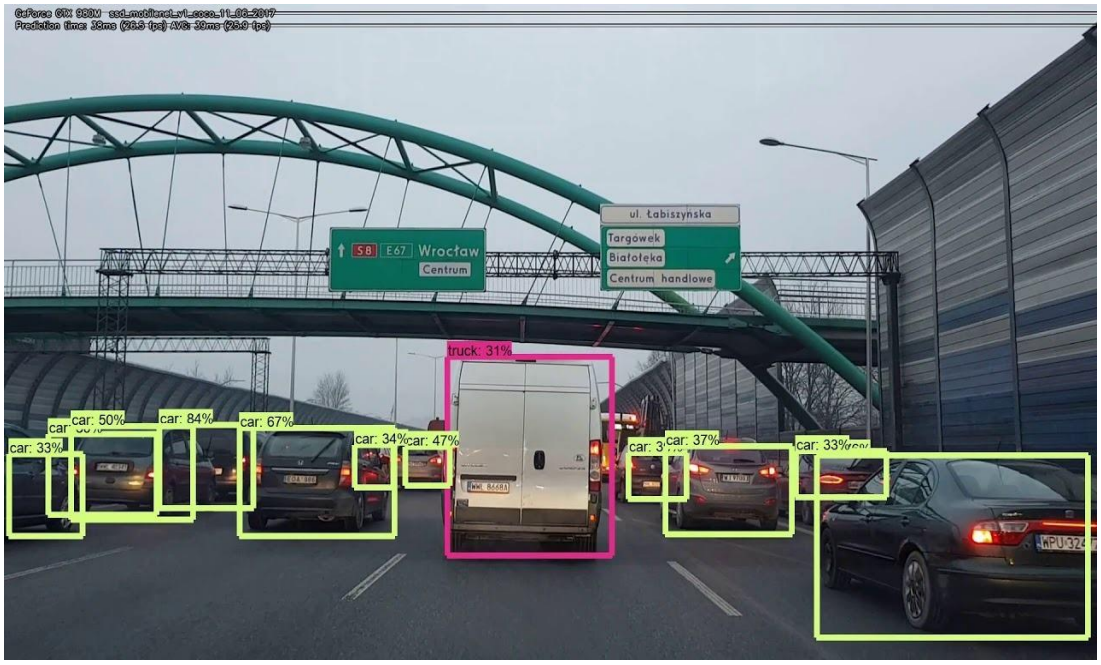

Fig 2.7 Sample Output using SSD

2.2.6 Ultrasonic Range Finder using 8051

The Approach in [6] implements a range finder using 8051 controller. Generally, the distance can be measured using pulse echo and phase measurement method. Here, the distance is measured using pulse echo method. The HC –SR04 ultrasonic module transmits a signal to the object, then receives echo signal from the object and

produces output signal whose time period is proportional to the distance of the object. The mechanism of the ultra sonic sensor is similar to the RADAR (Radio Detection and Ranging).This circuit calculates the distance of the object based on the speed of the sound wave at normal temperature and displays the distance on SSD.

Ultrasonic rangefinder can find the distance between an object from itself using ultrasonic sensor. HC – SR04 Ultrasonic Module works on the principle of SONAR and is designed to measure the range of the object in small embedded projects. It offers excellent range detection with high accuracy and stable readings. The operation of the module is not affected by the sunlight or black material. The ultrasonic rangefinder can measure distances up to 2.5 meters at accuracy of 1 cm.

The ultrasonic module sends a signal to the object, then picks up its echo and outputs a wave form whose time period is proportional to the distance. The objective of this research work is to measure the obstacle distance from reference. This system used in automotive parking sensors and obstacle warning systems and can be used in terrain monitoring robots. This circuit finds a lot of application in projects like automotive parking sensors, obstacle warning systems, terrain monitoring robots, industrial distance measurements etc. Ultrasonic sensor HC-SR04 has stable performance and high ranging accuracy that makes it a popular module in electronic market .Compared to the Sharp IR ranging module, HC-SR04 is more inexpensive than it. But it has the same ranging accuracy and longer ranging distance.

2.2.7 Microsoft COCO: Common Objects in Context

Microsoft introduces a new large-scale dataset called COCO (Common Objects in Context) that addresses three core research problems in scene understanding: detecting non-iconic views (or non-canonical perspectives) of objects, contextual reasoning between objects and the precise 2D localization of objects [7]. For many categories of objects, there exists an iconic view. For example, when performing a web-based image search for the object category “bike,” the top-ranked retrieved examples appear in profile, unobstructed near the center of a neatly composed photo.

Image Classification: The task of object classification requires binary labels indicating whether objects are present in an image. Early datasets of this type comprised images containing a single object with blank backgrounds, such as the MNIST handwritten digits or COIL household objects . Caltech 101 and Caltech 256 marked the transition to more realistic object images retrieved from the internet while also increasing the number of object categories to 101 and 256, respectively. Popular datasets in the machine learning community due to the larger number of training examples, CIFAR-10 and CIFAR-100 offered 10 and 100 categories from a dataset of tiny 32×32 images . While these datasets contained up to 60,000 images and hundreds of categories, they still only captured a small fraction of our visual world.

Object detection Detecting an object entails both stating that an object belonging to a specified class is present, and localizing it in the image. The location of an object is typically represented by a bounding box. Early algorithms focused on face detection using various ad hoc datasets. Later, more realistic and challenging face detection datasets were created. Another popular challenge is the detection of pedestrians for which several datasets have been created. The Caltech Pedestrian Dataset contains 350,000 labeled instances with bounding boxes. COCO dataset statistics are shown in Fig 2.8.

Image Collection

Common Object Categories

The selection of object categories is a non-trivial exercise. The categories must form a representative set of all categories, be relevant to practical applications and occur with high enough frequency to enable the collection of a large dataset. Other important decisions are whether to include both “thing” and “stuff” categories and whether fine-grained and object-part categories should be included. “Thing” categories include objects for which individual instances may be easily labeled (person, chair, car) where “stuff” categories include materials and objects with no clear boundaries (sky, street, grass). Since we are primarily interested in precise localization of object instances, we decided to only include “thing” categories and not

"stuff." However, since "stuff" categories can provide significant contextual information, we believe the future labeling of "stuff" categories would be beneficial.

Non-iconic Image Collection

Given the list of object categories, the next goal was to collect a set of candidate images. We may roughly group images into three types, iconic-object images, iconic-scene images and non-iconic images. Typical iconic-object images have a single large object in a canonical perspective centered in the image. Iconic-scene images are shot from canonical viewpoints and commonly lack people. Iconic images have the benefit that they may be easily found by directly searching for specific categories using Google or Bing image search. While iconic images generally provide high quality object instances, they can lack important contextual information and non-canonical viewpoints. Fig 2.9 shows samples of annotated images in the MS COCO dataset.

Image Annotation

Since there were 2.5 million object instances, there was a need for efficient annotation method. Microsoft employed the following methods for this.

- Category Labeling
- Instance Spotting
- Instance Segmentation
- Annotation Performance Analysis

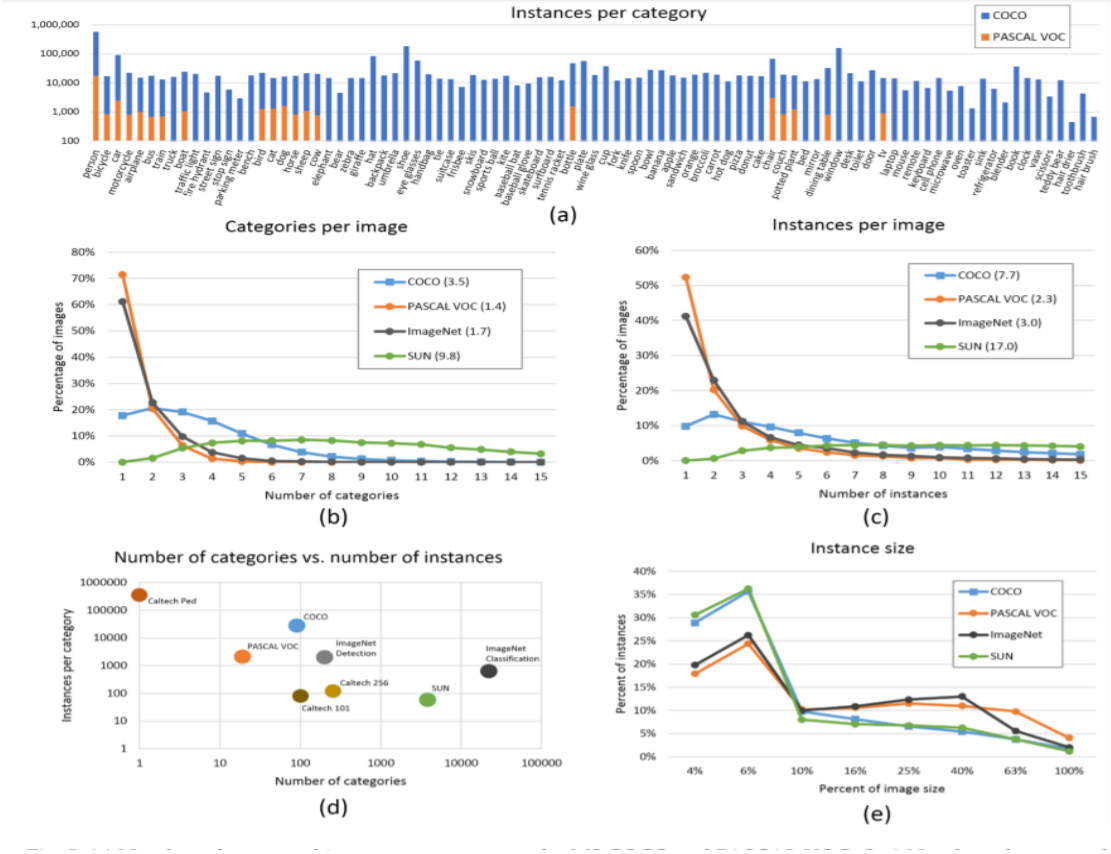

Fig 2.8 Dataset Statistics

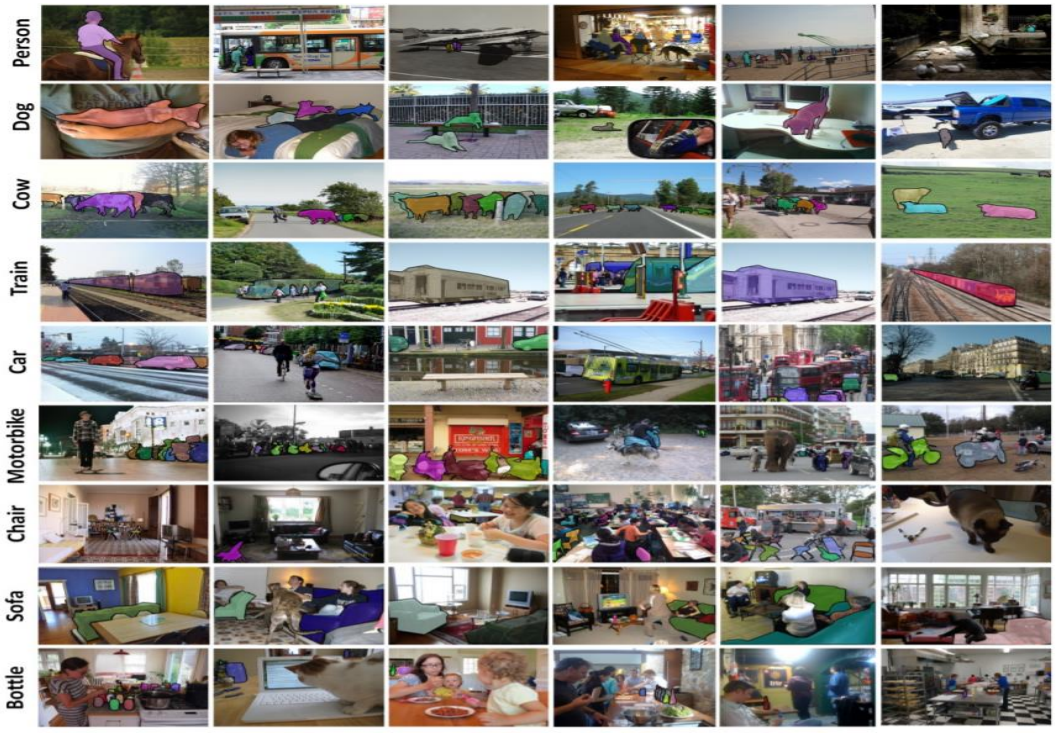

Fig 2.9: Samples of annotated images in the MS COCO dataset.

CHAPTER 3

REQUIREMENTS SPECIFICATION

A requirements specification is a description of a software system to be developed. Software requirements specification is a rigorous assessment of requirements before the more specific system design stages, and its goal is to reduce later redesign. It should also provide a realistic basis for estimating product costs, risks, and schedules

3.1 Functional Requirements

1 As soon as the device is turned on the connection should be established between the phone and the device.

2 The information received via the echo detection which is called pulse echo detection by the proximity sensor should be processed by the micro controller with in a given time period.

3 The message produced by the micro controller is sent to the mobile app which in turn converts this text information into speech, so proper Implementation of text to speech algorithms are required.

4 The app should be provided with necessary permissions to access the camera and bluetooth for the effective operation.

5 The usage of the app or the device shouldn't affect the software or hardware performance.

6 The operation of the device should be particular to the connected mobile and it shouldn't interfere with the other devices or the any other mobile device

3.2 Non-Functional Requirements

3.2.1 Usability:

1. The app should be easy to use. It should be developed to be used in hands free mode. The functioning of the app shouldn't affect the other running apps in the background.
2. The device should notify the used in a timely manner so it's convenient for the user

3.2.2 Reliability:

1. This software will be developed with machine learning, feature engineering and deep learning techniques. So, in this step there is no certain reliable percentage that is measurable.
2. As the user is completely dependent on the app for the movement in the environment, the app should be reliable and shouldn't crash very often. Even if it crashes a backup solution should be provided.

3.2.2 Performance:

1. Response time should be as low as possible.
2. The processing power of the device should be to an extent that it can identify the object , process the information from the micro controller and convert that text information into speech and all this tasks should be thinking instantaneously.

3.2.4 Supportability:

The System should require Java, XML, Python, Tensorflow, embedded C, Bluetooth connection.

3.3 Hardware Requirements

- Processor: Qualcomm snapdragon 600 series and above/Mediatek MT6223 and above

- RAM: 4 GB and above
- Camera: 8 MP and above
- Bluetooth: Version 3 and above
- Micro controller: NodeMCU
- Ultrasonic sensor: HC-SR04

3.4 Software Requirements

- Operating System: Android Lollipop and above/XTos
- Libraries: Tensorflow Lite
- Tools: Spyder/Jupyter/Anaconda/Android Studio
- Languages: Java, Python, XML, Embedded C

CHAPTER 4

SYSTEM ANALYSIS AND DESIGN

Systems development is 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.

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 Analysis and Design (SAD) mainly focuses on −

- Systems
- Processes
- Technology

4.1 System Architecture

The System Architecture is shown in Fig 4.1.

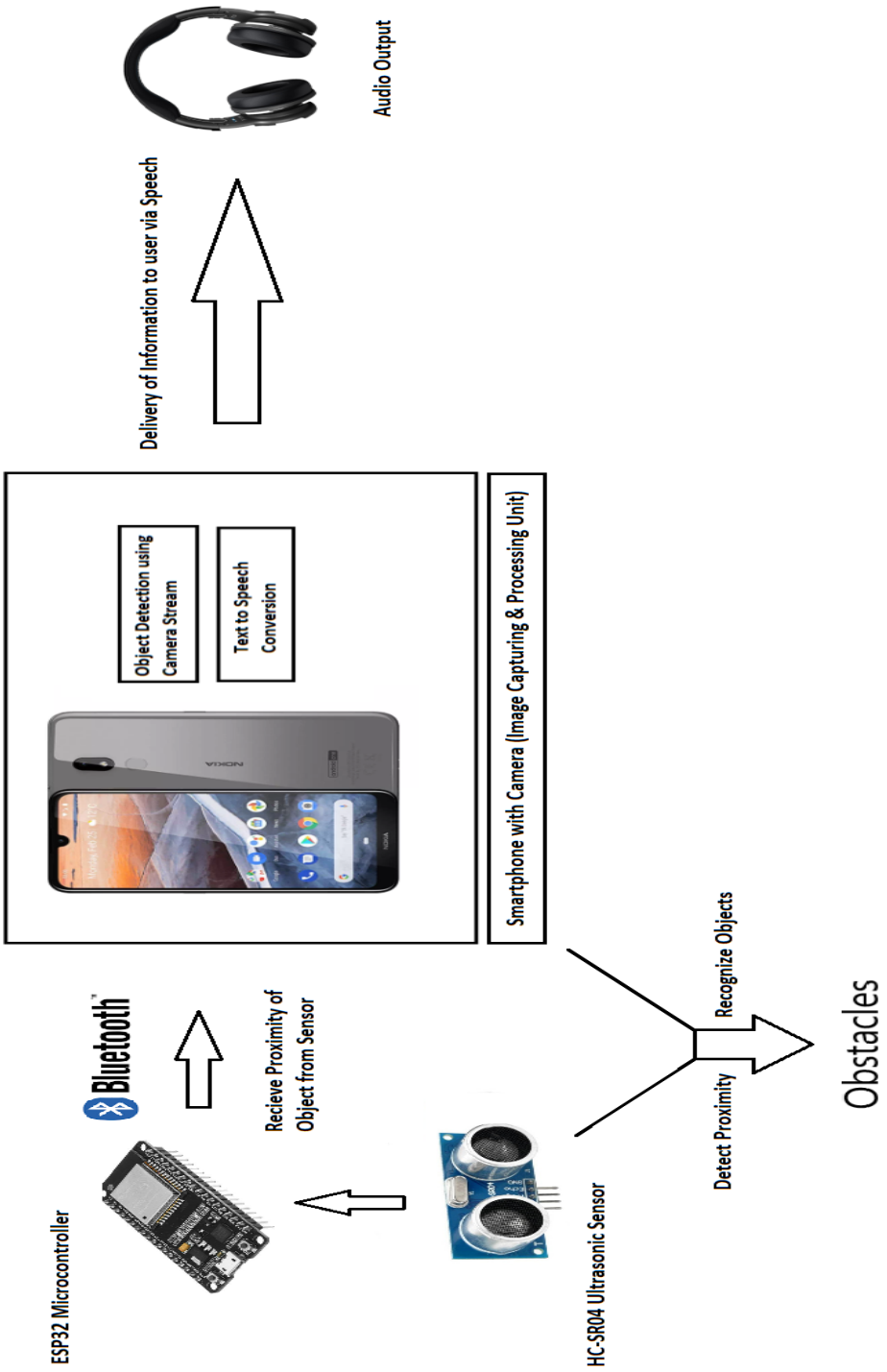

Fig 4.1 System Architecture

The proposed system consists of the following:

- HC-SR04 Ultrasonic Sensor
- ESP32 Microcontroller
- Smartphone (Image Processing & Capturing Unit)
- Audio Output Device

The HC-SR04 Ultrasonic Sensor uses Echo-location to determine the proximity of the object. The echo is then processed by the control circuit to calculate the time difference between the signal being transmitted and received. This time is used to calculate the distance between the sensor and the reflecting object. This data is then passed to the Microcontroller.

The Microcontroller receives data from the sensor and passes it to the Smartphone via Bluetooth. The Smartphone acts as an Image Processing and powerful processing unit and delivers the output as audio.

4.2 Dataset

The Dataset used is COCO (Common Objects in Context).

COCO is a large-scale object detection, segmentation, and captioning dataset. It consists of images, bounding boxes and labels, and is split into Train, Test and Validation sets. Objects are labelled using per-instance segmentations to aid in precise object localization.

4.3 Flowchart

The Flowchart is shown in Fig 4.2.

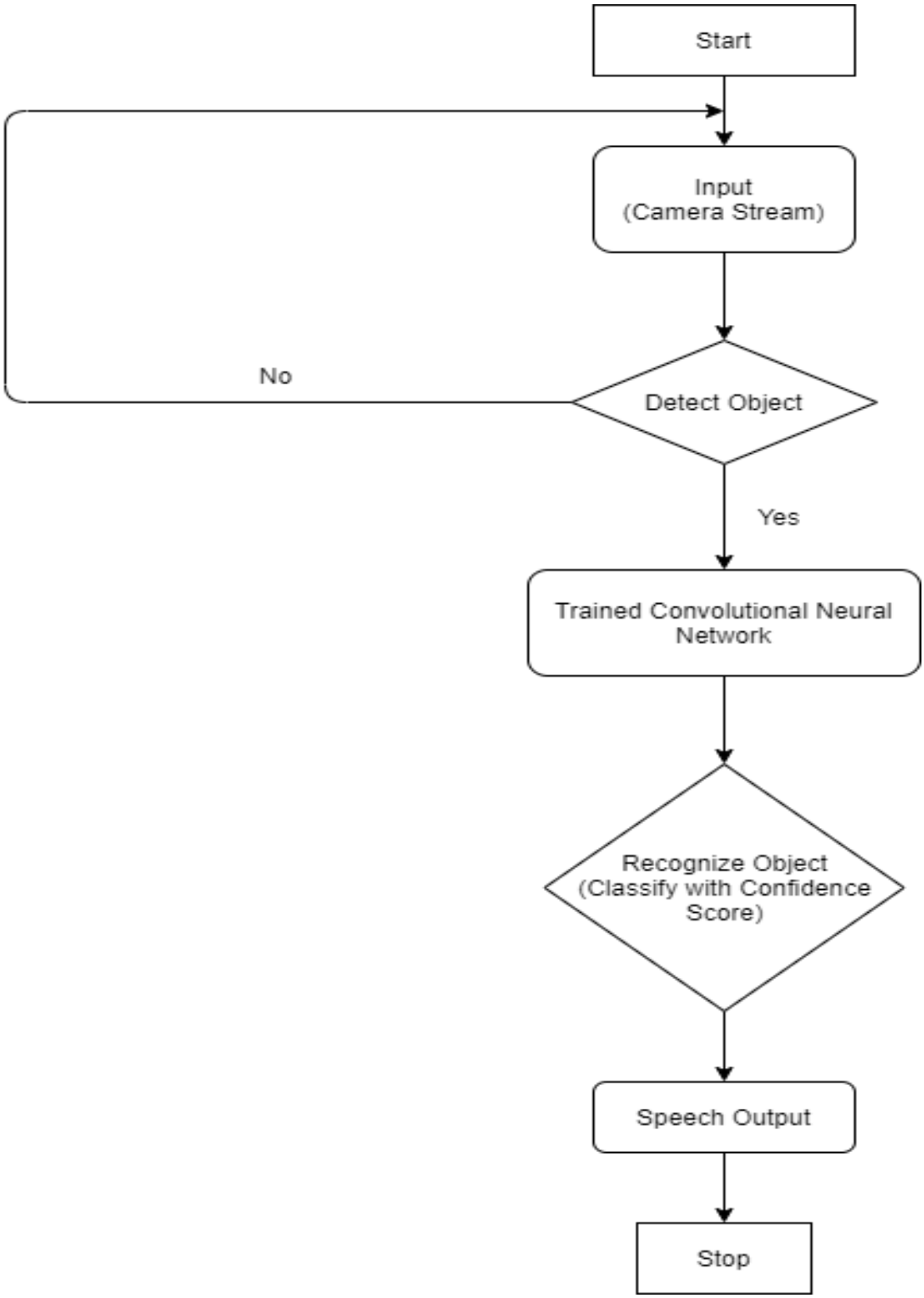

Fig 4.2 Flowchart

The Camera stream is passed as the input which contains cropped images of the object. If an object or a part of the object is identified, then the stream is passed as input to the trained CNN.

If no object is recognised, then a new camera stream is passed as input. The CNN is trained to identify the object based on the dataset and training model.

It classifies the object with a confidence score which is based on which part of the object it could identify and how accurate it is. This output is in the form of speech which helps the Visually Impaired to know which object is in front of them.

4.4 Use Case Diagram

The Use Case Diagram is shown in Fig 4.3.

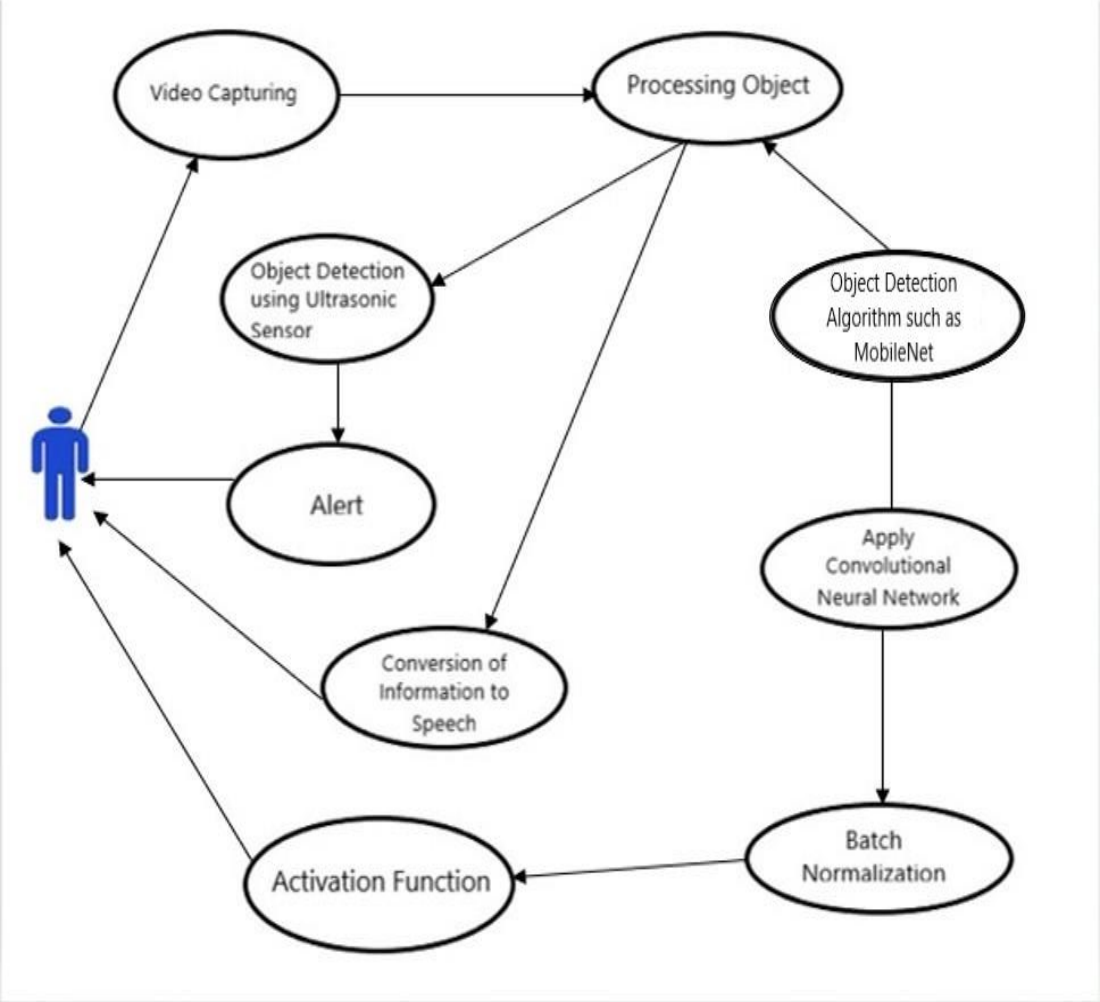

Fig 4.3 Use Case Diagram

The system shown is the Visual Aid App. The actor in the system is the user of the app i.e a visually impaired person.

The user can start the app and this starts the video capturing too. The video is then fed to a processing object which runs on an object detection algorithm such as MobileNet.

After the object is detected the distance of the object is calculated via an ultrasonic sensor and this info is then combined and the output is given in audio format.

The processing object internally applies convolutional neural network and batch normalization to give the detected object as output. This is a continuous process and the app goes back to capturing video and following the subsequent steps.

4.5. Sequence Diagram

The Sequence Diagram is shown in Fig 4.4.

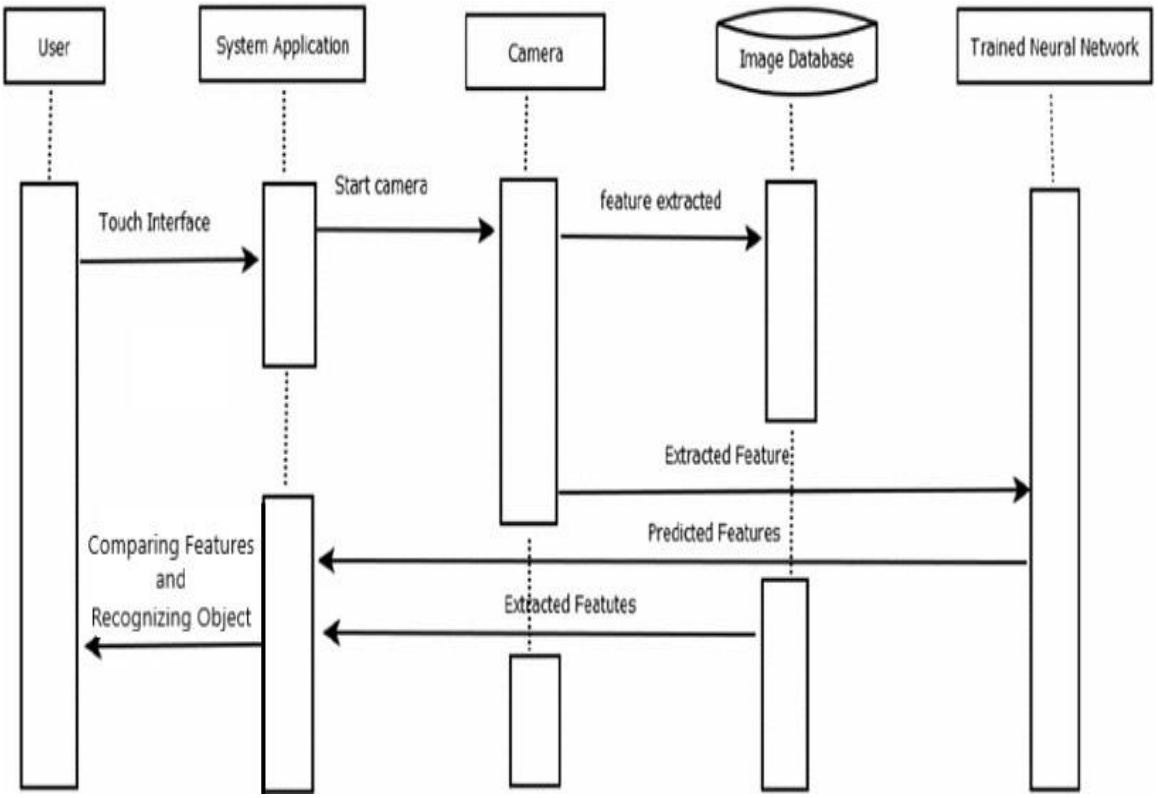

Fig 4.4 Sequence Diagram

Step 1: User launches the App using a voice command/touch interface.

Step 2: The Application launches the Camera

Step 3: The Camera detects objects by extracting certain features. These extracted features are stored in an Image Database

Step 4: The Extracted features are sent to a trained neural network.

Step 5: The Extracted features and Predicted Features are compared and the object is classified/recognized based on the highest confidence score

Step 6: The information is then given to the user along with the object's proximity in the form of an audio output.

CHAPTER 5

IMPLEMENTATION

5.1 Ultrasonic Range Finder

- Send HIGH pulse for 10 micro seconds
- Wait until the module transmits 40 KHz pulses. When 8th pulse is transmitted echo pin becomes HIGH TIMER1 starts counting.
- TIMER1 value is equal to the time taken by the signal to go forward and comeback so we need to take only half time.
- Distance Measured:
- The speed of the ultrasonic pulse is nothing but a function of the speed of sound which is 340.29 m/s or 34029 cm/s.

5.2 Object Detection and Localization

- The objective of the Convolution Operation is to extract the high-level features such as edges, from the input image. ConvNets need not be limited to only one Convolutional Layer. Convolution and it's implementation are shown in Fig 5.1 and Fig 5.2 respectively.
- Make a window of size much smaller than actual image size. Crop it and pass it to ConvNet (CNN) and have ConvNet make the predictions.
- Keep on sliding the window and pass the cropped images into ConvNet.
- After cropping all the portions of image with this window size, repeat all the steps again for a bit bigger window size.
- The Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data through dimensionality reduction.

- Flattening of the image into a column vector.
- Backpropagation applied to every iteration of training.
- Classify them using the Softmax Classification technique.
- At the end, you will have a set of cropped regions which will have some object, together with class and bounding box of the object.

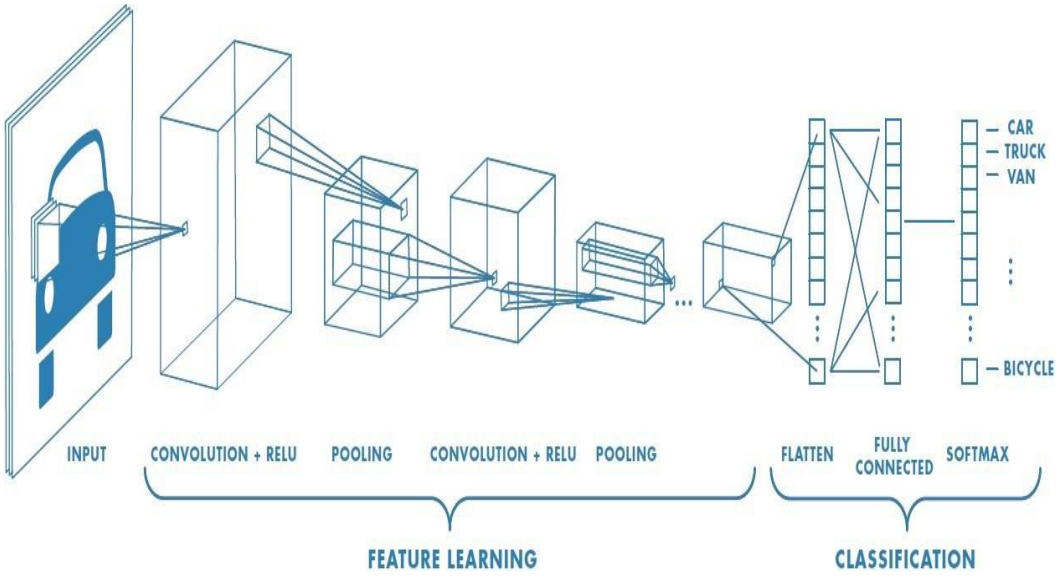

Fig 5.1 Convolution

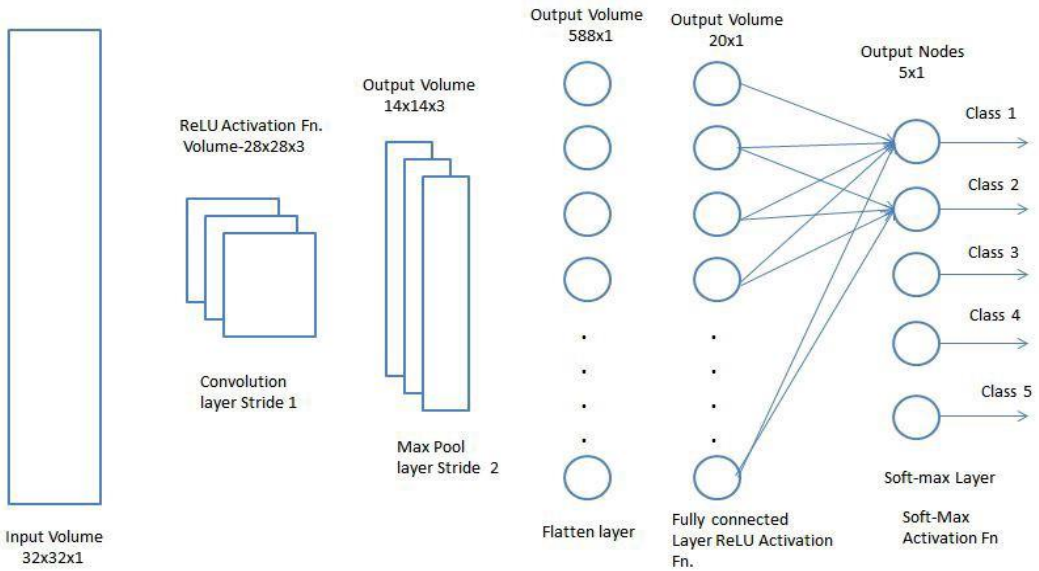

Fig 5.2 Convolution Implementation

Variables/Parameters:

For a Standard Convolutional Layer;

- Input - DF x DF x M, Feature Map F.
- Output - DG x DG x N, feature map G, where
- where DF is the spatial width and height of a square input feature map
- M is the number of input channels (input depth)
- DG is the spatial width and height of a square output feature map
- N is the number of output channel (output depth)
- The standard convolutional layer is parameterized by convolution kernel K of size DK x DK x M x N, where
- DK is the spatial dimension of the kernel assumed to be square
- M is number of input channels and N is the number of output channels.
- The output feature map for standard convolution assuming stride one and padding is computed as:

$$G_{k,l,n} = \sum_{i,j,m} K_{i,j,m,n} \ . \ F_{k+i-1,l+j-1,m}$$

- The Computational Cost for a Standard Convolution: DK x DK x M x N x DF x DF

5.3 Experimental Setup

5.3.1 Hardware

HC-SR04 Ultrasonic Sensor

The HC-SR04 Ultrasonic Distance Sensor is a sensor used for detecting the distance to an object using sonar. The HC-SR04 uses non-contact ultrasound sonar to measure the distance to an object and consists of two ultrasonic transmitters (basically

speakers), a receiver, and a control circuit. The transmitters emit a high frequency ultrasonic sound, which bounce off any nearby solid objects, and the receiver listens for any return echo. That echo is then processed by the control circuit to calculate the time difference between the signal being transmitted and received. This time is used to calculate the distance between the sensor and the reflecting object. This data is then passed to the Microcontroller. The HC-SR04 is shown in Fig 5.3.

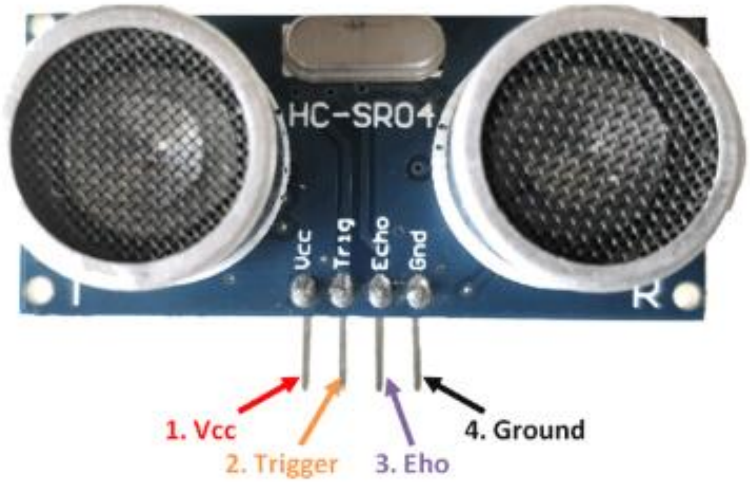

Fig 5.3 HC-SR04 Ultrasonic Sensor

As shown above the HC-SR04 Ultrasonic (US) sensor is a 4 pin module, whose pin names are Vcc, Trigger, Echo and Ground respectively. This sensor is a very popular sensor used in many applications where measuring distance or sensing objects are required. The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver. The sensor works with the simple high school formula that,

Distance = Speed × Time

The Ultrasonic transmitter transmits an ultrasonic wave, this wave travels in air and when it gets objected by any material it gets reflected back toward the sensor this reflected wave is observed by the Ultrasonic receiver module as shown in the picture below. This process is called Echo-Location and is shown in Fig 5.4.

Fig 5.4 Echo-Location

Features:

- Operating voltage: +5V
- Theoretical Measuring Distance: 2cm to 450cm
- Practical Measuring Distance: 2cm to 80cm
- Accuracy: 3mm
- Measuring angle covered: <15°
- Operating Current: <15mA
- Operating Frequency: 40H

ESP32 Microcontroller

ESP32 is low-cost, low-power, system-on-a-chip microcontroller with integrated Wi-Fi and dual-mode Bluetooth. It is programmed to receive Data from the Sensor and is connected to the Smartphone via Bluetooth for Information Transfer. The ESP32 Microcontroller is shown in Fig 5.5.

Fig 5.5 ESP32 Microcontroller

Features:

Processors

- CPU: Xtensa dual-core (or single-core) 32-bit LX6 microprocessor, operating at 160 or 240 MHz and performing at up to 600 DMIPS
- Ultra low power (ULP) co-processor

Wireless connectivity

- Wi-Fi: 802.11 b/g/n
- Bluetooth: v4.2 BR/EDR and BLE (shares the radio with Wi-Fi)

Peripheral interfaces

- 12-bit SAR ADC up to 18 channels
- 2 × 8-bit DACs
- 10 × touch sensors (capacitive sensing GPIOs)
- 4 × SPI
- 2 × I²S interfaces
- 2 × I²C interfaces

- 3 × UART
- SD/SDIO/CE-ATA/MMC/eMMC host controller
- SDIO/SPI slave controller
- Ethernet MAC interface with dedicated DMA and IEEE 1588 Precision Time Protocol support
- CAN bus 2.0
- Infrared remote controller (TX/RX, up to 8 channels)
- Motor PWM
- LED PWM (up to 16 channels)
- Hall effect sensor
- Ultra low power analog pre-amplifier

Security

- IEEE 802.11 standard security features all supported, including WFA, WPA/WPA2 and WAPI
- Secure boot
- Flash encryption
- 1024-bit OTP, up to 768-bit for customers
- Cryptographic hardware acceleration: AES, SHA-2, RSA, elliptic curve cryptography (ECC), random number generator (RNG)

Power management

- Internal low-dropout regulator
- Individual power domain for RTC
- 5μA deep sleep current

- Wake up from GPIO interrupt, timer, ADC measurements, capacitive touch sensor interrupt.

Smartphone

The Smartphone is used as a multipurpose device. It is used for taking the Input from the Camera Stream and is used to detect objects. It is preferred as a processing unit over the Raspberry Pi, because of better processing power.

It receives the proximity data from the Microcontroller and outputs this information as Speech to the User.

5.3.2 Software

Arduino IDE

Arduino Integrated Development Environment (IDE) is used to write and upload the program that receives the Proximity Information to the ESP32 Microcontroller.

The Arduino IDE is a cross-platform application (for Windows, macOS, Linux) that is written in functions from C and C++. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards.

The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware. By default, avrdude is used as the uploading tool to flash the user code onto official Arduino boards.

Android Studio

Android Studio is the IDE used to develop the Android Application with all of it's features. Java as Google's preferred language for Android app development. Java is still supported, as is C++.

Features

- Gradle-based build support
- Android-specific refactoring and quick fixes
- Lint tools to catch performance, usability, version compatibility and other problems
- ProGuard integration and app-signing capabilities
- Template-based wizards to create common Android designs and components
- A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations
- Support for building Android Wear apps
- Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine.
- Android Virtual Device (Emulator) to run and debug apps in the Android studio.

Tensorflow/Tensorflow Lite

Tensorflow is an open source artificial intelligence library, using data flow graphs to build models. It allows developers to create large-scale neural networks with many layers. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production.

TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units). TensorFlow

is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS.

Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.

TensorFlow computations are expressed as stateful dataflow graphs. The name TensorFlow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors.

TensorFlow Lite is the lightweight version which is specifically designed for the mobile platform and embedded devices. It provides machine learning solutions to mobile with low latency and small binary size.

TensorFlow Lite consists of two main components:

- The TensorFlow Lite interpreter, which runs specially optimized models on many different hardware types, including mobile phones, embedded Linux devices, and microcontrollers.
- The TensorFlow Lite converter, which converts TensorFlow models into an efficient form for use by the interpreter, and can introduce optimizations to improve binary size and performance.

TensorFlow Lite is designed to make it easy to perform machine learning on devices, "at the edge" of the network, instead of sending data back and forth from a server. For developers, performing machine learning on-device can help improve:

- Latency: there's no round-trip to a server
- Privacy: no data needs to leave the device
- Connectivity: An Internet connection isn't required
- Power consumption: network connections are power hungry. TensorFlow Lite works with a huge range of devices, from tiny microcontrollers to powerful mobile phones.

CHAPTER 6

RESULTS AND DISCUSSION

6.1 Object Detection

Fig 6.1 Object Detection in Visual Aid Android Application

Fig 6.1 shows object detection in the android application developed named Visual Aid using Android Studio. The app uses Tensorflow Lite as a base to run the model created. It shows the detected object along with its confidence score. The algorithm used (MobileNet) gives an accuracy around 70 percent.

6.1.1 Performance Indicators

- **Accuracy**: AP (Average precision) is a popular metric in measuring the accuracy of object detectors like Faster R-CNN, SSD, etc. Average precision computes the average precision value for recall value over 0 to 1.
    - **Precision**: Measures how accurate is your predictions. i.e. the percentage of your predictions are correct.
    - **Recall:** Measures how good you find all the positives. For example, we can find 80% of the possible positive cases in our top K predictions.
- **AP** is averaged over all categories. Traditionally, this is called "mean average precision" (mAP). We make no distinction between AP and mAP (and likewise AR and mAR) and assume the difference is clear from context.
- **Mult-adds**: The number of multiply-add operations.
- **Million parameters**: Number of parameters in millions.

6.1.2 Experimental Results

The algorithm used gives an accuracy of about 70%. This is shown in Fig 6.2.

The following graphs show the comparison of the mobile net algorithm in comparison to other algorithms in terms if the performance indicators mentioned. Parameter Findings are shown in Fig 6.3.

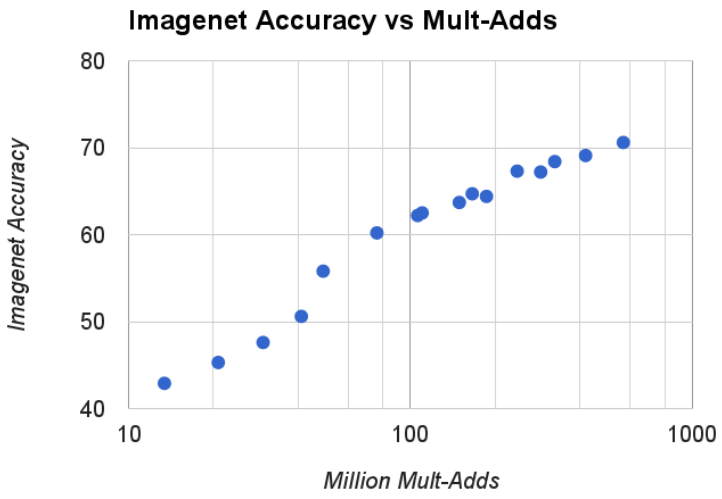

Fig 6.2 Findings from mobile net paper of its accuracy

Mobile Net algorithm uses depth-wise convolution and pointwise convolution which significantly reduces the number of multiply-add operations and gives around the same accuracy as other high performing algorithms such as YOLO ,GoogleNet etc

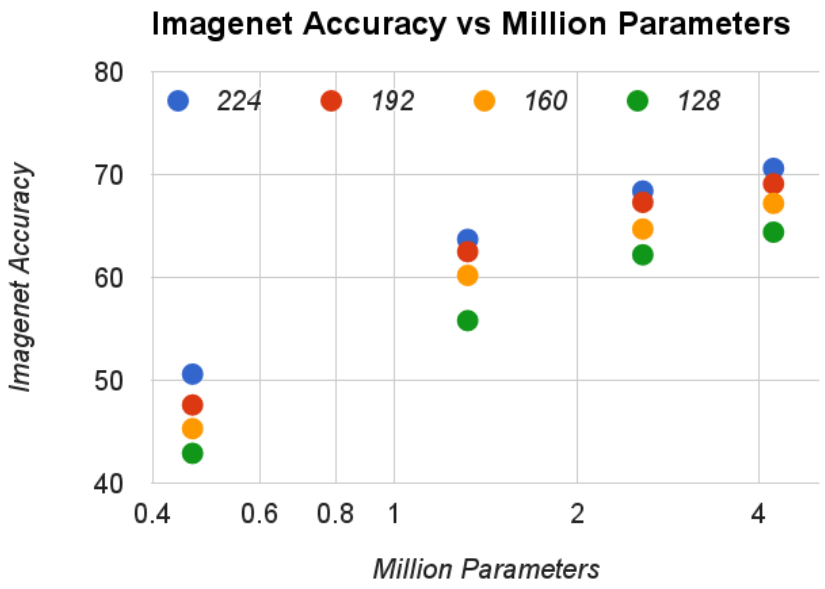

Fig 6.3 Findings from mobile net paper of its parameters

Table 6.1 Comparison of Algorithms for object detection

Model	**ImageNet Accuracy**	**Million Mult-Adds**	**Million Parameters**
1.0 MobileNet-224	70.6%	569	4.2
GoogleNet	69.8%	1550	6.8
VGG 16	71.5%	15300	138

6.2 IoT

6.2.1 Performance Indicators

- **Distance**: The distance up to which the ultrasonic sensor can measure.
- **Accuracy**: The correctness of the distance measured.

6.2.2 Experimental Results

The IOT module consisting of microcontroller and ultrasonic performs well in the mentioned performance indicators.

The ultrasonic sensor HCSR04 can measure objects up to a radius of 400 cm.

The microcontroller communicates with the android app using Bluetooth Low Energy and the modules performs as required.

The sensor measures distances accurately but gets distorted some times due to external factors.

CHAPTER 7

TESTING

Testing is a very important part of any project. It's the main thing to carry out to see if a part of the project is working as expected, if it's handling exception as expected if any exception occurs during execution. Testing should be carried out in various phases in the development of the project like

- **Unit Testing**, to test out each class or component that has been developed in the project is working as expected.
- **Integration Testing**, the part where all or some of the components are integrated and tested out if the components are working together fine as expected and see if any exception happens in any of the component the other components aren't affected by that and the execution doesn't stop. Etc.

Junit has been used to carry out unit testing in android app. Functionality, Performance and Compatibility testing was carried out for the IoT part.

In Functionality testing, we tested out if the IoT is functioning as required. In Performance Testing as the main task was to get the distance between the objects, we tested out how fast and accurate were the results and few adjustments were made to the sensor and code to improve the performance and in Compatibility Testing we tested out the compatibility of micro controller with the ultrasonic sensor and the connection to establish between the app and the micro controller through Bluetooth

During the development process as both IoT component and Android part were going in parallel, IoT part was mocked in Android while testing out the App with the help of Mockito.

IoT and Android app were tested out separately after that both the parts were integrated and integration testing was carried out on the whole component.

CHAPTER 8

CONCLUSION AND FUTURE SCOPE

8.1 Conclusion

Through this project, we try to build a device which detects an object and also measures the distance between the object and the device. The output is in the form of speech which tells you what the object is in the line of vision and distance in centimeters.

During the implementation of this project we came across several challenges like image quality, positioning of the micro controller, building an appropriate model for the object detection part. The image quality may vary in the real-life scenarios, for example, the image gets darker in a cloudy day and gets blurry in a rainy or foggy day, etc. A difficult task would be to recognize partially blocked objects, especially for objects with most of their surfaces blocked.

8.2 Contribution

Implementation has been successfully completed and it aims to help the visually impaired, enabling them to be aware of their surroundings, to walk independently with minimum human intervention/support and thereby ensuring their safety.

8.3 Future Scope

The robustness of the model needs to be improved to get good accuracy with image variations of brightness/contrast/blur, etc. So, another thing that can help is to manually add image variations to input image set such that the model is less sensitive to the image variations.

While capturing the images sometimes problem arises with an unusable image, due to blurring caused by focus and improper holding of the camera and depth of the field. To a limit this can be solved by using a deblurring and an auto focus system.

Right now, the model detects multiple objects but the IoT module acts as a bottleneck since it only measures the distance of a single object. So, improvements need to be made at the IoT side to make the sensor detect multiple objects and also the distance measured by the IoT sensor should be matched with the object that is detected

Since the object detection is carried out by a mobile, a case for the mobile can be designed which can have the IoT part and design it in such a way the mobile itself will act as a power source for the IoT sensor.

The model that has been used for the object detection part can be improved to achieve better accuracy and results.

REFERENCES

[1] Shankar Sivan, Gopu Darsan, Computer Vision based Assistive Technology For Blind and Visually Impaired People, July 2016

[2] An IoT based Obstacle Detection and Alerting System in Vehicles using Ultrasonic Sensor

[3] Object Detection and Its Implementation on Android Devices, Stanford University

[4] YOLOv3: An Incremental Improvement

[5] MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, Google Inc.

[6] Ultrasonic RangeFinder using 8051, International Journal for Research in Applied Science & Engineering Technology (IJRASET)

[7] Microsoft COCO: Common Objects in Context

[8] https://www.tensorflow.org/lite/models/object_detection/overview

[9] https://www.instructables.com/id/Distance-Measurement-Using-HC-SR04-Via-NodeMCU/

[10] http://www.circuitstoday.com/ultrasonic-range-finder-using-8051