AN ANDROID BASED REAL TIME OBJECT CLASSIFICATION AND DETECTION

BY

Sadia Khanom
ID: 151-15-5005

Nadim Mahmud
ID: 151-15-5016

Shajedul Islam Sagor
ID: 151-15-5030

Shohanur Rahman Shohan
ID: 151-15-5351

This Report Presented in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science and Engineering

Supervised By

Dr. Syed Akhter Hossain
Professor and Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

MAY 2019

APPROVAL

This Project titled "**An Android Based Real Time Object Classification and Detection**" submitted by **Sadia Khanom**, ID No: 151-15-5005, **Nadim Mahmud**, ID No: 151-15-5016, **Shajedul Islam Sagor**, ID No: 151-15-5030 and **Shohanur Rahman Shohan**, ID No: 151-15-5351 to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 03 May 2019.

BOARD OF EXAMINERS

[illegible]

Dr. Syed Akhter Hossain
Professor and Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Chairman

[illegible]

Nazmun Nessa Moon
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Internal Examiner

[illegible]

Abdus Sattar
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Internal Examiner

[illegible]

Dr. Mohammad Shorif Uddin
Professor
Department of Computer Science and Engineering
Jahangirnagar University

External Examiner

DECLARATION

We hereby declare that, this project is under the supervision of **Dr. Syed Akhter Hossain, Professor and Head, Department of Computer Science and Engineering,** Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

Supervised by:

[illegible]

Dr. Syed Akhter Hossain
Professor and Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Submitted by:

Sadia Khanom

Sadia Khanom
ID: 151-15-5005
Department of CSE
Daffodil International University

Nadim Mahmud

Nadim Mahmud
ID: 151-15-5016
Department of CSE
Daffodil International University

Sagor

Shajedul Islam Sagor
ID: 151-15-5030
Department of CSE
Daffodil International University

Shohan

Shohanur Rahman Shohan
ID: 151-15-5351
Department of CSE
Daffodil International University

ACKNOWLEDGEMENT

We want to pay our gratitude to the Almighty for enabling us to prepare the report successfully. Then we would like to express our sincere gratitude and cordial thanks to some specific persons who helped us in preparing this report.

Firstly, we want to mention our Supervisor **Dr. Syed Akhter Hossain, Professor and Head**, Department of CSE, for giving this opportunity to prepare the Project report on "An Android Based Real Time Object Classification and Detection". Undoubtedly, the experience of doing this report will help us immensely in our next higher-level courses. We would like to thank him for the valuable instructions and helpful advices in preparing this report.

Secondly, we would like to express our heartiest gratitude to other faculty member and the staff of CSE department of Daffodil International University

Next, this project report would not have been possible without the dedication and contribution of our friends. Their valuable opinions and contribution is what made this Project report possible.

Finally, we must acknowledge with due respect, the constant support and patience of our parents.

ABSTRACT

The ambition behind this project is to develop and design a app primarily for use in the object detection. This online based object detector entitled “An Android Based Real Time Object Classification and Detection” provides convenience for the users to know the name of the project. It overcomes the drawbacks of the long-established queuing system. User can know the name of an object in different language. Object detection is a very important task for different applications including autonomous driving, face detecting, video surveillance, etc. Now android is one of the largest platforms in the world that run in several smart phones and tablets from various manufacturers like Google, Samsung, and HTC etc. The project we have developed is a app for Android platform. After implementation of all functions, the system is tested in different stages and it works successfully as a prototype. We will make it offline based and also try to add sound, so that any blind people can hear the object name and also use it as their walking assistant.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

“An Android Based Real Time Object Classification and Detection” is considered to be a type of Android based app; this app is designed to be used for every citizen. It can detect objects and show the name of that object. In this app, there are a lot of language. For that reason, anyone can see the object name in their own

1.2 Motivation of work

Now-a-days many people don’t know the name of all object. If they know the name of an object in one language, they don’t know the other name of that object in other language. So we tried to solve this problem.

To solve this problem, we build a real time platform where user can show the name of an object in many languages. This will help them to learn many new things.

1.3 Objectives

- Users can operate it via computer/smartphones/tabs with the help of internet.
- Object can be detected in any positions.
- Detected object’s name will be shown in English language.
- Without English, user can see the name of that object in any other language.
- Overall providing a much better user satisfaction platform.

1.4 Expected Outcome

This project is to develop a app which will help people to learn the name of an object in many languages. It will surely make life easier for relatively busy users who want to know new things. We expect the project will be a time saver in the long run for users. In an era of digitization, its going to be a huge step forward for users to know new things.

1.5 Report Layout

Chapter 1: Introduction

In this chapter the discussions are about the motivations, objectives and the expected outcome of the project. Later part the report’s layout are being followed.

Chapter 2: Background

In this chapter the discussion is about the background circumstances of our project. We also talk about the related works, comparison to other candidate systems, the scope of the problem and challenges of the project.

Chapter 3: Requirement Specification

In this chapter we talked all about the requirements like business process modeling, the requirement collection and analysis, the use case model of the project and their description, the logical relational database model and the design requirements.

Chapter 4: Design Specification

This chapter is all about the prototype of the project, front-end design, back-end design, interaction design and UX and the implementation requirements.

Chapter 5: Implementation and Testing

This chapter includes the implementation of database, front-end designs, interactions and the test results of the project.

Chapter 6: Conclusion and Future Scope

We discussed about the conclusion and the scope for further developments.

CHAPTER 2

BACKGROUND

2.1 Introduction

“An android based Object Classification And Detection” is very instructive android based application. We can update the name of the objects in this system. The users can obtain their necessary information from here. This project will reduce money expense and consume time.

2.2 Related Work

There are many Android based application available in Bangladesh. They are:

- Object detection machine learing
- TFLite
- Object detector
- Miaz
- Object recognition
- Giorgio cam

2.3 Comparative Studies

Our implemented app is quite different from the existing app. Users can easily know the name of an object in English and also in other languages. In the existing apps, there is no option for know the object name in any other language.

2.4 Scope of Problem

- It is hard to keep pace with so much data for a singular business model.
- It is hard to keep pace with any position of an object.
- It is hard to make it offline apps.

2.5 Challenges

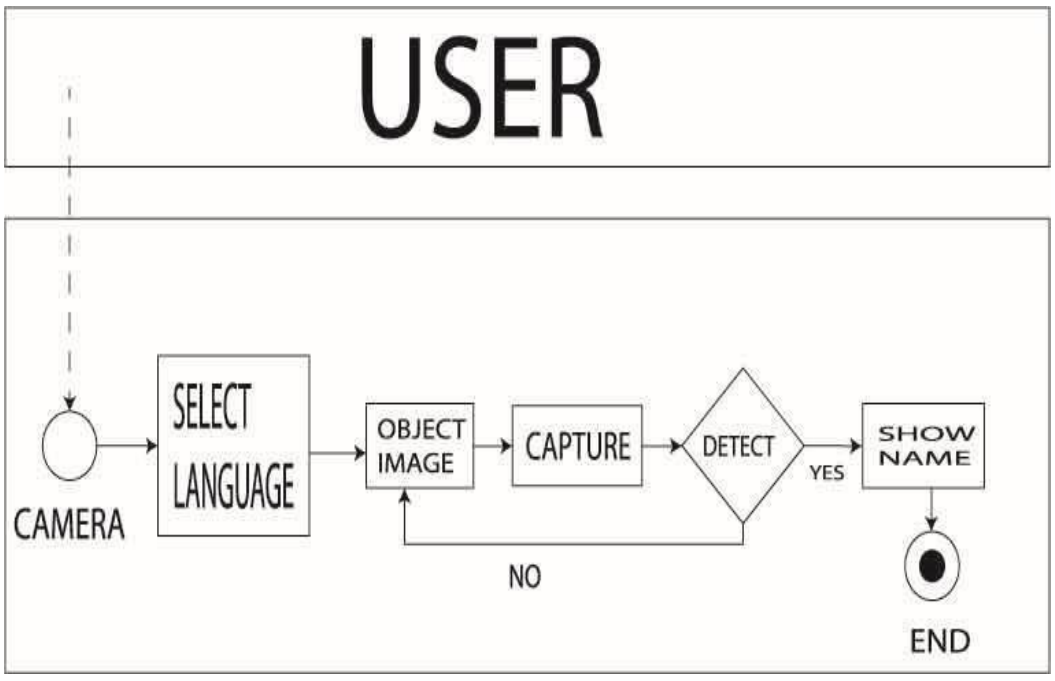
- User friendly.
- Giving many languages.
- Detecting any position of object.
- Making a secured platform.
- Beautiful and effective designs.
- Creating dynamic data passing channel.

CHAPTER 3

REQUIREMENT SPECIFICATION

3.1 Business Process Modeling

Business Process Model (BPM) is a business model that is generally prepared by business analyst to conceptualize the interaction of software and the user to provide a clear picture of the business process. In this process we can see that how the app work when a user tries to know the name of an object. When a user opens the camera, he/she needs to select the language, which language he/she wants to show the name of the object. Then he/she needs to capture photo and it will show the result. And Figure 3.1 shows the BPM for object detector.



Figure` 3.1: Business Process Modeling

3.2 General System Requirement

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.

In this paper, we developed a CNN based model and then implemented it with Tensorflow and Android. Our model is trained with KITTI benchmark. The KITTI data-set has over 10 Gigabytes of well-labeled data for object detection purpose. After training, our model is able to detect objects in view of camera on the Android device.

The input to the model is a 1242-pixel width, 375-pixel height image from KITTI data-set containing labeled cars, pedestrians, cyclists as targets to be detected and other objects that we don't care. We use a SqueezeDet layer and then a ConvDet layer to generate tens of thousands of bounding box coordinates (for localization), confidence score (for detection) and class scores (for classification). All this information is sent into a Non-Maximum Suppression (NMS) filter to predict the final detection results. Similarly, the input to our app is a camera stream, then we use inference interface to help us call the model pretrained and installed on our android device to produce the same type information (bounding box coordinates, confidence score and class scores). As above, a NMS filter implemented in the app facilities the final prediction.

3.3 Use Case Model

Following use case model will represent the relationship of primary actors like users with the system and various aspects of uses of the software briefly. And Figure 3.2 shows the use case of object detector.

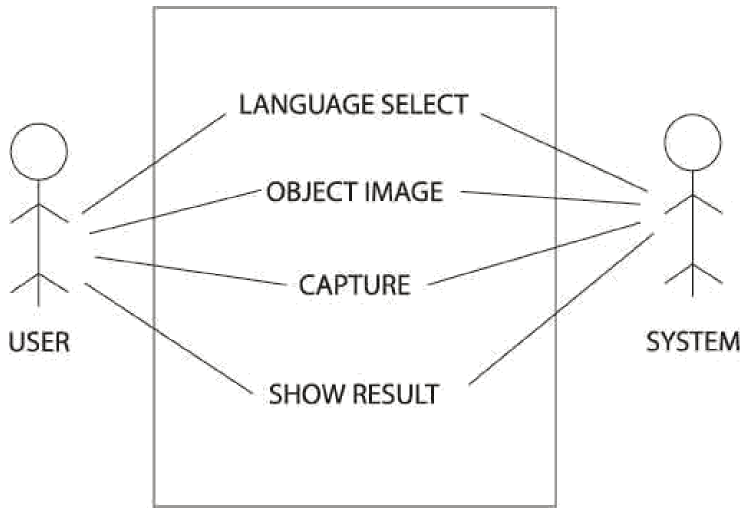


Figure 3.2: Use case diagram

3.3.1 Use Case Description

Use Case: Object Detector

Actor: User, System

Purpose: Know the name of the object

Description:

- At first user needs to select the preferable language.
- Then user needs to capture an image of an object.
- Then it will show the name of the object.

3.4 Logical Data Model

The term Logical Data Modeling is a process used to define and analyze requirements needed to support the business process within the scope of corresponding information systems in organizations. The Entity-Relationship model or Entity-Relationship diagram (ERD) is a logical data models, it includes the entity, attributes and relationship. Level can be varied so it is a multivalued attribute in the diagram. And Figure 3.3 shows the ERD of object detector.

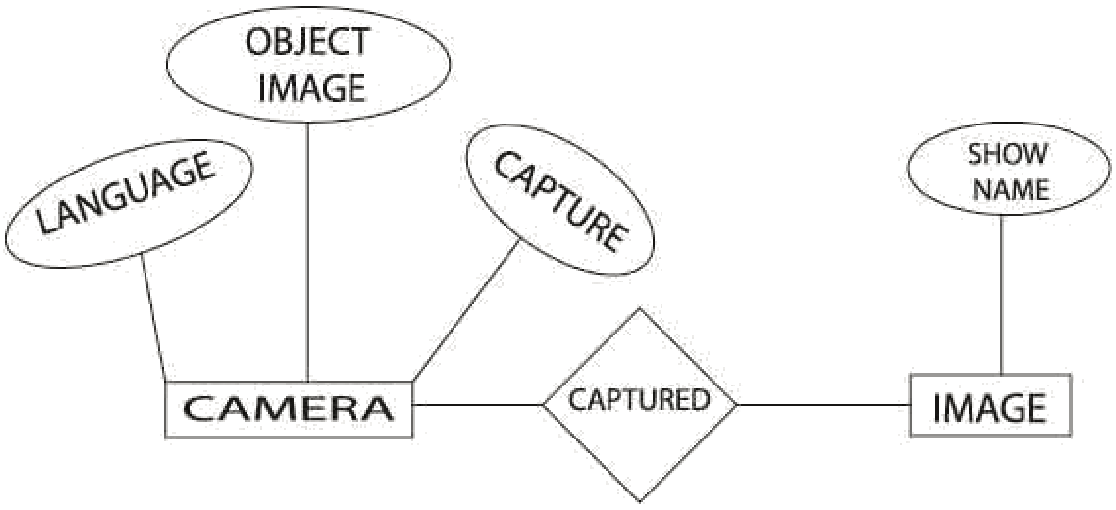


Figure 3.3: ER diagram

3.5 Design Requirement

Convolutional Neural Network (CNN) usually stands for the neural network which contains one or more convolutional neural layers. Each neural layer can be regarded as a combination of several spatial filters. These filters are used for extracting features from pictures. Some well-known filters are Histogram of Oriented Gradients (HOG) and color histograms, etc. A typical input for a convolutional layer is a 3 dimensional Grid. They are height (H), width (W) and channels (C). Here each channel represents a filter in the convolutional layer. The input of first layer usually has a shape of (H, W,
3) where 3 stands for the RGB channels for the raw pictures. Figure 3.4.1 shows the CNN workflow diagram.

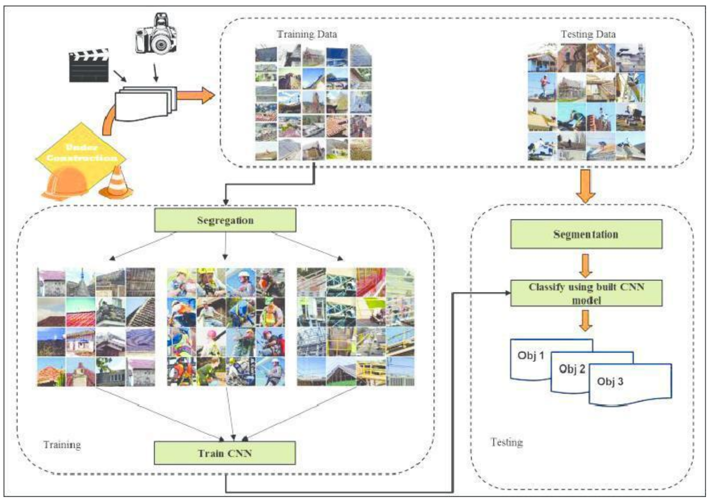


Figure 3.4.1: CNN Workflow

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. And Figure 3.4.2 shows the diagram of R-CNN workflow.

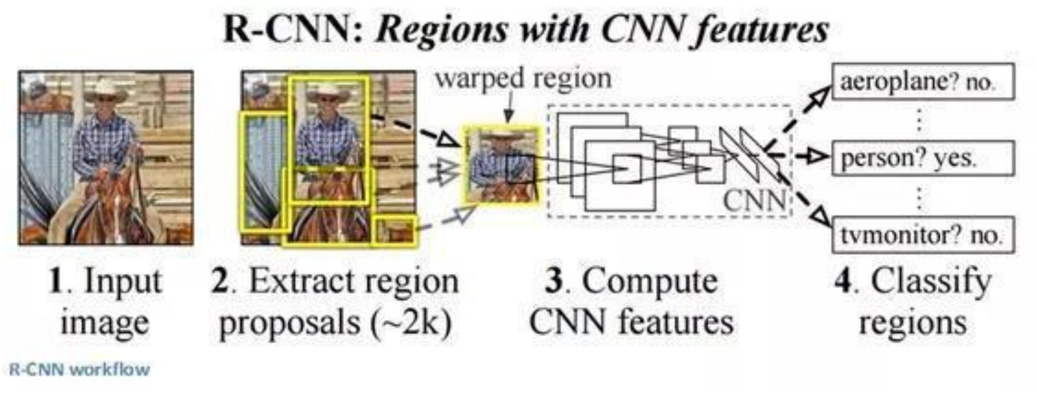


Figure 3.4.2: R-CNN Workflow

CNN Model

In this section, the CNN model to detect objects and the implementation of android app are elaborated in detail.

The model has the benefit of small model size, good energy efficiency and good accuracy due to the fact that it's fully convolutional and only contains a single forward pass. The overview of this object detection model is as following in Figure 3.4.3.

The CNN model we adopted is called SqueezeDet. The SqueezeDet model is a fully convolutional neural network for object detection. It's based on SqueezeNet architecture that extracts feature maps from image with CNN. Then another convolutional layer is used to find bounding box coordinates, confidence score and class probabilities. Finally, a multi-target loss is applied to compute final loss in training phase and a NMS filter is enforced to reduce the number overlapping bounding boxes and generate final detection in evaluation phase. And Figure 3.4.3 shows the diagram of SqueezeDet Architecture.

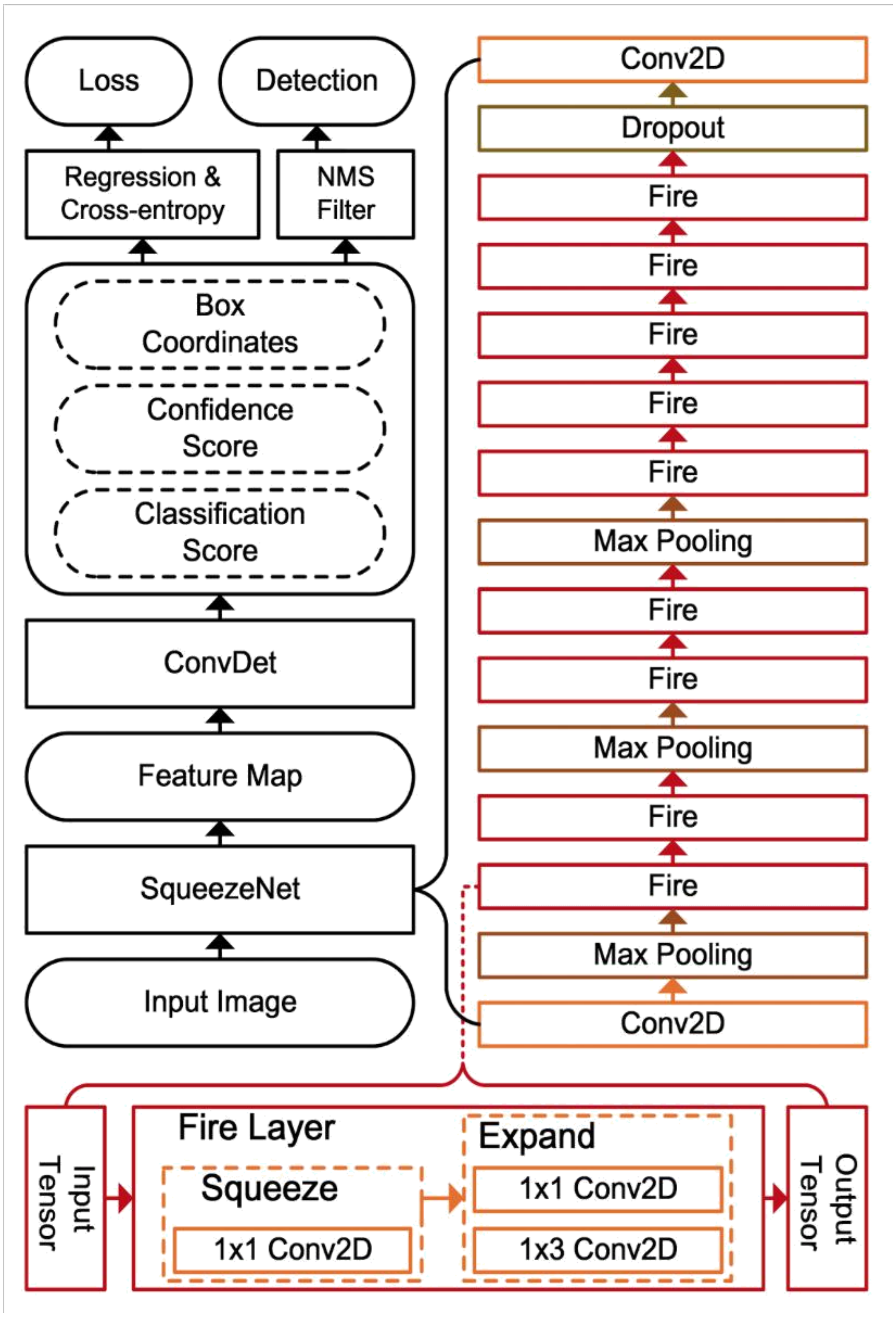
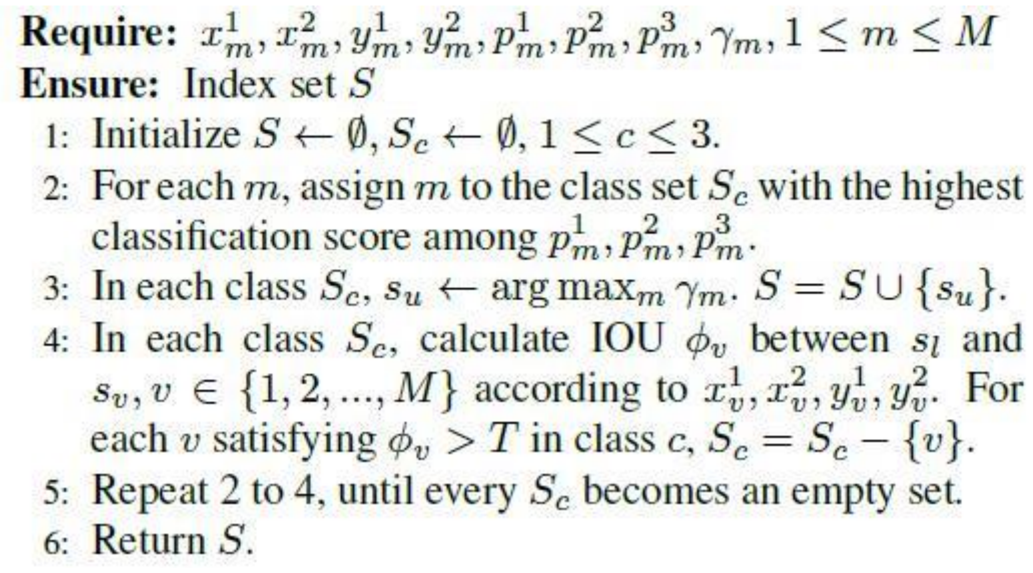


Figure 3.4.3: SqueezeDet Architecture

Android Implementation

For the implementation of CNN model in Android device, we used the interface provided by “Tensorflow An-Algorithm Android Camera Demo”

<u>Non-Maximum Suppression Algorithm</u>



```
Require: $x_m^1, x_m^2, y_m^1, y_m^2, p_m^1, p_m^2, p_m^3, \gamma_m, 1 \leq m \leq M$
Ensure: Index set $S$
  1: Initialize $S \leftarrow \emptyset, S_c \leftarrow \emptyset, 1 \leq c \leq 3$.
  2: For each $m$, assign $m$ to the class set $S_c$ with the highest
     classification score among $p_m^1, p_m^2, p_m^3$.
  3: In each class $S_c$, $s_u \leftarrow \arg\max_m \gamma_m$. $S = S \cup \{s_u\}$.
  4: In each class $S_c$, calculate IOU $\phi_v$ between $s_l$ and
     $s_v, v \in \{1, 2, ..., M\}$ according to $x_v^1, x_v^2, y_v^1, y_v^2$. For
     each $v$ satisfying $\phi_v > T$ in class $c$, $S_c = S_c - \{v\}$.
  5: Repeat 2 to 4, until every $S_c$ becomes an empty set.
  6: Return $S$.
```

First, the CNN model parameters need to be trained and saved into a proto buffer file. Basically, the way to save the CNN graph is to freeze all variables into constants with well-trained values and save them by their names. Then with Android interface tool (called “InferenceInterface”). Android app can load tensor with values, run the graph and read tensor output values. However, current interface omly support loading values and readingoutputs in the format of 1-D array. So, the input node/output node in the graph should be designed to be 1 D array to accommodate that. 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 the detected results are marked with boxes in real time. To accommodate the 8 fps of the default frame rate in Android device, we need the total processing time to be less than 125 ms. The overall app architecture is as shown in Figure 3.4.4.

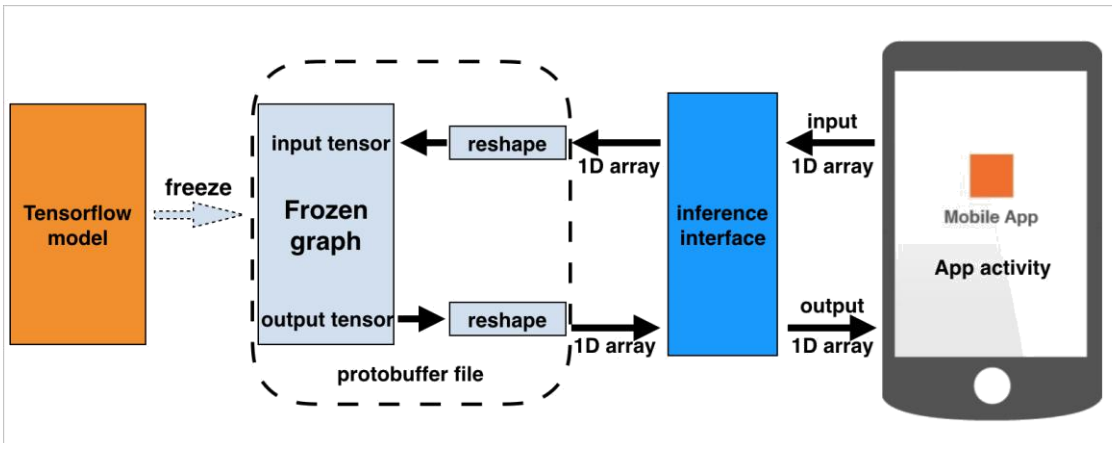


Figure 3.4.4: Android App Architecture

CHAPTER 4

DESIGN SPECIFICATION

4.1 Front-end Design

Front-end design is the main user interface that show the output of a system. It contains the beauty of a software. It is the main point where users directly interacts with the system. Front-end design is very important for mobile application based programs. The output of the design is view to mobile base emulator.

4.1.1 Apps Layout

Figure 4.1.1 shows the diagram of the apps layout. Here is capture option, language select option.

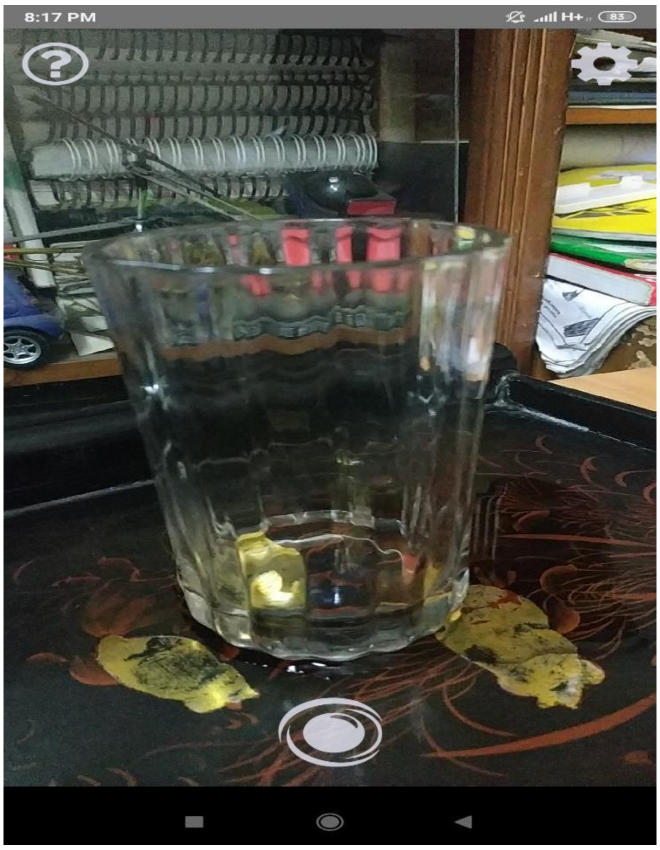
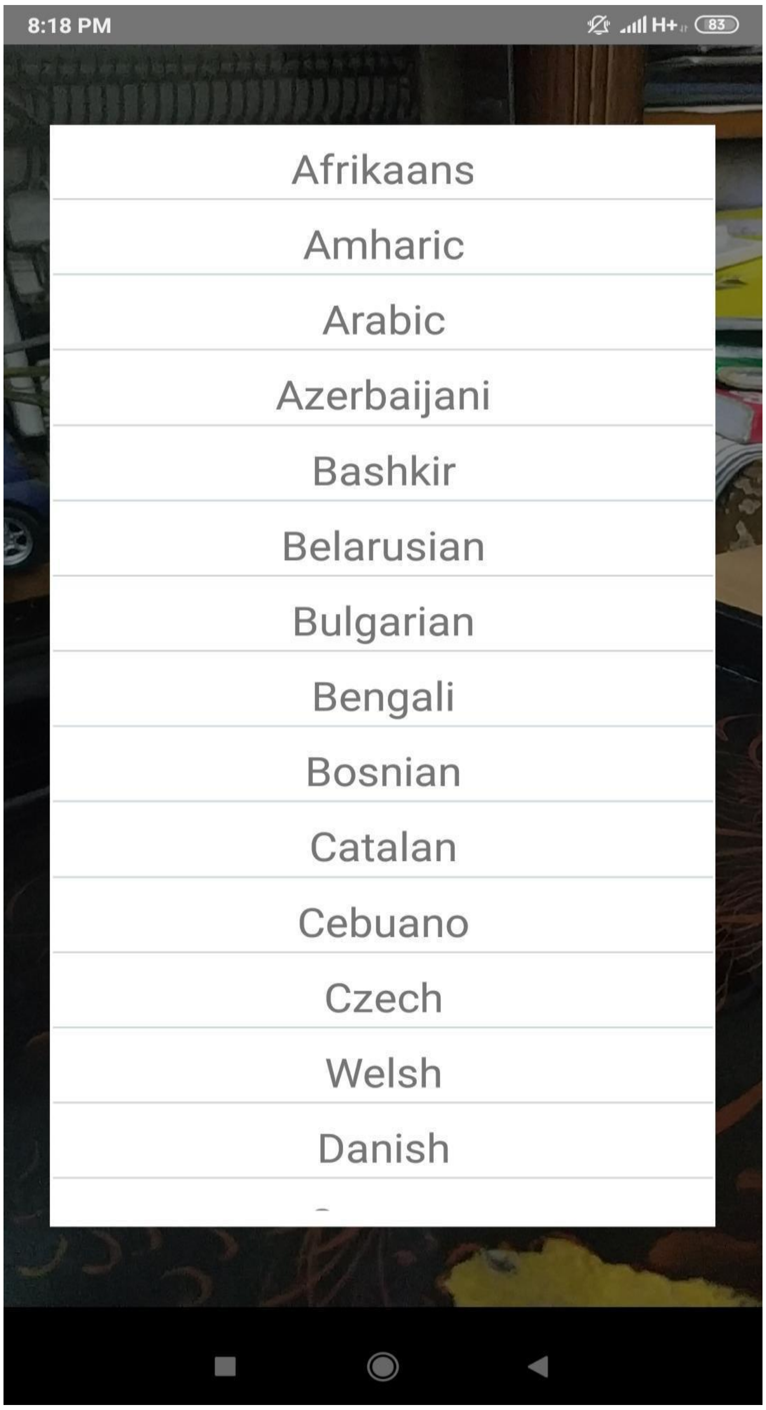


Figure 4.1.1: Apps layout

4.1.2 Different Language

Figure 4.1.2 shows the diagram of different types of language. From that option, user can select their preferable language.



Afrikaans
Amharic
Arabic
Azerbaijani
Bashkir
Belarusian
Bulgarian
Bengali
Bosnian
Catalan
Cebuano
Czech
Welsh
Danish

Figure 4.1.2: Different language

4.1.3 Object

Figure 4.1.3 shows the object which a user wants to capture. When a user wants to know the name of an object, then he/she needs to select the object.

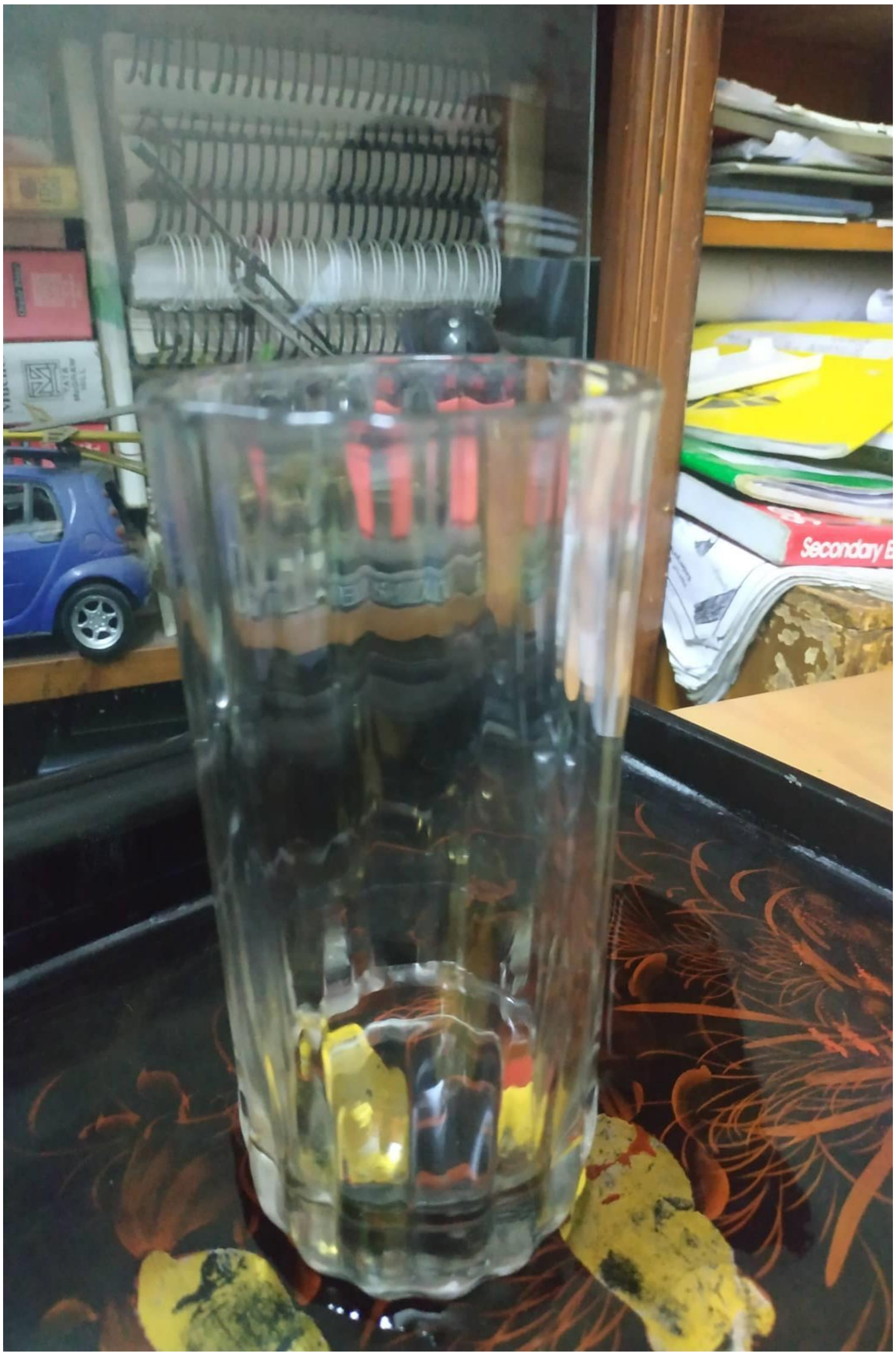


Figure 4.1.3: Object

4.1.4 Capture Photo

Figure 4.1.4 shows the fiagram of capturing photo of the object.

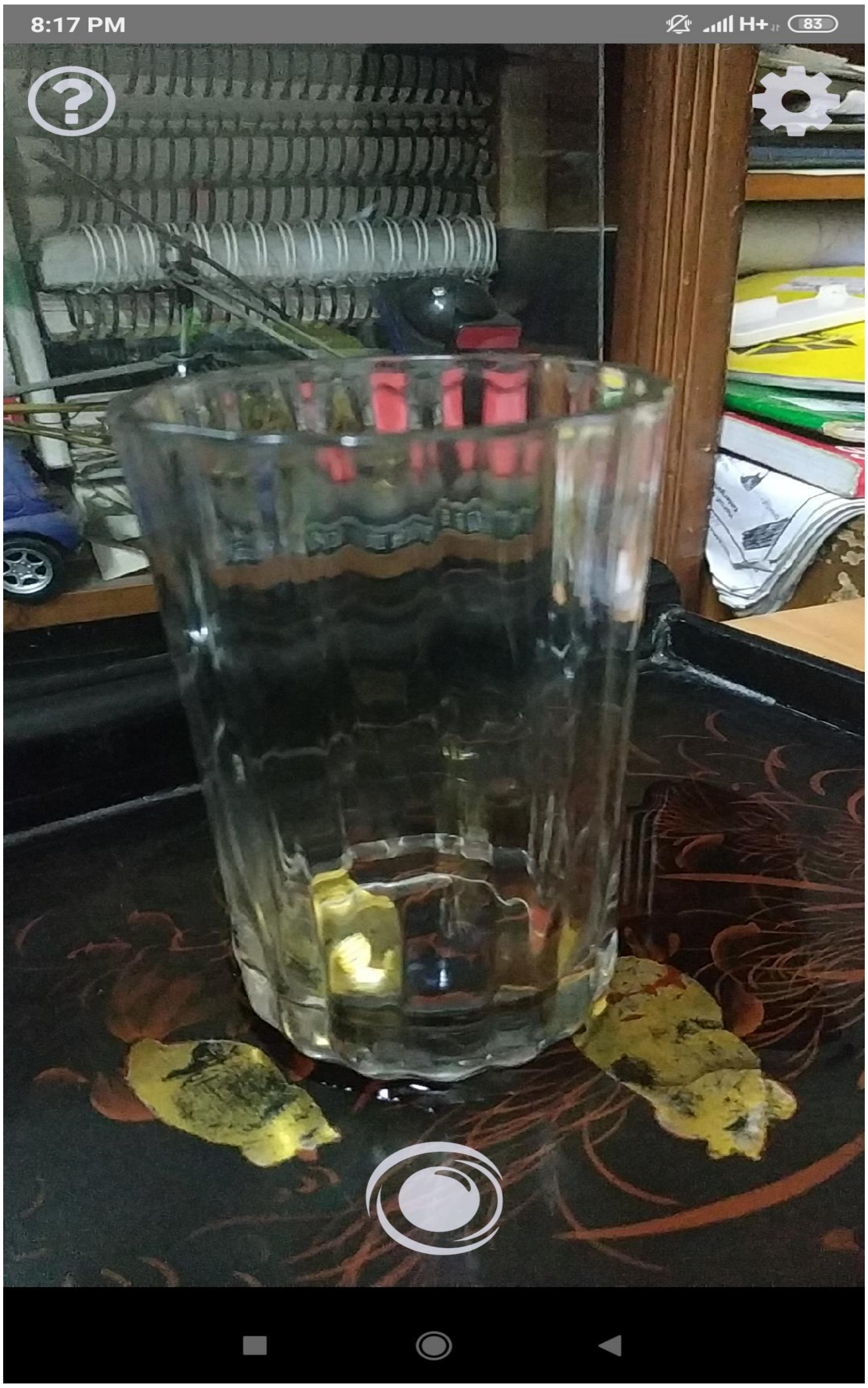


Figure 4.1.4: Capture Photo

4.1.5 Output

Figure 4.1.5 shows the diagram of the output of our apps. Here a user can show the name of the object in English and also in other one language.

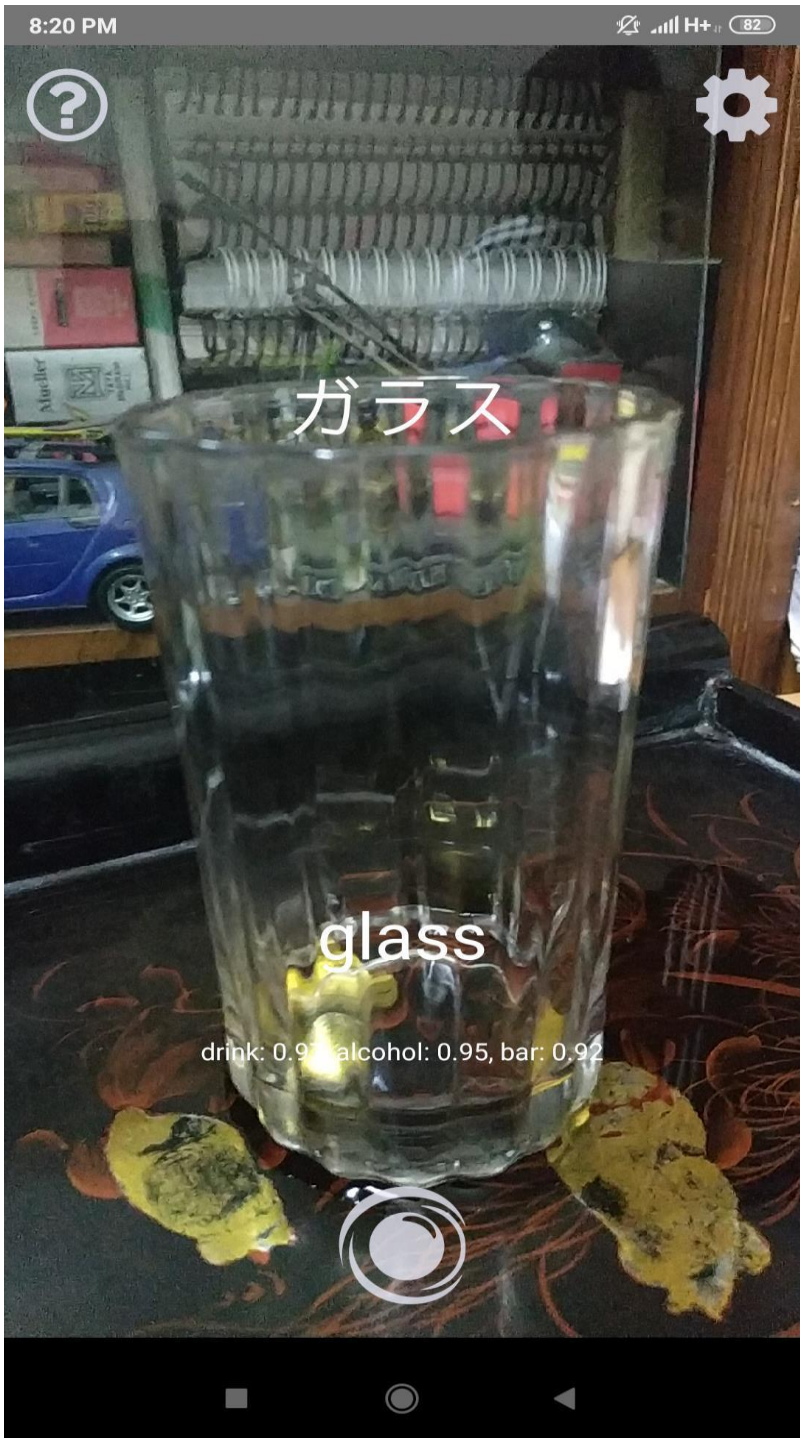


Figure 4.1.5: Output

4.2 Back-end Design

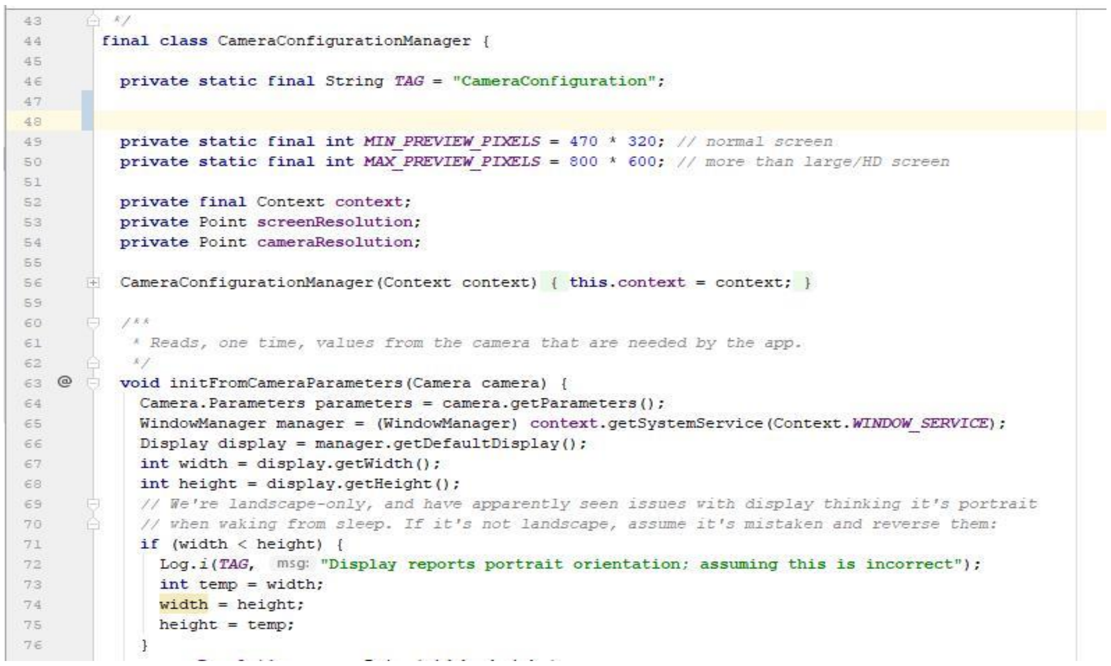
In this part we will discuss about coding. Most of the cases we used Java, XML and JavaScript. Back-end is such a part where all the logics are worked behind the system. The user cannot interact with this part or anything. The back-end design is the part where the actual work happens. This part is the most crucial part for a developing a system.

4.3 Main Camera Section

A class which deals with reading, parsing, and setting the camera parameters which are used to configure the camera hardware.

4.3.1 Camera Configuration Manager

This is bigger than the size of a small screen, which is still supported. The routine below will still select the default (presumably 320x240) size for these. This prevents accidental selection of very low resolution on some devices. In Figure 4.3.1 show that



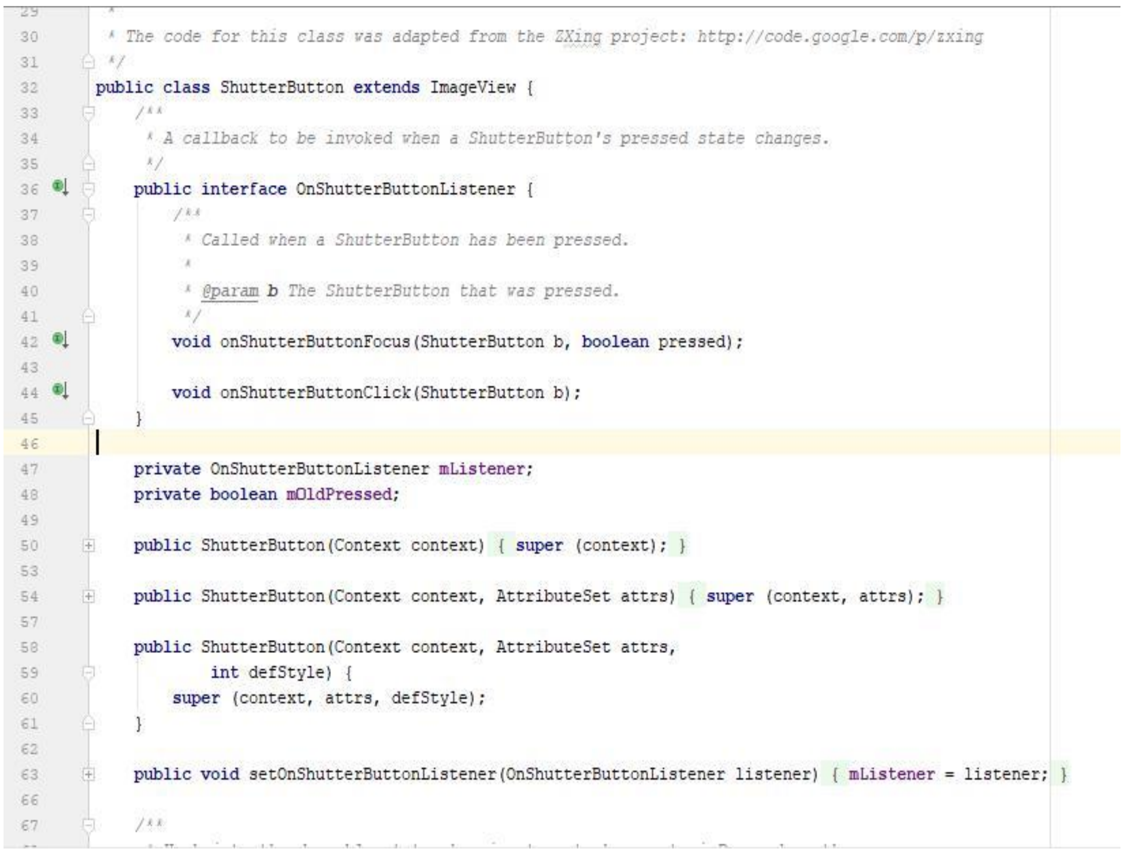
```
 */
final class CameraConfigurationManager {

  private static final String TAG = "CameraConfiguration";


  private static final int MIN_PREVIEW_PIXELS = 470 * 320; // normal screen
  private static final int MAX_PREVIEW_PIXELS = 800 * 600; // more than large/HD screen

  private final Context context;
  private Point screenResolution;
  private Point cameraResolution;

  CameraConfigurationManager(Context context) { this.context = context; }

  /**
   * Reads, one time, values from the camera that are needed by the app.
   */
  void initFromCameraParameters(Camera camera) {
    Camera.Parameters parameters = camera.getParameters();
    WindowManager manager = (WindowManager) context.getSystemService(Context.WINDOW_SERVICE);
    Display display = manager.getDefaultDisplay();
    int width = display.getWidth();
    int height = display.getHeight();
    // We're landscape-only, and have apparently seen issues with display thinking it's portrait
    // when waking from sleep. If it's not landscape, assume it's mistaken and reverse them:
    if (width < height) {
      Log.i(TAG,  msg: "Display reports portrait orientation; assuming this is incorrect");
      int temp = width;
      width = height;
      height = temp;
    }
```

Figure 4.3.1: Camera Configuration Manager

4.3.2 Shutter Button

A button designed to be used for the on-screen shutter button. It's currently an {@code ImageView} that can call a delegate when the pressed state changes.

When clicking using a trackball button, the view system changes the the drawable state before posting click notification. Figure 4.3.2: Shows Shutter Button

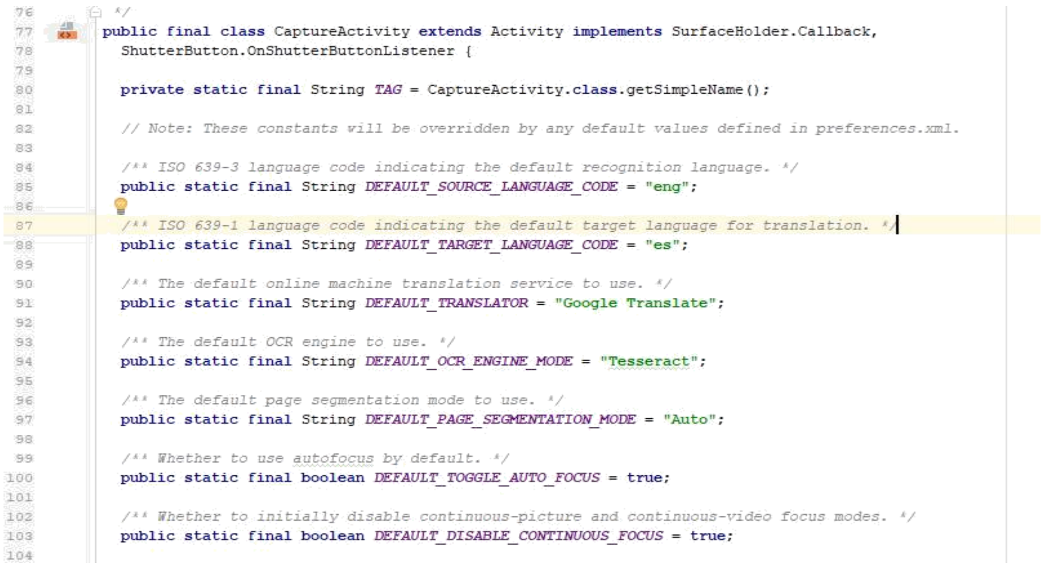


```
 *
 * The code for this class was adapted from the ZXing project: http://code.google.com/p/zxing
 */
public class ShutterButton extends ImageView {
    /**
     * A callback to be invoked when a ShutterButton's pressed state changes.
     */
    public interface OnShutterButtonListener {
        /**
         * Called when a ShutterButton has been pressed.
         *
         * @param b The ShutterButton that was pressed.
         */
        void onShutterButtonFocus(ShutterButton b, boolean pressed);

        void onShutterButtonClick(ShutterButton b);
    }

    private OnShutterButtonListener mListener;
    private boolean mOldPressed;

    public ShutterButton(Context context) { super (context); }

    public ShutterButton(Context context, AttributeSet attrs) { super (context, attrs); }

    public ShutterButton(Context context, AttributeSet attrs,
            int defStyle) {
        super (context, attrs, defStyle);
    }

    public void setOnShutterButtonListener(OnShutterButtonListener listener) { mListener = listener; }

    /**
```

Figure4.3.2: Shutter Button

4.3.3 Capture Activity

This activity opens the camera and does the actual scanning on a background thread. It draws a viewfinder to help the user place the text correctly, shows feedback as the image processing is happening, and then overlays the results when a scan is successful. In Figure 4.3.3 shows Capture Activity

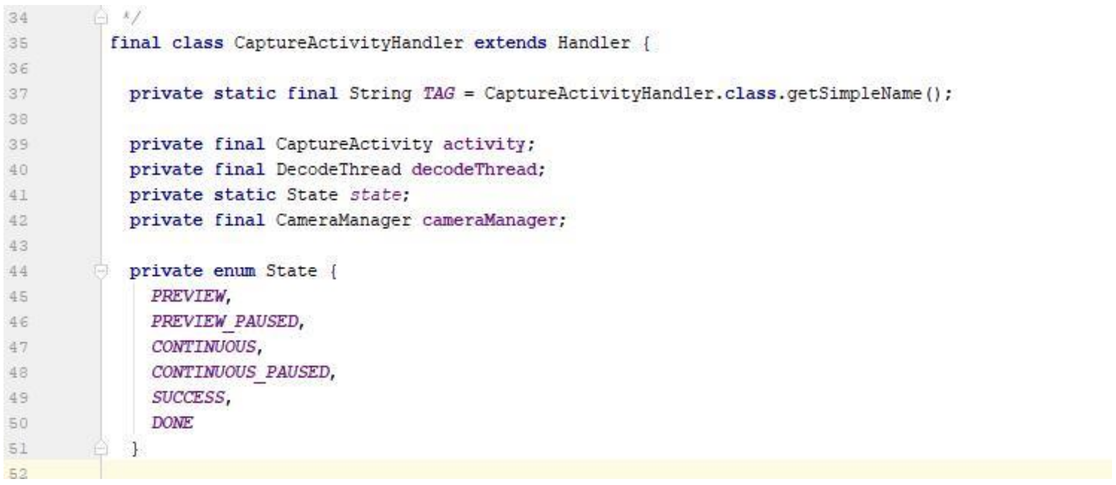


```java
 */
public final class CaptureActivity extends Activity implements SurfaceHolder.Callback,
  ShutterButton.OnShutterButtonListener {

  private static final String TAG = CaptureActivity.class.getSimpleName();

  // Note: These constants will be overridden by any default values defined in preferences.xml.

  /** ISO 639-3 language code indicating the default recognition language. */
  public static final String DEFAULT_SOURCE_LANGUAGE_CODE = "eng";

  /** ISO 639-1 language code indicating the default target language for translation. */
  public static final String DEFAULT_TARGET_LANGUAGE_CODE = "es";

  /** The default online machine translation service to use. */
  public static final String DEFAULT_TRANSLATOR = "Google Translate";

  /** The default OCR engine to use. */
  public static final String DEFAULT_OCR_ENGINE_MODE = "Tesseract";

  /** The default page segmentation mode to use. */
  public static final String DEFAULT_PAGE_SEGMENTATION_MODE = "Auto";

  /** Whether to use autofocus by default. */
  public static final boolean DEFAULT_TOGGLE_AUTO_FOCUS = true;

  /** Whether to initially disable continuous-picture and continuous-video focus modes. */
  public static final boolean DEFAULT_DISABLE_CONTINUOUS_FOCUS = true;

```

Figure 4.3.3: Capture Activity

4.3.4 Capture Activity Handler

In Figure 4.3.4 is a class handles all the messaging which comprises the state machine for capture.

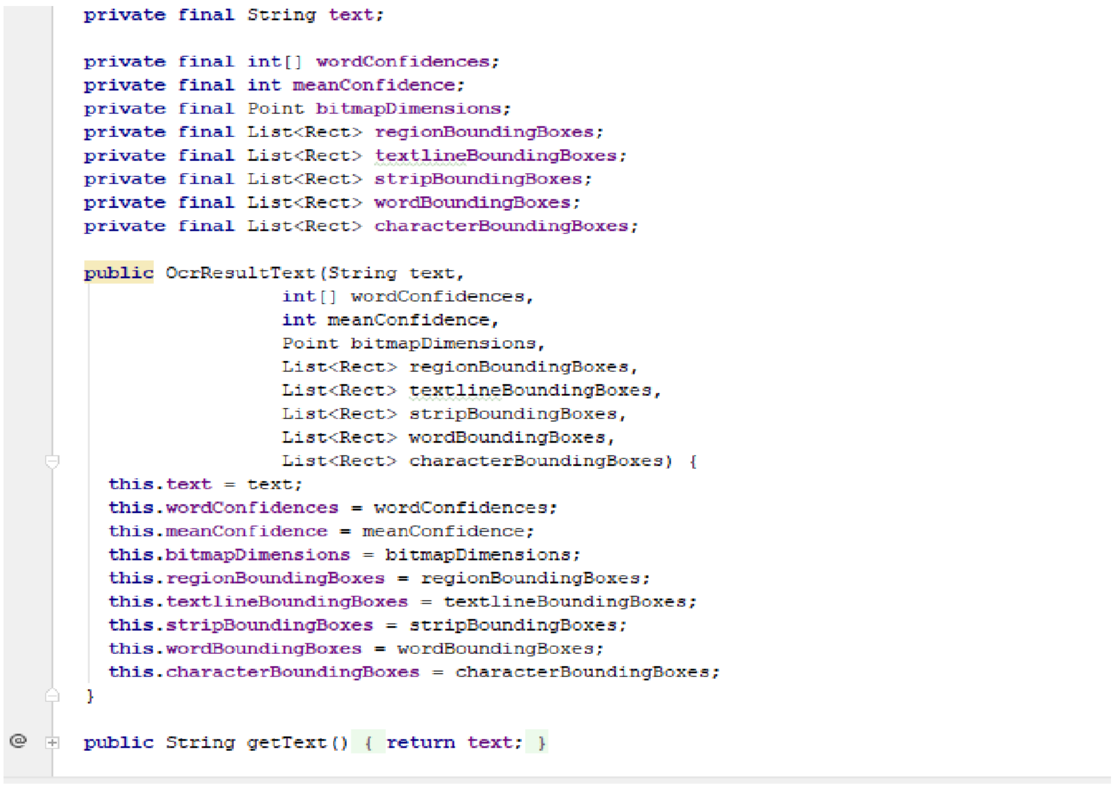


```java
 */
final class CaptureActivityHandler extends Handler {

  private static final String TAG = CaptureActivityHandler.class.getSimpleName();

  private final CaptureActivity activity;
  private final DecodeThread decodeThread;
  private static State state;
  private final CameraManager cameraManager;

  private enum State {
    PREVIEW,
    PREVIEW_PAUSED,
    CONTINUOUS,
    CONTINUOUS_PAUSED,
    SUCCESS,
    DONE
  }

```

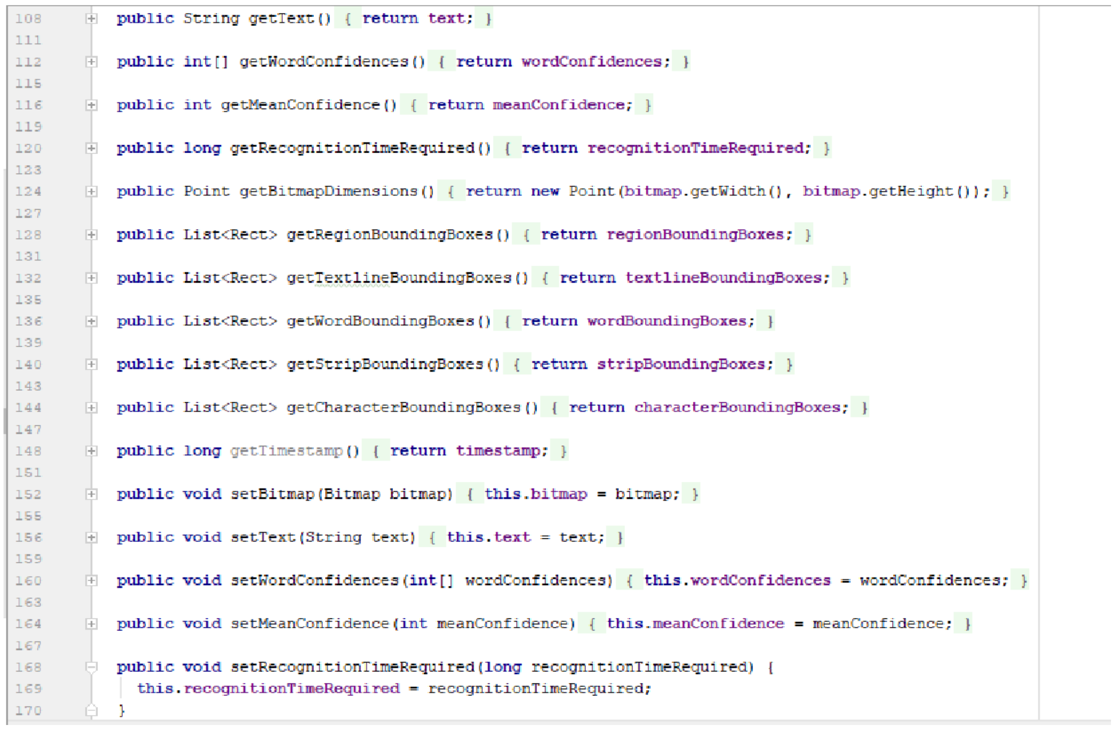
Figure 4.3.4: Capture Activity Handler

4.3.5 Result Of Text

Figure 4.3.5 is an Encapsulates text and its character/word coordinates resulting from squeezenet.



```java
private final String text;

private final int[] wordConfidences;
private final int meanConfidence;
private final Point bitmapDimensions;
private final List<Rect> regionBoundingBoxes;
private final List<Rect> textlineBoundingBoxes;
private final List<Rect> stripBoundingBoxes;
private final List<Rect> wordBoundingBoxes;
private final List<Rect> characterBoundingBoxes;

public OcrResultText(String text,
                     int[] wordConfidences,
                     int meanConfidence,
                     Point bitmapDimensions,
                     List<Rect> regionBoundingBoxes,
                     List<Rect> textlineBoundingBoxes,
                     List<Rect> stripBoundingBoxes,
                     List<Rect> wordBoundingBoxes,
                     List<Rect> characterBoundingBoxes) {
  this.text = text;
  this.wordConfidences = wordConfidences;
  this.meanConfidence = meanConfidence;
  this.bitmapDimensions = bitmapDimensions;
  this.regionBoundingBoxes = regionBoundingBoxes;
  this.textlineBoundingBoxes = textlineBoundingBoxes;
  this.stripBoundingBoxes = stripBoundingBoxes;
  this.wordBoundingBoxes = wordBoundingBoxes;
  this.characterBoundingBoxes = characterBoundingBoxes;
}

public String getText() { return text; }
```

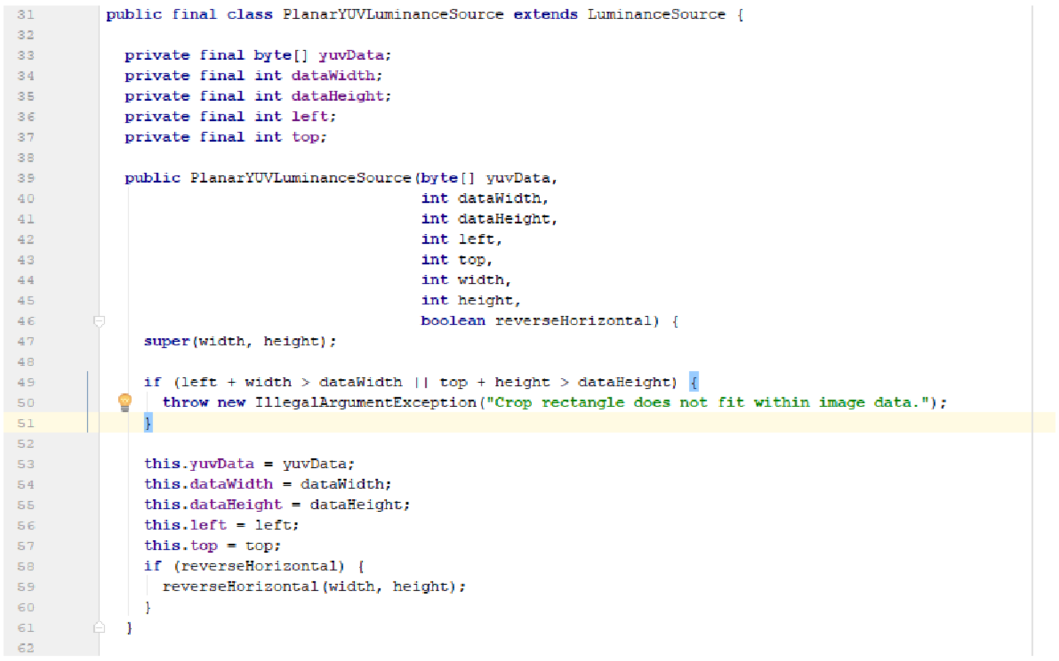


```java
public String getText() { return text; }

public int[] getWordConfidences() { return wordConfidences; }

public int getMeanConfidence() { return meanConfidence; }

public long getRecognitionTimeRequired() { return recognitionTimeRequired; }

public Point getBitmapDimensions() { return new Point(bitmap.getWidth(), bitmap.getHeight()); }

public List<Rect> getRegionBoundingBoxes() { return regionBoundingBoxes; }

public List<Rect> getTextlineBoundingBoxes() { return textlineBoundingBoxes; }

public List<Rect> getWordBoundingBoxes() { return wordBoundingBoxes; }

public List<Rect> getStripBoundingBoxes() { return stripBoundingBoxes; }

public List<Rect> getCharacterBoundingBoxes() { return characterBoundingBoxes; }

public long getTimestamp() { return timestamp; }

public void setBitmap(Bitmap bitmap) { this.bitmap = bitmap; }

public void setText(String text) { this.text = text; }

public void setWordConfidences(int[] wordConfidences) { this.wordConfidences = wordConfidences; }

public void setMeanConfidence(int meanConfidence) { this.meanConfidence = meanConfidence; }

public void setRecognitionTimeRequired(long recognitionTimeRequired) {
  this.recognitionTimeRequired = recognitionTimeRequired;
}
```

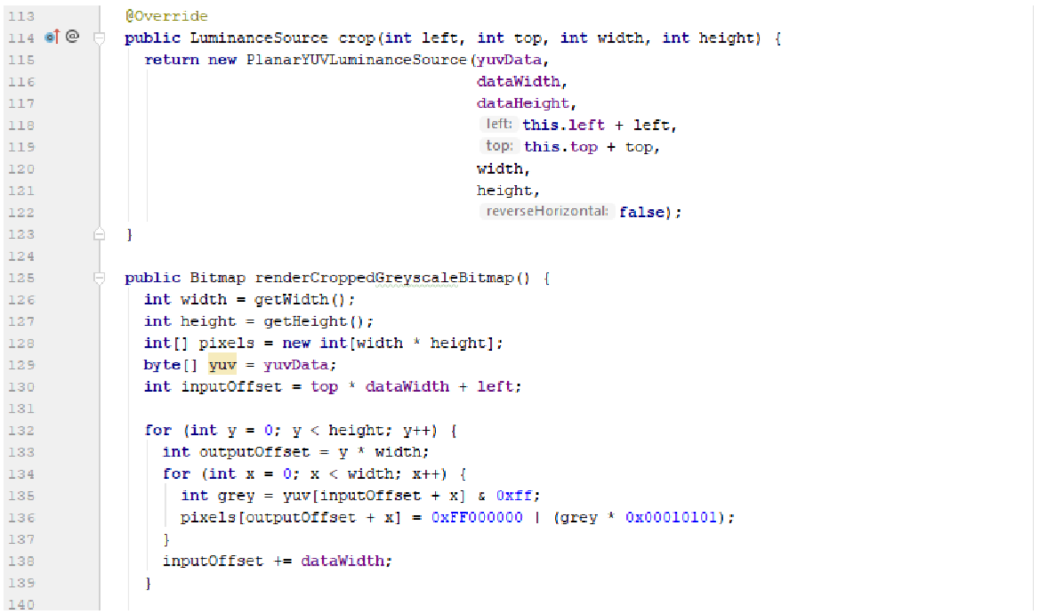
Figure 4.3.5: Result of Text

4.3.6 PlanarYUVLuminanceSource

This object extends LuminanceSource around an array of YUV data returned from the camera driver with the option to crop to a rectangle within the full data. This can be used to exclude superfluous pixels around the perimeter and speed up decoding. It works for any pixel format where the Y channel is planar and appears first, including YCbCr_420_SP and YCbCr_422_SP. Figure 4.3.6 Shows Planar YUVLuminance Source.



```java
public final class PlanarYUVLuminanceSource extends LuminanceSource {

  private final byte[] yuvData;
  private final int dataWidth;
  private final int dataHeight;
  private final int left;
  private final int top;

  public PlanarYUVLuminanceSource(byte[] yuvData,
                                  int dataWidth,
                                  int dataHeight,
                                  int left,
                                  int top,
                                  int width,
                                  int height,
                                  boolean reverseHorizontal) {
    super(width, height);

    if (left + width > dataWidth || top + height > dataHeight) {
      throw new IllegalArgumentException("Crop rectangle does not fit within image data.");
    }

    this.yuvData = yuvData;
    this.dataWidth = dataWidth;
    this.dataHeight = dataHeight;
    this.left = left;
    this.top = top;
    if (reverseHorizontal) {
      reverseHorizontal(width, height);
    }
  }

```



```java
  @Override
  public LuminanceSource crop(int left, int top, int width, int height) {
    return new PlanarYUVLuminanceSource(yuvData,
                                        dataWidth,
                                        dataHeight,
                                        left: this.left + left,
                                        top: this.top + top,
                                        width,
                                        height,
                                        reverseHorizontal: false);
  }

  public Bitmap renderCroppedGreyscaleBitmap() {
    int width = getWidth();
    int height = getHeight();
    int[] pixels = new int[width * height];
    byte[] yuv = yuvData;
    int inputOffset = top * dataWidth + left;

    for (int y = 0; y < height; y++) {
      int outputOffset = y * width;
      for (int x = 0; x < width; x++) {
        int grey = yuv[inputOffset + x] & 0xff;
        pixels[outputOffset + x] = 0xFF000000 | (grey * 0x00010101);
      }
      inputOffset += dataWidth;
    }

```

Figure 4.3.6: Planar YUVLuminance Source.

4.4 Implementation of Requirements

- UI design implemented with SqueezeDet
- Our model is trained with KITTI benchmark. The KITTI data-set has over 10 Gigabytes of well-labeled data for object detection purpose.
- Invalid data input displays error message.
- Required data fields are checked to get full information.
- Java, XML for specific design.

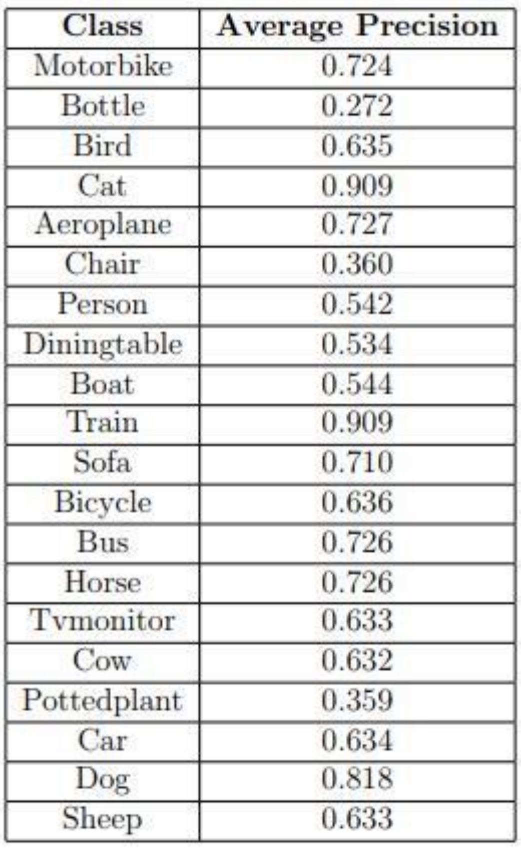
CHAPTER 5

IMPLEMENTATION AND TESTING

5.1 Implementation of Database

The data-set we use is The KITTI Vision Benchmark Suite, which is made for academic use in the area of autonomous driving. For our target, we use the object detection data-set, which contains 7481 training images and 7518 test images. Total 80256 objects are labeled for this data-set. The distribution of object number in the training data-set is shown in Table 5.1 51865 objects are labeled

Table 5.1: Average detection precisions



Class	Average Precision
Motorbike	0.724
Bottle	0.272
Bird	0.635
Cat	0.909
Aeroplane	0.727
Chair	0.360
Person	0.542
Diningtable	0.534
Boat	0.544
Train	0.909
Sofa	0.710
Bicycle	0.636
Bus	0.726
Horse	0.726
Tvmonitor	0.633
Cow	0.632
Pottedplant	0.359
Car	0.634
Dog	0.818
Sheep	0.633

5.2 Implementation of Front-end Design

It is very tough and challenging to create a simple UI design for the users, we tried to make as simple as possible. In this project, we trained a CNN object detection model at desktop platform and applied the trained model into a mobile platform. As a baseline, we have a running Android app that runs our CNN model trained by Tensorflow.

This deep compression method are not explored in this project due to time constraint, but it worth looking into in future development. Smaller model is not only beneficial for storage capacity, it should also be beneficial for computing efficiency.

5.3 Implementation of Interaction

Here to make our system we have implemented responsive UI for better user experience. The system design of our app is user friendly. Any kind of user can operate it easily.

5.4 Testing Implementation

Testing implementation is a procedure of testing upcoming implementation of a system, where the tester or system architect will see cases and specification, whether it is implemented or have limitations.

Table 5.2: Test case evaluation

Test Case	**Test Input**	**Expected outcome**	**Obtained outcome**	**Pass / fail**	**Tested on**
1. Detect the full object	Open app and focus on an object	Successfully detected and show the name of the object	Successfully detected and show the name of the object	Pass	15-09-2018
2. Detect The part of an object	Open app and focus on a part of an object	Successfully detected and show the name of the object	Successfully detected and show the name of the object	Pass	02-10-2019

3. Show the result in any language	Select the language and focus on an object	Successfully detected and show the name of the object on that specific language	Successfully detected and show the name of the object on that specific language	Pass	25-02-2019

5.5 Test Result and Report

The test was externally done by our supervisor and some of our qualified classmates in extreme situations and the result came out very satisfactory. We also tested the system ourselves a lot of time throughout the development of the project. We found problems and we tried our best to solve those to create a better and hassle free user interface. So finally we are very satisfied and confident enough about the testing and outcomes.

We have added some of the test cases above and more others we did which we could not add. But we can assure that the system is already usable at its best condition.

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 Discussion and Conclusion

We have successfully implemented the system “An Android Based Real Time Object Classification and Detection” with the help of various links and tools. We have been able to provide a system which is running online. We have been successful in our attempt to take care of needs of people. Finally, we hope that this will go a long way in popularizing the organization and making its work enrollment.

6.2 Scope for Further Development

The project is largely created to satiate user appetite and user experience. So the design and functionality is limited to user interaction only. But we aspire to create a better admin experience too including more features in the dashboard and give a better visual experience. We are looking forward to adding

- Add voice
- Make it offline app
- Make IOS app

REFERENCES

[1] R-CNN (Object Detection) available at << https://medium.com/coinmonks/review-r-cnn-object-detection-b476aba290d1 >>, last accessed on 10-12-2018 at 01:00pm.

[2] Faster R-CNN (object detection) implemented by Keras for custom data from Google's Open Images Dataset V4, available at << https://towardsdatascience.com/faster-r-cnn-object-detection-implemented-by-keras-for-custom-data-from-googles-open-images-125f62b9141a>>, last accessed on 28-01-2019 at 09:15pm.

[3] Understanding Fast RCNN, available at << https://www.analyticsvidhya.com/blog/2018/10/a-step-by-step-introduction-to-the-basic-object-detection-algorithms-part-1/ >>, last accessed on 06-11-2018 at 10:20pm.

[4] Single Shot Detector (SSD), available at << https://cv-tricks.com/object-detection/faster-r-cnn-yolo-ssd/ >>, last accessed on 28-02-2019 at 11:00pm.

[5] R-CNN: Regions with CNN diagram, available at << https://arxiv.org/pdf/1311.2524.pdf >>, last accessed on 24-12-2018 at 10:30pm

[6] Non-Maximum Suppression Algorithm << http://cs231n.stanford.edu/reports/2017/pdfs/627.pdf>>, last accessed on 16-02-2019 at 09:30pm

[7] Trainning label of object data set, available at <<http://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d >>, last accessed on 06-01-2019 at 10.00am

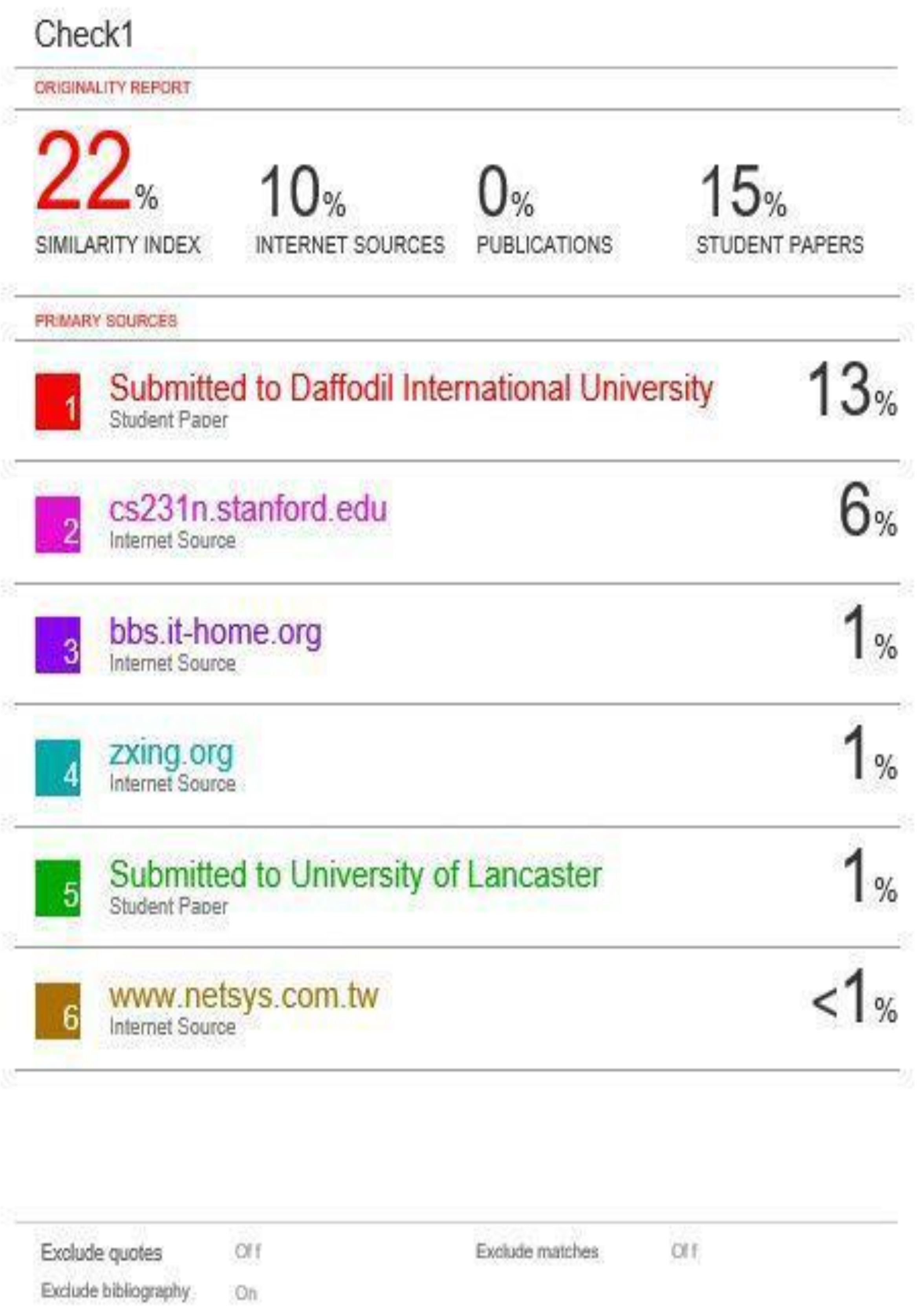
[8] Benchmark Results on KITTI, available at << https://github.com/chuanqi305/SqueezeNet-SSD >>, last accessed on 30-03-2019 at 11.15am

[9] Average precision Diagrams designed at <<https://www.cse.iitb.ac.in/~pratikm/projectPages/objectDetection/downloads/report.pdf >>, last accessed on 30-3-2019 at 12.00pm

APPENDIX

From Fall-2017 semester we had started our journey to make a system, where the users can find the name of any object in any language. For implementing and monitoring our system we followed the model with all the hard work and spending a lot of time we finally were able to reach our ambition at last. So we believe that our “An Android Based Real Time Object Classification and Detection” will be a positive and effective thing for the users. And we will continuously upgrade our system as much as possible.

PLAGARISM REPORT



Check1

ORIGINALITY REPORT

22%	10%	0%	15%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Daffodil International University Student Paper	13%
2	cs231n.stanford.edu Internet Source	6%
3	bbs.it-home.org Internet Source	1%
4	zxing.org Internet Source	1%
5	Submitted to University of Lancaster Student Paper	1%
6	www.netsys.com.tw Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		