Table of Contents

Contents

[Abstract 2](#_Toc24015439)

[Introduction 2](#_Toc24015440)

[Problem Statement 2](#_Toc24015441)

[Previous Solutions 3](#_Toc24015442)

[Aim 3](#_Toc24015443)

[Technologies Used 3](#_Toc24015444)

[Libraries Used 3](#_Toc24015445)

[Methodology 4](#_Toc24015446)

Image Classification

*Semester Initial Project Report*

*Group Members: F16CS06 | F16CS18 | F16CS72*

# Abstract

Image classification has been so far carried out almost exclusively using sophisticated and expensive software solutions and devices. Mobile devices have evolved to the extent that they can be used for such purpose, but the algorithms and systems that provide such functionalities in real time are rare. We will show the theoretical basis of the image classification, implementation of the algorithm using the android studio and google libraries and the practical experience in such an approach.

# Introduction

Image classification is a complex process, the accuracy of which is mainly related to the characteristics of the dataset, complexity of the problem under analysis, and the robustness of the classification algorithm.

Image classification refers to the task of extracting information classes from a multiband raster image. The resulting raster from image classification can be used to create thematic maps. Depending on the interaction between the analyst and the computer during classification, there are two types of classification: supervised and unsupervised.

ML Kit is a mobile SDK that brings Google's machine learning expertise to Android and iOS apps in a powerful yet easy-to-use package. Whether you're new or experienced in machine learning, you can implement the functionality you need in just a few lines of code. There's no need to have deep knowledge of neural networks or model optimization to get started. On the other hand, if you are an experienced ML developer, ML Kit provides convenient APIs that help you use your custom TensorFlow Lite models in your mobile apps.

# Problem Statement

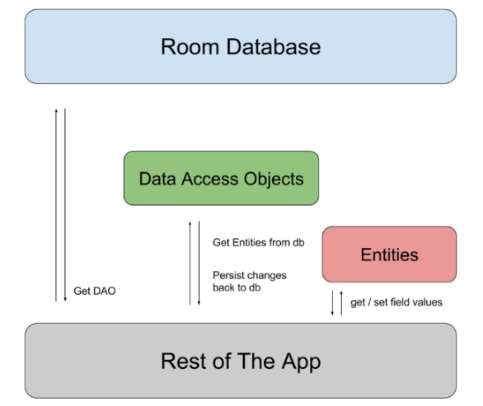
We see different things everyday that we come across for the first time. And we know the thing by its name the second time we see that. But what would happen if you don’t have to wait for the 2nd time? Using Image Classification, we can use our android device’ camera to ensure that we have the correct name of every objects at once we see the object.

# Previous Solutions

There is certain many different application available on Play Store, all waiting to be downloaded and be used. But the main drawback of these application is that they need very high-speed internet support in order to get the data from their servers.

This problem gives birth to delay in catching and integrating the information for a useful purpose. Plus, in countries, like ours, we do not have stable and high-speed internet connection all the time, therefore, such application, that are already available in the market, are not feasible to be used by local users.

# Aim

In order to fight with all the above-mentioned problems, we must use a reliable, light bandwidth, ORM Database that can be accessed without the being in a range of high-speed internet connection. As of now, Android’s ORM Database arranges a vast quantity of reusable data formats and sheets that allow different independent and dependent libraries and repositories to work upon and function without any hindrances.

We aim to use, Android’s ORM database, along with Firebase Machine Learning Libraries, integrated with a preplanned and pre-trained Machine Learning App that would allow our database to use many algorithms and implement them in our image classification app

# Technologies Used

We have tried to use as a smaller number of technologies as possible. Still, our technologies will be:

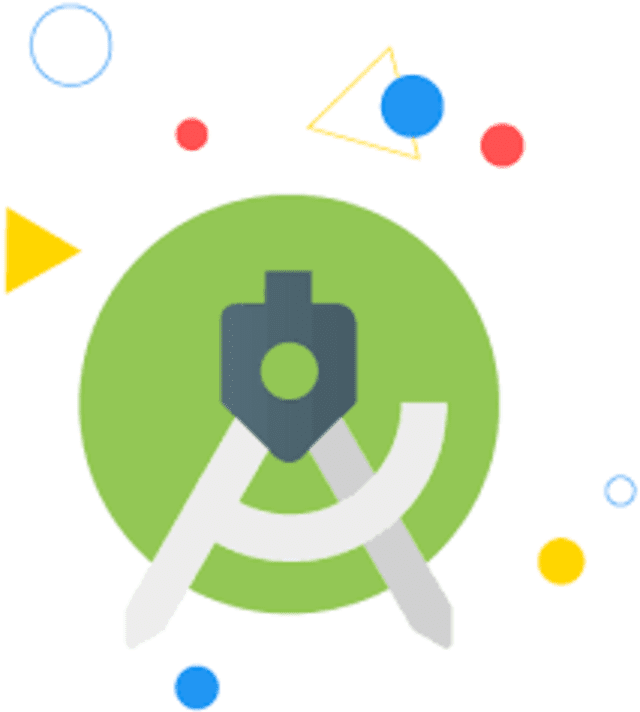
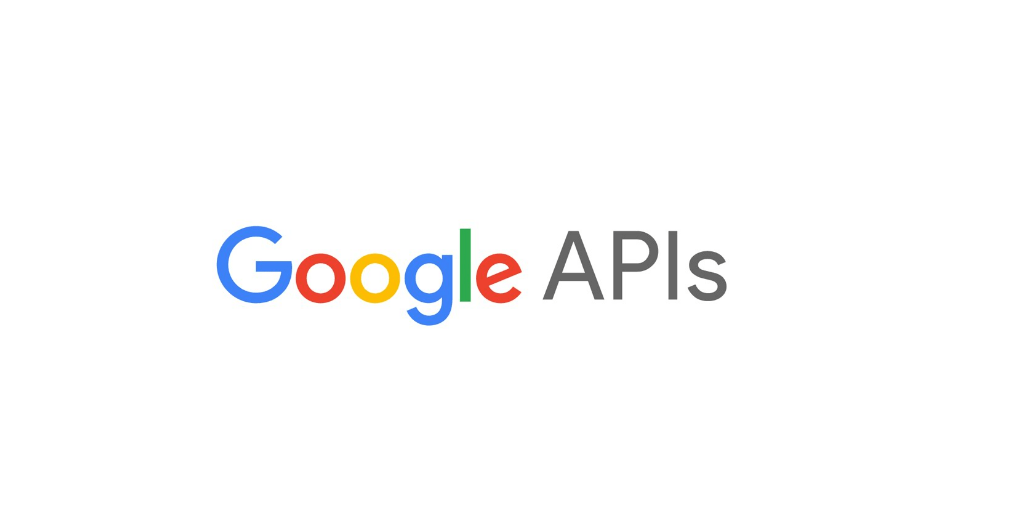
 

Figure 2Android Development Kit Figure 3 Google APIs

Figure 2 Firebase Figure 4 TensorFlow Lite

# Libraries Used

Libraries used in this project will be:

* Google ML Kit
* Firebase Optical Engine
* Android Firebase Dependencies
* Android Object-Relational Mapping (ORM) Database
* Google Face Detection API

# Methodology

The app will use the Room database to get the data access objects, or DAOs, associated with that database. The app will then use each DAO to get entities from the database and save any changes to those entities back to the database. Finally, the app will use an entity to get and set values that correspond to table columns within the database.

The resulting data will then be shared with the Firebase ML Framework in order to make it work and identify from the ORM Database. Once done, the ORM database will send the result to out Java code and we will see the out on our screen as in the images below:

Data shown on the screen

Result sorted and sent to Java and XML code for output

Returns the result to Firebase ML Framework

ORM Checks within its records

Data sent to ORM Database

Data Captured by Camera

As for Image detection is concerned, on pressing the button, you will be taken to your mobile’s camera and Google’s ORM Database will retrieve the image and slice it. It will then scan the image, slice by slice and will examine it for any traces of a human face. Then it will send the data to the .JSON file in our internal server. The Application will then show a resulting image for the given image with an addition on a small red square indicating the presence of human face. At a maximum level, the application can detect 07 (Seven) faces at once.