Android Application For Image and Face Recognition using Watson Service

STUDENT NAME

Mohamed BOUDOUAR

Examiner
Pr. Oussama REDA

January 2019

Mobile and Cloud Computing
Master IAO
Department of Computer Science
Faculty of sciences
Mohammed V university of Rabat

Abstract

Mobile computing in conjunction with Mobile web services drives a strong approach where the limitations of mobile devices may possibly be tackled. Mobile Web Services are based on two types of technologies; SOAP and REST, which works with the existing protocols to develop Web services. Through this project we will see how to take advantage of cloud based solutions for the aim of designing mobile applications using the android platform and keeping the user experience (UX) running smoothly. with the help of the concurrency mechanism of android platform.

Table of contents

1 Introduction

Objects recognition is the task of identifying an already detected object as a known or unknown object using machine learning techniques. With the IBM Watson Visual Recognition service, creating mobile applications that can accurately detect and analyze objects in images is easier than ever.

The purpose of this project was the design and implementation process of an Android application which uses a cloud web service. The idea of the application was to use one of the IBM Watson services. Specifically, Our project involved us to get familiar with the android concurrency techniques such as AsyncTask in one hand, and consuming cloud based solutions of IBM platform on the other side. I basically focused on developing an intuitive, clean and simple android application for visual recognition and face recognition. Where the scanner application serves as a source of data which allows users to choose a picture from the gallery, and performing the calls to the back-end service of IBM Watson and with the help of Watson visual recognition service, the result of the service gives us a pretty good idea about the image's content.

The theoretical part of the project discussed the Android platform, its components and the IBM Watson services. Then we will proceed to the project design, to show the different stages and parts of the application with the requirements of the Watson services. At the end we will conclude by the goals achieved and some perspectives of the project.

2 Literature Review

Android operating system **:** is one of the most widely used mobile Operating System these days. it is based on the Linux kernel which is primarily designed for smartphones and tablets. Since Android is an open source it has become the fastest growing mobile operating system.[1]

***Android components*:**

By design, There are following four main components that can be used within an Android application, and we distinguish between[2] :

1. *Activities* : They dictate the UI and handle the user interaction to the smartphone screen.
2. Services : They handle background processing associated with an application.
3. Broadcast Receivers :They handle communication between Android OS and applications.
4. Content Providers : They handle data and database management issues.

Android Concurrency :

Concurrency by definition is the ability to run several programs or several parts of a program in parallel. If a time consuming task can be performed asynchronously or in parallel, this improve the throughput and the interactivity of the application. Android provides several API's to do asynchronous processing, one of the efficient ways to deal with UI threads use Android AsyncTasks, which is the best known of the asynchronous API, it is easy to implement and returns results on the main thread and Speaking of the implementation, the AsyncTask executes everything in doInBackground() inside of another thread, which does not have access to the GUI where the views are[3].

IBM Watson service : *The Watson Visual Recognition service is a powerful business tool, as it facilitates the creation of smart applications that can extract information from visual data without the need to develop AI software. moreover Watson Visual Recognition service helps developers to accurately analyze, classify, and train images using machine learning. by using a set of built-in models in service that provides highly accurate results without training[4].*

3 Functional Requirements

1. ***On the first launch of the application, the user is presented with a principal screen with a menu to choose either objects recognition or human face recognition.***
2. ***We have to provide the user the choice to navigate through the gallery of the phone.***
3. ***Display the selected image in the screen of our application.***
4. ***provide the user with a button to send the image to backend service of the IBM cloud.***
5. ***the user should be able to see the result of the processing***

4 Project Design

4.1 Architecture Overview

The design of the intended product is explained through the architecture below. The figure explains the overall interactions of the user and the cloud service.

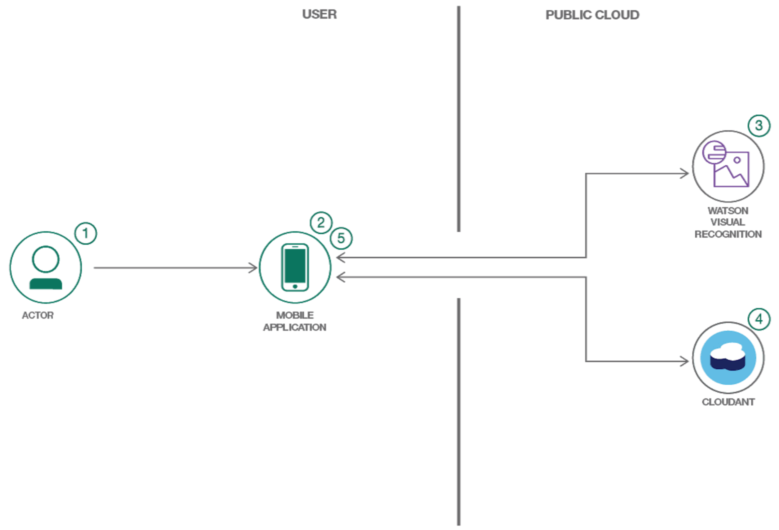


Figure 42 Architecture Overview

4.2 Project requirements

To be able to develop an android application, which interacts with the watson service, the following requirements are mandatory:

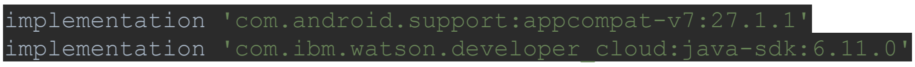
1. an IBM Cloud account
2. Android Studio.
3. device or emulator running Android.

4.3 Activating the Visual Recognition Service

In Watson services Platform, the Visual Recognition service must be manually activated before it can be used in an app. So we log in to the IBM console and navigate to Services > Watson. In the page that opens, then we press the Create Watson service button. After that From the list of available services shown next, we choose Visual Recognition. Once the service is ready, an API key will be generated.

4.4 Project Setup

we will be using the Watson Java and Android SDKs while interacting with the Visual Recognition service. Therefore, we have to provide the following dependencies to our application module's build.gradle file:



```
implementation 'com.android.support:appcompat-v7:27.1.1'
implementation 'com.ibm.watson.developer_cloud:java-sdk:6.11.0'
```

4.4.1 Setup Manifest file

1. Additionally, our application will need access to the external storage media, so we have to add the request WRITE_EXTERNAL_STORAGE permission.
2. we will need also the INTERNET permission.

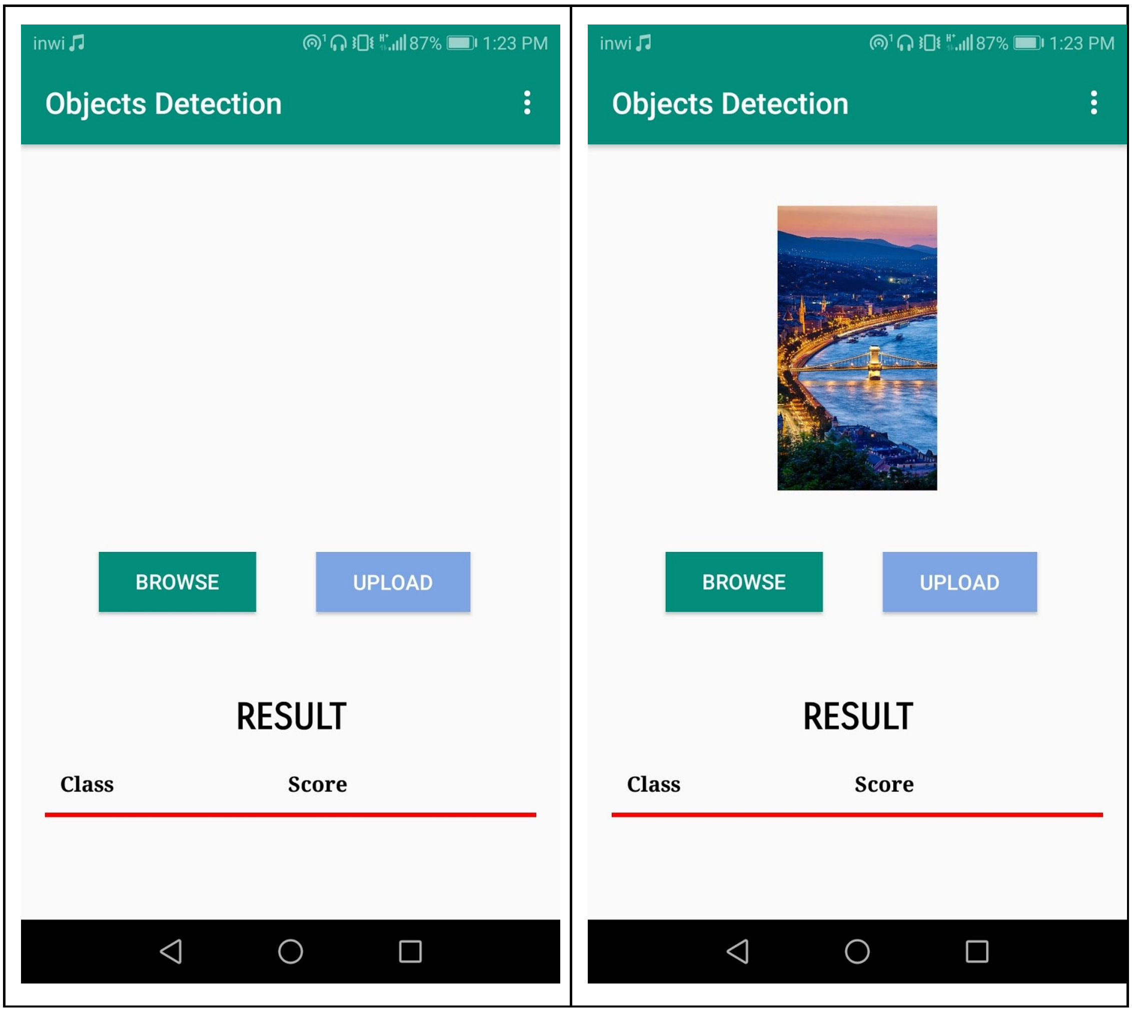
5 Implementation

5.1 Using Asynctask

By its definition, AsyncTask enables proper and easy use of the UI thread. This class allows us to perform background operations and publish results on the UI thread without having to manipulate threads and/or handlers, In our case the class is useful to send the image to the watson service, and display the result in our UI thread, Therefor, I have used two classes for Object Recognition and Face Recognition.

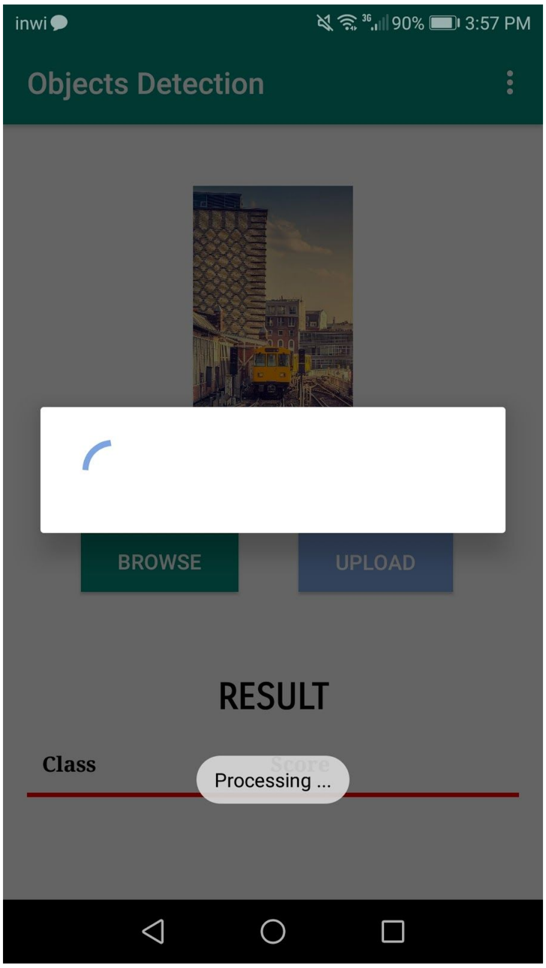
5.2 Welcome Screen

We started the project by creating the welcome stage for the user, for the aim of choosing the image from the gallery and display it in our screen.



5.3 Objects detector

To process items in the picture, we need to pass the path of the selected image to AsyncTask, then we send the image to the service to be processed. In addition we have to display Dialogue to inform the user about the process for the sake to not block the UI.



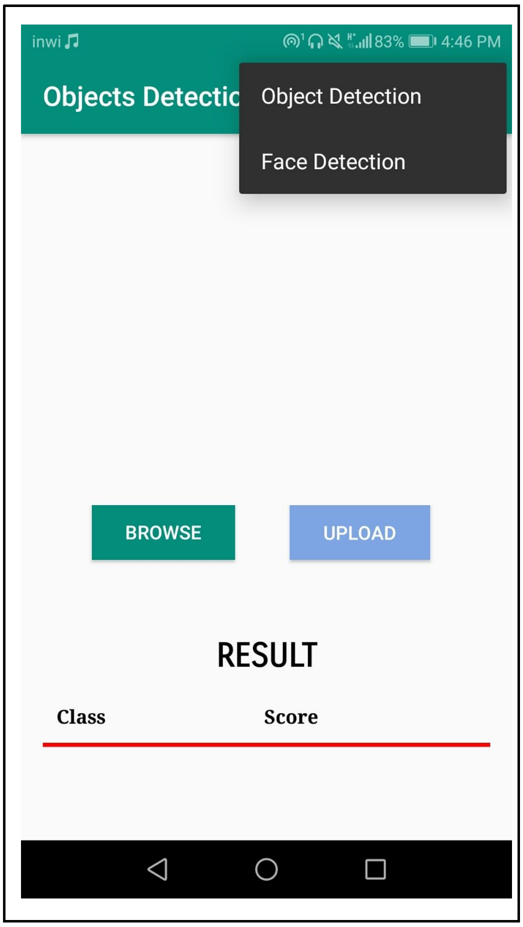
to authenticate to the API. we pass either a bearer token in an Authorization header or an apikey. In our case we used the apiKey, with the use of the SDK, it manages the lifecycle of the tokens. the we pass the image to the classify() method of the VisualRecognition object. After creating a ClassifyImagesOptions object by using then ClassifyImagesOptions.Builder class.

5.4 Application Overview

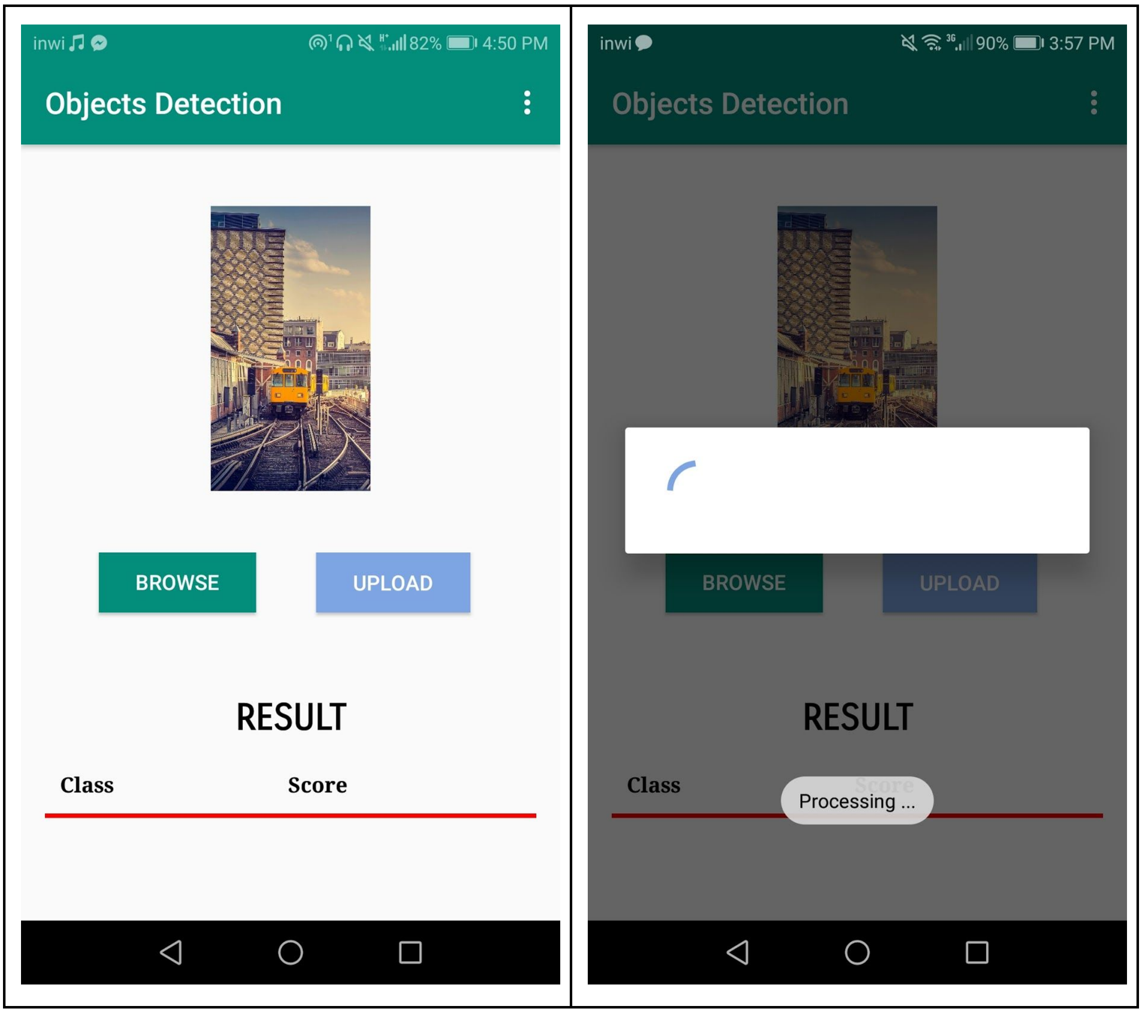
In the following section we will present screens and functionalities implemented

5.4.1 Object Recognition

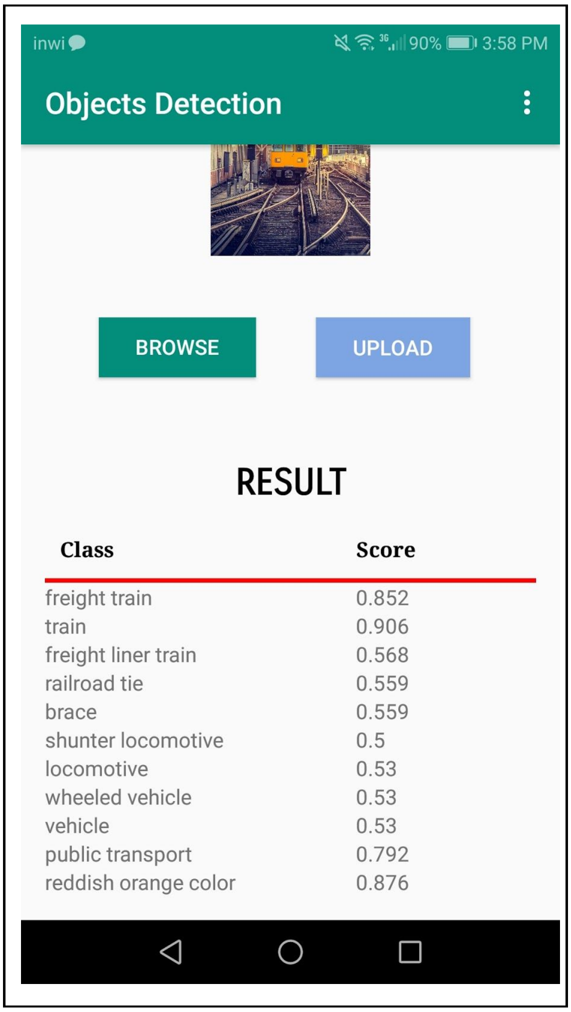
1. we start by Choosing either Objects Detector or Face Recognition through the menu:



2. Then we choose objects detection, we browse through the gallery to select the image :



3. Display the result to the user :

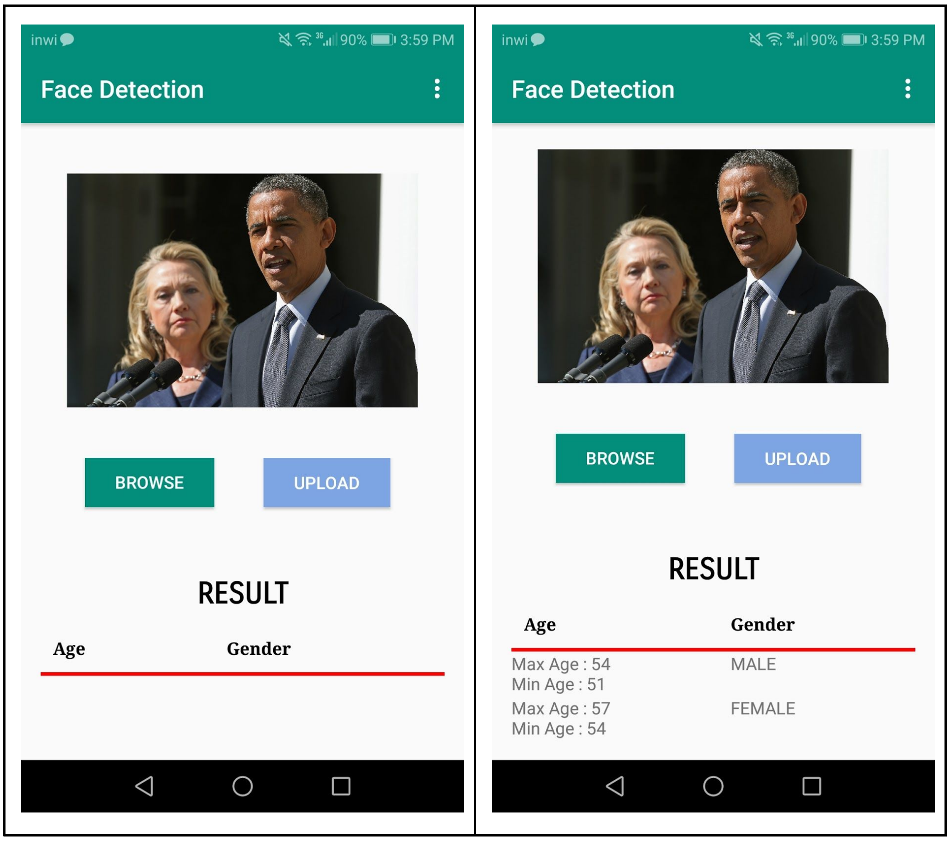


RESULT

Class	Score
freight train	0.852
train	0.906
freight liner train	0.568
railroad tie	0.559
brace	0.559
shunter locomotive	0.5
locomotive	0.53
wheeled vehicle	0.53
vehicle	0.53
public transport	0.792
reddish orange color	0.876

5.4.2 Face Recognition

1. we choose the image from the gallery
2. we upload the image.
3. we display the results to the user.



RESULT

Age	Gender

RESULT

Age	Gender
Max Age : 54 Min Age : 51	MALE
Max Age : 57 Min Age : 54	FEMALE

6 Conclusions & Future Work

In the project, I have investigated realization of an android application with IBM cloud service and dealing with Android concurrency mechanism, and it has turned out to be an excellent opportunity for me to get familiar with cloud based solutions of IBM. therefore certainly It will help me in my next challenges and future work.

References

[1] https://www.tutorialspoint.com/android/

[2] www.techplayon.com/applications-component/

[3] https://android.jlelse.eu/8-ways-to-do-asynchronous-processing-in-android-and-counting-f634dc6fae4e

[4] https://www.ibm.com/watson/

[5] https://code.tutsplus.com/tutorials/using-the-ibm-watson-visual-recognition-api-in-android-apps--cms-29542

Appendix : Code Source:

DoInBackground :

```
protected ClassifiedImages doInBackground(File... files) {
        // Creating an object of type Option using the key provided by IBM watson
        IamOptions options = new IamOptions.Builder()
                .apiKey("len5TuXtFKngGr8bTrUjleNly03D9zJTBV_IS1Yo6aMQ")
                .build();

        VisualRecognition service = new VisualRecognition("2018-03-19", options);
        InputStream imagesStream = null;
        try {
            // creating an image stream object from the file contains the image 
selected
            imagesStream = new FileInputStream(files[0]);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        // Classifier
        ClassifyOptions classifyOptions = new ClassifyOptions.Builder()
                .imagesFile(imagesStream)
                .imagesFilename(files[0].getName())
                .classifierIds(Arrays.asList("default"))
                .build();
        publishProgress(10);
        ClassifiedImages result =null;
        int flag=-1;
        while (flag<0){
            result = service.classify(classifyOptions).execute();
            flag=1;
        }
        …
        …….
```