Counting Crops using ML and CV

A Project Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of

Bachelor of Technology

by

Kundan Pal
(111901028)

INDIAN INSTITUTE
OF TECHNOLOGY
PALAKKAD

COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY PALAKKAD

CERTIFICATE

*This is to certify that the work contained in the project entitled "**Counting Crops using ML and CV**" is a bonafide work of **Kundan Pal (Roll No. 111901028)**, carried out in the Department of Computer Science and Engineering, Indian Institute of Technology Palakkad under my guidance and that it has not been submitted elsewhere for a degree.*

Mrinal Kanti Das

Assistant Professor

Department of Computer Science & Engineering

Indian Institute of Technology Palakkad

Acknowledgements

I would like to thank Dr. Mrinal Kanti Das for providing his constant support throughout the project. I would like to extend my thanks to Dr. Mrinal and Dr. Srimanta for providing me the opportunity to work on this project.

Contents

Chapter 1

Introduction

Counting Crops using Ml and CV : Main functionality of this project is to count the number of crops present in the provided image. The image has to be taken as input from the application and this image input will go to Yolo model which will do image processing give feedback about the image. The logic to count the crops is implemented in python using Machine learning and Computer vision logic which follows the approach of YOLO model.

1.1 Task

My task is to develop a simple android application. The purpose of this application is to provide an interface so that user can get the result from the YOLO model. This application will send image data to the server and will get the result from the server.
YOLO(you only look once) uses real-time object recognition algorithm for image processing.

1.2 Expected Requirements

This application will take image from the user via accessing phone's camera and will display the number of crops present in the image given. In the back-end it will make an API call to a python interface which will process the image and will give number of crops as output. This output will be displayed in user's application interface. A server needs to be setup separately from the application which will take the image for the processing. we need this server for making API call and to process the image using YOLO model.

Chapter 2

Review of Prior Works

2.1 Observation

The logic for counting crops in python is already implemented in Yolo-model. The python code takes image as input and process it. And after processing it provides number of crops as output.

Previously seniors have tried to develop an android application for this problem which works fine for specific phone. However it does not support all of the android phones.

2.2 Proposed Solution

To overcome the issue of compatibility of android app we can develop a website for the same which will be the easiest solution for this problem. A website will also be able to tasks which we expect from and application such as making API calls and capturing image by using phone's camera.

However thinking of this problem in a real world scenario this project will mostly -be user by the farmers and I believe that the farmers are more comfortable in using android application compared to a website because once an app is installed it can be used frequently and it is more user friendly. So I will be working on developing an android application.

Chapter 3

Application

3.1 Design

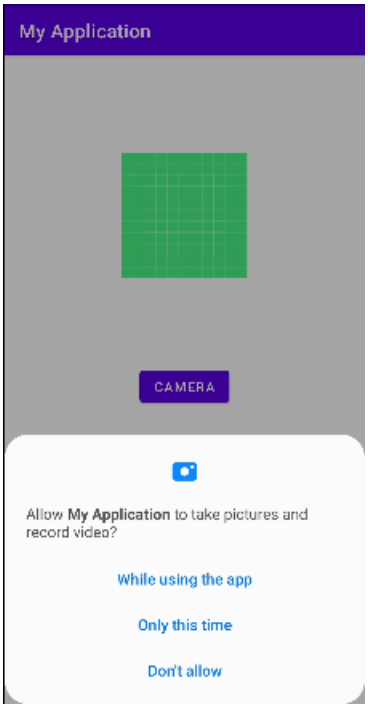
In this section we will discuss the basic design of our application. I am using android studio for developing the application.

3.1.1 Permission

On opening the app for the first time user will be asked for camera permission. A pop up will appear just after opening the application. Now user can either allow or deny it.
If user denies the permission then application will ask it again before using camera button (we will discuss the camera button in homepage section). And if user denies the permission at the run-time of the camera then user will have to give camera permission from the settings manually in order to use camera.

objective

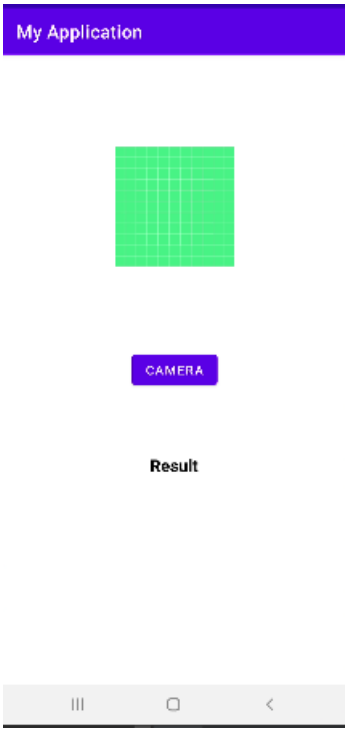
The main use of implementing asking permission is to prevent application from crashing if the camera permission is not give. Because if app try to use camera without permission application will crash. And also this functionality makes application more user friendly and improve the ease of using the application.



3.1.2 Homepage

After opening application user will see the main homepage which consists three entities which are following :

- Image View : This is image view is to show image taken from the camera by the user.
- Camera Button : After clicking this button application will try to open the camera. If the camera permission is not given it will ask for permission otherwise it will redirect to the phone's camera to take the image.
- Text View : This text view is basically for displaying output which application will receive from the server (we will discuss the control flow of the server returning output will be discussed in further sections).



3.2 Work Flow

User wants the result from the application after capturing the image. To show the result application will send this image to a server and server will process image using YOLO model and will return it to the application. And this result which app will receive from the server will display to the user.
Let's discuss it in detail in further subsection : client and server side.

(For better understanding assume that till now user has captured the image and waiting to see the results.)

3.2.1 Client Side

After clicking on camera button at the homepage user will be redirected to the phone's camera for capturing image. If user does not take image we will show a toast message that "Image is not taken" and if user successfully captures the image we will display this in

image view. So now basically application has image stored.
Further we will convert image in Base64 encoding before sending it to the server.

Base64 encoding is used so that image can be treated as a string. sending image as string is efficient way of sending image to the server without uploading.

Http Request

We create a request body for sending the http post request. This request body will contain our image(base64 format) and the URL of the server. We will receive the URL for the server from the ngrok application. (we will discuss about ngrok in further section).
Now to send this request to the server I am using okhttp library. We will en queue our request in the queue using okhttp client. After this it will wait for request to be addressed in the server. If request is successfully executed and server returns the response then we display the result in the text view of the application.

3.2.2 Server Side

For establishing a local server to handle http request and running YOLO model I am using flask server in python. This flask server runs locally using VS code on local host IP : http://127.0.0.1:5000/ In flask server two type of Http requests are handled.

- Get request : This type of request is used to get some data or response from the server. we are implementing this so we can check our flask server is running successfully.
- Post request : This type of requests are used to send some data by the client and get some response from the server.

Our application will use post request for sending base64 image to the server.

Handling of post request

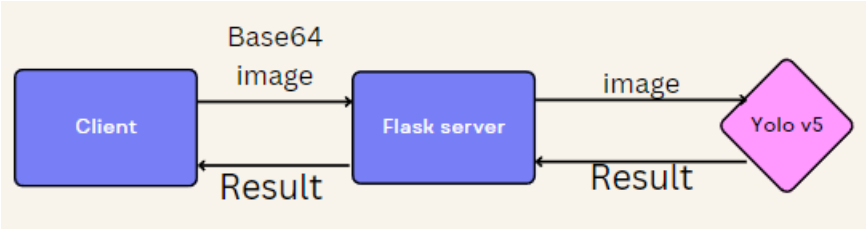
After flask server receives the base64 image from the client it converts it back to the image and stores it in the server.

YOLO model testing

For testing purpose I am using YOLO v5 of object detection Link. To process the image using available Yolo model we run a terminal command from python using Os library.

```
python detect.py --weights yolov5s.pt --img 640 --conf 0.25 --source image.jpg --save-txt
```

By using this command we are also storing the results of the Yolo model processing. In results it produces a image file with bounding boxes. It also produces a .txt file which consists the id of object present in the image, we can get the object name corresponding to the ids from dictionary we declared in flask server itself. We will get the results in flask server by reading the file which Yolo model produced. And server will return it to the client as json response. Json format is used for transmitting data in web applications because it supports many languages.

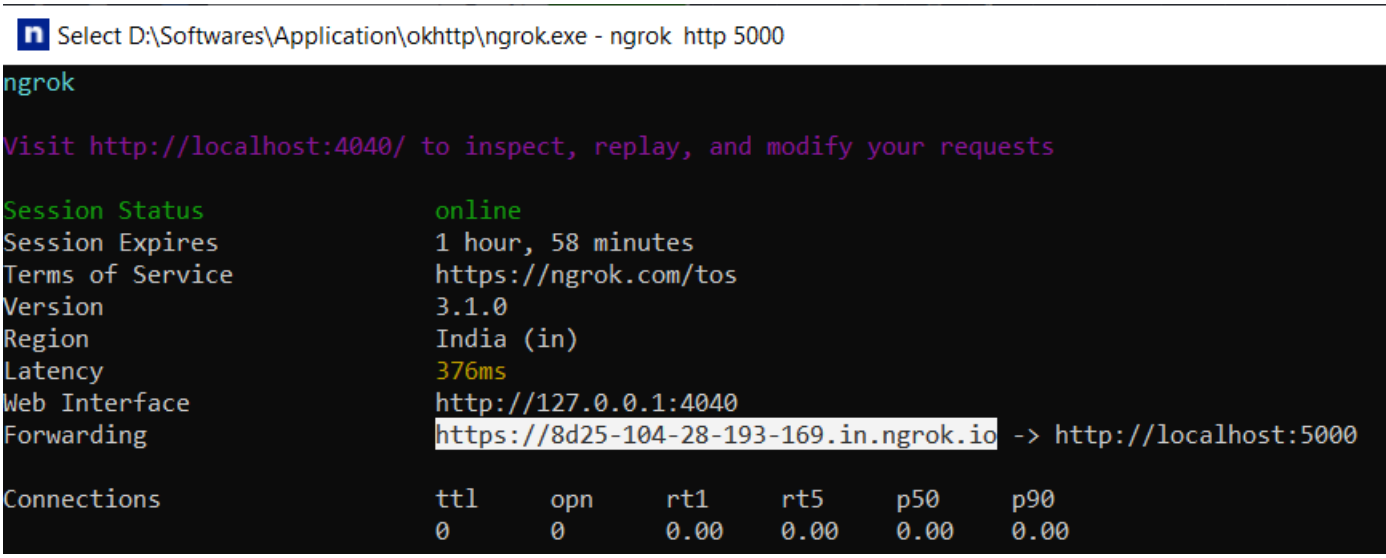


3.2.3 Need of ngrok application

We are running flask server in local machine so the local host Ip is only available inside machine only. Now to use server Url that can be accessed from a separate device we need a global URL. So to convert our local host IP to global URL I am using ngrok applicationLink. After opening this application we use this command to get the global URL

```
ngrok http 5000
```

(5000 represents the port number on which our local host IP runs)



This URL is generated whenever we start a new session. so we need to change the URL on which we are sending the post request form the client side.

Chapter 4

Future Work

I have tested this application for Yolo v5 for object detection. Further this application can be tested for the Yolo model for counting crops which was developed previously.

4.1 Design

Till now in our application only basic design is there to do the task and test the application. Further work can be done in terms of UI and UX to improve user experience.

4.2 Server Hosting

We can host our flask server using some premium services so that application can be used globally and can handle multiple http requests.

- Till now we are only taking image input from the camera only. Further we can implement logic such that image can be taken from the local storage as well.

4.3 Conclusion

The desired application is developed that integrates Yolo model processing. The server logic in flask is implemented for accomodating Yolo model. I have also tested this applica-

tion in following phones

- Samsung A50
- Redmi Note 10 pro

So the compatibility issue is resolved by making application more robust.

4.4 Repository

To see the work done by me on this project please follow this github