Fruit detection using YOLOv4

Solution Methodology

Business Overview

In the rapid development of technology, significant concerns are given to the food we consume. In the agriculture industry, one of the most cost-demanding factors is skilled labour. The industry is moving towards automation to decrease the cost of work and to increase quality. Robotic harvesting can provide potential solutions to many such problems faced by the industry. There are numerous challenging tasks to be fulfilled by the upcoming technology, one of them being an accurate fruit detection system. Different technologies have been used in the past for fruit recognition using emerging computer vision technologies.

The particular project discusses building a robust model for fruit detections. There can be many advanced use cases for this. Some of them are:

1. You are working in a warehouse where lakhs of fruits come in daily, and if you try to separate and package each of the fruit boxes manually, it will require a lot of workforce. So, you can build an automated system that can detect fruits and separate them for packaging.
2. You are the owner of a vast orchid. If you want to harvest fruits from it manually, it will require a lot of workforces too. You can build a robot or a self-driving truck that can detect fruits on specific trees and harvest them for you.

Aim

To build a robust fruit detection system using YOLOv4.

Tech stack

- Language: Python
- Object detection: YOLOv4
- Data annotation: LabelImg
- Environment: Google Colab

Approach

1. **Data collection and Labeling with LabelImg**

   To create a custom object detector, we need an excellent dataset of images and labels so that the sensor can efficiently train to detect objects.

   We can do this in two ways.

a. Using Google's Open Images Dataset

   We can gather thousands of images and their auto-generated labels within minutes. Explore that dataset here!

b. Creating your dataset and then labelling it manually

   We will create a dataset manually by collecting images from google image scraper or manually clicking them and then marking them using an image annotation tool, LabelImg.

2. **Building a YOLOv4 Object Detector with Darknet in the Cloud**

   a. Enabling GPU within your notebook

   b. Cloning and Building Darknet

      We are downloading AlexeyAB's famous repository and adjusting the Makefile to enable OPENCV and GPU for darknet and then build darknet.

3. **Demo on Pretrained model**

   YOLOv4 is trained on the coco dataset, which has 80 classes that it can predict. We will take these pre-trained weights to see how it gives results on some of the images.

4. **Customise YOLOv4 with the different command-line flags**

   a. Threshold Flag

   b. Output Bounding Box Coordinates

   c. Don't Show Image

   d. Multiple Images at Once

5. **Preparing dataset for training Yolo**

   a. Labelled Custom Dataset
   b. Custom .cfg file

      Edit the yolov4.cfg for your custom model training using the colab editor

   c. obj.data and obj.names files

      Create a new file within a code or text editor called obj.names where you will have one class name per line

      Example for multiclass obj.names file:

obj.names

```
1   apple
2   mango
3   orange
4   watermelon
5   pomegranate
6   other
```

You will also create an obj.data file and fill it in accordingly

obj.data

```
1   classes = 6
2   train = data/train.txt
3   test = data/test.txt
4   names = data/obj.names
5   backup = /content/gdrive/MyDrive/Object_Detection/backup
```

d. train.txt and test.txt file

The train.txt and test.txt files hold the relative paths to all our training images and validation images.

6. **Train Your Custom Object Detector**
7. **Training Yolo model from a checkpoint**
8. **Model Evaluation using Mean Average Precision**
9. **Predictions on images**
10. **Predictions on video**

Project Takeaways

1. Understanding Object detection
2. Transfer learning and pre-trained models
3. Understanding Problem statement
4. Google's open image dataset
5. Data labelling with LabelImg
6. Introduction to YOLOv4

7. Introduction to Google Colab
8. Installing prerequisites for YOLOv4
9. Building YOLOv4 object detector
10. Understanding how to use pre-trained models of YOLO
11. Customizing YOLOv4 using command line flags
12. Preparing dataset for training YOLOv4
13. Training Custom Object Detector with YOLO
14. Training YOLO model using checkpoints
15. Understanding Mean Average Precision
16. Storing and displaying detected classes in an image
17. Predictions on a video