Recognize and Locate "康师傅香辣牛肉面" and "康师傅卤香牛肉面" Using YoloV2

1552746 崔鹤洁 2018.6.4

1. Sample Collection & Labelling

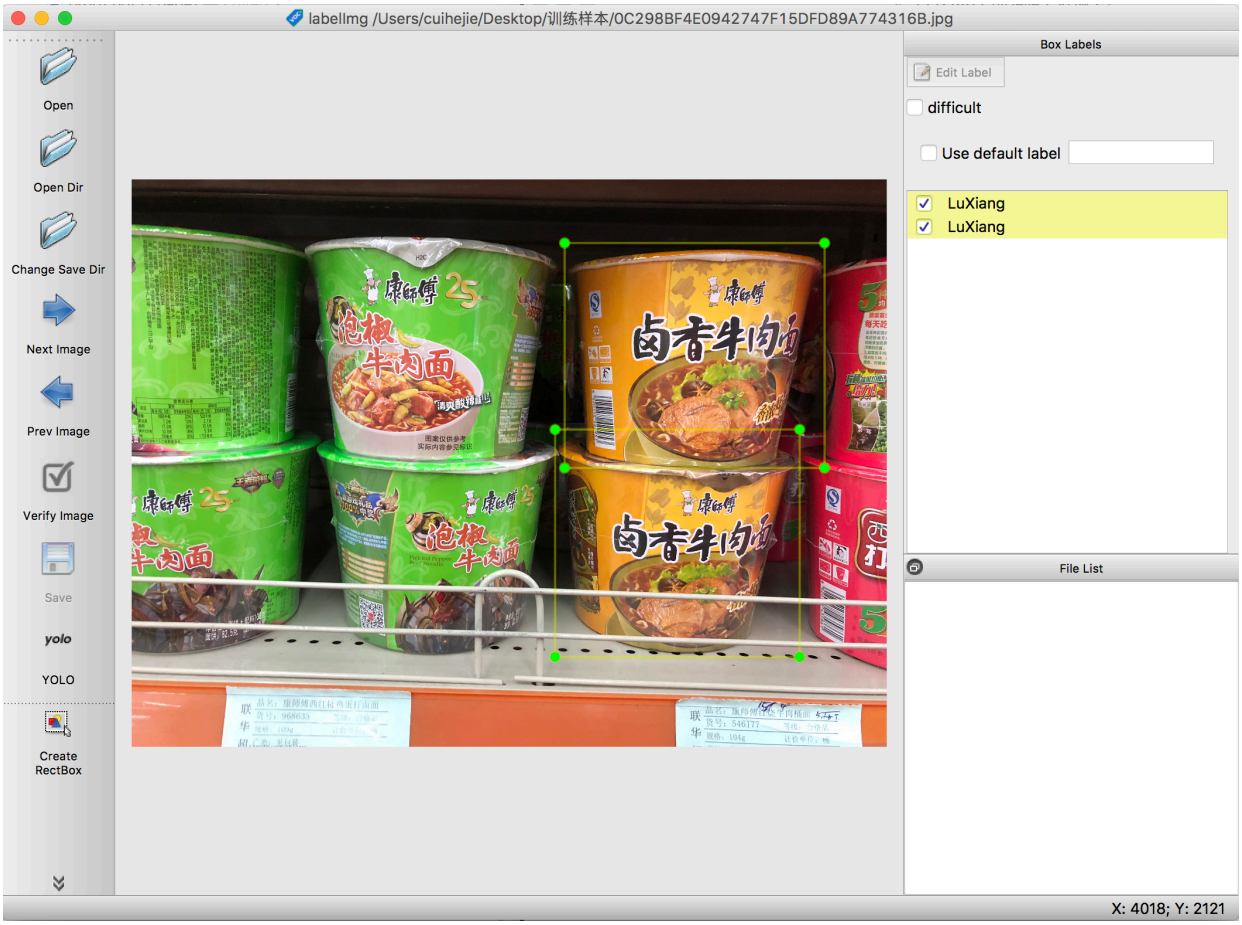
- **Collection**

The training samples are collected at LianHua supermarket using smart phone's camera.

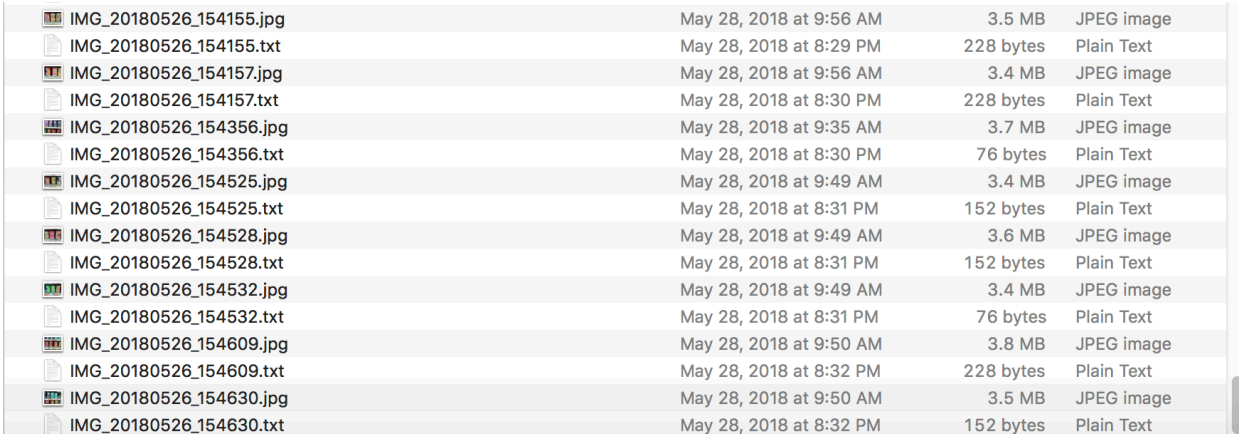
The total number of samples is 349. (270 for training and 79 for validation)

- **Labelling**

Tool: LabelImg (LabelImg is a graphical image annotation tool and label object bounding boxes in images)



Output: txt file storing position of bounding boxes



IMG_20180526_154155.jpg	May 28, 2018 at 9:56 AM	3.5 MB	JPEG image
IMG_20180526_154155.txt	May 28, 2018 at 8:29 PM	228 bytes	Plain Text
IMG_20180526_154157.jpg	May 28, 2018 at 9:56 AM	3.4 MB	JPEG image
IMG_20180526_154157.txt	May 28, 2018 at 8:30 PM	228 bytes	Plain Text
IMG_20180526_154356.jpg	May 28, 2018 at 9:35 AM	3.7 MB	JPEG image
IMG_20180526_154356.txt	May 28, 2018 at 8:30 PM	76 bytes	Plain Text
IMG_20180526_154525.jpg	May 28, 2018 at 9:49 AM	3.4 MB	JPEG image
IMG_20180526_154525.txt	May 28, 2018 at 8:31 PM	152 bytes	Plain Text
IMG_20180526_154528.jpg	May 28, 2018 at 9:49 AM	3.6 MB	JPEG image
IMG_20180526_154528.txt	May 28, 2018 at 8:31 PM	152 bytes	Plain Text
IMG_20180526_154532.jpg	May 28, 2018 at 9:49 AM	3.4 MB	JPEG image
IMG_20180526_154532.txt	May 28, 2018 at 8:31 PM	76 bytes	Plain Text
IMG_20180526_154609.jpg	May 28, 2018 at 9:50 AM	3.8 MB	JPEG image
IMG_20180526_154609.txt	May 28, 2018 at 8:32 PM	228 bytes	Plain Text
IMG_20180526_154630.jpg	May 28, 2018 at 9:50 AM	3.5 MB	JPEG image
IMG_20180526_154630.txt	May 28, 2018 at 8:32 PM	152 bytes	Plain Text

2. Install Darknet

- **Clone the darknet git repository**

```
git clone https://github.com/pjreddie/darknet.git
```

- **Change the first line of the `Makefile` in the base directory**

```
GPU=1
CUDNN=1
OPENCV=1
```

- **Compile C code**

```
cd darknet
make
```

3. YoloV2 Fine-Tuning

- **Download pretrained convolution weights**

For training we use convolutional weights that are pre-trained on Imagenet. We use weights from the extraction model.

```
wget https://pjreddie.com/media/files/darknet19_448.conv.23
```

- **Change the configuration parameters**

```
#cfg/yolov2-voc.cfg
[net]
batch=64
```

```
subdivision=8
max_batches=20000
the last [convolutional]
filters=35
[region]
classes=2

#cfg/voc.data
classes=2
train=/media/cuihejie/Data/darknet/data/train.txt
valid=/media/cuihejie/Data/darknet/data/test.txt
names=/media/cuihejie/Data/darknet/data/voc.names
backup=/media/cuihejie/Data/dakenet/backup

#data/voc.names
xiangla
luxiang
```

- **Generate the training and validation file**

The number of training samples: 270

The number of validation samples: 79

```
import xml.etree.ElementTree as ET
import pickle
import os
from os import listdir, getcwd
from os.path import join
import re

wd = getcwd()

folder = 'JPEGImages'
paths = listdir(folder)
paths = list(filter(lambda x: re.match(r'.+\.jpg$', x), paths))
index = 0
train = open('train.txt', 'w')
test = open('test.txt', 'w')

for path in paths:
    if index < 270:
        train.write("%s\n"%join(wd, folder, path))
    else:
        test.write("%s\n"%join(wd, folder, path))
    index += 1
train.close()
test.close()
```

- **Train the model**

```
./darknet detector train cfg/voc.data cfg/yolov2-voc.cfg
darknet19_448.conv.23
```

4. Save as Video

Write the following script to merge frames and export video files.

```
import cv2
import argparse
import os

# Arguments
dir_path = 're'
output = 'test.mp4'
images = os.listdir(dir_path)
images.sort(key=lambda x:int(x[2:-4]))

# Determine the width and height from the first image
image_path = os.path.join(dir_path, images[0])
frame = cv2.imread(image_path)
cv2.imshow('video',frame)
height, width, channels = frame.shape

# Define the codec and create VideoWriter object
fourcc = cv2.VideoWriter_fourcc(*'mp4v') # Be sure to use lower case
out = cv2.VideoWriter(output, fourcc, 30.0, (width, height))
for image in images:
    image_path = os.path.join(dir_path, image)
    frame = cv2.imread(image_path)
    out.write(frame) # Write out frame to video
    cv2.imshow('video',frame)
    if (cv2.waitKey(1) & 0xFF) == ord('q'): # Hit `q` to exit
        break

# Release everything if job is finished
out.release()
cv2.destroyAllWindows()
```

5. Environment

```
Ubuntu: 16.04
CUDA: 9.0
GPU: GTX 1080 Titan
```

6. Result & Reflection

The best result shows after 10000 batches but not at 20000 batches.

I guesses that for just 349 images (270 for training and 79 for validation), the accuracy is enough high at 10000 batches. Therefore, more batches make no contribution, it just learns something that we do not want.