BÁO CÁO
ĐỒ ÁN CUỐI KÌ

Ứng dụng YOLO
vào bài toán nhận dạng đối tượng

-- [★] --

Giáo viên hướng dẫn:

Ngô Minh Nhựt

Sinh viên thực hiện:

Vũ Thế Huy

Mục lục

A. Thông tin sinh viên

STT	Tên	MSSV	Email
1	Vũ Thế Huy	1512213	vuthehuy1997@gmail.com

B. Tóm tắt nội dung

Nội dung	Nội dung chi tiết	Tham khảo	Mức độ hoàn thành (%)
I. Cài đặt YOLO và sử dụng mô hình có sẵn	1. Giới thiệu về YOLO	https://ai.hblab.vn/2017/10/yolo-you-only-look-once.html	
	2. Cài đặt YOLO về máy	https://pjreddie.com/darknet/yolo/	100
	3. Sử dụng mô hình có sẵn	Tự xây dựng	100
II. Hiểu và huấn luyện để nhận dạng thêm một loại đối tượng mới	1. Thêm dữ liệu mới	Tự xây dựng Kết hợp sử dụng tool để lấy bounding box https://github.com/ManivannanMurugavel/Yolo-Annotation-Tool-New-	100
	2. Cách huấn luyện trong YOLO	Tự xây dựng và tham khảo cách huấn luyện từ https://medium.com/@manivannan data/how-to-train-yolov3-to-detect-custom-objects-ccbcafeb13d2	100
III. Áp dụng kết quả huấn luyện để xây dựng một ứng dụng hoàn thiện	Xây dựng ứng dụng từ kết quả huấn luyện cho tập dữ liệu mới	Tự xây dựng	100

C.Nội dung

I. Cài đặt YOLO và sử dụng mô hình có sẵn

1. Giới thiệu về YOLO

a. Khái niệm

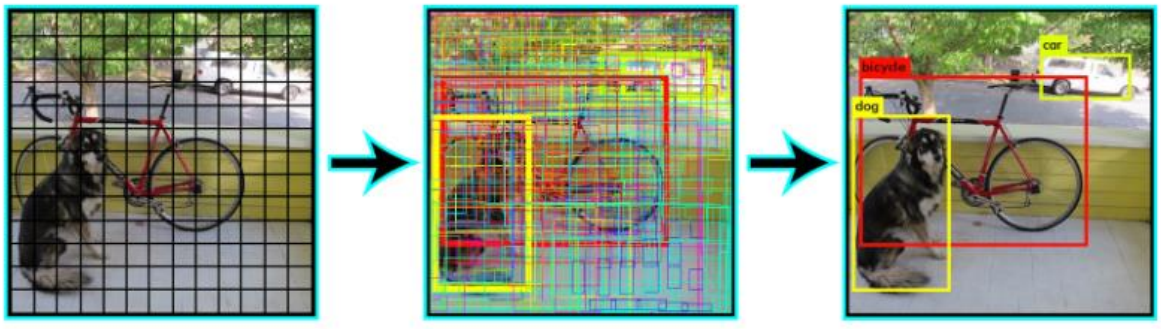
You only look once(Yolo) là một trong những thuật toán nhận diện vật thể nhanh nhất thời điểm hiện tại. Mặc dù không phải là phương pháp chính có độ xác cao nhất tuy nhiên Yolo vẫn là được ứng dụng rất nhiều trong những dự án thực tế khi mà độ chính xác không phải là ưu tiên hàng đầu.

b. Những lợi thế của YOLO :

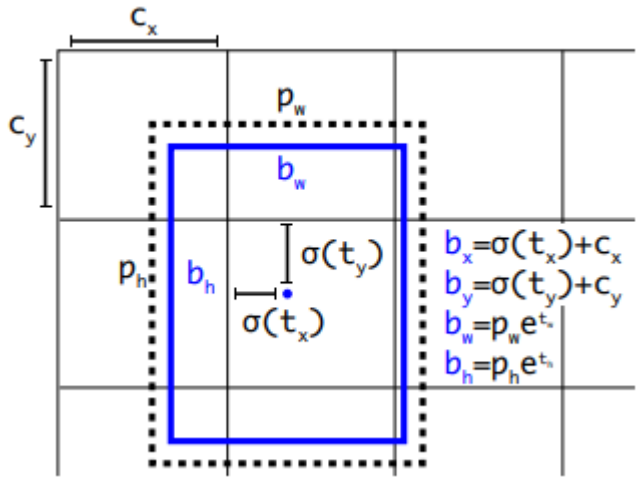
- "Hiểu" được khái quát về đại diện của mỗi đối tượng. Qua đó có thể phát hiện và phân loại chính xác vật thể trong đời thực và các tác phẩm hội hoạ.
- Gọn nhẹ
- Open source
- **NHANH**

Những Region Proposal Classification network khác (như Fast RCNN) thực hiện việc phát hiện trên các đề xuất khu vực (*Region proposal*), do đó sau cùng sẽ phải thực hiện dự đoán nhiều lần cho các region khác nhau, scale khác nhau trong một ảnh.

YOLO có một cách tiếp cận hoàn toàn khác, chỉ sử dụng duy nhất một neural network cho toàn bộ ảnh. Kiến trúc của YOLO giống với FCNN (Full Convolution Neural Network) hơn, và hình ảnh (kích thước $n \times n$) chỉ được truyền qua FCNN một lần duy nhất, sau đó sẽ trả về ouput là $m \times m$ prediction. Hình ảnh đầu vào sẽ được chia thành các ô lưới (grid cell), và dự đoán các bounding box và xác suất phân loại cho mỗi grid cell. Các bounding box này được đánh trọng số theo xác suất đã dự đoán.



c. Cách YOLO hoạt động



YOLO chia ảnh thành các grid và dự đoán bounding box của các đối tượng theo mỗi grid

d. Cách YOLO nhận dạng

Thông qua mô hình mạng cụ thể, từ input đã scale từ ảnh đầu vào thành n*n*3 (n ở YOLO thường là 418) tạo ra đầu ra có kích thước grid_number * grid_number * (4+1+num_class)*3(grid_number: số lượng grid trên mỗi chiều, num_class: số class mà yolo có thể detect).

Kết quả là trên mỗi grid dự đoán 3 boxs với các thông số lần lượt là 4 độ lệch hộp giới hạn, 1 tỉ lệ là đối tượng và num_class dự đoán lớp.

e. Cách YOLO huấn luyện

Neural net sẽ tính toán từng ảnh (có thể lặp lại 1 ảnh) để tối ưu hàm mất mát. Việc tối ưu này sẽ giúp neural net tìm ra 1 bộ trọng số tốt nhất để biểu diễn dữ liệu của bạn và giúp nhận dạng các ảnh mới.

2. Cài đặt YOLO về máy

Tải dữ source code về máy

```
\DoAn>git clone https://github.com/pjreddie/darknet
```

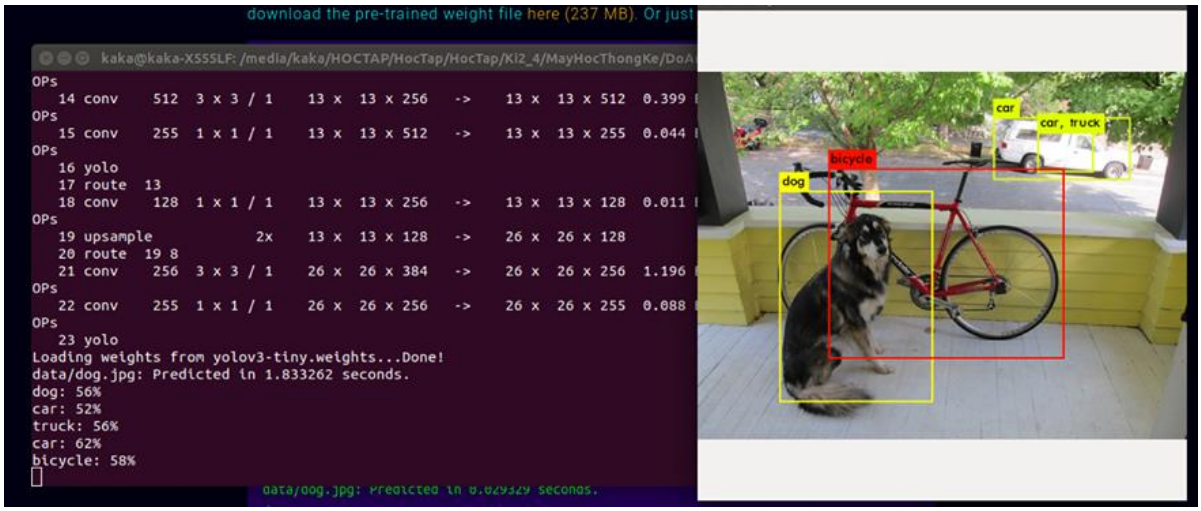
Chạy make để tạo file thực thi

```
\DoAn>cd darknet
```

```
\DoAn\darknet>make
```

Kiểm tra kết quả xem source code có hoạt đông ko

```
\DoAn\darknet>./darknet detect cfg/yolov3.cfg yolov3.weights data/dog.jpg
```



3. Sử dụng mô hình có sẵn

Để sử dụng mô hình có sẵn của darknet để xây dựng ứng dụng ta cần thêm một số hàm vào file python/darknet.py để tạo ra các hàm load trọng số và dự đoán độc lập, bên cạnh đó do kết quả trả ra là một mảng các object với các thông số về tọa độ và kích thước nên phải có một số hàm để vẽ bounding box thông qua file box.py. Và chương trình sẽ là một website cho phép người dùng upload ảnh và nhận kết quả detect thông qua file web.py.

a. Thêm hàm vào darknet.py

Thêm hàm trả về cấu trúc mạng và các thông số trong file config:

```
146   def load_app():
147       current_dir = os.getcwd()
148       current_dir = os.path.join(current_dir, "cfg/yolov3-tiny.cfg")
149       print(current_dir)
150       net = load_net(current_dir, "yolov3-tiny.weights", 0)
151       meta = load_meta("cfg/coco.data")
152       return net,meta
```

Để khi load chỉ cần chạy một lần

Thêm hàm trả về kết quả detect:

```
164   def detect_app(net,meta,filename):
165       r = detect(net, meta, filename)
166       print('Detected\nA')
167       return r
```

Cụ thể hóa và hỗ trợ fix khi có lỗi

b. Thêm file box.py

darknet.py | box.py | web.py

```
1   import darknet as dn
2   import os
3   import random
4   from PIL import Image, ImageFont, ImageDraw, ImageEnhance
5
6   # draw box for all of object that been returned from result
7   def all_box(draw, objects):···
18
19  # draw box for 1 object
20  def box(draw, object_detect, color):···
29
30  # draw rectangle for box
31  def rectangle_box(draw, x,y,width,heigh, color):···
34
35  # draw text for box
36  def text_box(draw, text ,x, y, color):···
44
45  # main function to draw box for detected object
46  def write_detect(net, meta, path, filename):···
59
60  if __name__ == "__main__":···
70
```

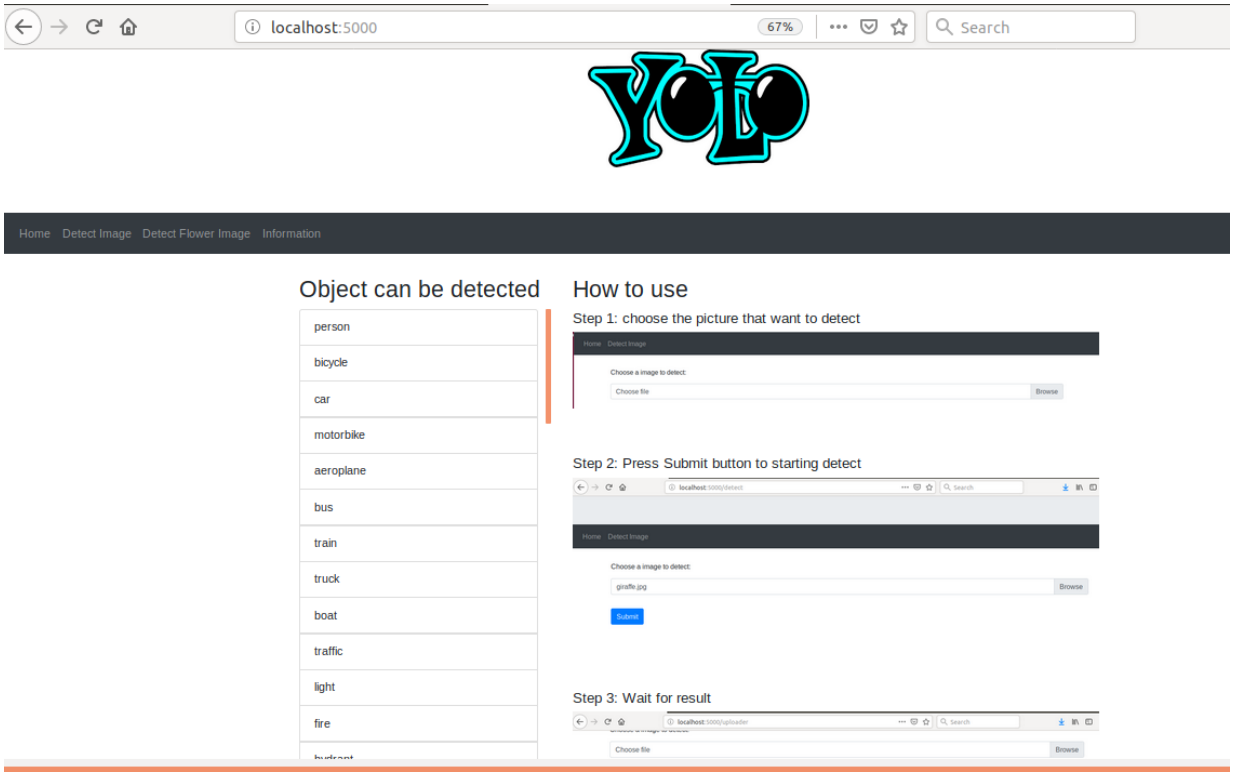
c. Thêm file web.py và các file html cần thiết

```
UPLOAD_FOLDER = os.path.join('python', 'static')
ALLOWED_EXTENSIONS = set(['txt', 'pdf', 'png', 'jpg', 'jpeg', 'gif'])

app = Flask(__name__)
app.config['SEND_FILE_MAX_AGE_DEFAULT'] = 0
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

net,meta = dn.load_app()
netf,metaf = dn.load_app_flower()


def allowed_file(filename):
    return '.' in filename and \
           filename.rsplit('.', 1)[1] in ALLOWED_EXTENSIONS


@app.route("/")
def main():
    return render_template('index.html')

@app.route('/detect')
def detectPage():
    return render_template('detect.html')

@app.route('/detectflower')
def detectFlowerPage():
    return render_template('detectflower.html')

@app.route('/uploader', methods = ['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        file = request.files['file']
```

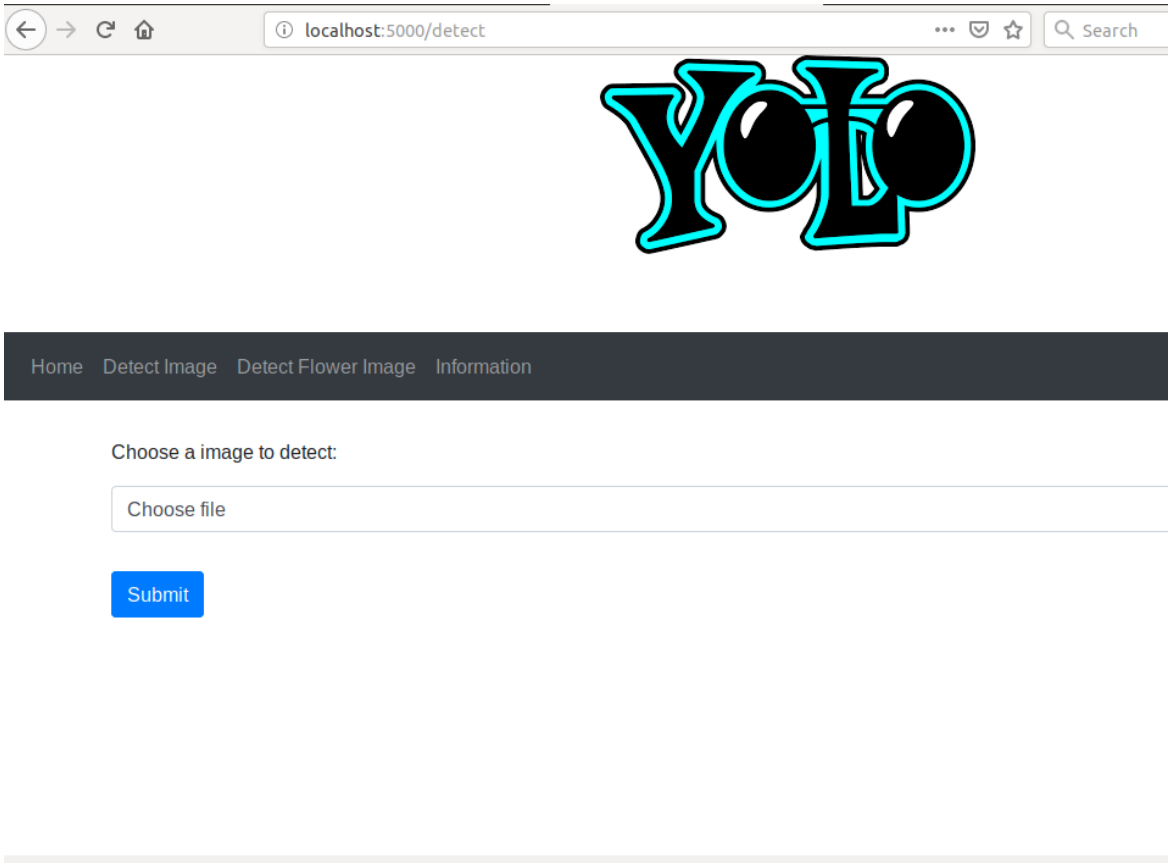
File này sẽ tạo deploy một website để sử dụng YOLO

d. Sơ lược về cấu trúc trang web

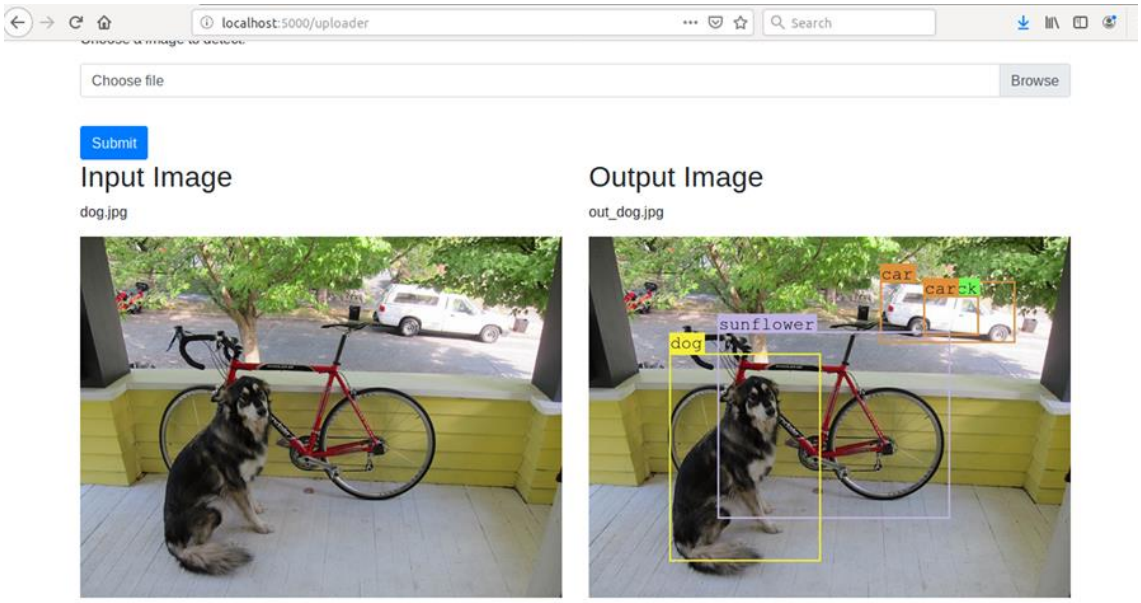
Trang chủ:



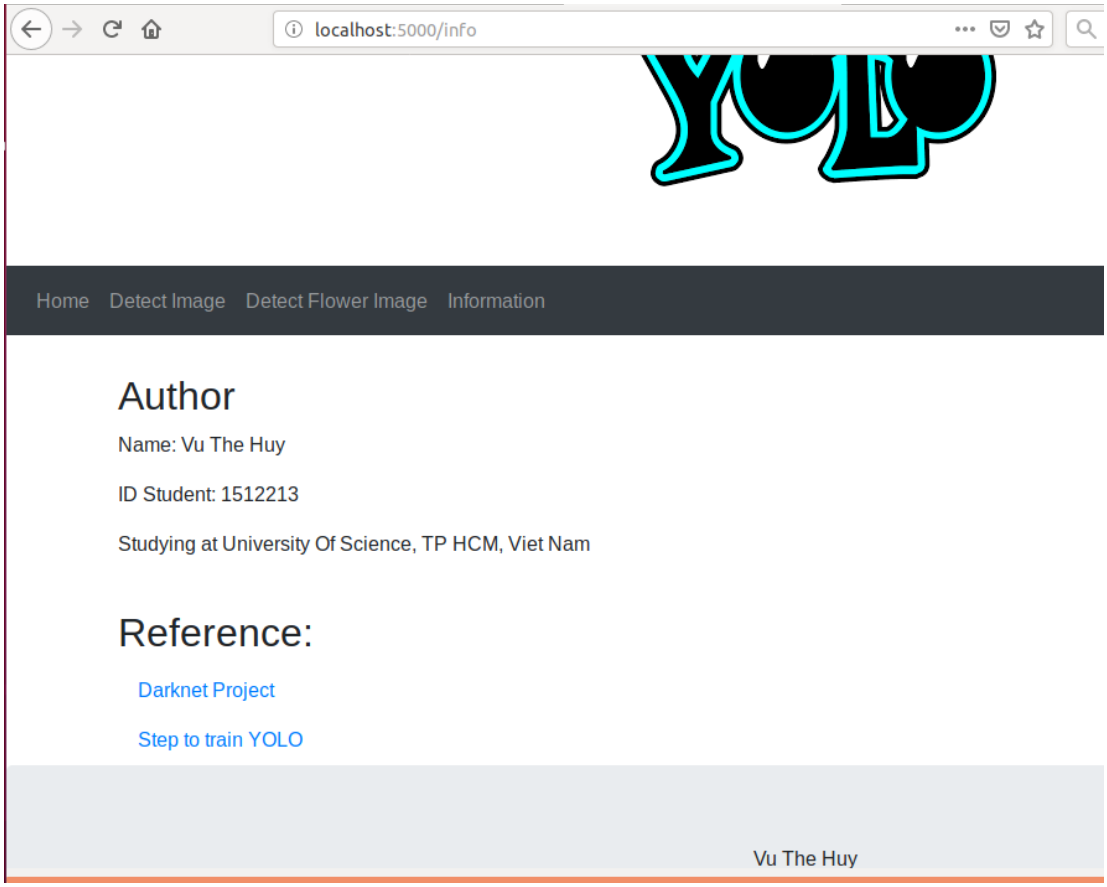
Trang detect:



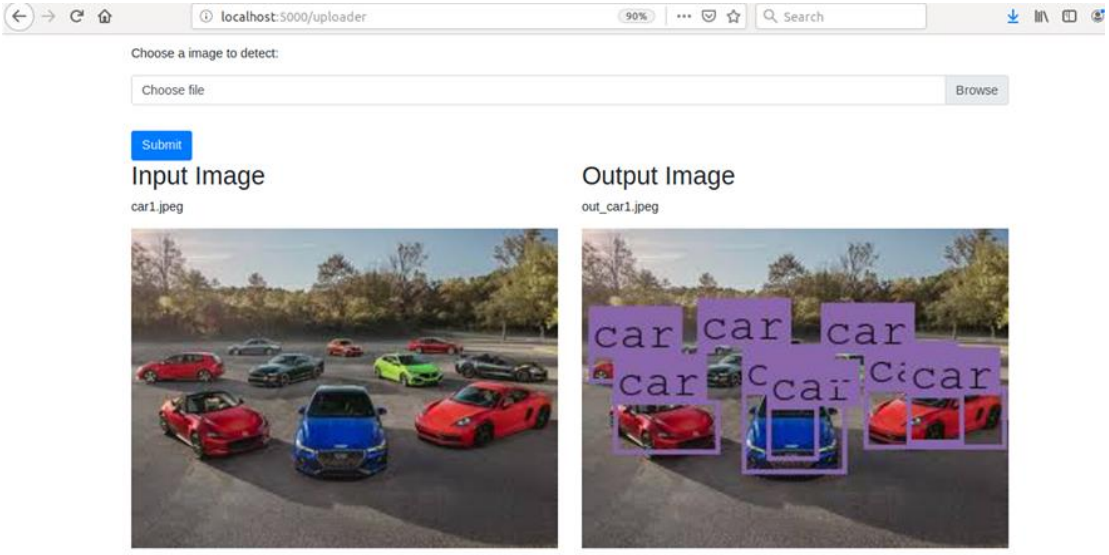
Sau khi detect

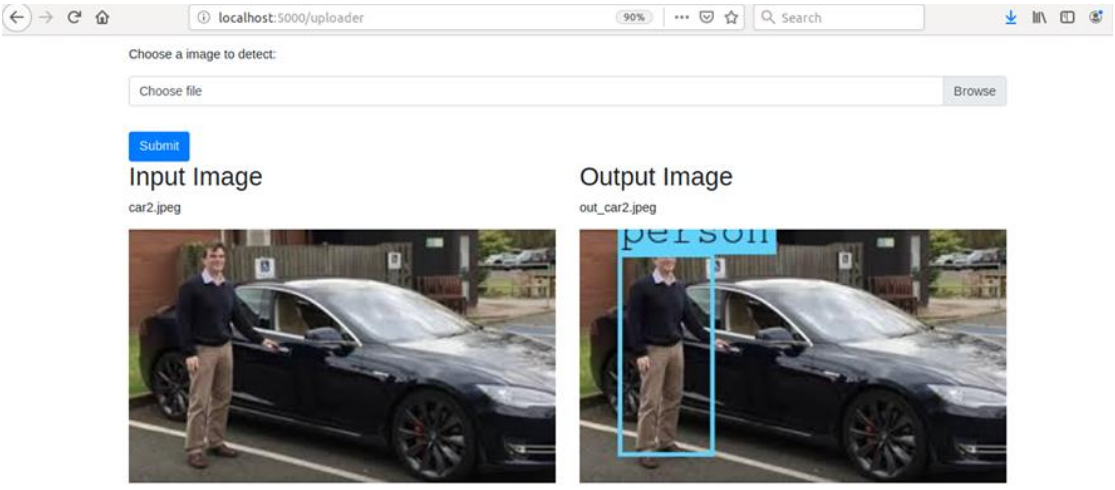


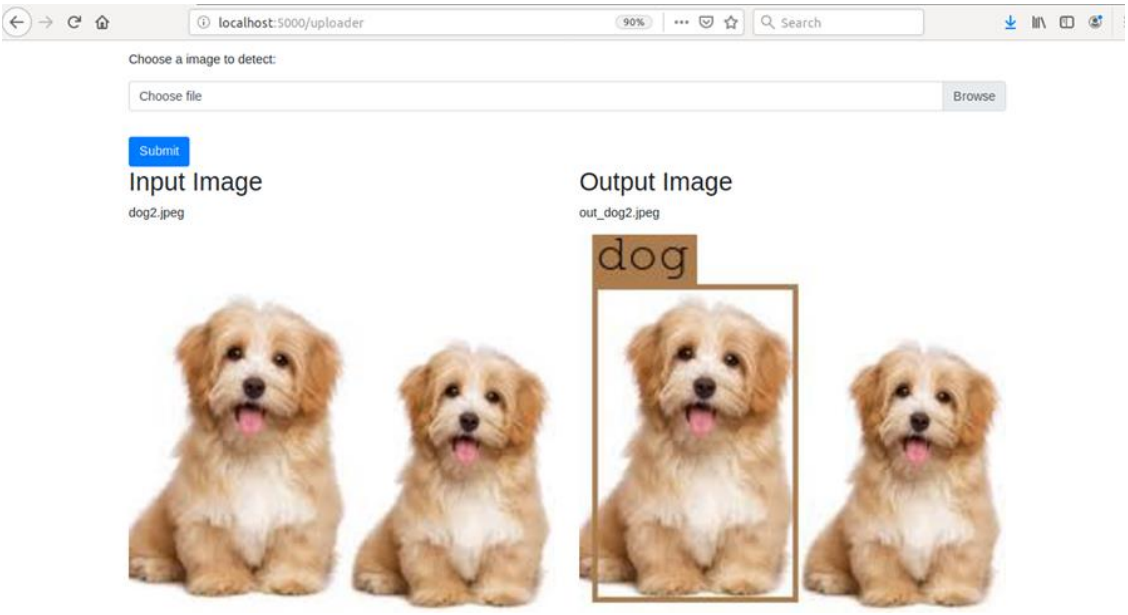
Trang information



e. Kết quả detect trong web







II. Hiểu và huấn luyện để nhận dạng thêm một loại đối tượng mới

Bài toán đặt ra: làm sao để ứng dụng YOLO để có thể phát hiện và phân biệt các loài hoa khác nhau để hỗ trợ mọi người khi cần biết hoa mà họ thấy là gì từ bức ảnh họ chụp.

1. Thêm dữ liệu mới

a. Mô tả dữ liệu:

- Dữ liệu là 520 bức ảnh về hoa hồng và hoa hướng dương.
- Trong ảnh có thể có một hoặc nhiều bông hoa các loại.

b. Thu thập dữ liệu

Dữ liệu được thu thập từ web https://images.google.com/

Hàm được dùng để lấy đường dẫn đến địa chỉ của ảnh

```
def getImageObject(searchName, count):
    result = []
    pageIndex = 0
    while len(result) < count:
        # use "tbs=sur:fmc" in the query for search the legal data.
        url = 'https://www.dreamstime.com/search.php?srh_field='+searchName+'&s_all=n&s_ph=n&s_il=n&s_video=n\
              &s_audio=n&s_ad=n&s_wp=y&s_sl0=y&s_sl1=y&s_sl2=y&s_sl3=y&s_sl4=y&s_sl5=y&s_rf=y&s_ed=y&s_orp=y\
              &s_orl=y&s_ors=y&s_orw=y&s_clc=y&s_clm=y&s_rsf=0&s_rst=7&s_st=new&s_sm=all&s_mrg=1&s_mrc1=y\
              &s_mrc2=y&s_mrc3=y&s_mrc4=y&s_mrc5=y&s_exc=&pg='+str(pageIndex)
        pageIndex += 1
        try :
            html_text = requests.get(url).text

            # Parse html text
            tree = BeautifulSoup(html_text, 'html.parser')
            imTable = tree.find('div', {'class': 'item-list'})
            if imTable == None:
                break

            arr = imTable.find_all('img')
            if arr == None:
                break
            for i in arr:
                result.append(i['src'])
        except:
            print("Can not request: " + url)
    return result
```

Sau khi có đường dẫn đến các ảnh, ta lưu ảnh về máy qua dòng code sau

```
    for i in range(len(flowerLinks)):
        for index, url in enumerate(flowerLinks[i]):
            urllib.request.urlretrieve(url.encode('utf-8').decode('ascii', 'ignore'), path + str(i) + '/' + str(index)
```

Thư mục chứa dữ liệu:

Sau đó dữ liệu sẽ được lọc tay để bỏ đi những ảnh không phù hợp

c. Đánh dấu bounding box cho tập dữ liệu

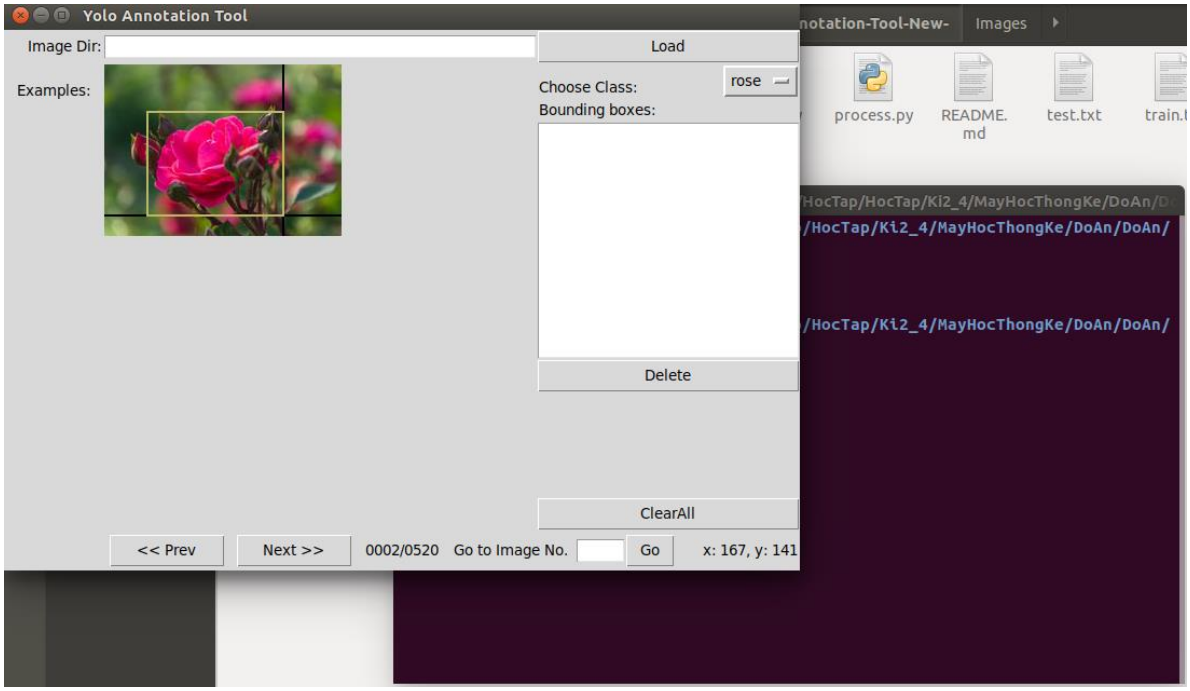
Bounding box được gán bằng cách chạy file main.py từ https://github.com/ManivannanMurugavel/Yolo-Annotation-Tool-New-

Có 2 thay đổi nhỏ ở hàm main.py

```
159        self.imageDir = os.path.join(r'./Images/rose_sunflower', '%s' %(self.category))
```

```
173        self.outDir = os.path.join(r'./Labels/rose_sunflower', '%s' %(self.category))
```

Việc này giúp lưu dữ liệu vào folder do mình tự tạo



d. Chuyển tập dữ liệu về định dạng file text để cung cấp dữ liệu cho quá trình huấn luyện

Để chuyển dữ liệu vào file text đầu tiên ta cần sao chép các nhãn đã lưu trữ dựa vào tập ảnh đưa vào cùng thư mục với thư mục lưu dữ liệu.

Tiếp đến ta chạy file process.py đã qua cập nhật để tạo ra file train.txt và test.txt

```
current_dir = os.getcwd() + '/Images/rose_sunflower'

# Percentage of images to be used for the test set
percentage_test = 10

# Create and/or truncate train.txt and test.txt
file_train = open('train.txt', 'w')
file_test = open('test.txt', 'w')

# Populate train.txt and test.txt
counter = 1
index_test = round(100 / percentage_test)
for pathAndFilename in glob.iglob(os.path.join(current_dir, "*.jpg")):
    title, ext = os.path.splitext(os.path.basename(pathAndFilename))

    if counter == index_test:
        counter = 1
        file_test.write(current_dir + "/" + title + '.jpg' + "\n")
    else:
        file_train.write(current_dir + "/" + title + '.jpg' + "\n")
        counter = counter + 1
```

2. Các

3. h huấn luyện trong YOLO

a. Tải tập Pretrained modal

Tải pretrained modal từ Imagenet là mô hình darknet53

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Lệnh tải modal:

```
DoAn\darknet> wget https://pjreddie.com/media/files/darknet53.conv.74
```

b. Tạo file configuration cfg/flower_yolov3-tiny.cfg tương tự như yolov3-tiny.cfg

Sao chép file yolov3-tiny.cfg và thảy đổi một số thông tin sau

- Dòng 6: đặt batch = 32. Số lượng ảnh sẽ dùng cho mỗi bước training
- Dòng 7: đặt subdivisions = 16. Mỗi batch sẽ lai được chia cho 16 để giảm yêu cầu cho CPU.
- Dòng 127: đặt filter = (class + 5) * 3. Cụ thể ở đây là filter = 21
- Dòng 135 : đặt class = 2(số lượng đối tượng muốn detect)
- Dòng 171: đặt filter = (class + 5) * 3. Cụ thể ở đây là filter = 21
- Dòng 177 : đặt class = 2(số lượng đối tượng muốn detect)

(đặt filter = (class + 5) * 3 là theo cấu hình của yolov3)

Một số tham số quan trọng khác có thể thay đổi :

- Dòng 8 và 9 : width và height: Có thể thay đổi nếu máy không đủ GPU ram để đọc hết ảnh.
- Learing rate : bước nhảy được sử dụng trong việc cực tiểu chi phí

c. Tạo file cfg/flower.data để cung cấp thông tin tập train, tập test và tên đối tượng

```
classes= 2
train  = data/flower_train.txt
valid  = data/flower_test.txt
names = data/flower.names
backup = backup/
```

File này có nghĩa là số class sẽ huấn luyện 2 class

Từ tập train và test lần lượt là data/flower_train.txt và data/flower_test.txt

Tên các class được đọc từ file data/flower.names

Backup sẽ được lưu trong thư mục backup

d. Tạo file data/flower.names để cung cấp thông tin tên đối tượng cho quá trình dự đoán

```
rose
sunflower
```

Tên của 2 class trong bài toán là rose(hoa hồng) và sunflower(hoa hướng dương).

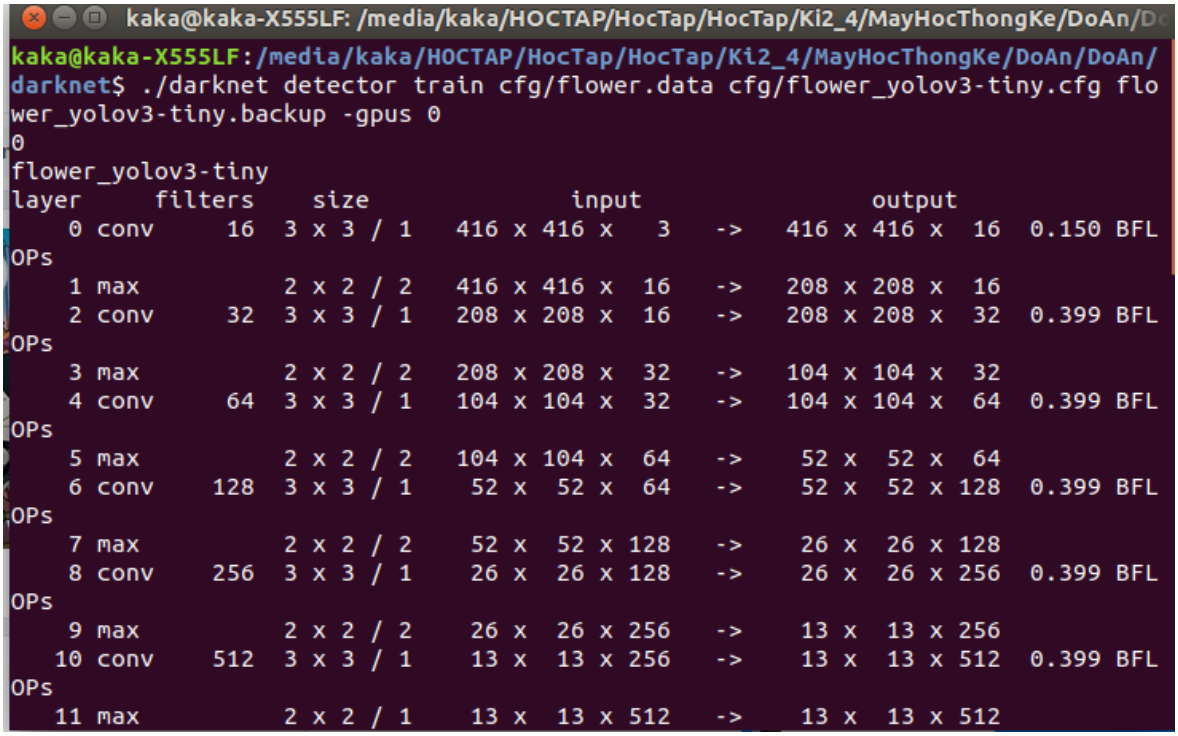
e. Huấn luyện

- ❖ Cài đặt cuda, opencv để hỗ trợ gpu cho bài toán
- ❖ Sửa lại file makefile

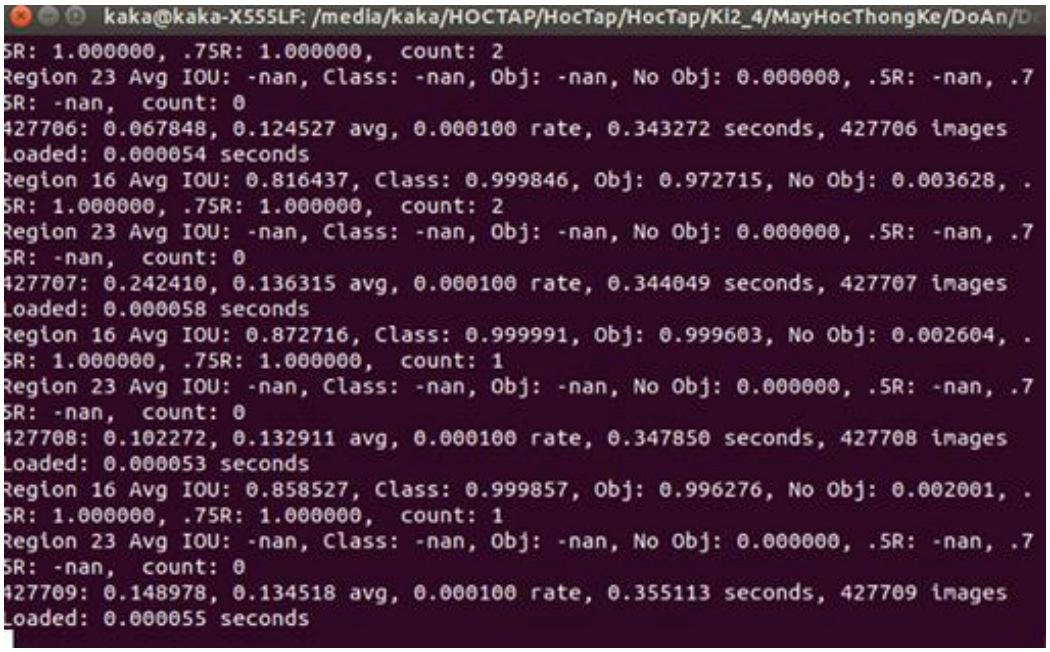
```
GPU=1
CUDNN=0
OPENCV=1
```

- ❖ Chạy lại lệnh make

Bắt đầu huấn luyện bằng lệnh detector train của darknet



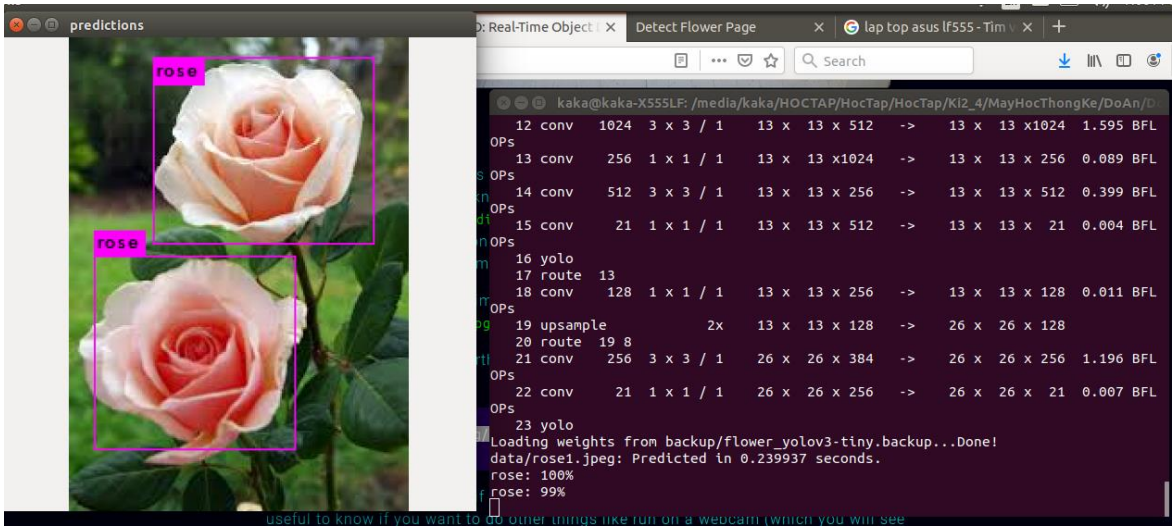
Trên terminal chương trình sẽ in ra một số thông tin trong đó quan trọng có số của vòng lặp, độ mất mắt trên tập train, độ mất mát trên tập test.

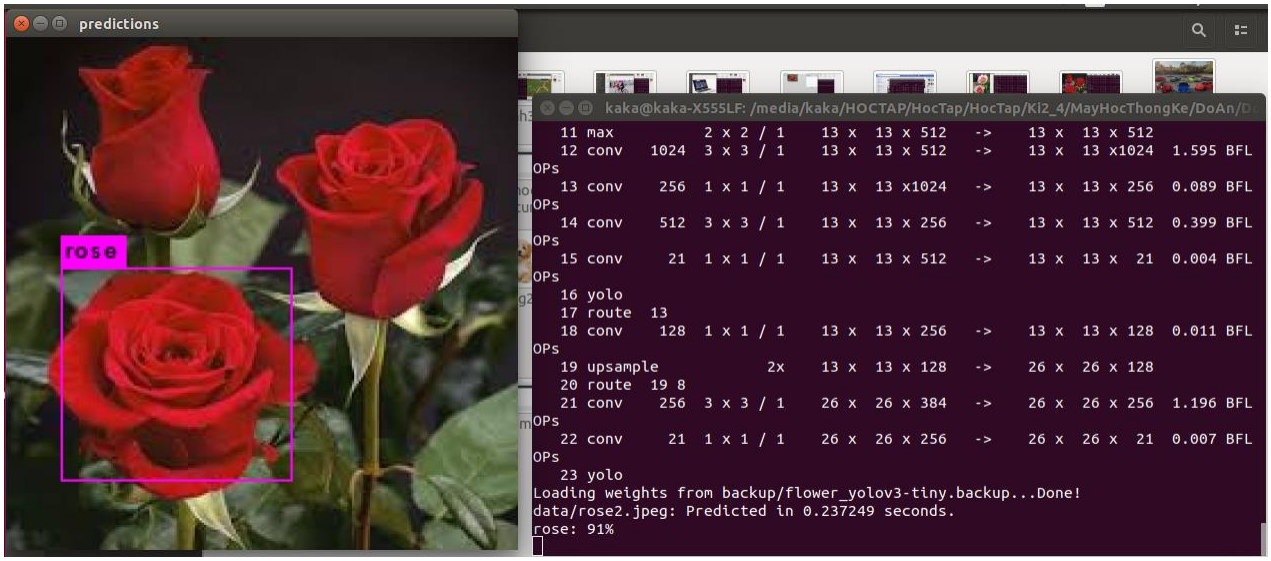


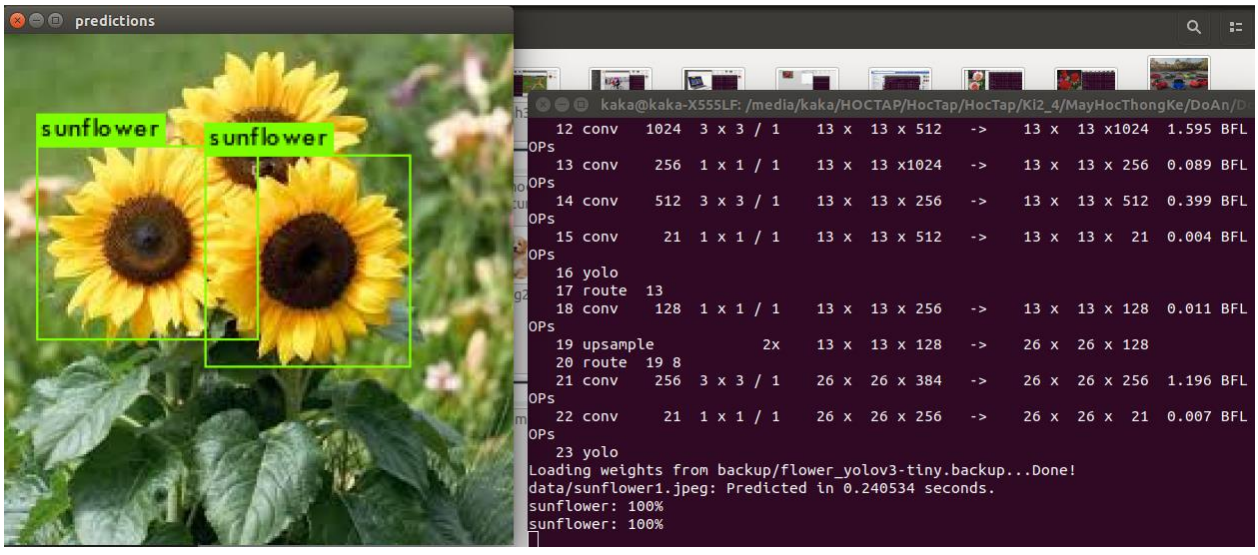
Sau khi chạy khoảng 300 000 epoch thì chi phí mới bắt đầu thấp nhanh hơn, và đến khi kết thúc thì chi phí thấp nhất

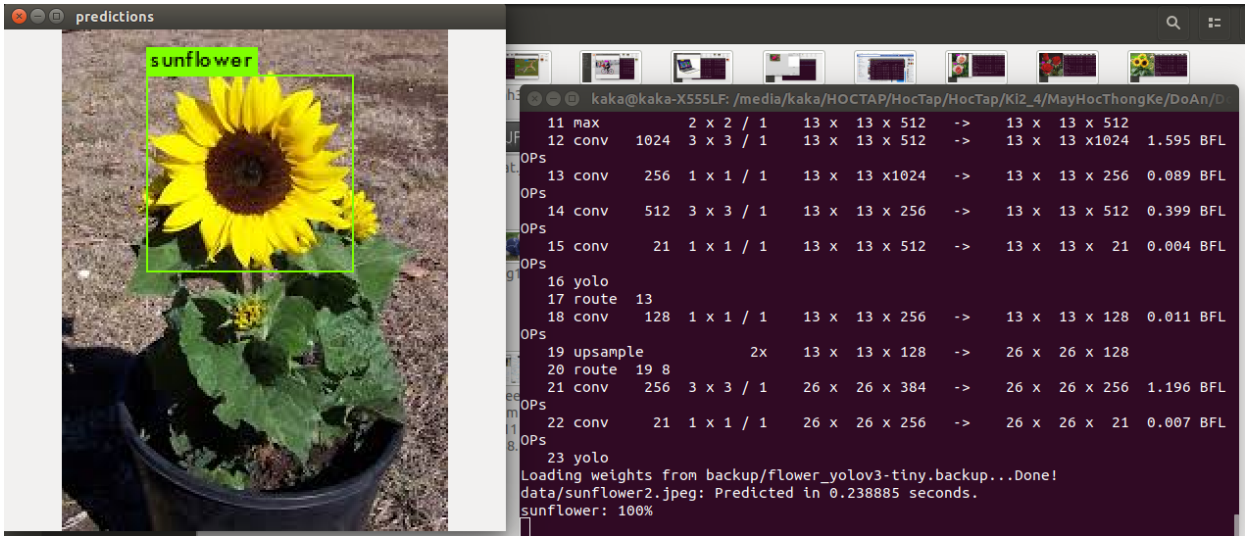
f. Kết quả

Một số kết quả thu được









g. Nhận xét

Yolo v3- tiny chạy nhanh, có thể detect khá ổn với dữ liệu không có nhiều biến đổi so với dữ liệu train.

III. Áp dụng kết quả huấn luyện để xây dựng một ứng dụng hoàn thiện

1. Thêm hàm def load_app_flower() để lấy net và meta từ file cfg của ứng dụng

```
154 def load_app_flower():
155     current_dir = os.getcwd()
156     current_dir = os.path.join(current_dir, "cfg/flower_yolov3-tiny.cfg")
157     print(current_dir)
158     net = load_net(current_dir, "backup/flower_yolov3-tiny.backup", 0)
159     meta = load_meta("cfg/flower.data")
160     print('Loaded net and meta\n')
161     return net,meta
```

Hàm này sẽ tương tự như hàm load_app() lấy net và meta cho file cấu hình của bài toán phân loại hoa

2. Thêm trang web detectflower vào web cũ

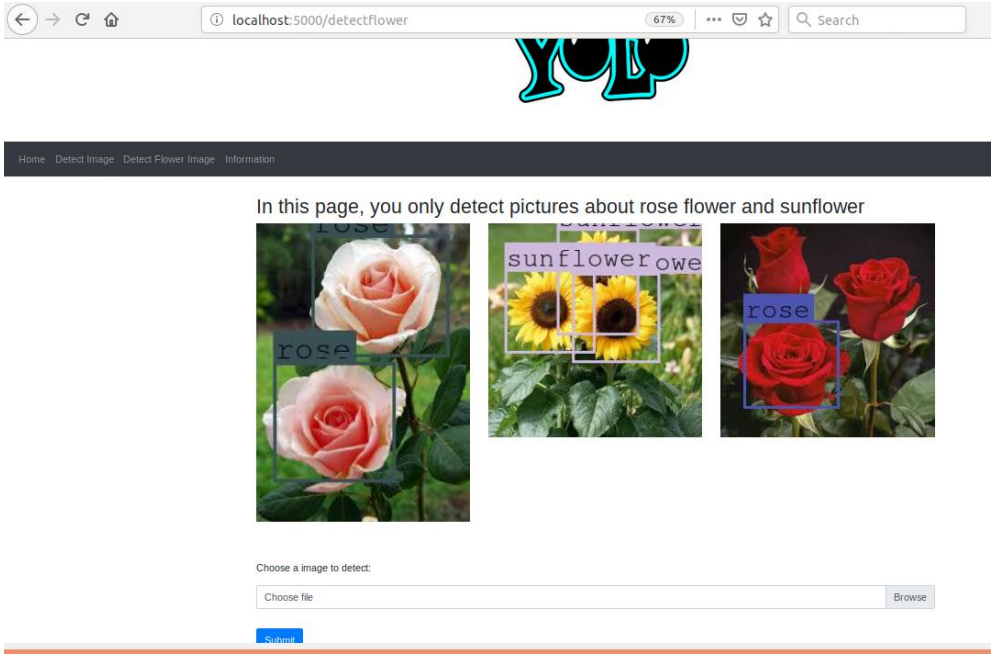
Khai báo link đến trang web

```
31  @app.route('/detectflower')
32  def detectFlowerPage():
33      return render_template('detectflower.html')
```

Khai báo hàm thực thi khi người dùng tải ảnh và submit ảnh

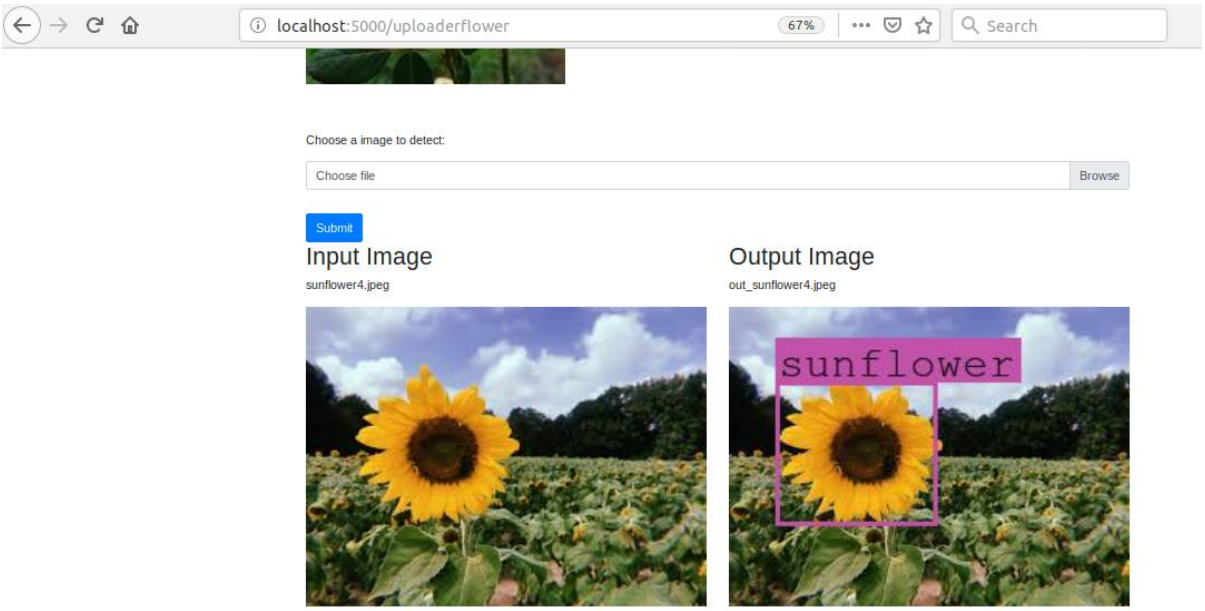
```
52  @app.route('/uploaderflower', methods = ['GET', 'POST'])
53  def upload__flower_file():
```

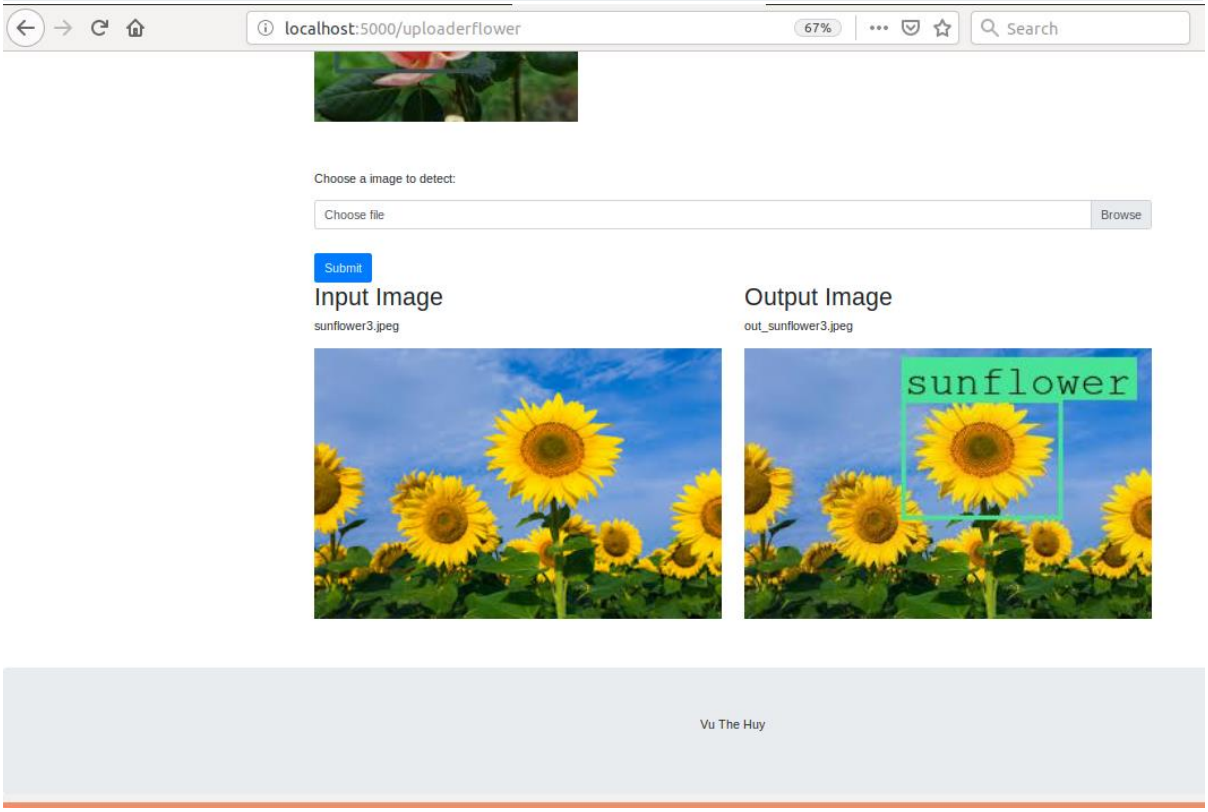
Ảnh về web

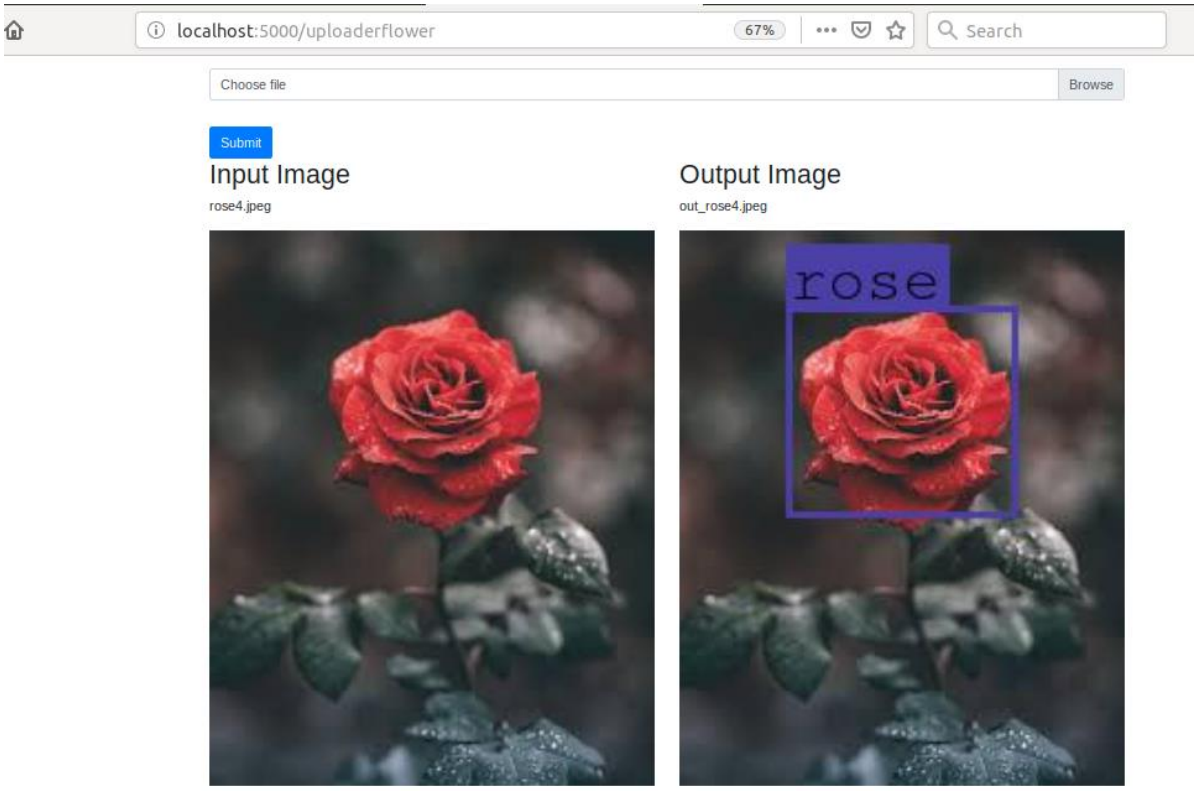


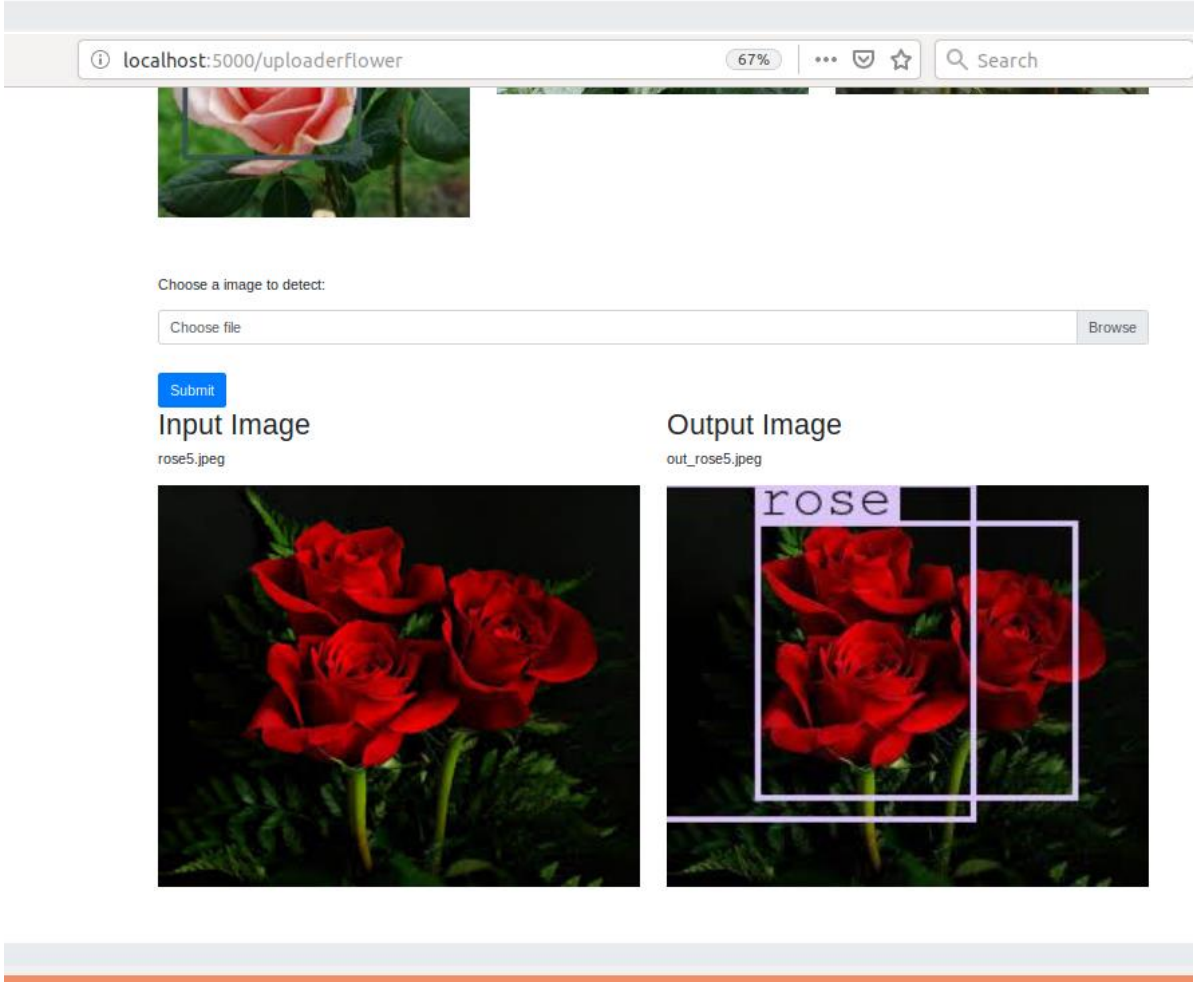
3. Kết quả

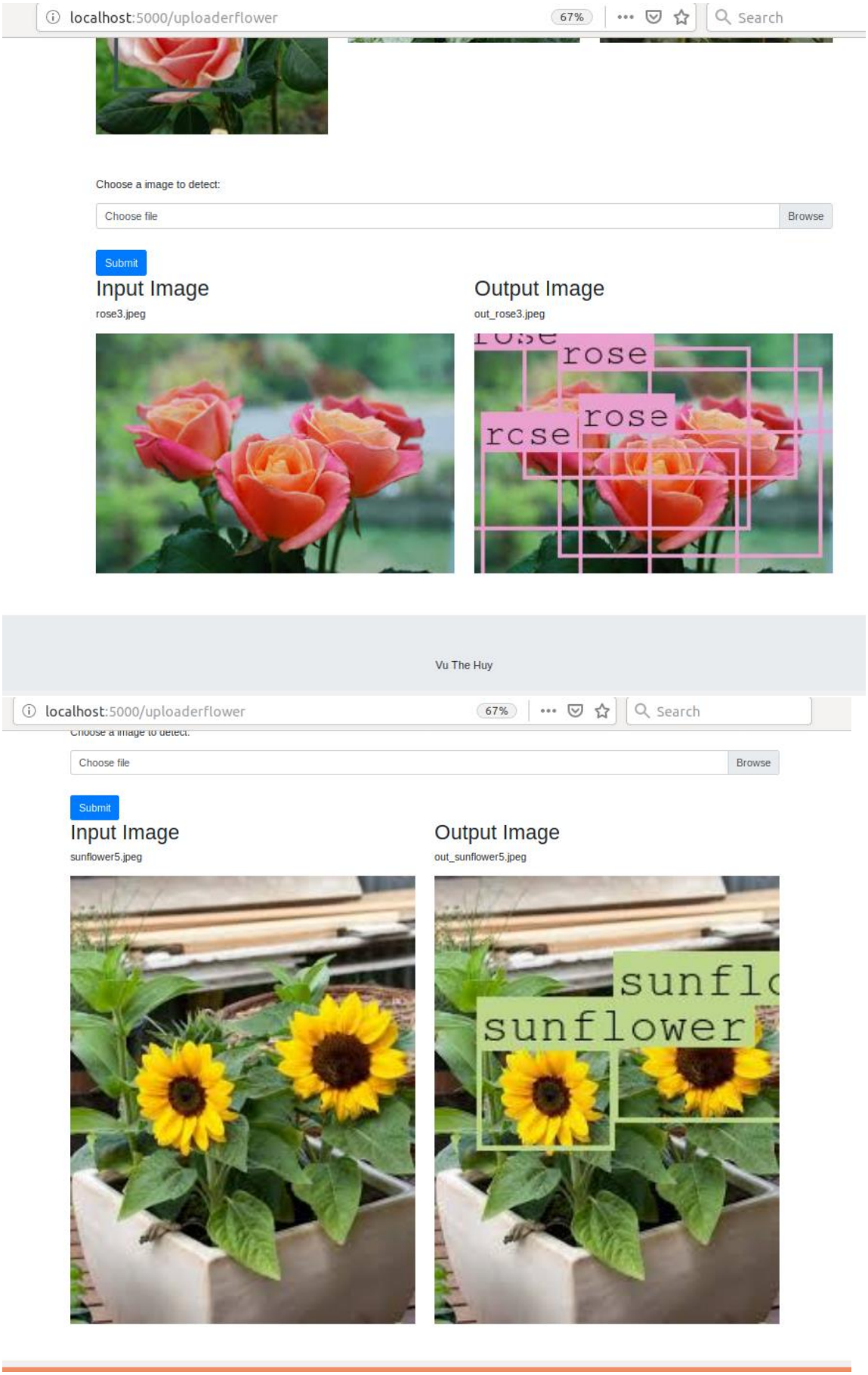
Một số kết quả chạy bằng kết quả huấn luyện






localhost:5000/uploaderflower
67%
Search
Choose file
Browse
Submit
Input Image
rose4.jpeg
Output Image
out_rose4.jpeg
rose


localhost:5000/uploaderflower
67%
Search
Choose a image to detect:
Choose file
Browse
Submit
Input Image
rose5.jpeg
Output Image
out_rose5.jpeg
rose


localhost:5000/uploaderflower
67%
Search
Choose a image to detect:
Choose file
Browse
Submit
Input Image
rose3.jpeg
Output Image
out_rose3.jpeg
rose
rose
rcse
Vu The Huy
localhost:5000/uploaderflower
67%
Search
Choose file
Browse
Submit
Input Image
sunflower5.jpeg
Output Image
out_sunflower5.jpeg
sunflower

4. Nhận xét

Mô hình có độ chính xác khá cao.

Có tốc độ nhanh

⇨ Có thể mở rộng bài toán ra nhiều loài hoa hơn sau khi cập nhật dữ liệu