Giảng viên hướng dẫn: Ths.Võ Thị Kim Anh

Nhóm 5

Tên thành viên: Hồ Thị Mỹ Duyên 1851050022

Lê Nguyễn Kim Ngân 1851010077

Hà Mỹ Duyên 1851010020

BÁO CÁO XỬ LÝ ẢNH

Chủ đề 4: Nhận dạng đối tượng ảnh, phân tích cảnh và hiểu cảnh

1. Cơ sở lý thuyết

Phát hiện đối tượng được coi là một trong những nhiệm vụ thách thức nhất trong lĩnh vực thị giác máy tính.

1.1. Object Detection

1.1.1. Định nghĩa Object Detection

Một trong những lĩnh vực quan trọng của Trí tuệ nhân tạo (Artificial Intelligence) là thị giác máy (Computer Vision). Computer Vision là một lĩnh vực bao gồm các phương pháp thu nhận, xử lý ảnh kỹ thuật số, phân tích và nhận dạng các hình ảnh, phát hiện các đối tượng, tạo ảnh, siêu phân giải hình ảnh và nhiều hơn vậy. Object Detection có lẽ là khía cạnh sâu sắc nhất của thị giác máy đo số lần sử dụng trong thực tế.

1.1.2. Các phương pháp của Object Detection

Bắt đầu sử dụng các phương pháp nhận diện đối tượng hiện đại trong các ứng dụng và hệ thống, cũng như xây dựng các ứng dụng mới dựa trên các phương pháp này.Việc triển nhận diện đối tượng sớm liên quan đến việc sử dụng các thuật toán cổ điển, giống như các thuật toán được hỗ trợ trong OpenCV, thư viện computer vision phổ biến. Tuy nhiên, các thuật toán cổ điển này không thể đạt được hiệu suất đủ để làm việc trong các điều kiện khác nhau.

Việc áp dụng đột phát và nhanh cóng của deep learning vào năm 2012 đã đưa vào sự tồn tại các thuật toán và phương pháp phát hiện đối tượng hiện đại và chính xác cao như R-CNN, Fast-RCNN, Faster-RCNN, RetinaNet và nhanh hơn nhưng rất chính xác như SSD và YOLO. Sử dụng các phương pháp và thuật toán này, dựa trên deep learning và cũng dựa trên việc học máy đòi hỏi rất nhiều kiến thức về toán học và việc học sâu. Có hàng triệu chuyên gia lập trình và các nhà phát triển phần mềm muốn tích hợp và tạo ra các sản phẩm mới sử dụng object detection. Nhưng công nghệ này xa tầm tay của họ và phức tạp để hiểu và sử dụng thực tế của nó.

ImageAI, một thư viện python cho phép các lập trình viên và các nhà phát triển phần mềm dễ dàng tích hợp các công nghệ thị giác máy hiện đại vào các ứng dụng hiện có và mới của họ, và chỉ cần sử dụng một vài dòng mã. ImageAI hỗ trợ một

danh sách các thuật toán học máy hiện đại nhất cho việc dự đoán hình ảnh,, nhận diện vật thể, phát diện video,...

1.2. Các thư viện

1.2.1. OpenCV

OpenCV là một trong những thư viện thị giác máy tính phổ biến nhất. OpenCV viết tắt cho Open Source Computer Vision Library. OpenCV là thư viện nguồn mở hàng đầu cho Computer Vision và Machine Learning, và hiện có thêm tính năng tăng tốc GPU cho các hoạt động theo real-time.

1.2.1.1. Các hàm dùng:

cv2.imread()

cv2.resize()

cv2.rectangle()

cv2.putText()

cv2.imshow()

cv2.waitKey()

cv2.destroyAllWindows()

cv2.dnn.readNet()

cv2.dnn.blobFormImage()

cv2.dnn.NMSBoxes()

cv2.FONT_HERSHEY_PLAIN()

1.2.2. Numpy

Numpy (Numeric Python): là một thư viện toán học phổ biến và mạnh mẽ của Python. Cho phép làm việc hiệu quả với ma trận và mảng, đặc biệt là dữ liệu ma trận và mảng lớn với tốc độ xử lý nhanh hơn nhiều lần khi chỉ sử dụng "core Python" đơn thuần.

1.2.2.1. Các hàm sử dụng:

np.random.uniform()

np.argmax()

1.3. YOLOv3

1.3.1. Định nghĩa yolov3

YOLOv3 sử dụng darknet53 làm backbone(có 53 lớp chập) và bản thân YOLOv3 có thêm 53 lớp chập nữa. Tổng cộng YOLOv3 ta có 106 lớp chập và đã đưa vào các lớp như: residual block, upsampling, skip connections.

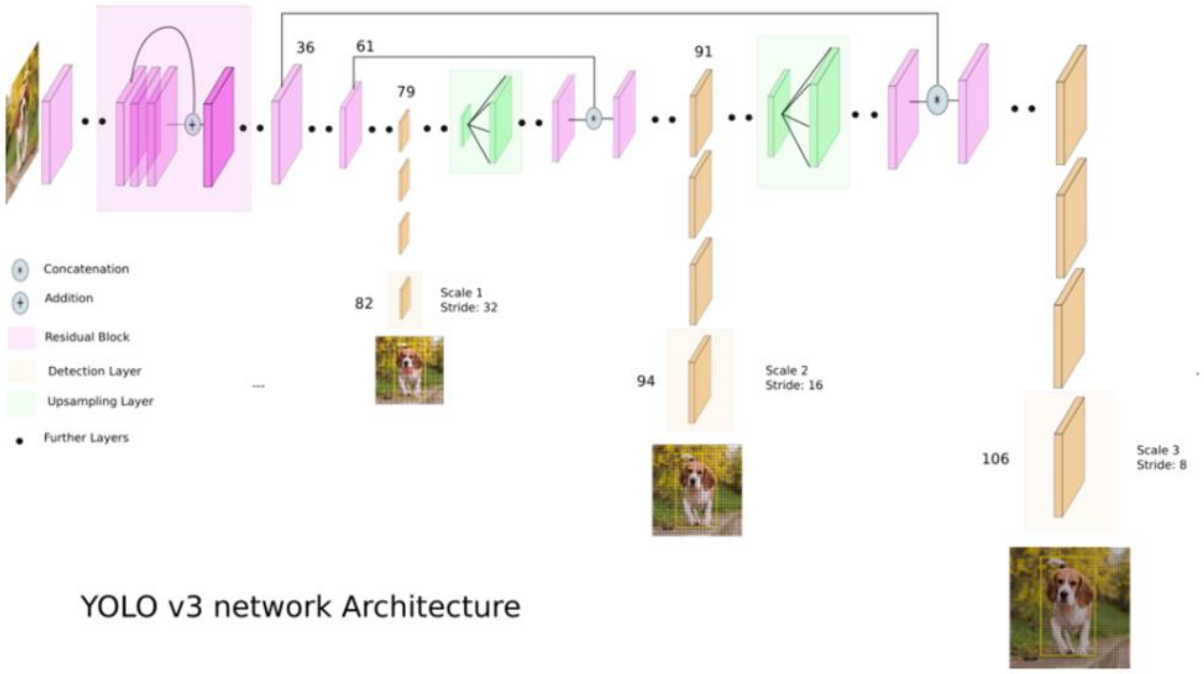
YOLOv3 là thuật toán phát hiện đối tượng theo thời gian thực để xác định các đối tượng cụ thể trong video, nguồn cấp dữ liệu trực tiếp hoặc hình ảnh. YOLO sử dụng các tính năng được học bởi mạng nơ-ron phức hợp sâu để phát hiện một đối tượng. Phiên bản 1-3 của YOLO được tạo ra bởi Joseph Redmon và Ali Farhadi.

Phiên bản đầu tiên của YOLO được tạo vào năm 2016 và phiên bản 3, được thảo luận nhiều trong bài viết này, được thực hiện hai năm sau đó vào năm 2018. YOLOv3 là phiên bản cải tiến của YOLO và YOLOv2. YOLO được triển khai bằng cách sử dụng thư viện học sâu Keras hoặc OpenCV.

Về độ chính xác thì YOLO có thể không phải là thuật toán tốt nhất nhưng nó là thuật toán nhanh nhất trong các lớp mô hình object detection. Nó có thể đạt được tốc độ gần như real time mà độ chính xác không quá giảm so với các model thuộc top đầu.

YOLO là thuật toán object detection nên mục tiêu của mô hình không chỉ là dự báo nhãn cho vật thể như các bài toán classification mà nó còn xác định location của vật thể. Do đó YOLO có thể phát hiện được nhiều vật thể có nhãn khác nhau trong một bức ảnh thay vì chỉ phân loại duy nhất một nhãn cho một bức ảnh.

1.3.2. Cấu trúc của YOLOv3



YOLO v3 network Architecture

1.4. Haar cascades

Về cơ bản là sử dụng các đặc trưng loại *Haar* và sau đó sử dụng thật nhiều đặc trưng đó qua nhiều lượt (*cascade*) để tạo thành một cỗ máy nhận diện hoàn chỉnh.

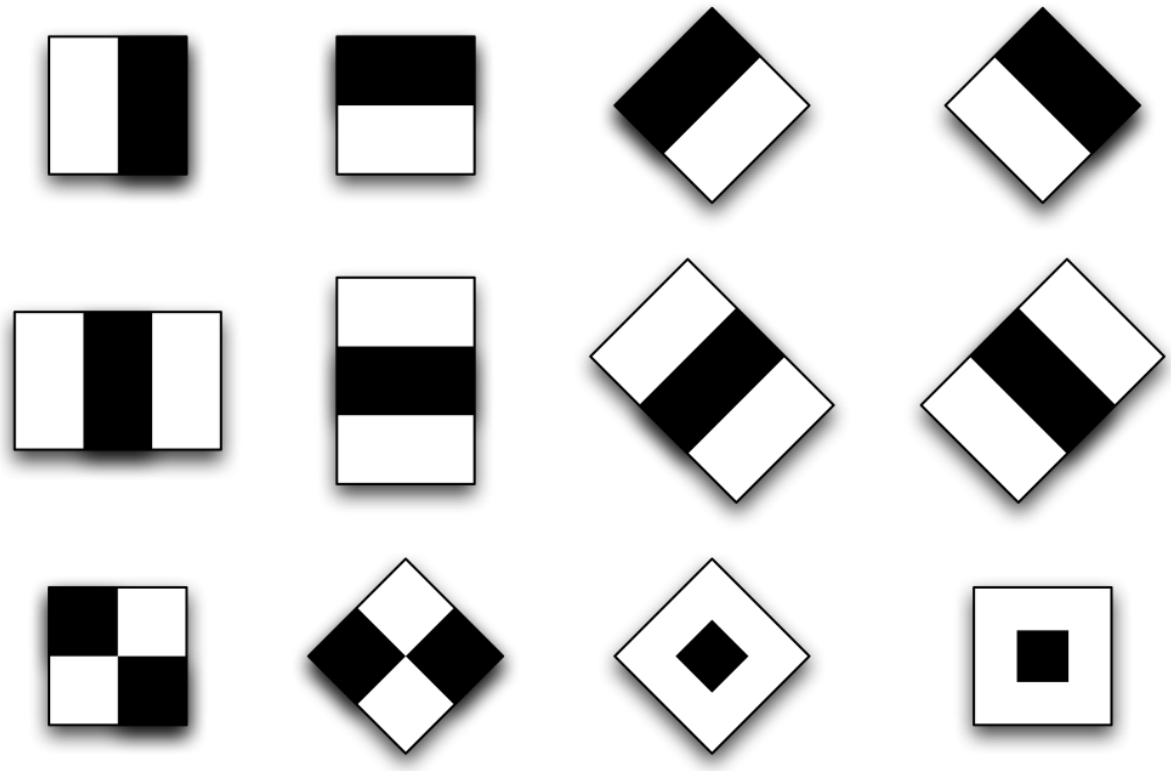
1.4.1. Phát hiện đơn giản theo màu sắc

Nó chỉ đơn giản là phân loại các đối tượng trong hình ảnh theo màu sắc. Tuy nhiện kết quả nhận diện có thể sai sót khi điều kiện ánh sáng thay đổi. Tất nhiên, loại phát

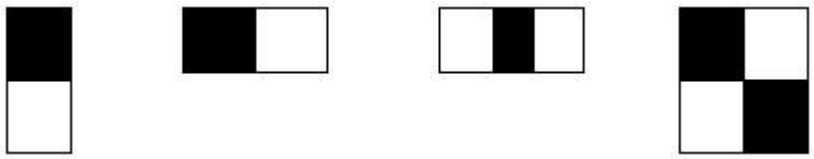
hiện này không phù hợp với hầu hết các ứng dụng trong thế giới thực, chỉ vì nhu cầu hiệu chỉnh và bảo trì liên tục.

1.4.2. Phát hiện theo tính năng haar-like

Các đặc trưng Haar-Like là những hình chữ nhật được phân thành các vùng khác nhau như hình:



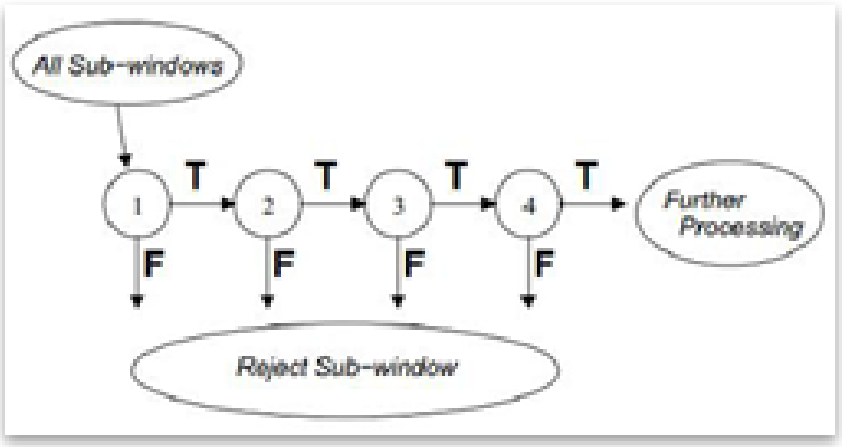
Đặc trưng do Viola và Jones công bố gồm 4 đặc trưng cơ bản để xác định khuôn mặt người. Mỗi đặc trưng Haar-Like là sự kết hợp của hai hay ba hình chữ nhật trắng hay đen như trong hình sau:



Một ví dụ về điều này sẽ là việc phát hiện khuôn mặt của con người. Thông thường, các khu vực xung quanh mắt tối hơn các khu vực trên má. Do đó, một ví dụ về tính năng haar-like để phát hiện khuôn mặt là một tập hợp hai khu vực hình chữ nhật lân cận phía trên vùng mắt và má.

1.4.3. Bộ lọc Cascade Classifier

Cascade Classifier bao gồm một danh sách các giai đoạn, trong đó mỗi giai đoạn bao gồm một danh sách những classifier yếu. Hoạt động trên nguyên tắc kết hợp tuyến tính các weak classifiers để hình thành một trong các classifiers. Viola và Jones dùng AdaBoost kết hợp các bộ phân loại yếu sử dụng các đặc trưng Haar-like theo mô hình phân tầng (cascade) như sau:



2. Trình bày ứng dụng coding

2.1. Giới thiệu PyCharm

Trình sửa code thông minh

- Giúp chúng ta viết mã chất lượng cao hơn
- Nó bao gồm các lược đồ màu (color schemes) cho từ khóa, lớp và hàm. Điều này giúp tăng khả năng đọc và hiểu mã.
- Giúp xác định lỗi một cách dễ dàng.

Hỗ trợ cho Thư viện Khoa học Python

- PyCharm hỗ trợ các thư viện khoa học của Python như Matplotlib, NumPy và Anaconda.
- Các thư viện khoa học này giúp xây dựng các dự án về Khoa học Dữ liệu và Học máy.
- Hỗ trợ các biểu đồ tương tác giúp các nhà phát triển hiểu dữ liệu tốt hơn.
- Nó có khả năng tích hợp với những công cụ khác nhau như IPython, Django và Pytest. Sự tích hợp này giúp thúc đẩy các giải pháp độc đáo.

Ưu điểm

- Cài đặt PyCharm rất dễ dàng.
- PyCharm là một IDE dễ sử dụng.
- Có rất nhiều plugin hữu ích và phím tắt hữu ích trong PyCharm.
- PyCharm tích hợp các tính năng của thư viện và IDE như tự động hoàn thành và tô màu.
- Nó cho phép xem mã nguồn trong một cú nhấp chuột.
- Tiết kiệm thời gian phát triển phần mềm
- Tính năng đánh dấu lỗi trong code giúp nâng cao hơn nữa quá trình phát triển.
- Cộng đồng các nhà phát triển Python vô cùng lớn và chúng ta có thể giải quyết các thắc mắc/ nghi ngờ của mình một cách dễ dàng.

Nhược điểm

- PyCharm không miễn phí và phiên bản Professional của nó khá đắt.
- Tính năng tự điền (auto-complete) sẽ không tốt cho các lập trình viên newbie
- Nó có thể gây ra sự cố trong khi sửa chữa các công cụ như venv.

3. Phân tích so sánh các phương pháp và ý tưởng nhận xét riêng của nhóm

3.1. Phân tích code

3.1.1. HAAR CASCADES

Đầu tiên chúng ta hãy nhập thư viện Python được sử dụng trong bài này:

```
import cv2
import matplotlib.pyplot as plt
```

Sau đó, chúng ta tải hình ảnh mà ta sẽ sử dụng trong bài này bằng cách sử dụng hàm imread của cv2:

```
img = cv2.imread("hinh8.jpg")
```

hinh8.jpg là tên file ảnh

Hình ảnh chúng em sử dụng ở đây là hình ảnh con đường có biển báo stop (biển báo dừng lại).

Để bắt đầu, chúng ta sử dụng thư viện CV2 để chuyển đổi hình ảnh của mình thàng thang màu xám và RGB

```
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
imaging_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

Sau đó, ta tiến hành thêm file XML của trình nhận diện đối tượng ảnh Haar Cascades

```
xml_data = cv2.CascadeClassifier('stop_data.xml')
```

stop_data.xml là tên file XML được sử dụng trong bài này

Kế tiếp ta tiến hành phân tích đối tượng trong file XML

```
detecting = xml_data.detectMultiScale(img_gray,minSize = (30, 30))

amountDetecting = len(detecting)

if amountDetecting != 0:
    for (a, b, width, height) in detecting: # Nếu có nhiều hơn 1 biển báo
        cv2.rectangle(imaging_rgb, (a, b), # Đánh dấu đối tượng
                      (a + height, b + width),
                      (0, 275, 0), 9)
```

Hàm detectMultiScale() để để kiểm tra xem có phát hiện đối tượng nào từ hình ảnh hay không và đánh dấu phần được phát hiện.

Ở đây ta sử dụng điều kiện if trong chương trình để kiểm tra đối tượng đó có được phát hiện hay không và nếu đối tượng được phát hiện sẽ tiến hành đánh dấu phần đối tượng được phát hiện bằng cách sử dụng vòng lặp for với các hàm cv2.

Sau đó, chúng ta vẽ biểu đồ phụ của hình ảnh bằng cách sử dụng hàm subplot () của plt và đưa ra các tham số trong đó. Cuối cùng, chúng ta đã sử dụng hàm plt.imshow () và plt.show() để hiển thị hình ảnh đầu ra.

```
plt.subplot(1, 1, 1)
plt.imshow(img_gray)
plt.show()
```

Và đây là kết quả ta nhận được sau khi chạy chương trình, có thể thấy chương trình nhận dạng được biển báo stop trong hình ảnh:

3.1.2. YOLOv3

Đầu tiên chúng ta hãy nhập thư viện Python được sử dụng trong bài này:

```
import cv2
import numpy as np
```

Sau đó, chúng ta tiến hành tải những file dataset của Yolo mà ta sẽ sử dụng trong bài này bằng cách sử dụng hàm readNet của cv2:

```
net = cv2.dnn.readNet("yolov3.weights", "yolov3.cfg")
classes = []
```

```
with open("coco.names", "r") as f:
    classes = [line.strip() for line in f.readlines()]
output_layers = net.getUnconnectedOutLayersNames()
colors = np.random.uniform(0, 255, size=(len(classes), 3))
```

yolov3.weights: Là file pretrain weights của Yolo.

yolov3.cfg: Là file config của YOLO.

Coco.names: là file chứa các đối tườn để nhận diện của Yolo.

Sau đó, chúng ta tải hình ảnh mà ta sẽ sử dụng trong bài này bằng cách sử dụng hàm imread của cv2:

```
img = cv2.imread("hinh8.jpg")
height, width, channels = img.shape
```

hinh8.jpg là tên file ảnh

Hình ảnh chúng em sử dụng ở đây là hình ảnh con đường có biển báo stop (biển báo dừng lại) và những chiếc ô tô.

Kế tiếp ta tiến hành nhận dạng đối tượng ảnh

```
blob = cv2.dnn.blobFromImage(img, 0.00392, (416, 416), (0, 0, 0), True,
crop=False)#phân tích ảnh dưới dạng nhị phân 1/255


net.setInput(blob)
outs = net.forward(output_layers)
```

Hàm blobFromImage() để phân tích ảnh dưới dạng nhị phân 1/255 và định dạng kích thước ảnh

Sau đó, chúng ta tiến hành hiển thị thông tin đối tượng sau khi nhận dạng được

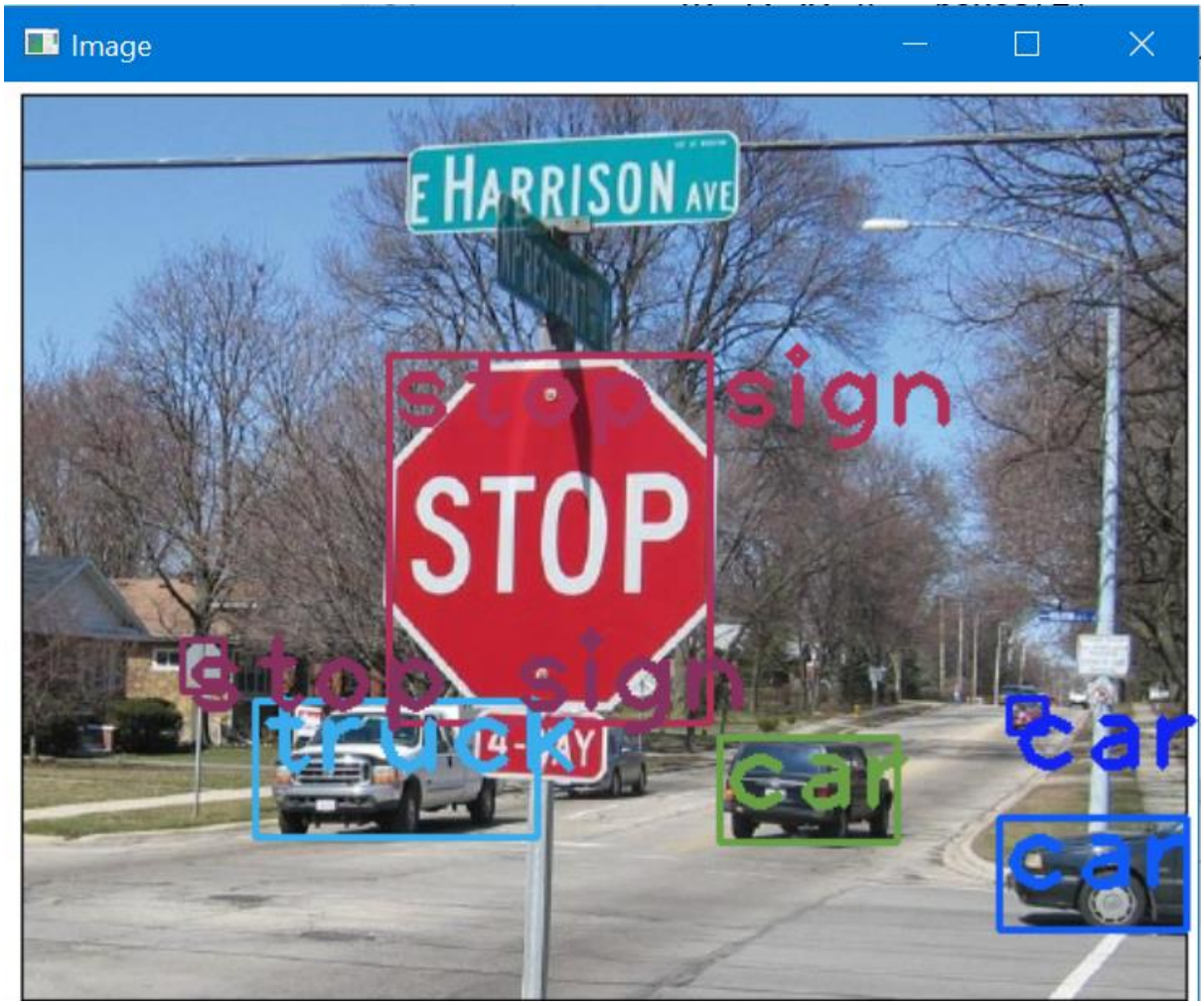
```
class_ids = []
confidences = []
boxes = []
for out in outs:
    for detection in out:
        scores = detection[5:]
        class_id = np.argmax(scores)
        confidence = scores[class_id]
        if confidence > 0.5:
            # Object detected
            center_x = int(detection[0] * width)
            center_y = int(detection[1] * height)
            w = int(detection[2] * width)
            h = int(detection[3] * height)
```

```
# Rectangle coordinates
            x = int(center_x - w / 2)
            y = int(center_y - h / 2)

            boxes.append([x, y, w, h])
            confidences.append(float(confidence))
            class_ids.append(class_id)

#định dạng màu chữ, fon chữ và khung nhận dạng đối tượng
indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)
print(indexes)
font = cv2.FONT_HERSHEY_PLAIN
for i in range(len(boxes)):
    if i in indexes:
        x, y, w, h = boxes[i]
        label = str(classes[class_ids[i]])
        color = colors[i]
        cv2.rectangle(img, (x, y), (x + w, y + h), color, 2)
        cv2.putText(img, label, (x, y + 30), font, 3, color, 3)
```

Cuối cùng, chúng ta đã sử dụng hàm cv2.imshow () để hiển thị hình ảnh trong đầu ra hàm cv2.waitKey() để không cho thoát cửa sổ lập tức mà người dùng phải nhấn phím bất kỳ để thoát và hàm cv2.destroyAllWindows()Nếu bạn có nhiều cửa sổ đang mở và bạn không cần mở những cửa sổ đó, bạn có thể sử dụng cv2.destroyAllWindows()để đóng tất cả những cửa sổ đó..

```
cv2.imshow("Image", img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Và đây là kết quả ta nhận được sau khi chạy chương trình:



3.2. So sánh hai phương pháp:

	YOLOv3	Haar Cascades
Dataset	file coco.names: khoảng 80 đối tượng có sẵn.	file xml
Dung lượng file dataset	nhẹ hơn	nặng hơn
Độ chính xác	cao	thấp hơn
code	phức tạp	đơn giản dễ hiểu
Tốc độ thực thi	nhanh	chậm

Đánh giá: Sau khi thực hiện hai phương pháp trên, nhóm chúng em cảm thấy mỗi phương pháp đều có ưu nhược điểm riêng. YOLOv3 nhận diện đối tượng có độ chính xác, tốc độ xử lý nhanh hơn. Nhưng về nhận diện khuôn mặt thì Haar Cascades có phần nổi trội hơn.

4. Tiến trình làm việc nhóm

4.1. Khó khăn

Trong quá trình làm bài tập lớn, chúng em đã gặp không ít khó khắn, trong đó khó khăn nhất là việc tìm tài liệu, đa phần các tài liệu tìm được là tài liệu tiếng anh nên chúng em có khó khăn về rào cản ngôn ngữ nhưng tụi em đã giải quyết được vấn đề này. Sau khi tìm được tài liệu phù hợp thì chúng em gặp phải một số vấn đề về code, trong quá trình code thì phát sinh khá nhiều lỗi nhưng chúng em đã tìm ra được lỗi của mình. Cuối cùng, chúng em chưa hoàn thành được phần nghiên cứu thêm đó là tự tạo dataset cho phần YOLO object detection, chúng em chưa thể chạy thành công phần dataset do mình tạo ra, hướng phát triển của chúng em là sẽ tiếp tục phát triển thêm phần nghiên cứu ấy.

4.2. Thuận lợi

Qua việc học lý thuyết và được làm cái bài tập thực hành nên chúng em đã có được một lượng kiến thức cơ bản, nhờ vậy tụi em đã áp dụng được vào bài tập lớn. Bên cạnh đó, nhóm em có 3 thành viên nên số lượng tài liệu chúng em chia nhau tìm hiểu cũng đạt kha khá kiến thức để vận dụng vào bài tập này.

5. Lời cảm ơn

Sau khi hoàn thành bài tập lớn, chúng em có thêm kiến thức trong lĩnh vực xử lý ảnh, đặc biệt là trong việc nhận dạng đối tượng trong ảnh. Chúng em xin gửi lời cảm ơn chân thành đến cô đã hướng dẫn chúng em thực hiện bài tập lớn này.