KHOA CÔNG NGHỆ THÔNG TIN

ĐẠI HỌC KHOA HỌC TỰ NHIÊN THÀNH PHỐ HỒ CHÍ MINH

ĐẠI HỌC QUỐC GIA TPHCM

HỆ THỐNG MULTIMEDIA

NHẬN BIẾT BIỂN BÁO TỐC ĐỘ

Thực hiện:

20C12007 – Trần Đình Lâm

20C11035 – Trương Thế Kiệt

20C11040 – Đặng Nhật Minh

BÁO CÁO ĐỒ ÁN CÀI ĐẶT

NĂM HỌC 2020-2021

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
227 Nguyễn Văn Cừ, Phường 4, Quận 5, TP.HCM
Điện Thoại: (08) 38.354.266 - Fax:(08) 38.350.096

BẢNG THÔNG TIN CHI TIẾT NHÓM

Tên nhóm:	**K2014**		
Số lượng:	**3**		
MSSV	**Họ tên**	**Email**	**Điện thoại**
20C12007	Trần Đình Lâm	tdlam123@gmail.com	0383522356
20C11035	Trương Thế Kiệt	truongthekiet709@gmail.com	
20C11040	Đặng Nhật Minh	minhdangnhat685@gmail.com	

BẢNG PHÂN CÔNG & ĐÁNH GIÁ HOÀN THÀNH CÔNG VIỆC			
Người thực hiện	**Công việc thực hiện**	**Mức độ hoàn thành**	**Đánh giá của nhóm**
20C12007 Trần Đình Lâm	Cài đặt model classification đã tham khảo, tìm hiểu và giải thích mô hình mạng CNN được đề xuất	70%	7/10
	Train và test model với tỉ lệ 0.8/0.2, Thử nghiệm chạy train bằng GPU		
20C11035 Trương Thế Kiệt	Chuẩn hóa dữ liệu để train mạng phân lớp	70%	7/10
	Thực hiện integrate hai model và detect dữ liệu từ camera/video/hình ảnh		
	Viết báo cáo		
	Cài đặt model classification đã tham khảo và training với bộ dữ liệu traning chỉ bao gồm những biển báo về tốc độ(6 loại) và test integration với model detection để so sánh độ chính xác		
20C11040 Đặng Nhật Minh	Chạy train mô hình yolo theo tutorial hướng dẫn là output file weight	70%	7/10
	Viết báo cáo		

MỤC LỤC

I. TỔNG QUAN

1. Mô tả bài toán

Trong các loại biển báo giao thông, các biển báo quy định về tốc độ xuất hiện với tần suất khá lớn. Khi di chuyển trên đường, chúng ta đôi khi không nhớ ra ý nghĩa hoặc không chú ý tới các biển báo tốc độ đó, dẫn đến việc vi phạm luật giao thông khi có thể chạy quá tốc độ. Mặc dù trên mỗi phương tiện giao thông đều có đồng hồ hiển thị tốc độ trên bảng điều khiển (dashboard) chung, tuy nhiên một số trường hợp người điều khiển không chú tâm vào tốc độ, cũng như không thể nhớ được đoạn đường đang chạy để có thể đối sánh và điều chỉnh lại ngay lập tức được.

Với mong muốn giải quyết vấn đề này bằng những kiến thức đã học về mạng neural và xử lý hình ảnh, nhóm quyết định sẽ nghiên cứu để tạo ra một ứng dụng với đầu vào là video từ camera hành trình, giúp người dùng biết được những biển báo tốc độ trên đường và đưa ra cảnh báo về giới hạn tốc một cách dễ dàng, nhanh chóng và trực quan nhất.

2. Các thư viện sử dụng trong đồ án

Tên thư viện	**Mục đích sử dụng**
os	làm việc với các tập tin và thư mục
pandas	làm việc với dataset
numpy	tính toán
matplotlib và seaborn	trực quan hóa dữ liệu bằng các dạng biểu đồ
PIL	thao tác trên hình ảnh
Sciki-learn sklearn	cung cấp cài đặt của các thuật toán thường dùng trong machine learning
Tensorflow và Keras	xây dựng mô hình neural network
Yolo v4	Thực hiện detect biển báo tốc độ và lấy bouding box

II. KIẾN TRÚC CHƯƠNG TRÌNH

1. Mô tả cấu trúc

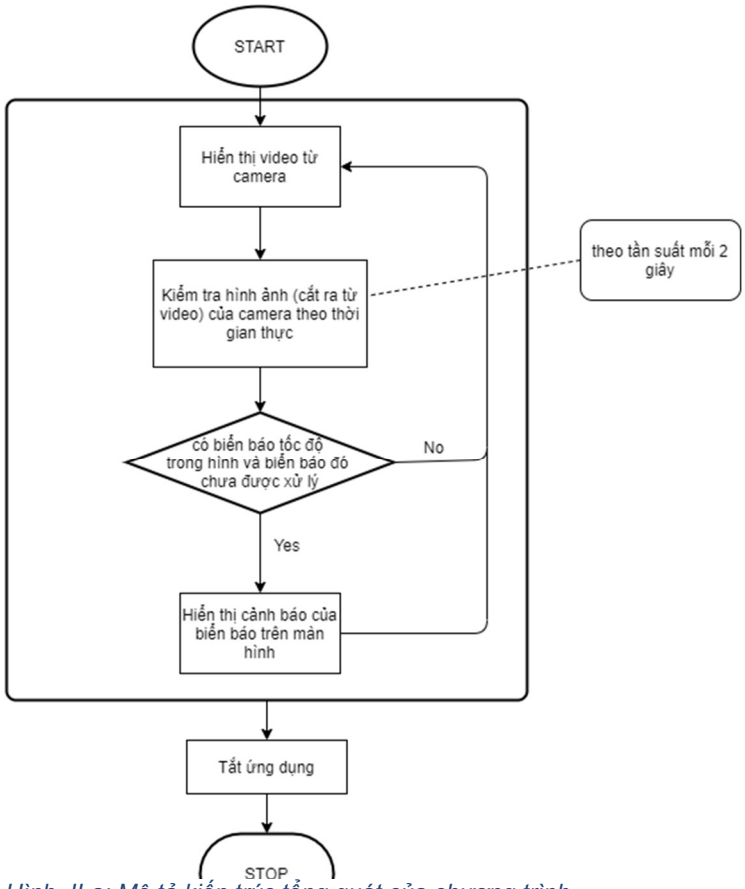
Cấu trúc chương trình ứng dụng gồm 2 thành phần chính, tương ứng với 2 nhiệm vụ chính để ứng dụng có thể nhận diện và cảnh báo cho người dùng, bao gồm:

- Kiểm tra vị trí của biển báo trên hình ảnh (hình ảnh từ video hoặc từ camera trực tiếp của phương tiện)
- Nhận diện và xác định nội dung tốc độ mà biển báo quy định.

Cụ thể, quy trình xử lý sẽ qua các bước sau:

- Bước 1: Hiển thị video từ camera
- Bước 2: Nếu xe chưa dừng, Lặp mỗi 2 giây
- Bước 3: Kiểm tra có biển báo tốc độ trong khung hình, và biển báo chưa được xử lý. Nếu không quay lại bước 2.
- Bước 4: Nếu có biển báo, tiến hành phân loại và hiển thị cảnh báo lên màn hình. Quay lại bước 2.
- Bước 5: Kết thúc.

Quy trình kết hợp của từng bước được thể hiện ở *Hình II-a*



Hình II-a: Mô tả kiến trúc tổng quát của chương trình

2. Một số lưu ý

- Khi chương trình chạy, giá trị input có thể là một tấm hình/video/camera được đưa qua mạng detection để lấy ra bounding box của những nói có biển báo tốc độ.
- Đối với input là giá trị từ video/camera các frame được xử lí 2s một lần để hạn chế việc delay.
- Sau khi lấy được bounding box. Có thể lưu lại những hình ảnh đó hoặc không.
- Sau đó tiến hành phần lớp từ những hình ảnh đã được cắt ra từ một frame.
- Sau khi phân loại xong sẽ ghi log lại giá trị tính toán được.

III. PHÂN TÍCH BỘ DỮ LIỆU GTSRB

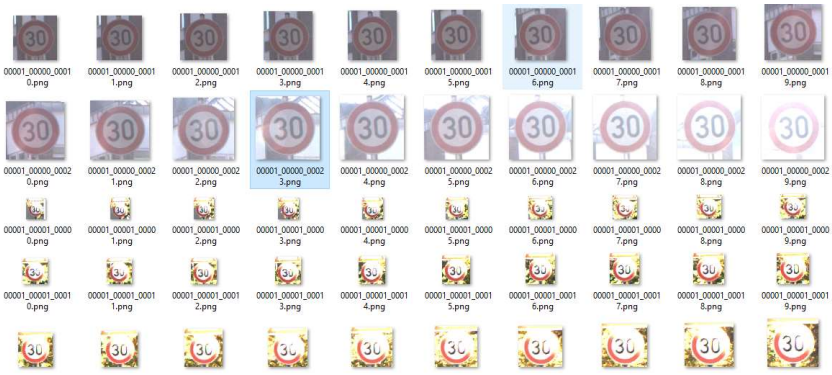
1. Bộ dữ liệu dùng để phân lớp

i. Cấu trúc Dataset GTSRB

- **German Traffic Sign Recognition Benchmark(GTSRB)** là bộ dữ liệu phổ biến và được dùng rộng rãi trong giới khoa học. Thông thường GTSEB được sử dụng để trainning và test các bài toán nhận diện và phân loại các biển báo giao thông đường bộ. Bộ dữ này được phần chia thành 43 loại với tổng số lượng hình ảnh cho bộ dữ liệu train lên đến 39209 hình.
- Các loại biển báo trong bộ data **GTSRG**:

Hình I: Các loại biển báo trong GTSRB

- Hình ảnh trong mỗi loại biển báo rất đa dạng về độ lớn, độ tương phản, noise và blurred.



Hình III-b: Độ đa dạng hình ảnh trong từng loại

ii. Chuẩn hóa data

- Do hình ảnh với size khác nhau, nên mỗi tấm ảnh từ nguyên gốc với size w x h x 3 sẽ được đưa về dạng được resize về 50_x50_x3.
- Sau khi chuẩn hóa xong thì dùng numpy để lưu lại 2 file vào folder "numpy", để không phải scan lại tập train.
- Lúc này ta có được tập train kích thước (39209, 50, 50, 3)

iii. Load data

- Tập dữ liệu input sau khi được chuẩn hóa xong sẽ được chia làm hai bộ là bộ train và bộ test.
- Bốn biến output tương ứng sau khi load data:
    - x_train có x_val tương ứng
    - y_train có y_val tương ứng

iv. Nguồn bộ dữ liệu

- https://www.kaggle.com/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign

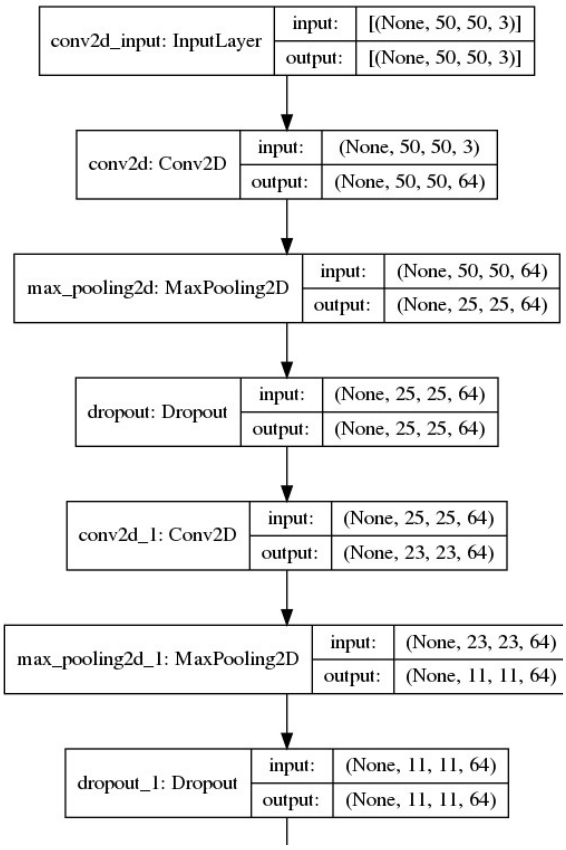
2. Bộ dữ kiệu dùng để phán hiện biển báo giao thông

- Bộ dữ liệu được tác giả đánh label sẵn được chia thành bốn loại là : speed limit, yield, mandatory, other
- Trong bài toán lần này mình chỉ cần một loại là biển báo tốc độ.
- Nguồn bộ dữ liệu https://onedrive.live.com/download?cid=A86CBC7F31A1C06B&resid=A86CBC7F31A1C06B%21121&authkey=AMUUk0Np4tqH3n4

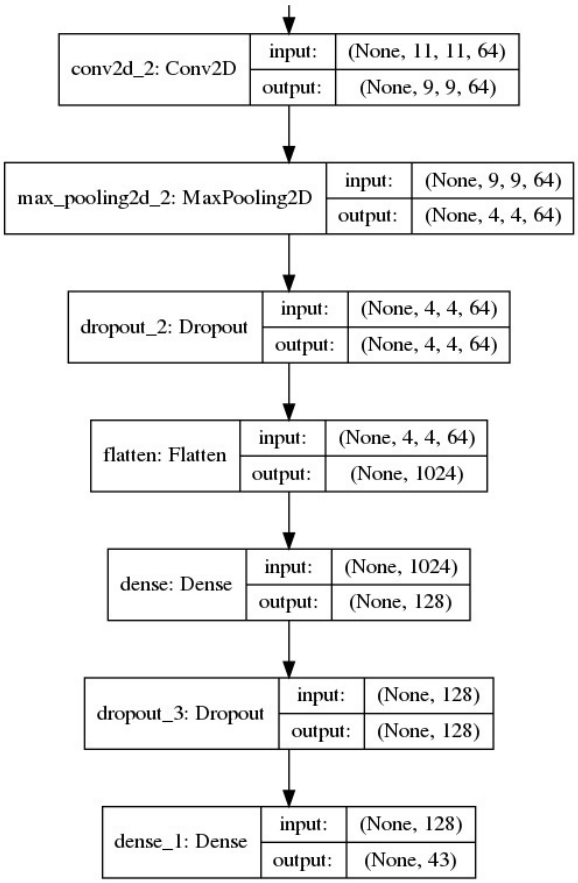
IV. CÀI ĐẶT MÔ HÌNH MẠNG

1. Cấu trúc mạng phân lớp biển báo nhóm đã cài đặt

Dựa theo cấu trúc một số mạng CNN được sử dụng tương ứng cho tập dữ liệu GTSRB trên Kaggle [3], cùng một số điều chỉnh mới, nhóm đã xây dựng mô hình mạng CNN như hình *IV-b* và *IV-c*:



Hình IV-b Cấu trúc 2 layer đầu của mạng



Hình IV-c Cấu trúc 4 layer sau của mạng

Theo đó, cấu trúc mạng bao gồm 6 layer chính:

- Layer 1 đến layer 3: Convolutional với kernel size 3x3, sau đó kết hợp max pooling 2x2 và dropout 50%
- Layer 4: Là một lớp Flatten để dàn phẳng output của layer 3
- Layer 5 và Layer 6: Fully-connected layer
- Đầu ra cuối cùng là vector 43 chiều, biểu thị 43 loại biển báo giao thông cần phân loại.

Sau khi cài đặt, các thông số mô tả mô hình biểu thị như hình *IV-d* sau:

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 50, 50, 64)        1792
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 25, 25, 64)        0
_________________________________________________________________
dropout (Dropout)            (None, 25, 25, 64)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 23, 23, 64)        36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 11, 11, 64)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 11, 11, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 9, 9, 64)          36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 4, 4, 64)          0
_________________________________________________________________
dropout_2 (Dropout)          (None, 4, 4, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 1024)              0
_________________________________________________________________
dense (Dense)                (None, 128)               131200
_________________________________________________________________
dropout_3 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 43)                5547
=================================================================
Total params: 212,395
Trainable params: 212,395
Non-trainable params: 0
```

Hình IV-d Các thông số của model cài đặt bằng Keras

Hình IV-i Layer 4,5,6

2. Huấn luyện pha detection object bằng Yolo

Để thực hiện train cho yolo, nhóm đã sử dụng một bộ data đã được đánh nhãn sẵn nhằm giúp tiết kiệm thời gian cho việc gán nhãn.

Việt training được thực hiện theo hướng dẫn ở NoteBook Google Colab theo đường dẫn: https://colab.research.google.com/drive/1VNc-Ywrs1XmfHcsq-BWpXZ5Zv_A2FcFn?usp=sharing&fbclid=IwAR22UkmNlEEFKXNZbHWQaUattjF14z_23_dRQnYybem5jfcI9fuowimPSYM

3. Kết hợp hai model vào 2 pha chính của ứng dụng

Ở pha detection, chương trình sử dụng thư viện OpenCV (version 4.5.2.54) để thực hiện tải nạp lại các bộ trọng số của mô hình cũng như những cấu hình cần thiết để chương trình có thể thực hiện detection.

Sau khi detect được khung hình chứa đối tượng đã nhắm đến (biển báo giao thông tốc độc), chương trình sử dụng thư viện Keras để tải nạp lại model đã train để sử dụng output từ model detection để xem biển báo đó thuộc loại nào.

Kết quả cuối cùng được xuất ra màn hình phục vụ quan sát cho người dùng cuối.

Chi tiết mã nguồn kết hợp 2 model đã có được nhóm lưu lại trong đường dẫn Github sau: https://github.com/TruongTheKiet/multimedia/blob/master/main.py

V. CHẠY CHƯƠNG TRÌNH VÀ ĐÁNH GIÁ KẾT QUẢ

1. Đánh giá về detection và get bouding box

Mô hình train yolo để get boudinging bõ của biển báo giao thông hoạt động tương đối chính xác, với hình ảnh độ lớn biển báo gần tương đồng với bộ dữ liệu train.

Kết quả của một thử nghiệm cụ thể từ hình ảnh input ngẫu nhiên có thể lấy ra được các hình ảnh biển báo tốc độ:

Ảnh đầu vào:

Hình IV-a Ví dụ đầu vào

Ảnh đầu ra:

Hình IV-b Kết quả đầu ra 1

Hình IV-c Kết quả đầu ra 2

2. Đánh giá về phần loại biển báo

Model có thể phân loại được các biển báo thuộc loại cảnh báo tốc độ. Tuy nhiên vẫn còn mốt số giới hạn như khi hình ảnh input đầu vào với chất lượng thấp gây nhiễu hoặc quá mờ, mô hình vẫn chưa thực hiện được chính xác việc phân loại.

3. Kết hợp hai mô hình lại với nhau

Việc kết hợp hai mô hình lại với nhau cũng mang lại kết quả tương đối với những hình ảnh input với chất lượng không quá kém.

Output của mô hình detection sẽ làm input để thực hiện làm phân lớp.

Hạn chế của app hiện tại là không thể thực hiện mượt mà được trên thời gian thực cũng như trên một video nào đó. Hạn chế do việc xử lí cần nhiều thời gian để hoàn thành nên sẽ đêm lại độ trê cũng như sự giật lag khi đang xem video/camera. Ví dụ cụ thể như trường hợp sau:

Input một hình ảnh:

Hình V-a Giá trị log ra là 30Km/h

VI. KẾT LUẬN

Với mục tiêu tìm hiểu, cài đặt và thực hiện training một model (train yolo cũng như một model nhóm cài đặt), nhóm đã thực hiện được, và tự đánh giá 70%, cụ thể:

- Sử dụng được các thư viện phổ biến trong lĩnh vực Computer Vision và Deep Learning.
- Giá trị đầu ra chưa đúng như mong đợi như lúc đầu đặt ra là có thể thực hiện trên thời gian thực.
- Đối với những hình ảnh input có biển báo có thể thực hiện được việc hiện thông tin biển báo tốc độ.

Bên cạnh đó, vì một số lí do nhóm không thực hiện phân loại luôn trên yolo mà lại sử dụng một model khác:

- Do bộ dataset đã đánh nhãn rồi để thực hiện training không có
- Để tiết kiệm thời gian, nhóm tái sử dụng lại mạng ở môn AI nâng cao để thực hiện phân loại ngoài ra thử nghiệm ở một số data train khác nhau như chỉ train data của biển báo tốc độ và đánh giá kết quả.

TÀI NGUYÊN SỬ DỤNG

- Source code model phân lớp: https://github.com/trandinhlam/document/tree/master/AI_NangCao/DoAnCaiDat/Project
- Source code training Yolo: https://colab.research.google.com/drive/1VNc-Ywrs1XmfHcsq-BWpXZ5Zv_A2FcFn?usp=sharing&fbclid=IwAR22UkmNlEEFKXNZbHWQaUattjF14z_23_dRQnYybem5jfcI9fuowimPSYM
- Source code integration 2 model: https://github.com/TruongTheKiet/multimedia/blob/master/main.py
- Bộ dữ liệu GTSRB: https://www.kaggle.com/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign
- Bộ dữ liệu training yolo : https://onedrive.live.com/download?cid=A86CBC7F31A1C06B&resid=A86CBC7F31A1C06B%21121&authkey=AMUUk0Np4tqH3n4
- File trọng số cho mô hình yolo: https://drive.google.com/file/d/1-cGUzkbdTl40t4Ef-DnITBa5iC7iAoXZ/view?usp=sharing
- File config: https://drive.google.com/file/d/1bnxwqyS9Bm-zTJkoDqC-AOxH9skzdi_Y/view?usp=sharing
- File model phân lớp: https://drive.google.com/file/d/1hC2iXeZfcXCpT-SRCGh0FgQhZ47r1Yky/view?usp=sharing
- File label cho mạng yolo: https://drive.google.com/file/d/1bt65Hr4VWr2vZldbquw5uNRZOpgwj9c-/view?usp=sharing

TÀI LIỆU THAM KHẢO

[1] "Keras - From Wikipedia, the free encyclopedia," [Online]. Available: https://en.wikipedia.org/wiki/Keras.

[2] "Module: tf.keras | TensorFlow Core v2.4.1," [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/keras?hl=tr.

[3] Kaggle, "Traffic sign Classification using CNN," [Online]. Available: view-source:https://www.kaggle.com/pritamaich/traffic-sign-classification-using-cnn.

[4] "NN-SVG," [Online]. Available: http://alexlenail.me/NN-SVG/LeNet.html.

[5] S. I. M. K. M. &. T. S. Saha, "An Efficient Traffic Sign Recognition Approach Using a Novel Deep Neural Network Selection Architecture," 2019.

[6] V. H. Tiệp, "Bài 36. Giới thiệu về Keras," machinelearningcoban.com, [Online]. Available: https://machinelearningcoban.com/2018/07/06/deeplearning/.

[7] https://medium0.com/@quangnhatnguyenle/yolov4-in-google-colab-train-your-custom-dataset-traffic-signs-with-ease-3243ca91c81d

[8] https://robocademy.com/2020/05/01/a-gentle-introduction-to-yolo-v4-for-object-detection-in-ubuntu-20-04/

[9] https://www.geeksforgeeks.org/python-opencv-capture-video-from-camera/