BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC
NGÀNH KHOA HỌC MÁY TÍNH

Đề tài

NHẬN DIỆN BIỂN BÁO GIAO THÔNG TRÊN THIẾT BỊ DI ĐỘNG

Sinh viên thực hiện : Huỳnh Bảo Quốc
Mã số : B1400516
Khóa : 40

Cần Thơ, 12/2018

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC
NGÀNH KHOA HỌC MÁY TÍNH

Đề tài

NHẬN DIỆN BIỂN BÁO GIAO THÔNG TRÊN THIẾT BỊ DI ĐỘNG

Giáo viên hướng dẫn:
PGS, TS Phạm Nguyên Khang

Sinh viên thực hiện:
Huỳnh Bảo Quốc
B1400516
Khoá:40

Cần Thơ, 12/2018

NHẬN XÉT CỦA GIẢNG VIÊN

LỜI CẢM ƠN

Đầu tiên em xin gởi lời cảm ơn đến thầy Phạm Nguyên Khang trong suốt khoảng thời gian làm luận văn đã chỉ dẫn về kỹ thuật cho đề tài lẫn cách làm việc, nhờ thầy em đã có hướng đi cho đề tài, cách làm việc tốt hơn. Mặc dù đã cố gắng hết khả năng nhưng không thể nào tránh khỏi những thiếu xót, nhưng nhờ sự thông cảm và dẫn đúng đường đi để em có thể khắc phục kịp thời để hoàn thiện báo cáo của mình, em xin chân thành cảm ơn thầy. Em cũng xin cảm ơn tất cả thầy cô trong bộ môn và khác bộ môn đã giúp em có kiến thức bốn năm ruỡi để có thể đáp ứng cho bài luận văn cuối cùng của một sinh viên. Em xin chân thành cảm ơn tất cả thầy cô, kính chúc sức khoẻ và ngày càng thành công.

Cần Thơ, ngày 12 tháng 12 năm 2018
Người viết
Huỳnh Bảo Quốc

MỤC LỤC

DANH SÁCH BIỂN BÁO

- Cấm ngược chiều

- Cấm dừng đỗ xe

- Giao nhau với đường không ưu tiên

- Giao nhau với vòng xuyến

- Trẻ em, học sinh qua đường

- Đường đi bộ

- Cấm rẽ.

- Người đi bộ cắt ngang

- Khu vực chợ đông người

- Cấm các phương tiện

- Chỗ ngoặc nguy hiểm

- Đường không băng phẳng

- Đường quẹo phải

DANH MỤC HÌNH

ABSTRACT

Today, new advances in technology science and technology have greatly benefited people's lives. Everything is almost automatic and the work performance is enhanced with the help of machines and equipment. One of the advanced technologies widely applied in life is identification technology. Data recognition includes sound recognition and image recognition. The objects of identification problems are very rich, such as facial recognition, voice, handwriting recognition, barcode identification ... Traffic signs are also one of them. This is a type of object with typical geometric characteristics, often encountered in daily life with the utility of giving information warnings to traffic participants. However, the traffic signs are not rules but only a system of symbols with the conventional meaning attached. Requires sign detection process to be accurate and fast to have a high accuracy identification model.

The thesis focuses on studying Cascade method for detection of signs, this method is quite popular for objects affected by environmental factors, partially obscured. Applying the Transfer Learning method to the problem of identifying signs, a common method in Deep Learning, using the world's famous short-lived neural networks to train a convolutional neural network with the goal of delivery signs information

TÓM TẮT

Ngày nay, những tiến bộ mới trong khoa học kỹ thuật công nghệ đã giúp ích rất nhiều cho cuộc sống của con người. Mọi thứ hầu như đều được tự động và hiệu suất công việc được nâng cao hơn với sự trợ giúp của máy móc, thiết bị. Một trong những công nghệ tiên tiến đang áp dụng rộng rãi trong đời sống chính là công nghệ nhận dạng. Nhận dạng dữ liệu bao gồm có nhận dạng âm thanh và nhận dạng hình ảnh. Các đối tượng của bài toán nhận dạng thì rất phong phú, ví dụ như nhận dạng khuôn mặt, tiếng nói, nhận dạng chữ viết tay, nhận dạng mã vạch … Biển báo giao thông cũng là một trong số đó. Đây là kiểu đối tượng có tính chất hình học đặc trưng, thường bắt gặp trong đời sống hằng ngày với công dụng là đưa ra những cảnh báo thông tin cho người tham gia giao thông. Tuy nhiên các biển báo giao thông thì không có quy luật mà chỉ là hệ thống các ký hiệu với ý nghĩa quy ước kèm theo, đòi hỏi quá trình phát hiện biển báo phải chính xác và nhanh để có một mô hình nhận dạng chính xác cao.

Luận văn tập trung nghiên cứu phương pháp Cascade cho việc phát hiện biển báo, phương pháp này khá phổ biến áp dụng cho các đối tượng biển báo bị tác động bởi yếu tố môi trường, bị che khuất một phần. Áp dụng phương pháp Transfer Learning cho bài toán nhận dạng biển báo, một phương pháp phổ biến trong Deep Learning, sử dụng các mạng nơ-ron tích chập đã huấn luyện nỗi tiếng trên thế giới để huấn luyện một mạng nơ-ron tích chập với mục tiêu dự đoán là biển báo giao thông.

PHẦN GIỚI THIỆU

1. Đặt vấn đề

Trên mỗi tuyến đường hiện nay có rất nhiều loại biển báo giao thông với đa dạng các loại biển báo như biển chỉ dẫn, biển báo nguy hiểm, biển báo cấm, biển báo hiệu lệnh với nhiều ý nghĩa khác nhau. Những biển báo này giúp người điều khiển phương tiện giao thông tham gia giao thông một cách an toàn, tránh các tai nạn không mong muốn. Do đó, đòi hỏi mọi người phải có khả năng nắm bắt và hiểu chi tiết về các loại biển báo để đảm bảo an toàn cho bản thân. Tuy nhiên, số lượng biển báo giao thông là cực kỳ lớn và đa dạng, gây khó khăn trong việc ghi nhớ ý nghĩa biển báo đối với người điều khiển phương tiện. Ứng dụng nhận diện biển báo giao thông được phát triển trên di động nhằm giải quyết vấn đề này. Thuận lợi khi phát triển ứng dụng này trên di động là tính dễ sử dụng, nhỏ gọn của điện thoại di động.

2. Lịch sử giải quyết vấn đề

Cũng nhìn nhận được về sự thiết yếu và cấp bách của vấn đề trên, nhiều nghiên cứu ở nhiều quốc gia đã tìm hiểu và giải quyết vấn đề với nhiều cách tiếp cận khác nhau và nhiều kết quả khả thi với hiệu quả cao. Ở vấn đề phát hiện biển báo, có một nghiên cứu giải quyết vấn đề bằng phương pháp YOLO Detection [21] đáp ứng thời gian thực với độ chính xác cao trên phương pháp này. Ở một nghiên cứu khác về vấn đề phát hiện biển báo mà không cần dữ liệu học, chỉ cần phương pháp xử lý ảnh cũng đáp ứng được thời gian thực đó là phương pháp phân đoạn ảnh [22], tuy nhiên phương pháp này nhiễu và độ chính xác với điều kiện môi trường (ánh sánh, che khuất đối tượng,…) khá cao. Song vấn đề phát hiện là vấn đề nhận dạng biển báo được tiếp cận với phương pháp Histogram of oriented gradient và phân loại SVM, phương pháp này đòi hỏi tập dữ liệu phải cần bằng và số chiều đặc trưng khá lớn, ở một nghiên cứu khác cho vấn đề nhận diện biển báo đã tiếp cận Học sâu để giải quyết vấn đề này, cụ thể là Mạng nơron tích chập (CNN) [6] bằng phương pháp Học chuyển giao (Transfer Learning) [4] với độ chính xác và chi phí thấp.

3. Mục tiêu đề tài

Nhận diện biển báo giao thông trên thiết bị di động đáp ứng thời gian thực. Huấn luyện mô hình phát hiện đối tượng bằng phương pháp Cascade Classification cho ra một tập tin xml để thư viện Opencv có thể sử dụng và phát hiện biển báo.Huấn luyện một mô hình nhận dạng sử dụng phương pháp Transfer Learning, huấn luyện lại tầng cuối của các mô hình nổi tiếng như ImageNet, CIFAR100… với mục tiêu của đề tài là biển báo giao thông ở tầng cuối của một mạng nơ-ron tích chập. Sử dụng tensorflow cho việc huấn luyện này bằng ngôn ngữ lập trình Python (phiên bản 3.x)

4. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu: Phương pháp phát hiện đối tượng biển báo bằng phương pháp Cascade Classification, phương pháp Transfer Learning cho việc nhận diện biển và lập trình trên ứng dụng di động để thực hiện việc phát hiện và nhận diện biển báo giao thông.

Phạm vi nghiên cứu: Tham khảo trong học phần thị giác máy tính, các bài báo về phương pháp Cascade Classification, phương pháp Transfer Learning, thư viện Tensorflow, thư viện OpenCV. Sử dụng ngôn ngữ lập trình Python, Java để phục vụ việc huấn luyện và xây dựng ứng dụng di động. [7][8][9][10][11][12]

5. Phương pháp nghiên cứu

Tham khảo giáo trình bài giảng Thị giác máy tính và Xử lý ảnh về các lý thuyết xử lý ảnh, không gian màu, pháp biến đổi trên ảnh, các phương pháp tìm biên, đặc trưng trong ảnh. Đọc hiểu tài liệu về các thư viện như OpenCV, Tensorflow để thực thi việc xử lý ảnh và các phương pháp lấy đặc trưng ảnh và huấn luyện mô hình. Đối với di động tìm hiểu các cộng cụ để xây dựng một ứng dụng di động cụ thể là Nền tảng Android với công cụ Android Studio bằng ngôn ngữ lập trình Java, và sử dụng công cụ PyCharm với ngôn ngữ lập trình Python để huấn luyện mô hình nhận dạng biển báo.

6. Kết quả đạt được

Xây dựng ứng dụng Android nhận diện biển báo giao thông với thời gian thực áp dụng phương pháp Cascade Classification và Transfer Learning. Nắm bắt hiểu rõ nguyên lý hoạt động các phương pháp và so sánh được phương pháp trên với phương pháp phổ biến được áp dụng với đề tài nhiều năm qua là HOG với SVM để thấy được sự cải tiến về độ chính xác và hiệu quả giữa hai cách làm đối với các nền tảng khác nhau đặc biệt là trên thiết bị di động

7. Bố cục luận văn

Phần nội dung

Ở những chương sau sẽ trình bày chi tiết các nội dung sau:

Chương 1: Mô tả lý thuyết về các vấn đề giải quyết bài toán.

Chương 2 :Thiết kế, cài đặt giải thuật.

Chương 3: Kiểm thử hệ thống và đánh giá độ chính xác, tốc độ của hệ thống.

Phần kết luận

Trình bày kết quả đạt được và hướng phát triển hệ thống

PHẦN NỘI DUNG

CHƯƠNG 1
MÔ TẢ BÀI TOÁN

1. Mô tả chi tiết bài toán

Bài toán phát hiện và nhận diện biển báo giao thông là bài toán kết hợp giữa xử lý ảnh và máy học, với mỗi ảnh xử lý là dữ liệu được thu từ máy ảnh có thể quan sát mọi gốc độ. Phương pháp được đề cập để giải quyết cho bài toán bao gồm bài toán phát hiện đối tượng, cụ thể phát hiện biển báo là bài toán quan trọng và được thực hiện đầu tiên để phát hiện ra vùng chỉ chứa biển báo để loại các vùng khác không phải biển báo, cung cấp vùng ảnh có chứa biển báo để bài toán nhận dạng đối tượng có thể sử dụng để nhận dạng và cho ra kết quả với thời gian nhanh chóng và chính xác cao. Xây dựng ứng dụng di động cho phát nhận dạng biển báo với thời gian thực để thông báo cho người điều khiển phương tiện giao thông.

2. Vấn đề và giải pháp liên quan đến bài toán

2.1. Phát hiện biển báo trong ảnh

Phát hiện biển báo là bài toán chính vì khi phát hiện vùng chứa đối tượng để giảm chiều đặc trưng, loại bỏ các nền. Nhưng phải chọn giải pháp để xử lý khi ảnh được nhận với tốc độ liên tục từ máy ảnh với nhiều yếu tố môi trường khác nhau như: ánh sáng, bị che khuất, biển báo không nguyên vẹn,… đòi hỏi phải xử lý phát hiện chính xác với tốc độ đáp ứng vấn đề thời gian thực trên di động.

Giải pháp cho vấn đề này thích hợp và hiệu quả bằng việc sử dụng phương pháp Cascade Classification. Ở phương pháp này khi cung cấp dữ liệu được xử lý càng nhiều thì độ chính xác càng cao, có thể phát hiện các biển báo bị che khuất, biển báo không nguyên vẹn và điều kiện ánh cao làm mờ biển báo.

2.2. Nhận dạng biển báo

Máy ảnh thu nhận ảnh với tốc độ liên tục, trên mỗi khung ảnh đòi hỏi phải nhận dạng chính xác đối tượng đã được phát hiện để có thể đáp ứng vấn đề về thời gian thực trên di động.

Giải pháp: xây dựng mô hình nhận dạng biển báo giao thông bằng phương pháp Transfer Learning, Transfer learning giống như mạng nơ-rôn, lấy cảm hứng từ việc con người có thể tổng hợp kiến thức từ các nguồn khác nhau để giải quyết vấn đề chưa được

gặp. Mục tiêu của Transfer Learning là huấn luyện một mô hình (target task) mới dựa trên những đặc trưng trích xuất được từ mô hình cũ (source domain). InceptionV3 được huấn luyện trên dữ liệu ImageNet. Tập dữ liệu lớn của ImageNet có nét tương đồng về biển báo, giao thông, đường bộ,.. với mô hình của chúng ta nên việc áp dụng Transfer Learning sẽ đạt được hiệu quả cao. Hơn nữa, Transfer Learning cũng đã đạt được những kết quả vô cùng khả quan trong thời gian gần đây [2][3]

2.3. Xây dựng ứng dụng Android chạy thử mô hình

Trong lập trình di động, cụ thể là hệ điều hành Android, việc sử dụng các mô hình được huấn luyện ở một nền tảng khác như Windows, Linux,.. là một việc khó khăn vì khi huấn luyện, mô hình được lưu lại với một định dạng khác mà Android không thể tích hợp được. Cần có một định dạng chung cho việc sử dụng mô hình và dự đoán đối tượng.

Giải pháp: với phương pháp phát hiện đối tượng thư viện OpenCV có thể sử dụng tập tin đã huấn luyện với đinh dạng xml, ta có thể sử dụng hàm detectMultiScale() của thư viện OpenCV để thực hiện việc phát hiện đối tượng. Một giải pháp cho vấn đề nhận diện biển báo bằng việc sử dụng thư viện TensorFlow cho việc huấn luyện lại một mạng nơron tích chập bằng phương pháp Transfer Learning như **Chương 1, phần 2.2 Nhận diện biển báo** có đề cập tới trên nền tảng Windows, việc huấn luyện này sẽ cho ra một mô hình có định dạng mà Android sử dụng cùng thư viện TensorFlow có thể sử dụng được với định dạng **protobuffer (**filename.pb), sử dụng hàm dự đoán trong thư viện tensorflow để dự đoán biển báo với mô hình trên.

2.4. Thời gian thực/bộ nhớ hạn hẹp trên di động

Phát hiện và nhận dạng là hai công việc làm tăng độ trễ của ứng dụng vì hai công việc thực hiện xử lý từng khung ảnh và liên tục.

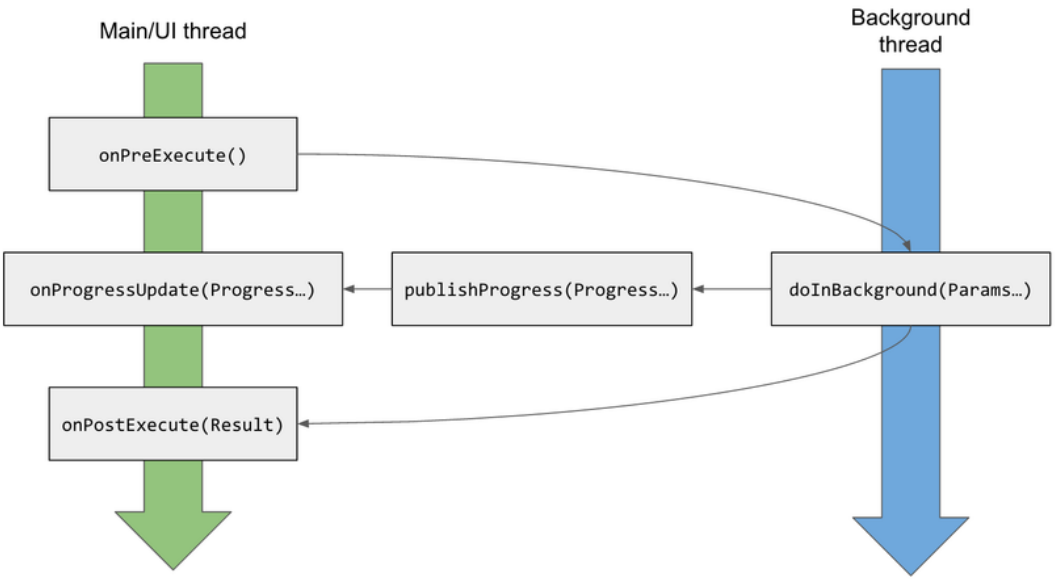
Để tối ưu vấn đề này ở phương pháp phát hiện biển báo sử dụng Cascade Classification khi huấn luyện tăng giá trị kích thước của sổ dò tìm biển báo (width, height), giảm kích thước từng khung ảnh để việc dò tìm đối tượng nhanh hơn, điều chỉnh giá trị các đối số trong hàm detectMultiScale() như:

- scaleFactor: dao động từ 1.05 đến 5

- neighbor: từ 3 đến 6.

Sử dụng tensorflow để tối ưu hoá mô hình nhận dạng trước khi sử dụng trên di động cho việc dự án biển báo. Áp dụng kỹ thuật lập trình đa luồng cho việc chia luồng thực hiện hai công việc phát hiện và nhận diện biển báo và giao tiếp giữa các luồng bằng lớp **AsyncTask** trong Android, lớp này được tích hợp sẵn các phương thức để thực hiện cho

việc chạy nền (Hình 1).Thực hiện bài toán phát hiện và nhận dạng song song bằng cách chạy nền trong BackgroundThread (Hình 1) và hiển thị kết quả ở MainThread (Hình 1).



Hình 1: Mô hình hoạt động đa luồng với AsyncTask trong Android

3. Mô tả giải pháp cho bài toán

Để xây dựng một ứng dụng trên thiết bị di động cần xác định hệ điều hành để thiết bị đó vận hành. Hiện nay, trên thị trường có rất nhiều hệ điều hành giành cho thiết bị di động như Android, IOS, Windows Phone… Android là một hệ điều hành thông dụng, phổ biến, được sử dụng cho rất nhiều thiết bị di động hiện nay như Smartphone, máy tính bảng. Bên cạnh đó, Android lại là hệ điều hành có mã nguồn mở nên được nhiều người phát triển vì vậy hệ điều hành được chọn để giải quyết bài toán trên là Android. Cùng với đó phần mềm Android Studio cũng được chọn là IDE để phát triển ứng dụng và thư viện OpenCV được sử dụng để giải quyết vấn đề nhận diện biển báo, Thư viện Tensorflow cho việc huấn luyện lại một mạng nơ-ron tích chập và sử dụng mô hình huấn luyện để nhận dạng trên Android.

3.1. Thư viện OpenCV và Tensorflow

OpenCV là một thư viện mã nguồn mở hàng đầu cho thị giác máy tính (computer vision), xử lý ảnh và máy học, và các tính năng tăng tốc GPU trong hoạt động thời gian thực. OpenCV được phát hành theo giấy phép BSD, do đó nó hoàn toàn miễn phí cho tất cả học thuật và thương mại. Nó có các interface C++, C, Python, Java và hỗ trợ Windows, Linux, Mac OS, iOS, và Android. OpenCV được thiết kế để tính toán hiệu quả với sự tập trung nhiều vào các ứng dụng thời gian thực. Được viết bằng tối ưu hóa C/C++, thư viện

có thể tận dụng lợi thế của xử lý đa lõi. Được sử dụng trên khắp thế giới, OpenCV có cộng đồng hơn 47 nghìn người dùng và số lượt tải vượt quá 6 triệu lần. Phạm vi sử dụng từ nghệ thuật tương tác, cho đến lĩnh vực khai thác mỏ, bản đồ trên web hoặc công nghệ robot.

Để sử dụng được openCV trên project Android ta cần import thư viện openCV vào project. Trước tiên ta cần chuẩn bị OpenCV sdk cho nền tảng android – được tải từ *opencv.org* và giải nén. Sau đó ta import openCV vào project theo các bước sau:

Bước 1: *File -> New-> Import module*. Chọn đường dẫn đến thư mục *sdk/java* trong thư mục ta vừa giải nén ở bước chuẩn bị.

Bước 2: Cập nhật lại file ***build.gradle*** (của module openCVLibrary). 4 trường compileSdkVersion, buildToolsVersion, minSdkVersion và targetSdkVersion sao cho trùng với ***build.gradle*** (của project).

Bước 3: *File -> Project Structure,* trong hộp thoại Project Structure chọn vào mục app ở phần Modules. Tiếp theo chọn tab Dependencies ở phía bên phải. Click vào dấu "+" màu xanh ở góc phải. Chọn *Module dependency* sau đó chọn OpenCV module trong hộp thoại *Choose Modules* vừa hiện ra.

Bước 4: Tạo thư mục *jniLibs* trong project ở đường dẫn app/src/main. Sau đó copy tất cả các thư mục trong mục sdk/native/libs của thư mục ta giải nén ở bước chuẩn bị vạo thư mục vừa tạo.

Bước 5: Ta sử dụng OpenCVLoader.initDebug() để khởi tạo code Java trước khi gọi Api OpenCV.

TensorFlow là một thư viện phần mềm mã nguồn mở dành cho máy học trong nhiều loại hình tác vụ nhận thức và hiểu ngôn ngữ. Nó hiện đang được sử dụng cho cả nghiên cứu lẫn sản xuất bởi đội khác nhau trong hàng tá sản phẩm thương mại của Google, như nhận dạng giọng nói, Gmail, Google Photos, và tìm kiếm, nhiều trong số đó đã từng sử dụng chương trình tiền nhiệm DistBelief của nó. TensorFlow nguyên thủy được phát triển bởi đội Google Brain cho mục đích nghiên cứu và sản xuất của Google và sau đó được phát hành theo giấy phép mã nguồn mở Apache 2.0 vào ngày 9/11/2015.

Để sử dụng TensorFlow cho việc huấn luyện mô hình trên Python ta cần cài đặt bằng cách mở **command-line** và nhập dòng lệnh **pip install tensorflow** để cài đặt, để cài đặt Tensorflow trên Android Studio cho việc sử dụng mô hình huấn luyện và dự đoán kết quả của biển báo:

Bước 1: Mở tập tin **build.gradle** trong **Android Studio**

Bước 2: Di chuyển đến dòng mã lệnh **dependencies{}** trong tập tin **build.gradle**

Bước 3: Nhập dòng lệnh **implementation 'org.tensorflow:tensorflow-android:1.4.0'** trong thân hàm của **dependencies{}** (ở bước 2)

Bước 4: Bấm vào nút **SyncGradle** để Android Studio tiến hành đồng bộ và tải thư viện Tensorflow phiên bản 1.4.0

3.2. Giải thuật Haar-like và AdaBoost

Haar-like và AdaBoost được xây dựng dựa trên sự kết hợp 4 phương pháp:

- Các đặc trưng Haar-like được đặt vào các vùng ảnh để tính toán các giá trị đặt trưng, từ giá trị đó đưa vào bộ phân loại AdaBoost xem có phải là biển báo hay không.

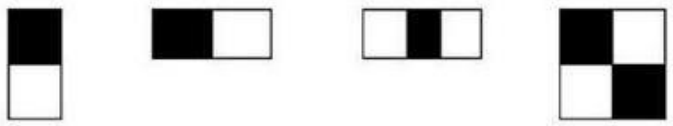
- Ảnh tích hợp (Integral Image) đây là công cụ giúp cho Haar-like tính toán nhanh hơn.

- AdaBoost sử dụng giá trị của Haar-like xem có phải là biển báo hay không.

- Cascade of Calassifiers là bộ phân loại tầng với mỗi tầng là AdaBoost làm tăng tốc bộ phân loại.

Các đặc trưng Haar-like

Haar-like có 4 đặc trưng cơ bản. Mỗi đặc trưng Haar-like là sự kết hợp của hai hay ba hình chữ nhật "trắng" hay "đen" như trong hình sau:

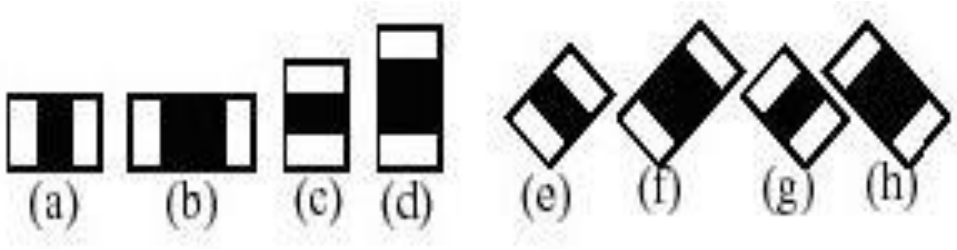


Hình 2: Các đặc trưng cơ bản

Để áp dụng các đặc trưng này vào bài toán xác định biển báo, 4 đặc trưng cơ bản được mở rộng ra, và được chia làm 3 tập đặc trưng như sau:

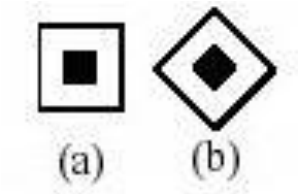
Hình 3: Đặc trưng cạnh

- Đặc trưng đường (line features)



Hình 4: Đặc trưng đường

- Đặc trưng xung quanh tâm (center-surround features)



Hình 5: Đặc trưng xung quanh tâm

Đặc trưng Haar-like có thể diễn đạt được tri thức về các đối tượng trong ảnh (bởi vì nó biểu diễn mối liên hệ giữa các bộ phận của đối tượng), điều mà bản thân từng điểm ảnh không diễn đạt được. Để tính giá trị các đặc trưng Haar-like, ta tính sự chênh lệch giữa tổng của các pixel của các vùng đen và các vùng trắng như trong công thức sau:

$f(x) = \text{Tổng}_{\text{vùng đen}}(\text{các mức xám của pixel}) - \text{Tổng}_{\text{vùng trắng}}(\text{các mức xảm của pixel})$

Ảnh tích hợp (Integral Image)

Để tính các giá trị đặc trưng Haar-like, ta phải tính tổng của các vùng pixel trên ảnh mất rất nhiều thời gian. Do đó, cần tạo Integral Image. Integral Image là một mảng 2 chiều với kích thước bằng với kích của ảnh cần tính các đặc trưng Haar-like, với mỗi phần tử của mảng này được tính bằng cách tính tổng của điểm ảnh phía trên (dòng 1) và bên trái (cột 1) của nó. Bắt đầu từ vị trí trên bên trái đến vị trí dưới, bên phải của ảnh, việc tính toán này chỉ dựa trên phép cộng số nguyên đơn giản, nên tốc độ thực hiện rất nhanh.

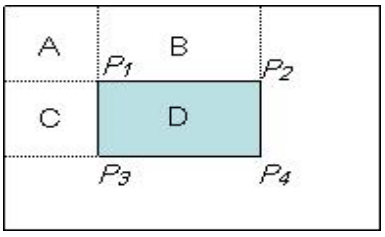
Sau khi đã tính được Integral Image, việc tính tổng các giá trị mức xám của một vùng bất kỳ nào đó trên ảnh thực hiện rất đơn giản theo cách sau:

Giả sử ta cần tính tổng các giá trị mức xám của vùng D như trong hình 9, ta có thể tính như sau:

$$D = A + B + C + D – (A+B) – (A+C) + A$$

Với A + B + C + D chính là giá trị tại điểm P4 trên Integral Image, tương tự như vậy A+B là giá trị tại điểm P2, A+C là giá trị tại điểm P3, và A là giá trị tại điểm P1. Vậy ta có thể viết lại biểu thức tính D ở trên như sau:

$$D = \underbrace{(x4,y4)}_{A+B+C+D} - \underbrace{(x2,y2)}_{(A+B)} - \underbrace{(x3,y3)}_{(A+C)} + \underbrace{(x1,y1)}_{A}$$



Hình 6: Ví dụ cách tính nhanh các giá trị mức xám của vùng D trên ảnh

Tiếp theo, để chọn các đặc trưng Haar-like dùng cho việc thiết lập ngưỡng, Viola và Jones sử dụng một phương pháp máy học được gọi là AdaBoost. AdaBoost sẽ kết hợp các bộ phân loại yếu để tạo thành một bộ phân loại mạnh. Với bộ phân loại yếu chỉ cho ra câu trả lời chính xác chỉ hơn viện đoán một cách ngẫu nhiên một chút, còn bộ phân loại mạnh có thể đưa ra câu trả lời chính xác trên 60%.

Thuật toán AdaBoost:

Với hk(x) là bộ phân loại yếu, được biểu diễn như sau:

$$h_k(x) = \begin{cases} 1 \; nếu \; p_k f_k(x) < p_k \theta_k \\ 0 \qquad nếu \; ngược \; lại \end{cases}$$

Trong đó:

x: mẫu hay cửa sổ con cần xét (X = (x1, x2, …, xn) là vector đặc trưng của mẫu).

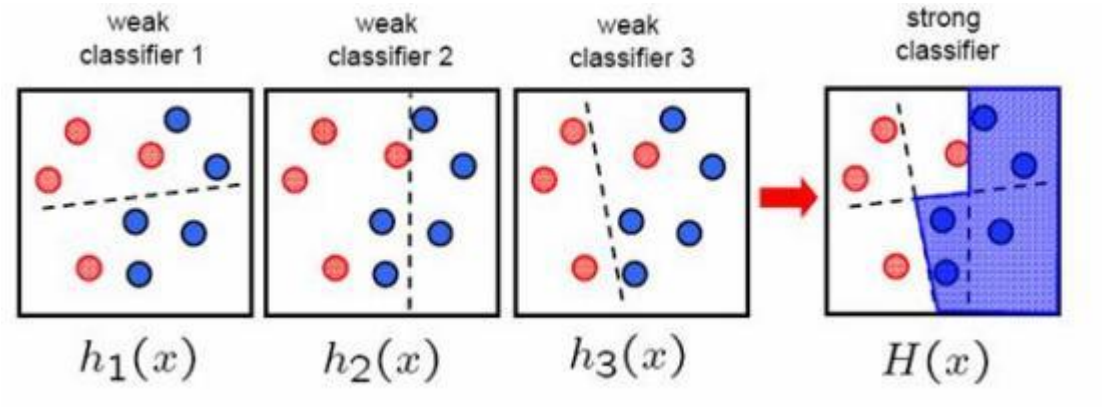
θ_k : ngưỡng.

fk: giá trị của đặc trưng Haar-like.

pk: hệ số quyết định chiều của bất phương trình

Công thức trên có thể diễn giải như sau nếu giá trị đặc trưng của mẫu cho bởi hàm đánh giá của bộ phân loại vượt qua một ngưỡng cho trước thì mẫu đấy là biển báo (gọi là object: đối tượng cần định vị), ngược lại thì mẫu là hình nền (không phải là đối tượng).

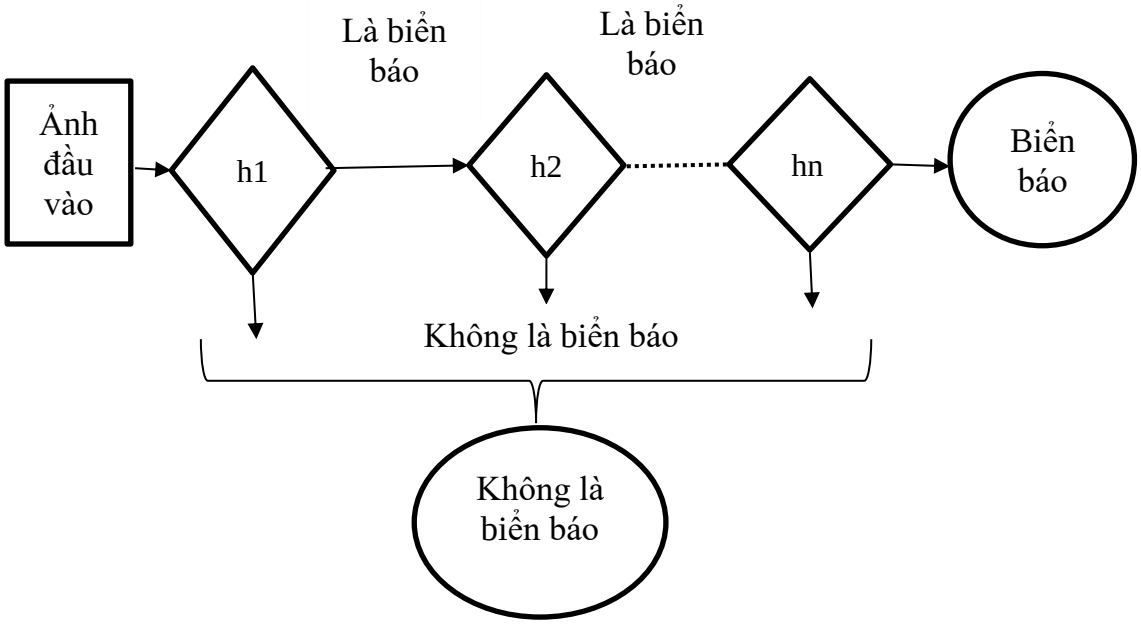
Trong thuật toán AdaBoost bộ phân loại mạnh được tạo ra từ tập hợp các bộ phân loại yếu.



Hình 7: Thuật toán AdaBoost.

Mô hình phân tầng Cascade of Classifiers

Mô hình phân tầng kết hợp các bộ phân loại yếu để xác định biển báo với đặc trưng đi vào từ h1, h2…hn. Nếu đặc trưng vào h1 là biển báo thì tiếp tục vào h2, ngược lại nếu không phải biển báo thì hiển thị kết quả không phải biển báo.



Hình 8: Mô hình phân tần kết hợp các bộ phân loại yếu để xác định biển báo

3.3. Mạng nơron tích chập và phương pháp Transfer Learning

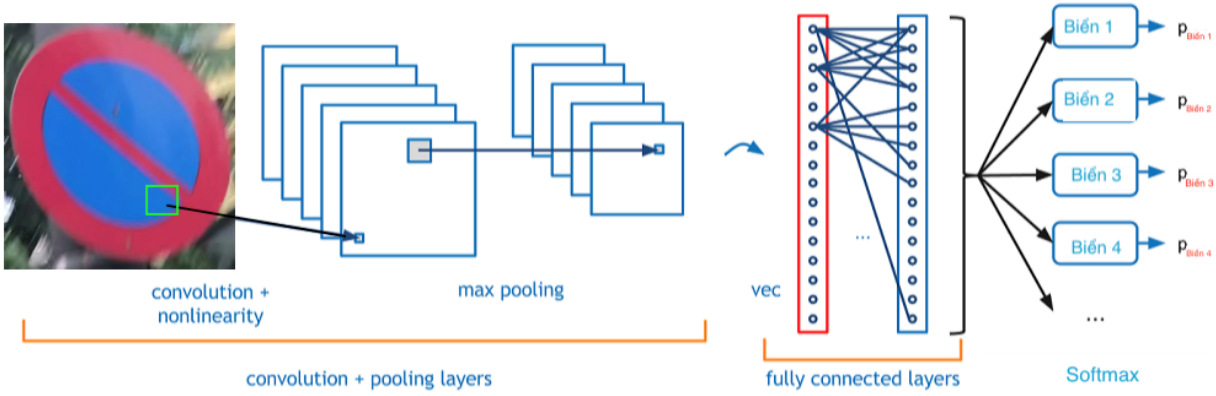
Mạng nơ rơn tích chập (gọi tắt là CNN) là mô hình do Yan LeCunn đề xuất, để giải quyết cho việc huấn luyện với dữ liệu là hình ảnh một cách hiệu quả.

Cấu trúc của CNN gồm 3 thành phần chính:

- Lớp tích chập (Convolution)

- Lớp giảm chiều (Pooling)

- Lớp kết nối đầy đủ (Fully Connected)



Hình 9. Mô hình mạng nơ rơn tích chập

Lớp tích chập là thành phần chính giúp CNN xử lí dữ liệu hình ảnh hiệu quả hơn so với các mạng nơ rơn (Neural Network). Thay vì sử dụng các lớp của mạng nơ rơn truyền thống, CNN sử dụng bộ lọc (filter) để xử lí và trả kết quả về cho lớp tiếp theo. Việc sử dụng các bộ lọc như vậy sẽ làm giảm đi thông tin cần lưu trữ của mô hình cũng như phù hợp với tính chất tự nhiên của hình ảnh là thông tin trên hình ảnh thường mang tính chất cục bộ, nên việc giữ lại một phần của hình ảnh trên một phạm vi nhỏ sẽ hiệu quả hơn là dùng lớp kết nối đầy đủ như các mạng nơ rơn truyền thống, ví dụ như những pixel nằm ở hai biên của một bức hình sẽ không mang nhiều thông tin bằng những pixel nằm gần nhau (*Local connectivity*). Hơn nữa, bằng cách sử dụng bộ lọc, chúng ta có thể đạt được tính chất *shift invariance* bằng cách sử dụng cùng một bộ lọc cho cùng một đặc trưng, từ đó giảm được khá nhiều lượng trọng số của mô hình cần lưu trữ.

Vị trí của các đặc trưng không còn quan trọng khi đã được trích xuất, dữ liệu chúng ta cần giữ lại là vị trí tương đối của các đặc trưng với nhau. Lớp giảm chiều (Pooling) được sử tính toán bằng cách lấy giá trị lớn nhất hoặc giá trị trung bình trong một cửa số nhỏ, nhằm giảm đi sự ảnh hưởng của vị trí các đặc trưng của ảnh, đồng thời làm giảm đi số lượng trọng số phải lưu, hạn chế học tủ (overfitting). Bộ lọc dành cho lớp giảm chiều thường dùng là khoảng 2x2 để không làm mất đi quá nhiều dữ liệu.

Lớp kết nối đầy đủ là lớp cuối của CNN, lớp này giống như các lớp trong các mạng nơ rơn thông thường. Đối với mô hình của chúng ta, lớp cuối cùng sẽ là lớp kết nối đầy đủ này, gồm 13 (13 lớp cho 13 loại biển báo) nút và hàm kích hoạt là *Softmax*.

Các mô hình được huấn luyện trước cần chọn là những mô hình có độ chính xác cao và việc cấu hình để sử dụng những mô hình không tốn quá nhiều thời gian. Đề tài này sẽ chọn trong các mô hình chiến thắng tại cuộc thi ImageNet. Inception của Google (vô địch 2014) và ResNet của Microsoft (vô địch 2015). Ở đây chúng ta sẽ sử dụng InceptionV3 của Google.

Tập dữ liệu của đề tài bao gồm 4022 ảnh tương chia đều 13 lớp (13 loại biển báo) được gán nhãn, mỗi lớp giao động từ 250 – 300 ảnh, Với lượng dữ liệu như vậy thì khá khiêm tốn nếu chúng ta xây dựng mô hình CNN từ đầu. Nhưng ở đây chúng ta sẽ áp dụng Transfer Learning để huấn luyện mô hình với dữ liệu không nhiều nhưng độ chính xác vẫn chấp nhận được.

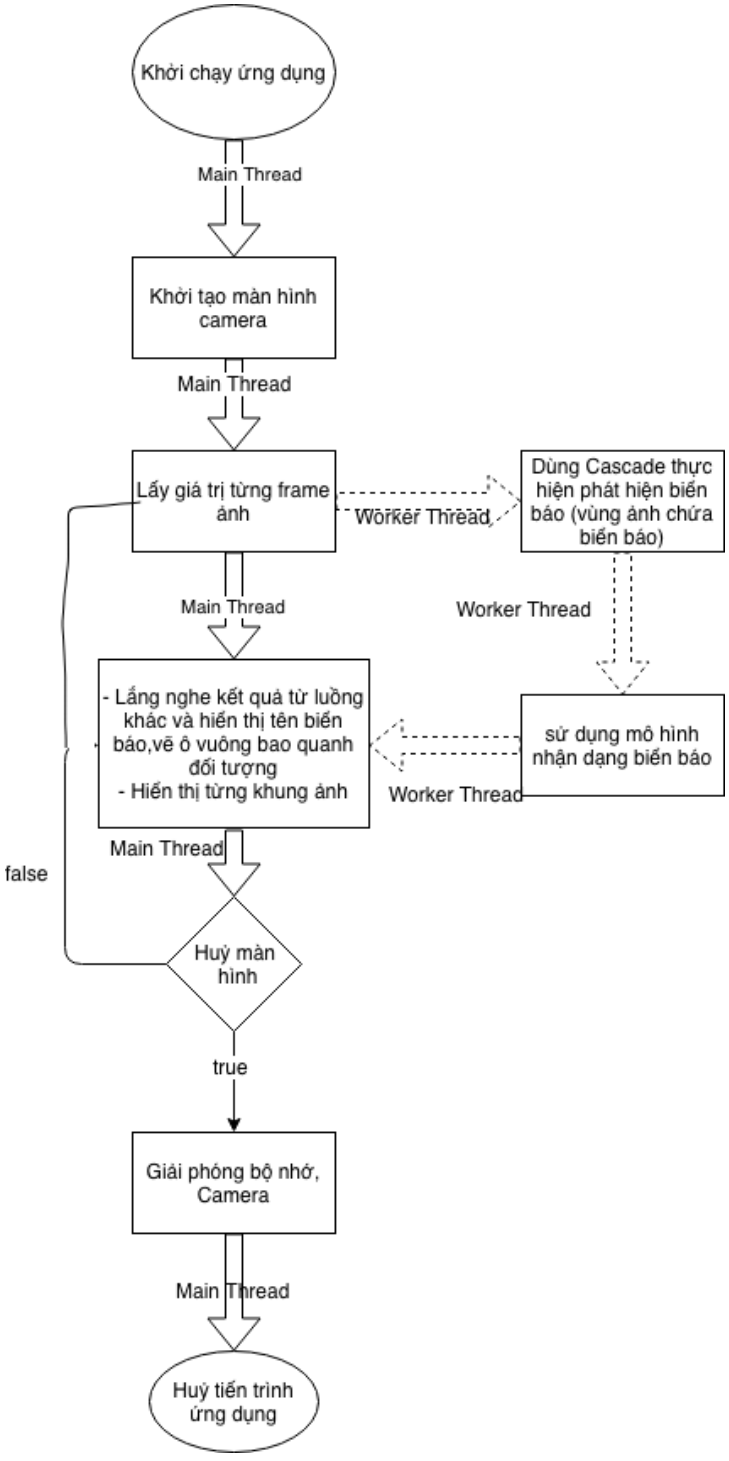
Transfer Learning: giống như mạng nơ-rôn, lấy cảm hứng từ việc con người có thể tổng hợp kiến thức từ các nguồn khác nhau để giải quyết vấn đề chưa được gặp. Mục tiêu của Transfer Learning là huấn luyện một mô hình (target task) mới dựa trên những đặc trưng trích xuất được từ mô hình cũ (source domain). InceptionV3 được huấn luyện trên dữ liệu ImageNet, với tổng số hình về hoa là 1924 tấm, nên ta có thể xem rằng không gian đặc trưng của InceptionV3 khá tương đồng với mô hình chúng ta sẽ xây dựng, việc áp dụng Transfer Learning sẽ đạt được hiệu quả cao. Hơn nữa, Transfer Learning cũng đã đạt được những kết quả vô cùng khả quan trong thời gian gần đây[2][3][4].

Phương pháp áp dụng đầu tiên là thêm một lớp *Softmax* vào cuối cùng của InceptionV3. Lớp cuối *Softmax* cuối gồm 13 nút tương đương với 13 lớp cần phân loại. Cách thay đổi này cũng khá là hiển nhiên do chúng ta cần đầu ra của mô hình là một phân phối xác suất của 13 thành phần.

Lý do chúng ta chỉ huấn luyện khối Inception cuối hơn là chỉnh sửa các lớp trên hay toàn bộ mô hình là do dữ liệu ban đầu của chúng ta hạn chế. Những đặc trưng được học bởi những khối Inception đầu sẽ mang tính chất cơ bản hơn không cụ thể như những khối cuối nên việc giữ nguyên các khối đầu và thay đổi những lớp cuối là hợp lí.

CHƯƠNG 2
THIẾT KẾ VÀ CÀI ĐẶT

1. Thiết kế hệ thống



Hình 10: Luồng hoạt động của hệ thống.

Mô tả chi tiết hệ thống (hình 10)**:**

Khi ứng dụng khởi chạy ứng dụng sẽ chạy một luồng mặc định đó là **Main Thread (UI Thread)** . Màn hình chính của ứng dụng sẽ vào trạng thái **onCreate()** (tham khảo hình 11), trong hàm này tiến hành khởi tạo Camera, Cấp quyền truy cập camera, đọc các mô hình đã huấn luyện cho việc sử dụng sau này.

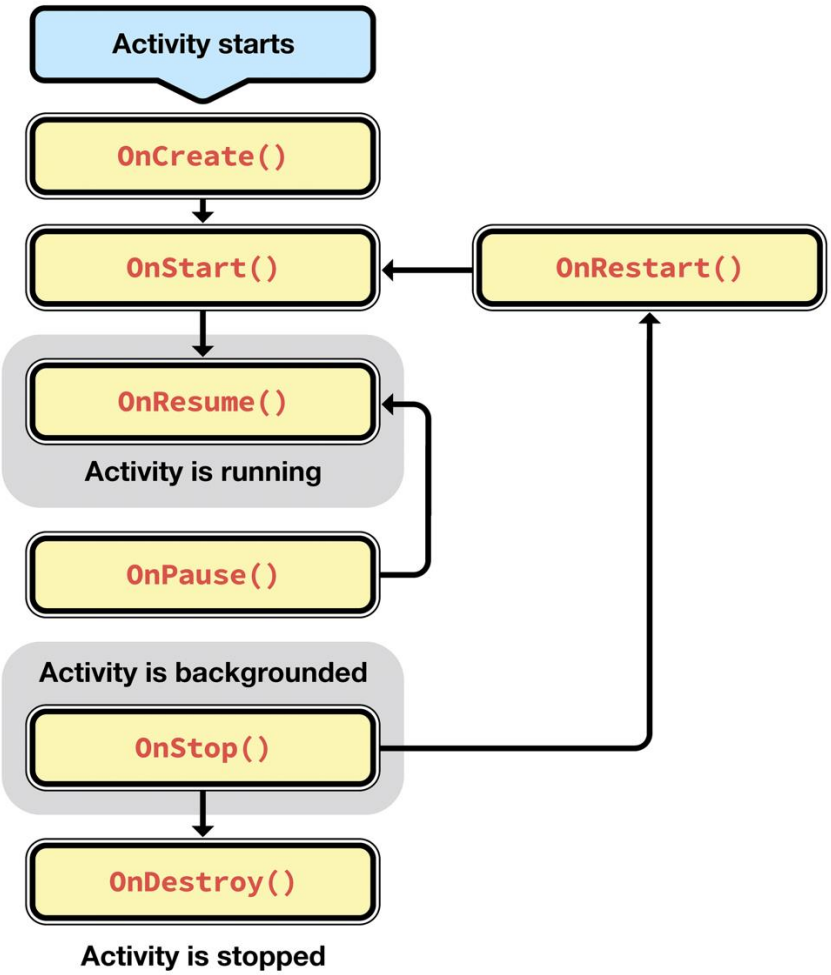
Từ Camera lấy giá trị từng khung ảnh để xử lý phát hiện và nhận dạng, xử lý đơn vị ảnh từ camera như kích cở, chuyển đổi Mat sang Bitmap…

Sử dụng mô hình phát hiện và nhận dạng trên ảnh đầu ra ở bước trước. Do quá trình phát hiện, nhận dạng biển báo sẽ trì hoãn một khoảng thời gian cho nên tiến hành khởi tạo một luồng mới để làm việc này (**Background Thread** hay còn gọi **Worker Thread**).

Ở Main Thread: Vẽ từng khung ảnh ra màn hình (FPS, kết quả nhận dạng, vùng chứa đối tượng, vùng chứa đối tượng khi cắt ra để huấn luyện…)

Ở Background Thread: Sử dụng tập tin xml và hàm detechMultiscale của OpenCV để phát hiện đối tượng nằm ở đâu trong khung ảnh. Thiết lập các đối số cho hàm này như: **scale** (1.05~2.0 càng nhỏ để tìm nhiều và không bỏ soát đối tượng, càng lớn khử phát hiện sai lầm), **neighbor** (3~6), minSize (kích thước nhỏ nhất mà window size của phương pháp cascade classsifier sẽ đi tìm, nhỏ hơn sẽ làm chậm). Sau khi có kết quả là vùng chứa biển báo tiến hành sử dụng mô hình nhận dạng biển báo bằng phương pháp transfer learning để ra kết quả mong muốn, song đó gởi dữ liệu từ **BackgroundThread** đến **Main Thread** như: ảnh đã phát hiện, nhãn của biển báo để **Main Thread** cập nhật giao diện. Ở các bước trên màn hình đang ở trạng thái **running**. Cho đế khi màn hình vào trạng thái **onDestroy()**(tham khảo hình 11), trong hàm này ta tiến hành giải phóng bộ nhớ Camera và các tham số để tránh tràn bộ nhớ.

Một màn hình trong lập trình android nó sẽ có một vòng đời tương ứng với mỗi trạng thái **onCreate()**, **onRestart()**, **onResume()**, **onStop()**, **onDestroy()**, **onPause()**. Tương ứng mỗi trạng thái mà xử lý sự kiện.



Hình 11: Vòng đời của một màn hình trong Android

2. Thiết kế và cài đặt giải thuật

Quy trình phát hiện và nhận dạng biển báo giao thông:

Bước 1: Tiền xử lý dữ liệu

Tiền xử lý dữ liệu bằng cách tách từng khung ảnh trong video, tách với tỉ lệ 5 khung trên giây để tránh các đối tượng tương đương liên tục giống nhau sẽ làm mất ý nghĩa dữ liệu. Sử dụng python và thư viện OpenCV để viết chương trình tách khung ảnh và cắt đối tượng để tạo ra tập dữ liệu dương(ảnh chỉ chứa biển báo), tập dữ liệu âm (ảnh nền không chứa biển báo, tốt nhất là ảnh được cắt ra từ dữ liệu ảnh thô) cho việc huấn luyện Cascade Classification.

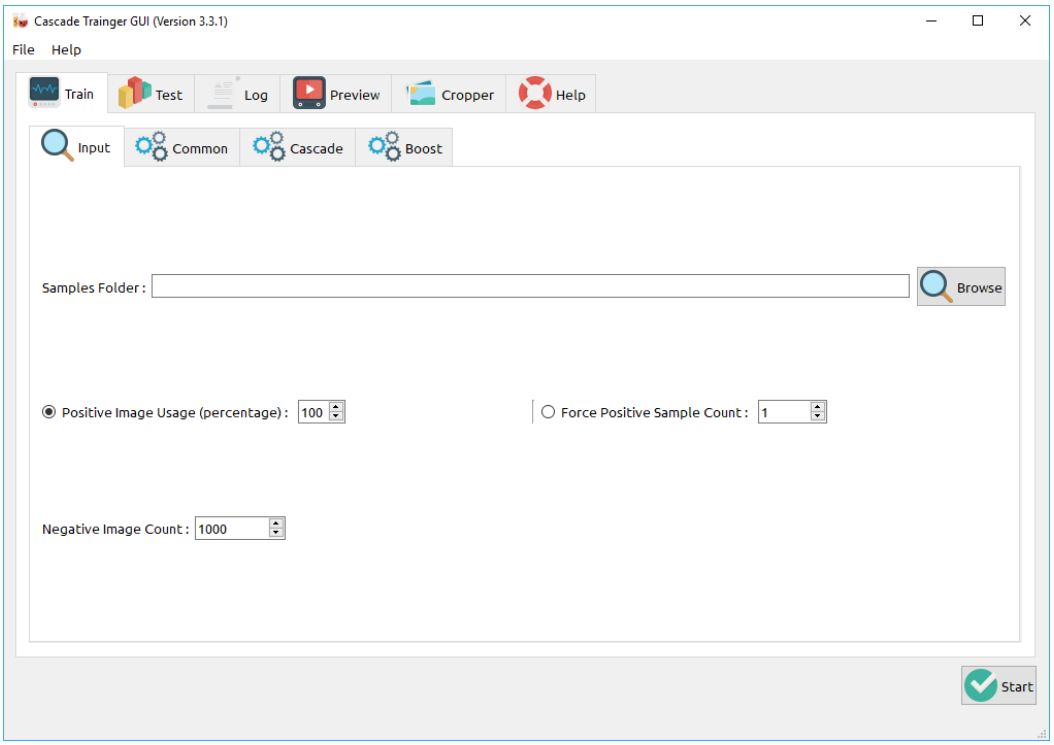
Đường dẫn đến chương trình tiền xử lý dữ liệu cho huấn luyện Cascade: https://github.com/huynhquoc09x11x1996/LuanVanKHMT

Chương trình có thể chạy trên mọi nền tảng, được viết bằng ngôn ngữ python, để có thể biết cách sử dụng ta có thể truy cập vào đường dẫn bên trên.

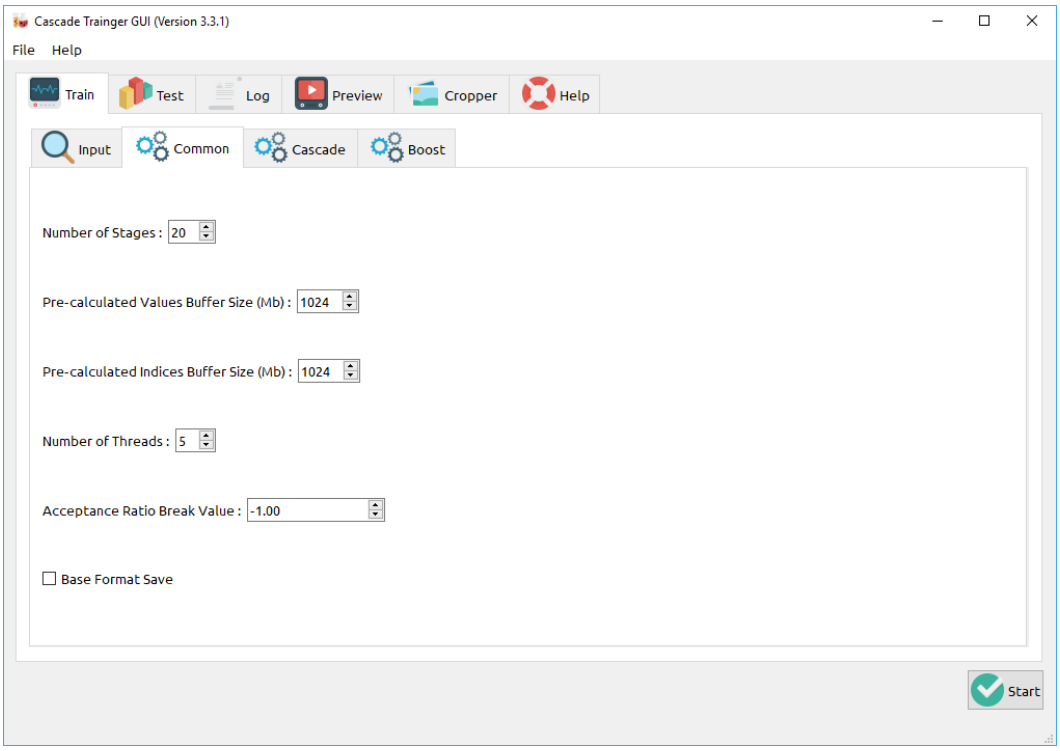
Bước 2: Huấn luyện mô hình

Huấn luyện mô hình phát hiện biển báo

Sau khi có tập dữ liệu dương và tập dữ liệu âm tiến hành huấn luyện Cascade cho việc phát hiện biển báo. Sử dụng công cụ để huấn luyện Cascade **CASCADE TRAINING GUI** [8]. Công cụ có giao diện trực quan dễ sử dụng, đầy đủ các tính năng huấn luyện và kiểm thử mô hình. Thiết lập các thông số cho quá trình huấn luyện như: số lượng ảnh dương trong tập ảnh dương**(Positive image usage)** là 2080 ảnh, số lượng ảnh âm trong tập ảnh âm**(Negative image count)** là 4066 ảnh**,** số lượng giai đoạn huấn luyện **(Stage)** là 20**,** chiều rộng và chiều cao của một cửa sổ dò **(wihth,height)** với kích cở 24x24**,** tỉ lệ false alarm **(maxFalseAlarm)** là 0.45**.** Tiếp theo là bắt đầu quá trình huấn luyện để cho ra một tập tin với định dạng XML.



*Hình 12: Giao diện [input] công cụ **CASCADE TRAINING GUI***



*Hình 13: Giao diện [common] công cụ **CASCADE TRAINING GUI***

Huấn luyện mô hình nhận dạng biển báo

Sử dụng lại tập dữ liệu dương tiến hành chuẩn bị gom nhóm từng các ảnh vào từng thư mục, tên thư mục là nhãn của lớp biển báo **(Hình 14)**, tạo một tập tin liện kê các nhãn của các lớp **(Hình 15)**. Sau đó tiến hành huấn luyện bằng phương pháp Transfer Learning với thư viện TensorFlow. Tiến hành thiết lập các đối số cho quá trình huấn luyện bằng việc mở **CommandLine** và nhập dòng lệnh như sau để cho ra một tập tin định dạng **protobuffer** (một định dạng chung của thư viện TensorFlow để sử dụng trên Android):

```
python tensorflow/examples/image_retraining/retrain.py
  --how_many_training_steps = số lần lặp
  --output_graph= đường dẫn chứa tập tin mô hình sau khi huấn luyện
    xong
  --output_labels= đường dẫn tới tập tin liệt kê các lớp
  --image_dir = đường dẫn tới thư mục các lớp
```

Hình 14: Ví dụ gom nhóm thư mục

Hình 15: Ví dụ liệt kê các lớp trong tập tin

Bước 3: Phát hiện và Nhận dạng biển báo

Di chuyển các tập tin mô hình vào thư mục **Assets** trong **Android Studio**, sử dụng hàm **detectMultiScale()** [10] trong thư viện OpenCV để tiến hành phát hiện biển báo của từng khung ảnh mà Android thu nhận từ máy ảnh với các đối số như **scaleFactor** (giá trị

từ 1.05 – 2 kiểu dữ liệu float), **neighBor**(giá trị từ 3- 6 kiểu dữ liệu Integer)**, minSize , maxSize.** Sau khi phát hiện biển báo kết quả trả về là một đối tượng có 4 giá trị vị trí góc trái trên của đối tượng **(pixel tại x,y)** và chiều dài chiều rộng của đối tượng **(width, height).** Tiến hành sử dụng thư viện TensorFlow để nhận dạng vùng ảnh chứa biển báo vừa cắt bởi 4 giá trị trên bằng các sử dụng lớp **TensorFlowInferenceInterface**[11][18] đầu vào lớp này là đường dẫn tới tập tin mô hình trong **Assets.** Sau quá trình trên kết quả sẽ trả ra là nhãn của biển báo của từng khung ảnh với nhãn đã được liệt kê trong tập tin mô hình **(Hình 15).**

3. Giao diện hệ thống

Hình 16. Logo ứng dụng.

Hình 17. Giao diện ứng dụng với màn hình nhận dạng biển báo trực tiếp từ camera

- Góc trái trên màn hình là hiển thị FPS của camera.
- Góc bên phải:
  - Hình trên: hiển thị vùng chứa đối tượng đã được cắt ra.
  - Hình dưới: hiển thị vùng chứa đối tượng sẽ nhận dạng.
- Bên dưới cùng là một dòng chữ hiển thị tên biển báo.
- Bề mặt camera sẽ hiển thị lại từng khung và vẽ một ô vuông quanh đối tượng

CHƯƠNG 3
KIỂM THỬ VÀ ĐÁNH GIÁ

1. Mục tiêu

- Xây dựng ứng dụng nhận diện biển báo giao thông trên nền tảng Android
- Phát hiện và nhận dạng biển báo trong ảnh với độ chính xác cao

2. Nghi thức kiểm tra

Đối với phương pháp phát hiện biển báo bằng Haar-Cascade ta tiến hành sử dụng lại tập tin xml sau khi huấn luyện trên **2000** ảnh dương và **4000** ảnh âm với **20** giai đoạn mỗi giai đoạn ngưỡng maxFalseAlarm là **0.45** và chiều rộng/cao cho cửa sổ dò tìm là **24x 24** để phát hiện trên **40** ảnh mới không có trong tập huấn luyện với **5** lần lặp để thống kê số biển báo phát hiện đúng trên mỗi ảnh, độ nhiễu trên mỗi ảnh.

Đối với nhận dạng biển báo bằng phương pháp Transfer Learning: Tiến hành lấy những ảnh đã được cắt đúng ở phần kiểm tra phát hiện đối tượng bên trên để thực hiện nhận dạng trên 40 ảnh mới.

So sánh giữa 2 phương pháp HOG + SVM và phương pháp Transfer Learning cho việc nhận dạng biển báo trên cùng 1 tập dữ liệu(**4000** ảnh cho **13 lớp**), áp dụng nghi thức kiểm tra **Hold-out** chia tập dữ liệu ra 2 phần:

- Tập huấn luyện: 2/3 tập dữ liệu (2666 ảnh)
- Tập kiểm tra 1/3 còn lại của tập dữ liệu (1334 ảnh)

Tiến hành kiểm tra độ chính xác mô hình và ma trận nhầm lẫn **(Confusion Matrix)** [24] sau khi dự đoán trên tập kiểm tra trên mỗi phương pháp.

Kiểm tra hoạt động của ứng dụng trên các thiết bị có phiên bản Android từ thấp tới cao (Android 4.x đến Android 8.0) để kiểm tra trên từng độ phân giải nhận ảnh của từng phiển bản Android.

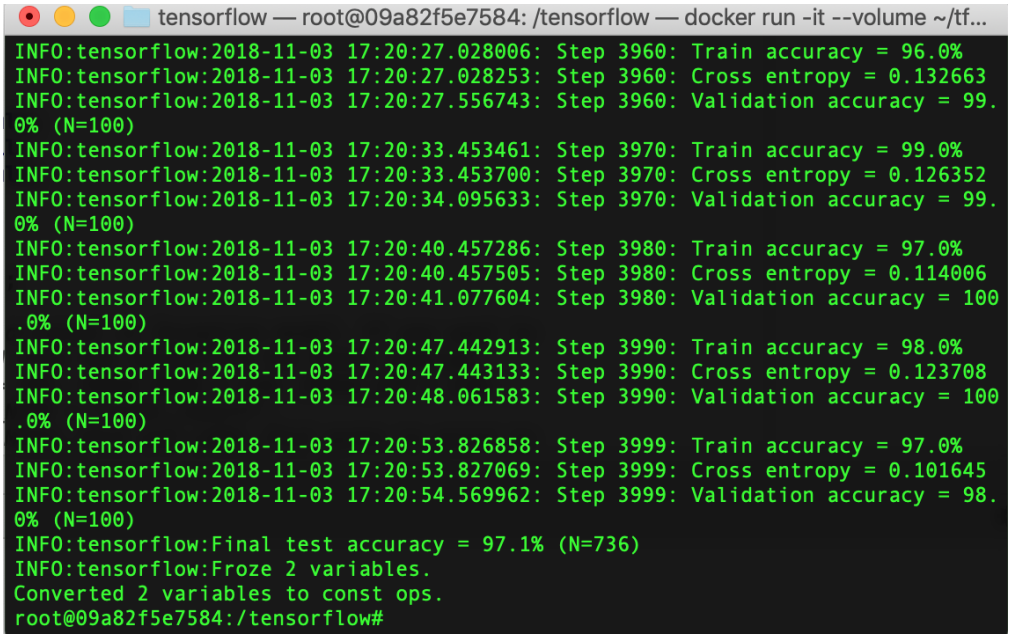
3. Kết quả kiểm tra

Kết quả phát hiện biển báo với 5 lần lặp: Số lượng phát hiện được chính xác đối tượng trên 40 ảnh cao, số lượng nhiễu có thể dùng ngưỡng lớn hơn giá trị chiều dài, chiều rộng trong quá trình huấn luyện **(24x24)** có thể là **100x100, 200x200…** để lấy ngưỡng có diện tích cao hơn sẽ loại bỏ được nhiễu. Bảng bên dưới thống kê qua 5 lần lặp kiểm tra phát hiện đối tượng với 40 ảnh mới không trong tập huấn luyện.

Lần lặp	Số ảnh phát hiện đúng
1	38/40 (95%)
2	36/40 (90%)
3	39/40 (97%)
4	36/40 (90%)
5	36/40 (90%)

Bảng 1: Thống kê độ chính xác mô hình phát hiện biển báo với phương pháp Cascade

Kết quả độ chính xác mô hình trên **2666** ảnh huấn luyện, **1334** ảnh kiểm tra là **97.1%** (Hình 18) với phương pháp **Transfer Learning** và trên cùng tập huấn luyện nhưng độ chính xác của phương pháp **HOG + SVM** là **95%**



Hình 18: Kết thúc quá trình huấn luyện với phương pháp Transfer Learning

```
Do chinh xac: 95%
[[154   0   0   0   0   0   0   0   0   0   0   1   0   1]
 [  1 144   0   1   2   0   0   1   0   1   0   1   0   7]
 [  0   2 282   0   0   0   2   0   2   4   0   0   0   3]
 [  0   1   3  31   1   0   0   0   0   1   0   0   0   0]
 [  0   5   0   0  48   0   0   0   0   0   0   0   0   2]
 [  0   0   0   0   0  49   2   0   0  11   0   0   0   0]
 [  0   2   1   0   0   0 134   0   0   6   1   0   0   0]
 [  0   1   0   0   1   0   0  92   0   0   0   2   0   2]
 [  0   0   1   2   0   0   0   0  47   0   0   0   0   0]
 [  0   0   6   1   2   7   3   1   1 150   0   0   0   1]
 [  0   0   0   0   0   0   0   1   0   0   7   0   0   0]
 [  1   2   0   0   1   0   0   2   0   0   1  62   0   2]
 [  0   1   0   0   0   0   0   0   0   0   0   1  10   0]
 [  0   0   0   1   0   0   0   0   1   0   1   0   0 997]]
```

Hình 19: Độ chính xác và ma trận nhầm lẫn với phương pháp HOG + SVM

Cả hai phương pháp trên cùng tập huấn luyện cho kết quả khá tương đồng, nhưng ưu thế ở phương pháp Transfer Learning về mặt thời gian huấn luyện, có thể huấn luyện mô hình với tập dữ liệu lớn hơn với thời gian nhanh hơn. Đặc biệt vì khi áp dụng phương pháp Transfer Learning bằng thư viện TensorFlow có thể chuyển đổi định dạng mà trên thiết bị di động có thể sử dụng mô hình là **protobuffer.**

Kết quả chạy ứng dụng trên các phiên bản Android khác nhau để kiểm tra việc nhận ảnh tốt của từng phiên bản để phục vụ cho việc phát hiện và nhận dạng biển báo tốt cho thấy rằng các thiết bị nhận ảnh với độ phân giải thấp như Android 4.4 và Android 5, các phiên bản từ Android 5 trở lên với độ phân giải **Full HD 1920x1080** nhận ảnh tốt và cho kết quả tốt trong quá trình phát hiện nhận diện biển báo.

PHẦN KẾT LUẬN

1. Kết quả đạt được:

Xây dựng thành công ứng dụng phát hiện và nhận diện đối tượng trên thiết bị di động đáp ứng thời gian thực với phương pháp Cascade và Transfer Learning, giúp hiểu biết hơn về các biển báo trên các tuyến đường. Ứng dụng cho phép người dùng thao tác đơn giản và kết quả phát hiện/ nhận diện biển báo lên đến **10~15 FPS,** áp dụng kỹ thuật đa luồng tận dùng nguồn tài nguyên của hệ thống, giảm chi phí tính toán.

2. Hướng phát triển:

Tối ưu hoá dòng lệnh, cải thiện hiệu năng để giảm thiểu việc tiêu hao năng lượng của một điện thoại, áp dụng các kỹ thuật xử lý ảnh để hỗ trợ người dùng sử dụng các thiết bị nhận ảnh kém và mờ bởi độ phân giải thấp. Tăng cường số lượng biển báo, dữ liệu để ứng dụng có thể nhận diện bất cứ biển báo trên đường trong nước và ngoài nước. Cải thiện giao diện người dùng. Áp dụng các phương pháp phát hiện biển báo, nhận diện biển báo khác như YOLO, phân vùng màu và phát hiện hình dạng biển báo để năng cao hiệu suất bởi đây là các bài toán phát hiện và nhận dạng đối tượng dành cho thời gian thực trên di động

TÀI LIỆU THAM KHẢO

[1] Tensorflow tutorials: Re-train your own dataset.

[2] DeepLearning: Transfer Learning.

[3] Mohsen Kaboli.“A Review of Transfer Learning Algorithms”, [Research Report]Technische Universität, München. 2017.

[4] Kushagra Pandyaa, Pranay Singhala, “Image Classification using Transfer Learning”, ISSN : 0974–5572, International Journal of Control Theory and Applications.

[5] Huu Tiep Vu, Website, “machinelearningcoban.com”, Perceptron Learning Algorithm, Softmax Regression, Multi-layer Perceptron và Backpropagation, Transfer Learning, Tích chập 2 chiều.

[6] Krizhevsky, Alex. "ImageNet Classification with Deep Convolutional Neural Networks" (PDF). Retrieved 17 November 2013.

[7] Giáo trình xử lý ảnh, bài giảng thị giác máy tính.

[8] Cascade Training GUI , http://amin-ahmadi.com/cascade-trainer-gui/

[9] Cascade Classifier documents.

[10] OpenCV for Android.

[11] Tensorflow for Android, Python.

[12] Android Programming: Better performance through threading

[13]"CS231n Convolutional Neural Networks for Visual Recognition". *cs231n.github.io*. Retrieved 2017-04-25.

[14] Onoda, T., Ratsch, G., M ¨ uller, K.R.: An asymptotic analysis of AdaBoost in the binary classifi- ¨ cation case. In: Proceedings of the 8th International Conference on Artificial Neural Networks, pp. 195–200 (1998)

[15] Freund, Y.: Boosting a weak learning algorithm by majority. Information and Computation 121(2), 256–285 (1995)

[16] J Greenhalgh, M. Mirmehdi, "Real-time detection and recognition of road traffic signs. Intelligent Transportation Systems", IEEE Transactions on, vol. 13, no. 4, pp. 1498-1506, 2012.

[17] S. Xu, "Robust traffic sign shape recognition using geometric matching", Intelligent Transport Systems IET, vol. 3, no. 1, pp. 10-18, 2009.

[18] B Xiong, O. Izmirli, "A road sign detection and recognition system for mobile devices", 112012 International Workshop on Image Processing and Optical Engineering. International Society for Optics and Photonics, 2012.

[19] A Gudigar, B N Jagadale, P K Mahesh et al., "Kernel Based Automatic Traffic Sign Detection and Recognition Using SVM" in Eco-friendly Computing and Communication Systems, Berlin Heidelberg:Springer, pp. 153-161, 2012.

[20] Krückhans, Martin. "Ein Detektor für Ornamente auf Gebäudefassaden auf Basis histogram-of-oriented-gradients"-Operators".

[21] Chung Yu Wang “Traffic Sign Detection using You Only Look Once Framework”

[22] Trương Quốc Bảo, Trương Hùng Chen và Trương Quốc Định. Phát hiện và nhận dạng biển báo giao thông đường bộ sử dụng HOG và Mạng nơron tích chập, 2.2 phân đoan ảnh.

[23] AlexanderShustanov. CNN for Real-Time Trafic Sign Recognition.

[24] Scikit-Learning documents for model avalution, https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html