ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

Báo Cáo Cuối Kì

Chủ đề: Thị giác máy tính - Nhận diện xe

Họ và tên: Lâm Tiên Điền An

Giảng viên hướng dẫn: Hà Lê Hoài Trung

Hồ Chí Minh, 2019

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

Báo Cáo Cuối Kì

Chủ đề: Thị giác máy tính - Nhận diện xe

Môn: Đồ án 1

Lớp: CE201

Giảng viên: Hà Lê Hoài Trung

Họ và tên: Lâm Tiên Điền An

MSSV: 15520002

Nội dung

Chương 1: Giới thiệu đề tài

1.1 Lý do chọn đề tài

Ngày nay, các xe tự hành đang được sự quan tâm rất nhiều của các công ty lớn trên giới như Tesla, Uber, Google… tập trung nghiên cứu phát triển và dần hoàn thiện. Xe tự hành sử dụng kết hợp nhiều thiết bị như GPS, Radar, camera,…đã được chạy thử nghiệm và bước đầu đạt được những thành công nhất định.Tuy nhiên, các xe tự hành này cũng còn gặp nhiều hạn chế do thời tiết, điều kiện giao thông,… nên chưa được đưa vào sản xuất đại trà.

Nhận diện xe là một thành phần rất quan trọng của hệ thống xe tự hành. Dựa vào những đánh giá và tìm hiểu về xe tự hành nhóm đã quyết định chọn đề tài "Nhận diện xe" để làm nghiện cứu. Với mong muốn xe có thể phát hiện được các xe khác khi tham gia giao thông để đưa ra lời cảnh báo nhằm hạn chế được các rủi ro tai nạn giao thông trong những tình huống người lái mất tập trung hoặc tầm nhìn của người lái bị hạn chế (ví dụ như do điểm mù).

1.2 Tình hình nghiên cứu trong và ngoài nước

1.2.1 Trong nước

Nhận diện xe là một thành phần rất quan trọng của hệ thống xe tự hành, ở nước ta còn rất hạn chế. Việt Nam đang ở những bước đi đầu tiên nghiên cứu và ứng dụng xe tự hành. Các viện nghiên cứu, các trường đại học lớn đang rất hào hứng tham gia vào lĩnh vực này. Để trở thành xe tự hành, những chiếc xe bình thường cần phải lắp thêm hàng loạt bộ cảm biến, cùng với đó là trung tâm lưu trữ dữ liệu cùng thuật toán để tự ra quyết định.

FPT software là một đại diện tiêu biểu cho việc nghiên cứu xe tự hành ở Việt Nam . Sau 6 tháng nghiên cứu và phát triển, ngày 31/10 các phần mềm này đã được đưa vào vận hành thử nghiệm trên ô tô chạy trong khuôn viên của công ty. Cụ thể vào ngày 31/10/2017, FPT đã công bố chiếc xe hơi thử nghiệm công nghệ tự hành đầu tiên ở Việt Nam.

Ảnh xe tự hành của FPT

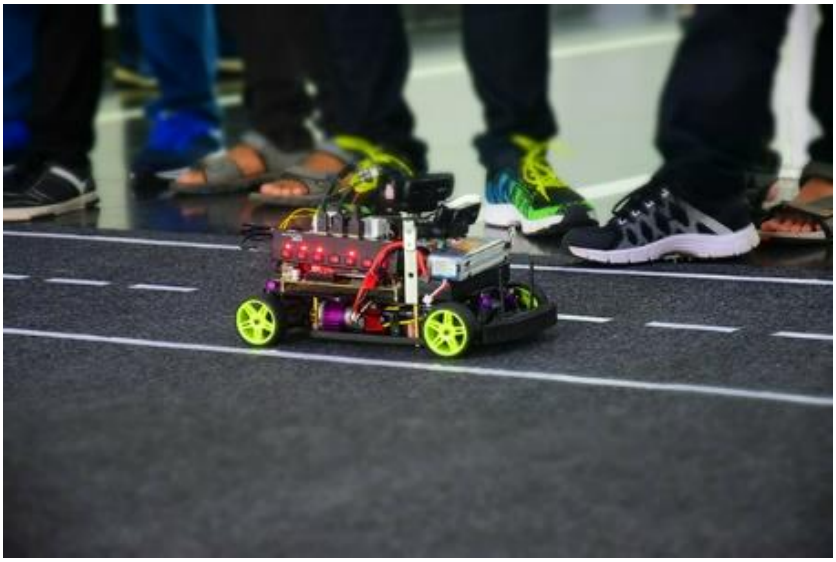
Hiện giao thông ở những thành phố lớn, đông đúc của Việt Nam như Hà Nội, thành phố Hồ Chí Minh vẫn còn là một thách thức để ứng dụng xe tự lái. Dự kiến, mất khoảng 10 năm nữa để cập nhật, phát triển công nghệ, xe tự lái mới có thể chạy trên đường phố ở Việt Nam.

Tuy nhiên, trước mắt, xe tự lái được nghiên cứu để ứng dụng trong sân golf, trong kho hàng, di chuyển giữa các toà nhà trong khu công nghệ cao Hoà Lạc, hay trong sân bay bằng xe tự lái là điều hoàn toàn khả thi ở Việt Nam.

1.2.2 Ngoài nước

Trên thế giới, thị giác máy tính và xe robot được phát triển từ khá lâu và không còn quá xa lạ. Hiện tại có rất nhiều hãng xe ô tô lớn đang đầu tư để phát triển ngành công nghệ tương lai này, có thể kể đến như Honda, Ford, Samsung,….

Bên cạnh các xe phát hiện đối tượng thì các xe có thị giác còn được sử dụng theo dõi các mục tiêu di động trong quốc phòng và xã hội.



Chương 2: Tổng quan Thị giác Máy tính

2.1 Thị giác Máy tính

Thị giác máy tính là một lĩnh vực trong Artificial Intelligence và Computer Science (Trí tuệ nhân tạo và Khoa học máy tính) nhằm giúp máy tính có được khả năng nhìn và hiểu giống như con người.

Tạo ra một chiếc máy nhìn được như cách con người nhìn là không đơn giản, không chỉ vì khó tạo ra một chiếc máy như vậy mà ngay cả chúng ta cũng chưa thực sự hiểu cách thức hoạt động của quá trình nhìn.

Quá trình mô phỏng thị giác con người này được chia thành 3 giai đoạn nối tiếp (tương tự cách con người nhìn): mô phỏng mắt (thu nhận - khó), mô phỏng vỏ não thị giác (xử lý - rất khó) và mô phỏng phần còn lại của bộ não (phân tích và nhận dạng các hình ảnh - khó nhất).

2.2 Thư viện của OpenCV

2.2.1 OpenCV là gì ?

OpenCV (Open Source Computer Vision) là một thư viện mã nguồn mở.

OpenCV đã được bắt đầu tại Intel vào năm 1999 bởi Gary Bradsky và bản phát hành đầu tiên xuất hiện vào năm 2000.

OpenCV được phát hành theo giấy phép BSD, do đó nó hoàn toàn miễn phí cho cả học thuật và thương mại.

OpenCV cung cấp cho chúng ta một hệ thống đơn giản, dễ dàng sử dụng giúp cho mọi lập trình viên có thể nhanh chóng tạo ra các ứng dụng về thị giác máy tính như các hệ thống giám sát an ninh, phân tích ảnh chụp để chuẩn đoán bệnh trong y tế, bảo mật, robot, xe hơi tự lái….

Nó không chỉ cung cấp các trình xử lý ảnh đơn giản mà còn kể cả các hàm bậc cao như nhận diện khuôn mặt, nhận diện vật thể, theo dõi các đối tượng.

2.2.2 Tại sao chọn thư viện OpenCV ?

OpenCV 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 và 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 lượng tải vượt quá 14 triệu lần.

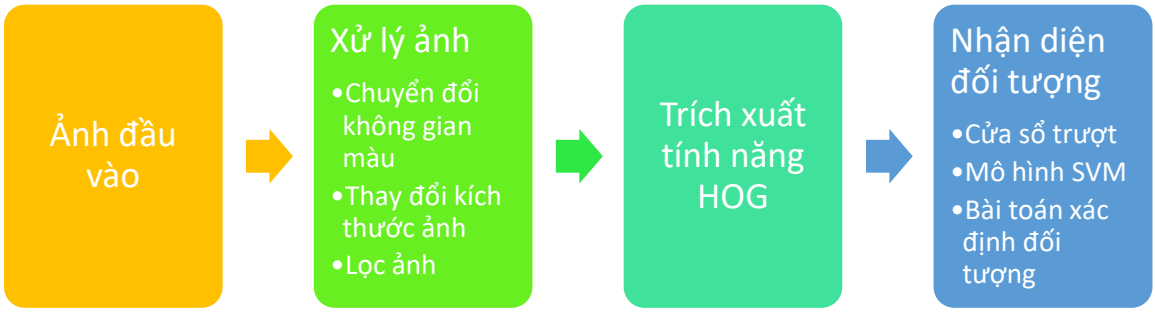
OpenCV cung cấp cho chúng ta thư viện với hơn 500 hàm và 2500 thuật toán được tối ưu để phát hiện và nhận dạng gương mặt, xác định các đối tượng, phân loại hành động của con người trong video, theo dõi chuyển 12 động của camera, theo dõi các đối tượng di chuyển, giải phóng các mô hình 3D của các đối tượng, tạo ra các điểm mây 3D từ các camera âm thanh nổi, ghép các hình ảnh với nhau để tạo ra độ phân giải cao hình ảnh của toàn bộ cảnh, tìm hình ảnh tương tự từ cơ sở dữ liệu hình ảnh, loại bỏ mắt đỏ khỏi những bức ảnh chụp bằng flash, theo dõi chuyển động của mắt, nhận diện cảnh quan.

Chương 3: Histogram of Oriented Gradients - HOG

3.1 Giới thiệu Histogram of Oriented Gradients

HOG là một mô tả tính năng được sử dụng trong thị giác máy tính và xử lý hình ảnh, dùng để xác định một đối tượng. Các khái niệm về HOG được nêu ra từ năm 1986 tuy nhiên cho đến năm 2005 HOG mới được sử dụng rộng rãi sau khi Navneet Dalal và Bill Triggs công bố những bổ sung về HOG. HOG tương tự như các biểu đồ edge orientation, scale-invariant feature transform descriptors(như sift, surf,..), shape contexts nhưng HOG được tính toán trên một lưới dày đặc các cell và chuẩn hóa sự tương phản giữa các block để nâng cao độ chính xác. HOG được sử dụng chủ yếu để mô tả hình dạng và sự xuất hiện của một object trong ảnh.

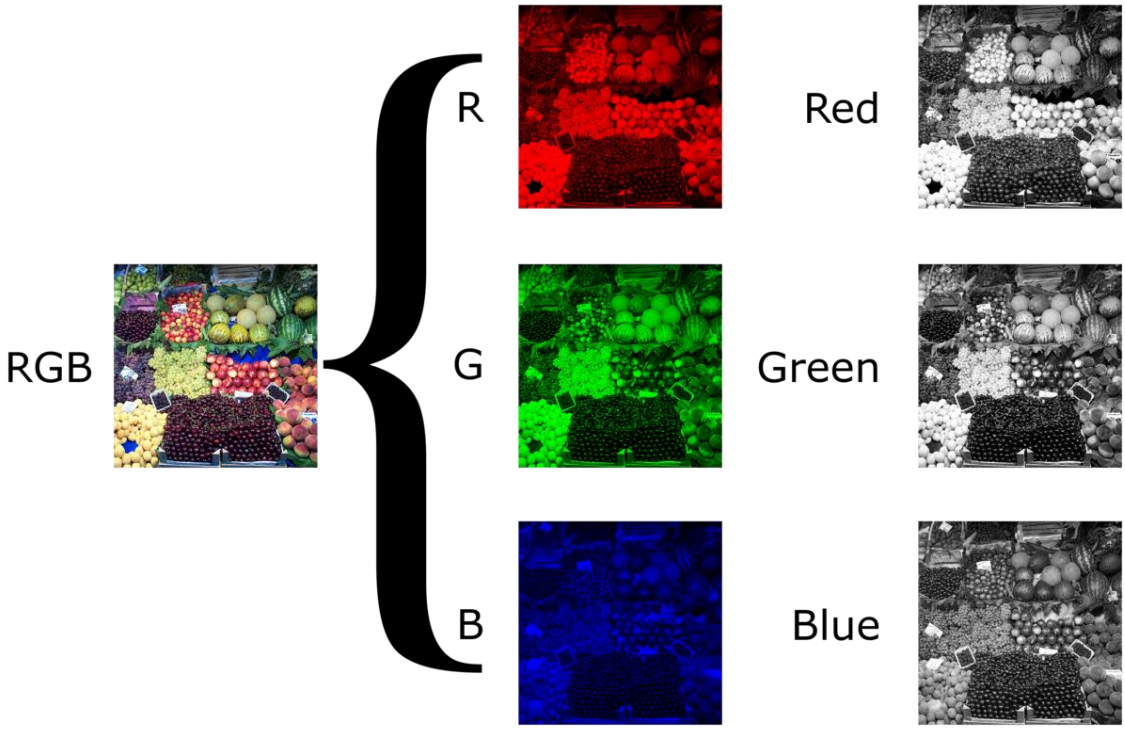
3.2 Sơ đồ giải thuật



3.3 Xử lý ảnh

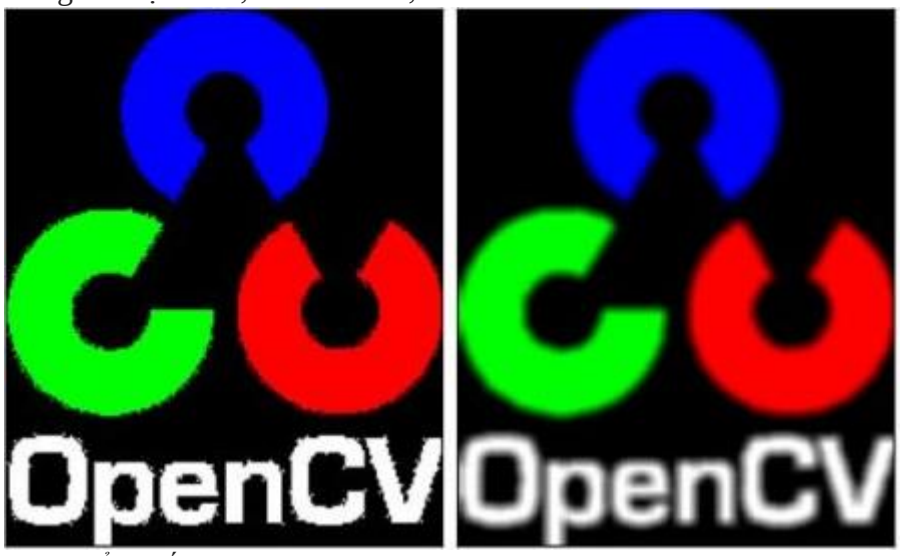
Để xác định đối tượng một cách chính xác hơn ta cần xử lý để hình ảnh có một kích thước chung.

Ta chuyển đổi không gian màu để cho máy tính có thể tính toán dễ dàng hơn. Các không gian màu gồm: RGB, YCrCb, YUV, Grayscale,…



Không gian màu RGB.

Lọc ảnh để làm giảm độ nhiễu, làm mờ ảnh,...



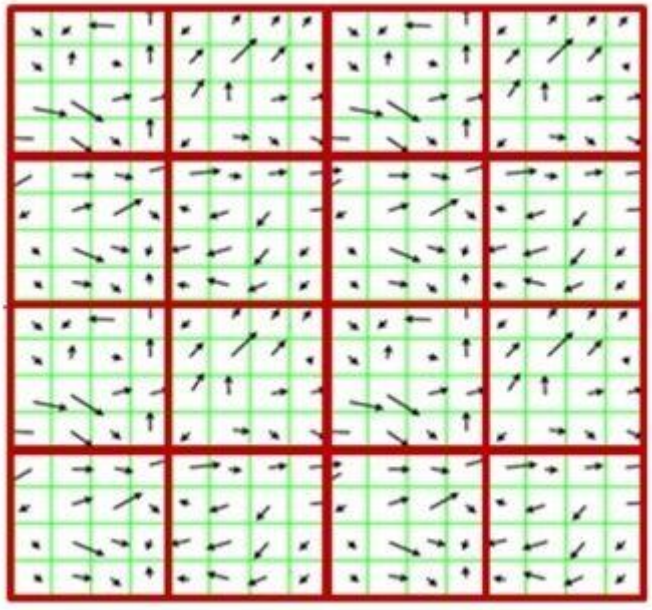
Ảnh gốc và ảnh sau khi bị làm mờ qua bộ lọc Tính trung bình

3.4 Trích xuất tính năng HOG

3.4.1 Tính toán gradient theo cả hướng x và y

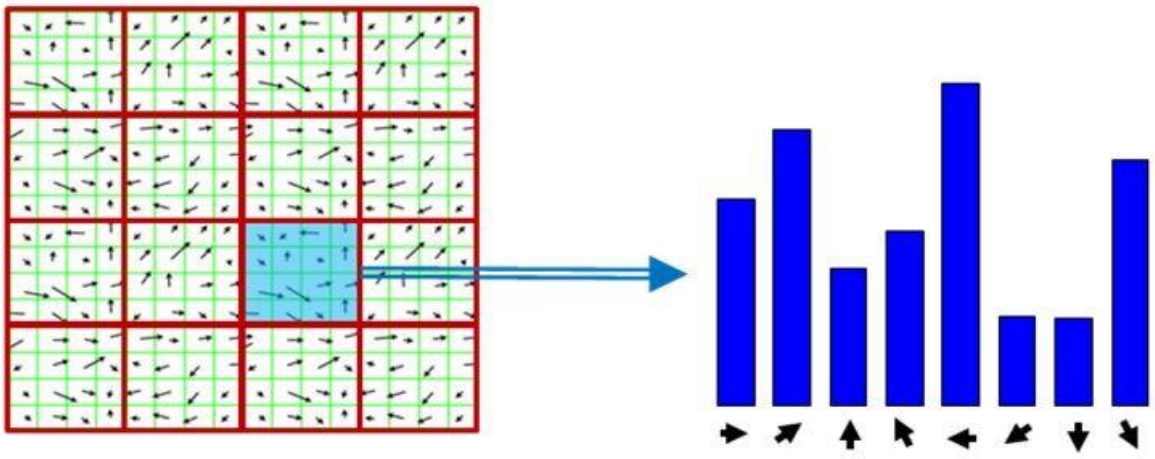
Lược đồ Gradients tính toán dựa trên thông tin về hướng và cường độ biến thiên màu /mức xám tại mỗi vùng trên ảnh.

Để lấy được hình ảnh gradient, chúng ta sẽ sử dụng tích chập(convolution): $G_x = I * D_x$ và $G_y = I * D_y$ với I là hình ảnh đầu vào, D_x là bộ lọc cho chiều x, và D_y là bộ lọc cho chiều y . Sau khi có các ảnh gradient, chúng ta có thể tính toán cường độ gradient của hình ảnh: $|G| = \sqrt{{G_x}^2 + {G_y}^2}$ Cuối cùng, định hướng của gradient cho mỗi pixel trong hình ảnh ban đầu được tính bằng cách: $\theta = arctan(\ G_x / G_y\ )$ Dự vào |G| và θ, chúng ta có thể tính được một biểu đồ cường độ gradient, trong đó cột của histogram dựa trên θ và trọng số của mỗi cột của biểu đồ được dựa trên |G|.



3.4.2 Lấy phiếu bầu cùng trọng số trong các cell

Bây giờ chúng ta cần chia hình ảnh của chúng ta thành các cell và block Một cell là một vùng hình chữ nhật được xác định bởi số điểm ảnh thuộc mỗi cell. Ví dụ: nếu ta có một hình ảnh 128 x 128 với pixel_per_cell = 4 x 4 thì sẽ có 32 x 32 = 1024 cell, pixel_per_cell = 32 x 32, sẽ có 4 x 4 = 16 cell. Với mỗi cell trong bức ảnh, ta cần xây dựng 1 biểu đồ cường độ gradient. Mỗi pixcel sẽ được vote vào vào biểu đồ, trọng số của mỗi vote chính là cường độ gradient tại pixel đó Cuối cùng, mỗi pixel đóng góp một phiếu bầu có trọng số vào biểu đồ - trọng lượng của phiếu chỉ đơn giản là cường độ gradient |G| tại pixel đó.Lúc này, chúng ta có thể thu thập và ghép các biểu đồ này để tạo ra feature vector cuối cùng. Tuy nhiên, ta sẽ chuẩn hóa các block để có được kết quả tốt hơn.

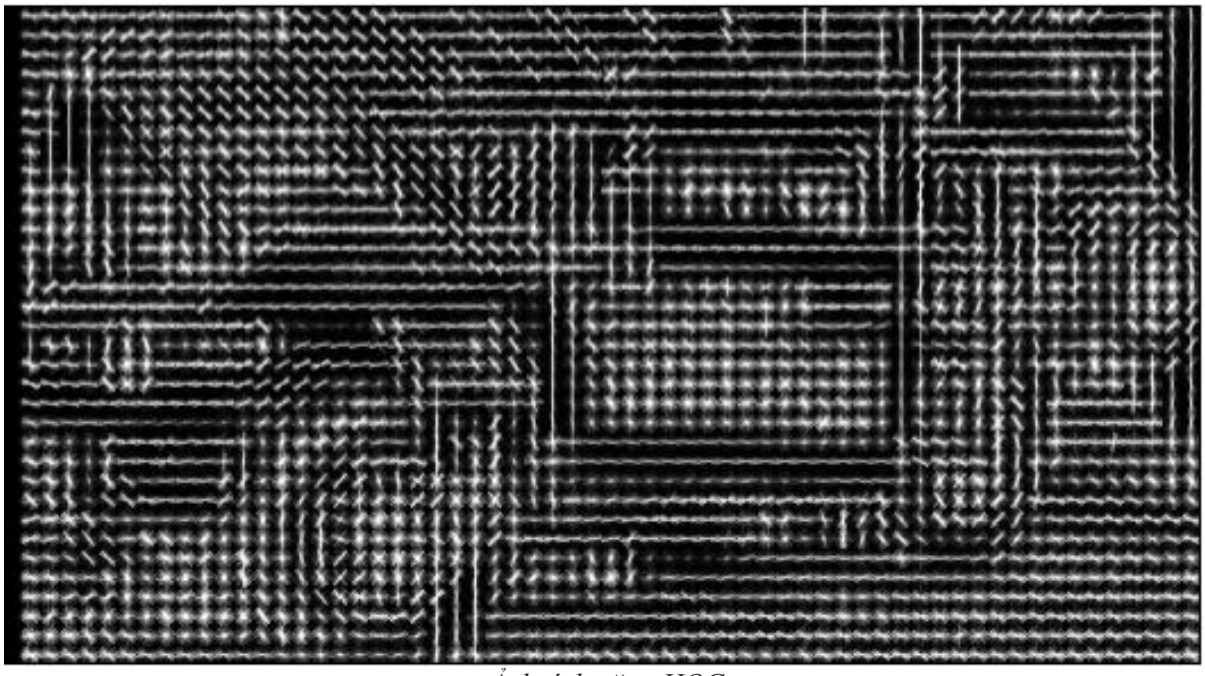


3.4.3 Chuẩn hóa các block

Một lần nữa, ta cần chia các block giống như chia cell ở phía trên. Đơn vị của ta không còn là các điểm ảnh nữa mà là các cell. Người ta thường sử dụng hoặc 2 x 2 hoặc 3 x 3 cell_per_block có được độ chính xác hợp lý trong hầu hết các trường hợp.Các block này sẽ chồng lên nhau. Ví du: ta có 3x3 cells và cell_per_block= 2x2 thì ta sẽ có 4 block. Tiếp đến, ta sẽ tiến hành thu thập và ghép các histogram của cell trong block.

3.4.4 Thu thập tất cả các biểu đồ cường độ gradient định hướng để tạo ra feature vector cuối cùng

Ảnh gốc



Ảnh tính năng HOG

3.5 Phát hiện đối tượng

3.5.1 Cửa sổ trượt – Sliding Window

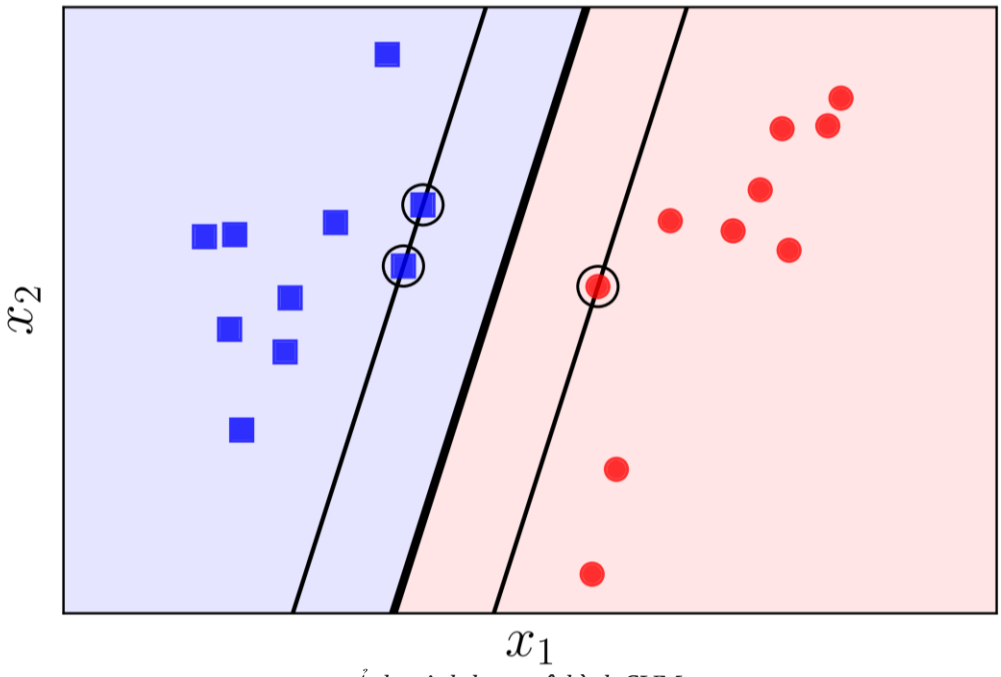
Cửa sổ trượt kiểm tra các phần dịch chuyển của ảnh và phát hiện hoạt động trên các hình ảnh sử dụng ảnh kim tự tháp. Phát hiện được đối tượng ở mức độ đa tỉ lệ (multiscale level).

Cửa sổ trượt giải quyết các vấn đề về vị trí bằng cách quét các vùng nhỏ hơn của hình ảnh lớn hơn và sau đó lặp lại quá trình quét trên các tỷ lệ khác nhau của cùng một hình ảnh. Công nghệ này cho phép mỗi hình ảnh được phân tách thành các phần, cho phép loại bỏ các phần không có khả năng chứa các đối tượng.

3.5.2 Mô hình SVM

SVM (Support Vector Machine) là một khái niệm trong thống kê và khoa học máy tính cho một tập hợp các phương pháp học có giám sát liên quan đến nhau để phân loại và phân tích hồi quy.

SVM là một thuật toán phân loại nhị phân, SVM nhận dữ liệu vào và phân loại chúng vào hai lớp khác nhau. Với một bộ các ví dụ luyện tập thuộc hai thể loại cho trước, thuật toán luyện tập SVM xây dựng một mô hình SVM để phân loại các ví dụ khác vào hai thể loại đó.

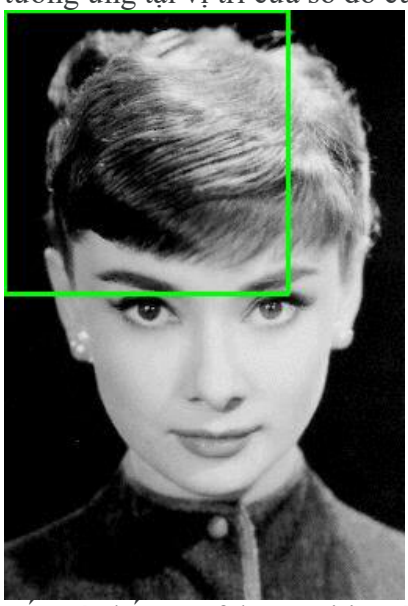


Ảnh minh họa mô hình SVM

3.5.3 Bài toán xác định đối tượng

Sau khi đã có "mô tả tính năng HOG" ta sẽ sử dụng vào bài toán xác định đối tượng.
Lấy ra số lượng P các hình ảnh chưa đối tượng và trích xuất HOG feature descriptor từ các hình ảnh này:

1. Lấy ra N các hình ảnh không chưa bất kì một đối tượng nào và trích xuất HOG feature descriptor từ các hình ảnh này. Trong thực tế thì N>>PN>>P
2. Huấn luyện mạng SVM trên các HOG feature descriptor trên tập dữ liệu từ bước một và bước 2
3. Đối với mỗi hình ảnh trong tập không chứa đối tượng, sử dụng phương pháp sliding window, tại mỗi vị trí cửa sổ tính toán giá trị HOG và sử dụng mô hình SVM đã huấn luyện ở trên để dự đoán kết quả. Nếu mô hình đưa ra kết quả sai, lưu lại giá trị HOG tương ứng tại vị trí cửa sổ đó cùng xác suất được dự đoán

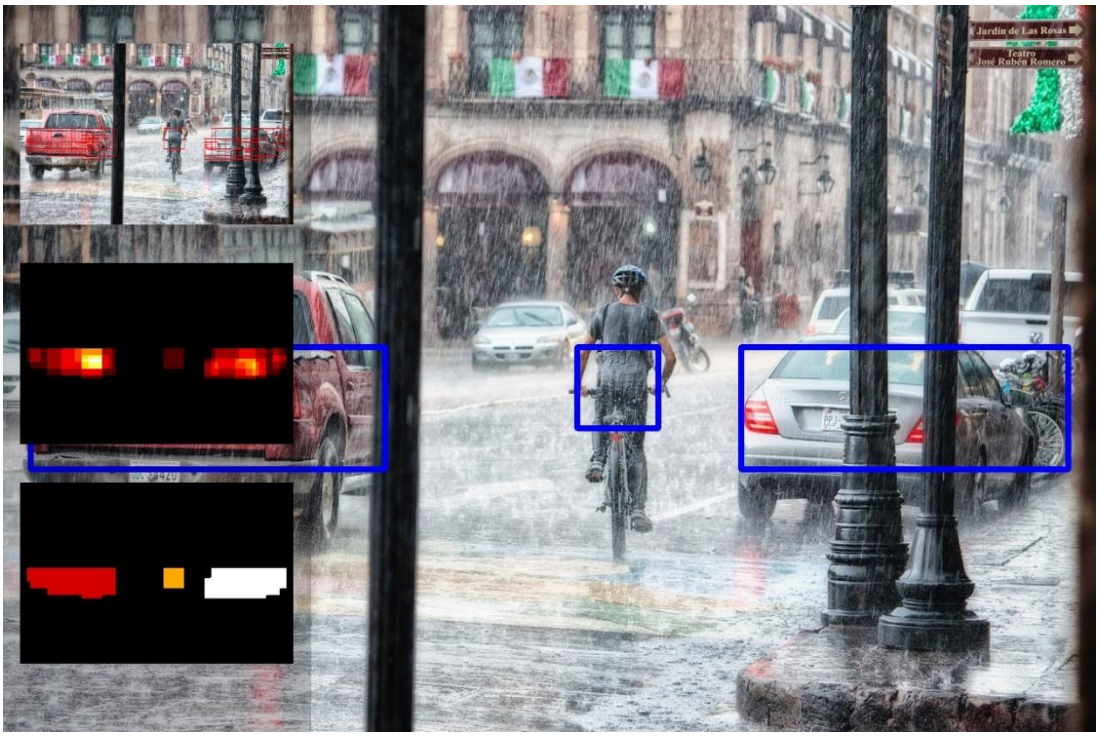


4. Lấy các kết quả false-positive tìm thấy ở bước 4 , sắp xếp theo giá trị của xác suất và huấn luyện lại mô hình SVM

5. Kết thúc

3.6 Kết quả thực nghiệm

Chương trình hiện tại chỉ xử lý được video ở định dạng mp4.
Kết quả khi dùng nhận diện đối tượng bằng HOG:


Jardín de Las Rosas
Teatro
José Rubén Romero

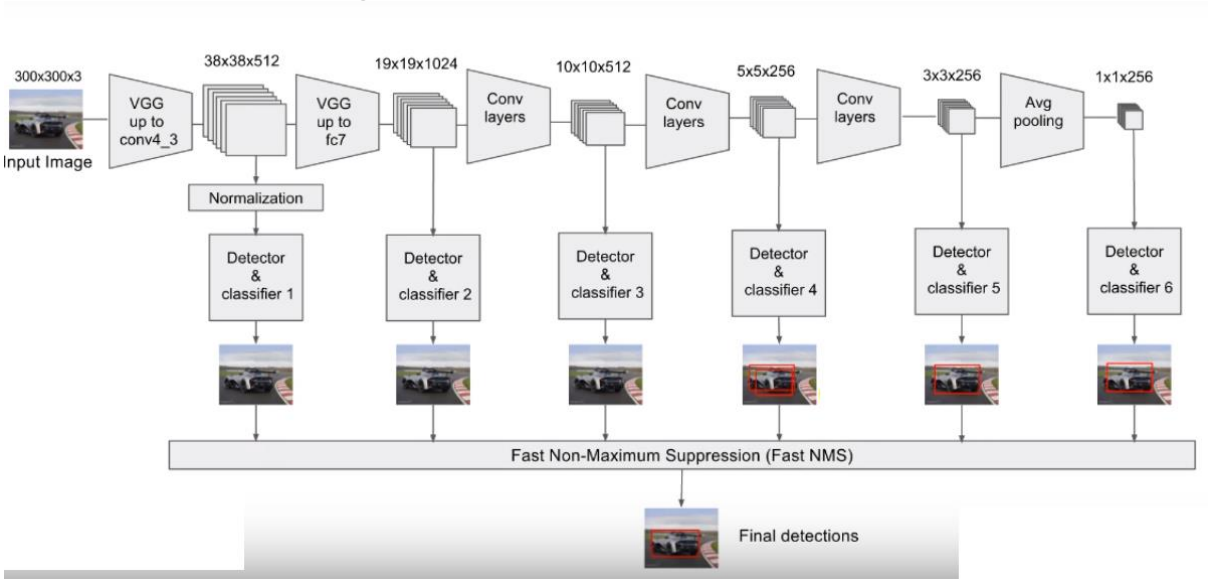
Chương 4: Single Shot MultiBox Detector - SSD

4.1 Giới thiệu Single Shot MultiBox Detector

Single Shot: là các nhiệm vụ của việc nội địa hoá và phân loại đối tượng được thực hiện trong một lần chuyển tiếp duy nhất của mạng.
MultiBox: là tên của một kỹ thuật để hồi quy bounding box được phát triển bởi Szegedy và cộng sự.
Detector: là một bộ dò tìm đối tượng cũng như phân loại các đối tượng được phát hiện.
SSD ban đầu được phát triển bởi Google. Nó là một thuật toán phổ biến trong việc xác định đối tượng. Thuật toán này đơn giản hơn so với Faster R-CNNs. tốc độ FPS nhanh hơn Faster R-CNNs. 22-46 FPS tùy thuộc vào biến thể sử dụng. SSD tốc độ chậm hơn YOLO nhưng có xu hướng chính xác hơn YOLO.

4.2 Mô hình

4.2.1 Mô hình hoạt động



4.2.2 Trích xuất bản đồ tính năng

SSD sử dụng VGG16 để trích xuất các bản đồ tính năng. Phát hiện các đối tượng bằng cách sử dụng lớp Conv4_3. VGG16 được phát triển năm 2014, là một biến thể sâu hơn nhưng lại đơn giản hơn so với kiến trúc convolution (từ gốc: convolutional structure) thường thấy ở CNN. Convolutional Neural Network (CNNs – Mạng nơ-ron tích chập) là một trong những mô hình Deep Learning tiên tiến giúp ta xây dựng được những hệ thống thông minh với độ chính xác cao. CNN được sử dụng nhiều trong các bài toán nhận dạng các đối tượng trong ảnh.
Mỗi prediction gồm 1 boundary box và 21 điểm cho mỗi class (một lớp phụ cho không có đối tượng). Chọn điểm số cao nhất làm lớp cho đối tượng bị chặn
Conv4_3 tạo tổng dự đoán là 38×38×4: bốn dự đoán cho mỗi ô, bất kể độ sâu của feature maps.
SSD dự trữ một lớp "0" để cho biết nó không có đối tượng.
Việc tạo nhiều dự đoán có chứa các hộp ranh giới và điểm tin cậy được gọi là multibox.

4.2.3 Dự đoán tích chập

SSD không sử dụng mạng đề xuất vùng được ủy quyền. Nó tính cả vị trí và điểm số class bằng cách sử dụng các bộ lọc tích chập nhỏ . Sau khi trích xuất các bản đồ tính năng, SSD áp dụng các bộ lọc chập 3×3 cho mỗi ô để đưa ra các dự đoán. (tính toán kết quả giống như các bộ lọc CNN thông thường.) Mỗi bộ lọc xuất 25 kênh: 21 điểm cho mỗi lớp cộng với một hộp

biên. Ví dụ, trong Conv4_3, chúng em áp dụng 4 bộ lọc 3×3 để ánh xạ 512 kênh đầu vào đến 25 kênh đầu ra.

$$(38x38x512) \xrightarrow{(4x3x3x512x(21+4))} (38x38x4x(21+4))$$

4.2.4 Bản đồ tính năng đa tỉ lệ

Phối cảnh làm thay đổi tỉ lệ của các đối tượng.

Ảnh minh họa

Ban đầu, mô tả cách SSD phát hiện các đối tượng từ một layer đơn lẻ. Trên thực tế, nó sử dụng nhiều lớp (bản đồ đa tính năng) để phát hiện các đối tượng độc lập. SSD sử dụng các layer có độ phân giải thấp hơn để phát hiện các đối tượng có kích thước lớn hơn.

SSD bổ sung thêm 6 lớp convolution phụ trợ sau VGG16. Năm trong số đó sẽ được thêm vào để phát hiện đối tượng. Ba trong số năm lớp đó, sẽ đưa ra 6 dự đoán thay vì 4. Tổng cộng, SSD đưa ra 8732 dự đoán khi sử dụng 6 lớp.

Bản đồ tính năng đa tỉ lệ tăng độ chính xác một cách đáng kể.

Prediction source layers from:						mAP use boundary boxes?		# Boxes
38 × 38	19 × 19	10 × 10	5 × 5	3 × 3	1 × 1	Yes	No	
✔	✔	✔	✔	✔	✔	74.3	63.4	8732
✔	✔	✔				70.7	69.2	9864
	✔					62.4	64.0	8664

Độ chính xác được đo bằng độ chính xác trung bình chính xác - mAP: độ chính xác của các dự đoán.

4.2.5 Các hộp ranh giới mặc định

Việc đào tạo với một hình dạng sẽ không ổn định, chỉ dự đoán đúng ở một loại đối tượng. Dự đoán với nhiều hình dạng hơn sẽ xác định được nhiều đối tượng hơn và ổn định hơn.

Trong thực tế, các hộp biên giới không có hình dạng và kích thước tùy ý. The ground truth boundary boxes có thể được phân chia thành các cụm với mỗi cụm được biểu diễn bằng một hộp ranh giới mặc định(trọng tâm của cụm).

Thay vì đoán ngẫu nhiên, chúng ta có thể bắt đầu phỏng đoán dựa trên các hộp mặc định đó dựa vào thuật toán phân cụm K-means. Để giữ mức độ phức tạp thấp, các hộp mặc định được chọn trước một cách thủ công. SSD cũng giữ các hộp mặc định ở mức tối thiểu (4 hoặc 6) với một dự đoán cho mỗi hộp mặc định. Các dự đoán hộp biên liên quan đến các hộp biên mặc định tại mỗi ô (Δcx, Δcy, w, h).

Đối với mỗi feature map layer, nó dùng chung một tập hợp các hộp mặc định, được căn giữa tại ô tương ứng. Các layer khác nhau sử dụng các bộ hộp mặc định khác nhau để tùy chỉnh phát hiện đối tượng ở các độ phân giải khác nhau.

SSD xác định giá trị scale cho từng lớp bản đồ đặc trưng (từ 0.1 đến 0.9). Đối với các layer tạo 6 dự đoán, SSD bắt đầu với 5 aspect ratio: 1, 2, 3, 1/2 và 1/3.

$$w = scale \cdot \sqrt{\text{aspect ratio}}$$

$$h = \frac{scale}{\sqrt{\text{aspect ratio}}}$$

Then SSD adds an extra default box with scale:

$$scale = \sqrt{\text{scale} \cdot \text{scale at next level}}$$

Công thức tính chiều rộng và chiều cao của các hộp mặc định.

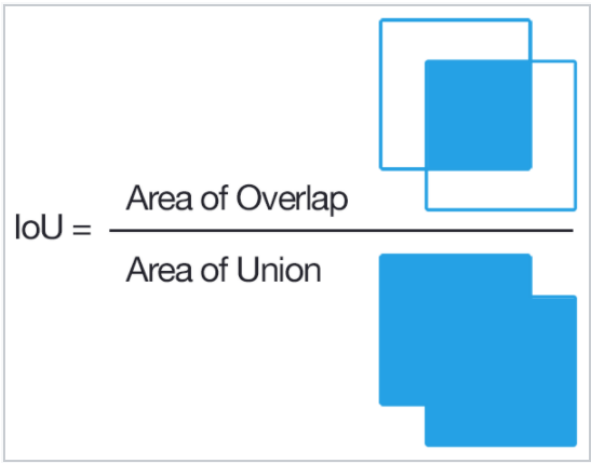
4.3 Đào tạo

4.3.1 Tập dữ liệu

Cần đào tạo và kiểm tra các tập dữ liệu với các ground truth bounding box và các nhãn lớp được gán (chỉ một nhãn cho mỗi hộp biên).
Bộ dữ liệu Pascal VOC và COCO là một điểm khởi đầu tốt.

4.3.2 Chiến lược phù hợp

Prediction SSD được phân loại là kết quả tích cực hoặc kết quả tiêu cực. SSD chỉ sử dụng các kết quả tích cực để tính toán chi phí trong hộp ranh giới không phù hợp.
Nếu default boundary box tương ứng (không phải predicted boundary box) có IoU với ground truth box lớn hơn 0.5, thì kết quả là tích cực. Nếu không, nó là tiêu cực. IoU-Intersection-over-Union, tỉ lệ diện tích chồng chéo lớn nhất trên diện tích chồng chập với một ground truth box.



Công thức tính IoU

Khi xác định được kết quả phù hợp, ta sử dụng predicted boundary box tương ứng để tính toán chi phí.
Chiến lược phù hợp (Matching stategy) khuyến khích mỗi dự đoán dự đoán các hình gần gũi hơn với hộp mặc định tương ứng. Do đó dự đoán đa dạng hơn và ổn định hơn trong việc đào tạo.

4.3.3 Hàm mất mát

Mỗi deep learning / neural network cần một objective function khác nhau để học hỏi. Sau khi kết nối các ground truth và các default box, đánh dấu các hộp mặc định còn lại làm nền, cần xây dựng objective function của SSD. Đưa các predicted box gần hơn với ground truth box.
Hàm mất mát có 2 phần: Confidence loss và Localization loss

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{loc}(x, l, g))$$

Trong đó:

l: predicted box.

g: ground truth box.

c:the number of classes plus the background class that indicates no object.

N: số default box phù hợp.

α : trọng số của localization loss.

4.3.3.1 Confidence loss

Confidence loss: Sự mất mát trong việc đưa ra một class prediction của mỗi đối tượng.

Đối với mỗi dự đoán tích cực, ta sẽ cho giá trị dựa theo điểm confidence của class tương ứng.

Ngược lại, ta sẽ cho giá trị theo điểm confidence của class "0": class "0" class không có đối tượng nào được phát hiện.

$$L_{conf}(x, c) = -\sum_{i \in Pos}^{N} x_{ij}^{p} log(\hat{c}_i^p) - \sum_{i \in Neg} log(\hat{c}_i^0) \quad \text{where} \quad \hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)}$$

$$x_{ij}^{p} = \begin{cases} 1 & \text{if } IoU > 0.5 \text{ between default box } i \text{ and ground true box } j \text{ on class } p \\ 0 & \text{otherwise} \end{cases}$$

4.3.3.2 Localization loss

Localization loss: Sự không phù hợp giữa ground truth box và predicted boundary box. Chỉ được tính trên các hộp tích cực (với một ground truth phù hợp)

Tính toán sự khác biệt giữa sai lệch chính xác và dự đoán (the correct and predicted) cho các tọa độ điểm trung tâm.

Công thức:

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \sum_{m \in \{cx, cy, w, h\}} x_{ij}^{k} \text{smooth}_{\text{L1}}(l_i^m - \hat{g}_j^m)$$

$$\hat{g}_j^{cx} = (g_j^{cx} - d_i^{cx})/d_i^w \qquad \hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy})/d_i^h$$

$$\hat{g}_j^{w} = \log\left(\frac{g_j^w}{d_i^w}\right) \qquad \hat{g}_j^{h} = \log\left(\frac{g_j^h}{d_i^h}\right)$$

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise,} \end{cases}$$

$$x_{ij}^{p} = \begin{cases} 1 & \text{if } IoU > 0.5 \text{ between default box } i \text{ and ground true box } j \text{ on class } p \\ 0 & \text{otherwise} \end{cases}$$

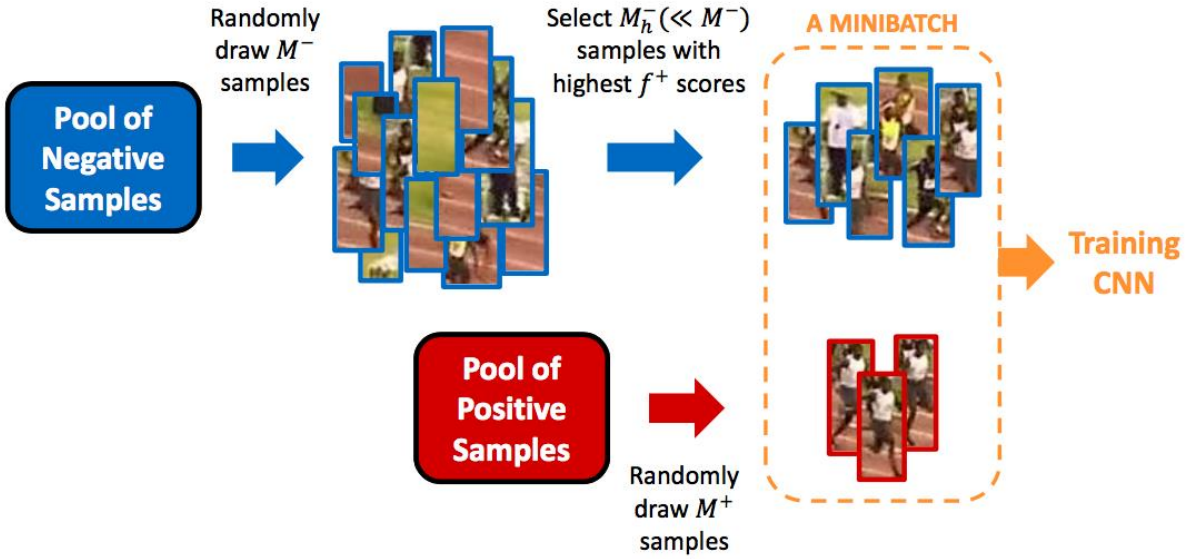
Trong đó: cx,cy: tọa độ trung tâm của default bounding box(d) với chiều rộng (w), chiều cao (h).

4.3.4 Khai thác mẫu tiêu cực

Sau khi kết hợp bước, hầu hết các hộp mặc định là tiêu cực, đặc biệt là khi số lượng các hộp mặc định có thể là lớn. Điều này tạo ra một sự mất cân đối đáng kể giữa các đào tạo tích cực và tiêu cực.

SSD vẫn yêu cầu lấy mẫu tiêu cực để nó có thể tìm hiểu điều gì cấu thành dự đoán xấu. Thay vì sử dụng tất cả các mẫu tiêu cực, ta sắp xếp những mẫu tiêu cực đó bằng tính toán confidence loss. SSD chọn mẫu tiêu cực với mức mất mát hàng đầu và đảm bảo tỷ lệ giữa các mẫu tiêu cực được chọn với mẫu tích cực tối đa là 3:1.

Điều này dẫn đến tối ưu hóa nhanh hơn và đào tạo ổn định hơn.



Ảnh ví dụ về Hard negative mining (từ Jamie Kang blog)

4.3.5 Tăng cường dữ liệu

Tăng cường dữ liệu là quan trọng trong việc cải thiện độ chính xác.

Tăng cường dữ liệu thông qua lật, cắt xén và biến dạng màu sắc. Để xử lý các biến thể trong các kích thước và hình dạng đối tượng khác nhau, mỗi hình ảnh đào tạo được lấy mẫu ngẫu nhiên theo một trong các tùy chọn sau:

Sử dụng bản gốc.

Lấy mẫu path với IoU là 0,1, 0,3, 0,5, 0,7 hoặc 0,9.

Lấy mẫu ngẫu nhiên một path.

Path được lấy mẫu sẽ có tỷ lệ khung hình từ 1/2 đến 2. Sau đó, nó được thay đổi kích thước thành một kích thước cố định và chúng tôi lật một nửa dữ liệu đào tạo. Ngoài ra, còn có thể áp dụng các biến dạng ảnh.

data augmentation	SSD300		
horizontal flip	✔	✔	✔
random crop & color distortion		✔	✔
random expansion			✔
VOC2007 test mAP	65.5	74.3	77.2

Hình ảnh cải tiến hiệu năng sau khi tăng cường dữ liệu

4.3.6 Triệt tiêu tối đa

SSD sử dụng triệt tiêu tối đa (Non-maximum suppression) để loại bỏ các dự đoán trùng lặp trỏ đến cùng một đối tượng.

SSD sắp xếp các dự đoán theo điểm tin cậy (confidence score). Bắt đầu từ những dự đoán có điểm cao điểm tin cậy cao, SSD đánh giá liệu có bất kỳ các predicted boundary box nào trước đây có IoU cao hơn 0,45 so với dự đoán hiện tại cho cùng một class hay không. Nếu được tìm thấy, dự đoán hiện tại sẽ bị bỏ qua. Tối đa, giữ 200 dự đoán hàng đầu cho mỗi hình ảnh.

4.4 Kết quả sau đào tạo

Sử dụng Nvidia Titan X trên thử nghiệm VOC2007, SSD đạt được 59 FPS với 74,3% mAP trên thử nghiệm VOC2007, so với Nhanh hơn R- CNN 7 FPS với mAP 73,2% hoặc YOLO 45 FPS với mAP 63,4%.

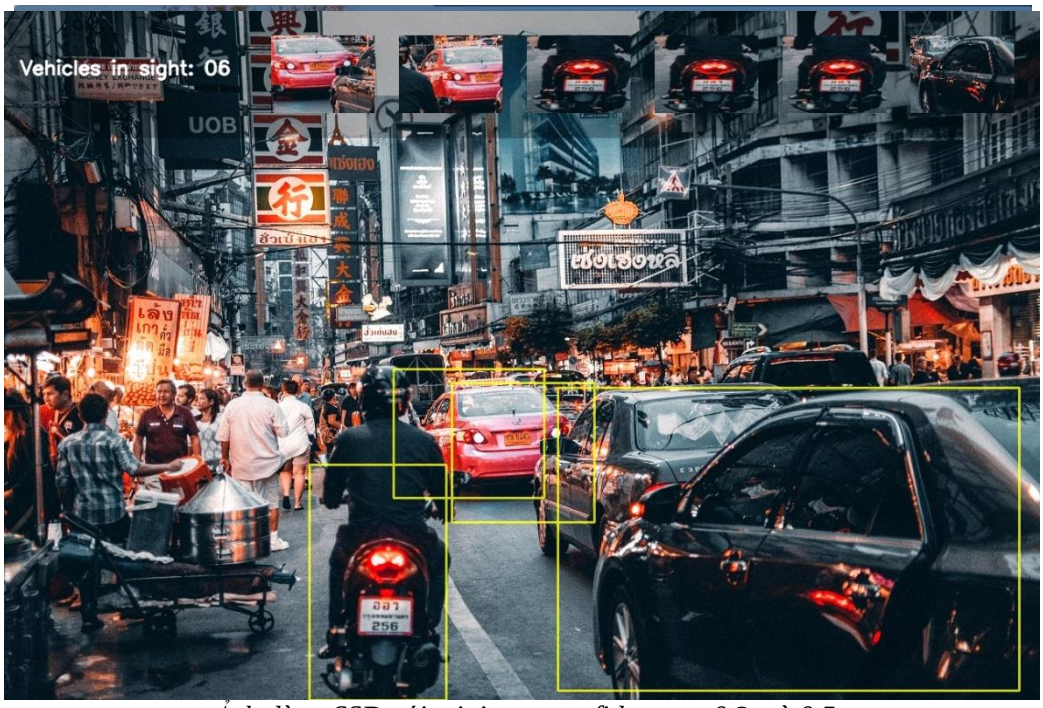
Đây là sự so sánh chính xác cho các phương pháp khác nhau. Đối với SSD, nó sử dụng kích thước hình ảnh 300 × 300 hoặc 512 × 512.

Method	data	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Fast [6]	07	66.9	74.5	78.3	69.2	53.2	36.6	77.3	78.2	82.0	40.7	72.7	67.9	79.6	79.2	73.0	69.0	30.1	65.4	70.2	75.8	65.8
Fast [6]	07+12	70.0	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	74.8	80.4	70.4
Faster [2]	07	69.9	70.0	80.6	70.1	57.3	49.9	78.2	80.4	82.0	52.2	75.3	67.2	80.3	79.8	75.0	76.3	39.1	68.3	67.3	81.1	67.6
Faster [2]	07+12	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
Faster [2]	07+12+COCO	78.8	84.3	82.0	77.7	68.9	65.7	88.1	88.4	88.9	63.6	86.3	70.8	85.9	87.6	80.1	82.3	53.6	80.4	75.8	86.6	78.9
SSD300	07	68.0	73.4	77.5	64.1	59.0	38.9	75.2	80.8	78.5	46.0	67.8	69.2	76.6	82.1	77.0	72.5	41.2	64.2	69.1	78.0	68.5
SSD300	07+12	74.3	75.5	80.2	72.3	66.3	47.6	83.0	84.2	86.1	54.7	78.3	73.9	84.5	85.3	82.6	76.2	48.6	73.9	76.0	83.4	74.0
SSD300	07+12+COCO	79.6	80.9	86.3	79.0	**76.2**	57.6	87.3	88.2	88.6	60.5	85.4	**76.7**	**87.5**	**89.2**	84.5	81.4	55.0	81.9	**81.5**	85.9	78.9
SSD512	07	71.6	75.1	81.4	69.8	60.8	46.3	82.6	84.7	84.1	48.5	75.0	67.4	82.3	83.9	79.4	76.6	44.9	69.9	69.1	78.1	71.8
SSD512	07+12	76.8	82.4	84.7	78.4	73.8	53.2	86.2	87.5	86.0	57.8	83.1	70.2	84.9	85.2	83.9	79.7	50.3	77.9	73.9	82.5	75.3
SSD512	07+12+COCO	**81.6**	**86.6**	**88.3**	**82.4**	76.0	**66.3**	**88.6**	**88.9**	**89.1**	**65.1**	**88.4**	73.6	86.5	88.9	**85.3**	**84.6**	**59.1**	**85.0**	80.4	**87.4**	**81.2**

Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	~ 1000 × 600
Fast YOLO	52.7	155	1	98	448 × 448
YOLO (VGG16)	66.4	21	1	98	448 × 448
SSD300	74.3	46	1	8732	300 × 300
SSD512	76.8	19	1	24564	512 × 512
SSD300	74.3	59	8	8732	300 × 300
SSD512	76.8	22	8	24564	512 × 512

Bảng tóm tắt về hiệu suất tốc độ trong Frame Per Second

4.5 Kết quả thực nghiệm

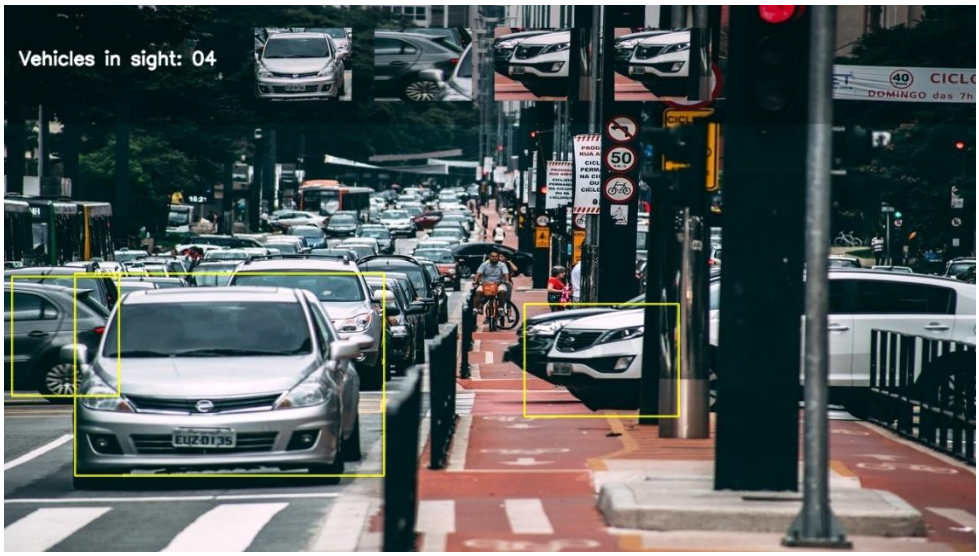


Ảnh dùng SSD với minimum confidence = 0.3 và 0.5

Ảnh dùng SSD với minimum confidence = 0.3 và 0.5

Ảnh dùng SSD với minimum confidence = 0.3

Ảnh dùng SSD với minimum confidence = 0.5



Ảnh dùng SSD với minimum confidence = 0.3

Ảnh dùng SSD với minimum confidence = 0.5

Chương trình hiện tại chỉ xử lý được video ở định dạng mp4.

Chương 5: Kết luận và hướng phát triển
5.1 Kết luận

5.1.1 Ưu điểm
Lập trình: Nâng cao được trình độ lập trình.
Tìm hiểu được một khía cạnh của lĩnh vực xe tự hành đó là nhận diện xe.
Phát triển kỹ năng làm việc nhóm, tăng khả năng tự học.

5.1.2 Khuyết điểm
Do phạm vi lý thuyết rộng và liên quan đến nhiều nghiên cứu trước đó nên việc tìm hiểu về lý thuyết về lĩnh vực này có nhiều điểm còn chưa rõ ràng. Cần phải có thêm thời gian để có thể tìm hiểu sâu hơn.
Yêu cầu về phần cứng khá cao, tốc độ của phần cứng chỉ hoạt động ở mức chấp nhận được.
Thời gian để tạo ra tập dữ liệu còn rất lâu.
Phần mềm vẫn chưa thể nhận diện đúng hết tất cả các đối tượng.

5.2 Hướng phát triển
Để triển khai trong thực tế, đề tài đòi hỏi cần phải cải tiến hơn nữa. Để có thể hoàn thiện đề tài hơn ta có thể phát triển theo những hướng như:

Nâng cấp CPU cho phần cứng.
Hướng tới việc liên kết qua server để các xe có thể giao tiếp với nhau.
Ngoài nhận diện đối tượng là xe, có thể mở rộng để nhận diện cái vật cản trên đường.

Tài liệu tham khảo

1. Bảo Anh. “Công nghệ xe tự hành của Việt Nam: Còn nhiều thách thức”: http://congluan.vn/khoa-hoc-cong-nghe/o-to-xe-may/cong-nghe-xe-tu-hanh-cua-viet-nam-con-nhieu-thach-thuc-39614
2. Trần Duy Quang. “Histogram of Oriented Gradients (HOG)”: https://vi.scribd.com/doc/138024664/Histogram-of-Oriented-Gradients-HOG
3. Eddie Forson. “Understanding SSD MultiBox—Real-Time Object Detection In Deep Learning”: https://towardsdatascience.com/understanding-ssd-multibox-real-time-object-detection-in-deep-learning-495ef744fab
4. Hao Gao. “Understand Single Shot MultiBox Detector (SSD) and Implement It in Pytorch”: https://medium.com/@smallfishbigsea/understand-ssd-and-implement-your-own-caa3232cd6ad
5. Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu , Alexander C. Berg. “Single Shot MultiBox Detector”: https://www.cs.unc.edu/~wliu/papers/ssd.pdf