MỤC LỤC

Mục tiêu nghiên cứu

Mục tiêu muốn xây dựng một hệ thống nhận diện và phát hiện đồ vật trong phòng nhằm tạo dựng 1 giải pháp hỗ trợ cho kiểm kê. Đề tài thực hiện các việc thu thập tập dữ liệu đồ vật, tiến hành phân loại và đánh giá. Sau đó cài đặt thuật toán kết hợp thư viện có sẵn như yolo, OpenCV. Thực hiện việc đánh giá trên các loại máy tính có cấu hình khác nhau. Cải thiện hiệu xuất hoặc độ chính xác của đối tượng khi có kết quả thử nghiệm.

1. Trình bày tổng quan

1.1. Nghiên cứu trong nước

Đề tài sử dụng yolo để nhận diện vật thể qua video :

Đề tài này đề cập đến các thuật toán của bài toán nhận diện vật thể cùng cách yolov3 hoạt động cũng như cài đặt môi trường để có thể chạy được mô hình yolov3 và nhận diện qua video . Cùng với đó là hướng dẫn cách train để nhận diện vật thể mới không có sẵn trong mô hình pre-train .

Link tham khảo : https://123doc.net//document/5433032-su-dung-deeplearning-yolo-de-phan-loai-doi-tuong-trong-video.htm

Đề tài nhận diện vật thể qua ảnh và video với yolo :

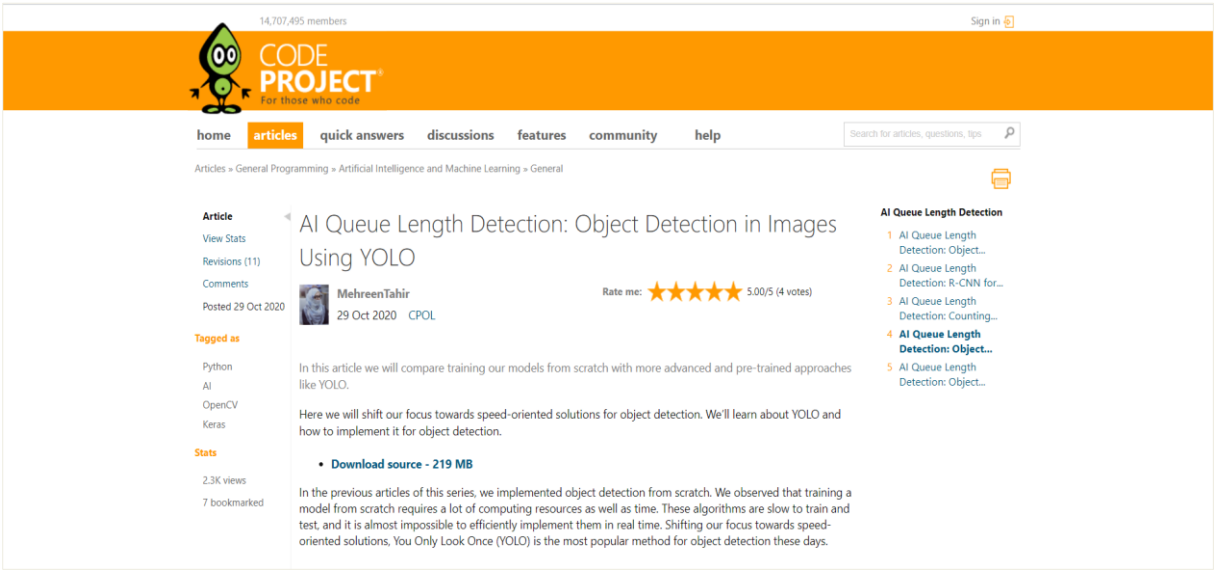
Đề tài này đề cập đến các thuật toán của bài toán nhận diện vật thể cùng cách yolov3 hoạt động cũng như cài đặt môi trường để có thể chạy được mô hình Yolov3 và nhận diện qua video và hình ảnh . Bên cạnh đó là giải thích dòng lệnh sử dụng trong mô hình một cách chi tiết .

Link tham khảo : http://daotao.sict.udn.vn/uploads/2020/08/1596612862-17it2-nguyenvu-nhandangdoituongbangyolo-1.docx

1.2. Nghiên cứu ngoài nước

Nhận diện vật thể qua ảnh với Yolo và Opencv:

Bài báo đã hướng dẫn khá chi tiết cách cài đặt và sử dụng yolo cũng như OpenCV để nhận diện vật thể trong ảnh . Hướng dẫn đã được 4 lượt đánh giá 5 sao trên trang Codeproject .Tuy nhiên trong bài này tác giả chưa đề cập đến cách train mô hình cũng như cách phát hiện vật thể real-time.

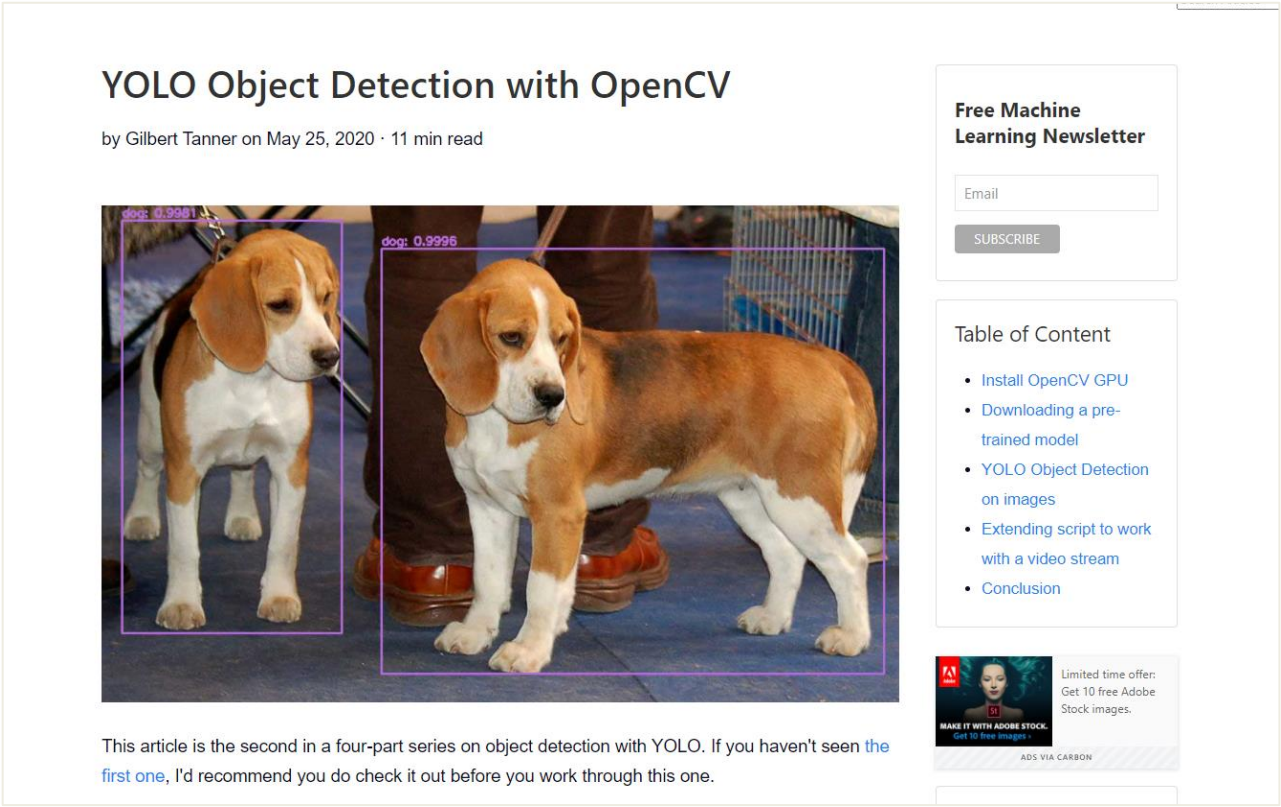


hình 1- Bài báo nhận diện vật thể trong ảnh

Link kham khảo: https://www.codeproject.com/Articles/5283661/AI-Queue-Length-Detection-Object-Detection-in-Imag

Nhận diện vật thể qua video và ảnh sử dụng Opencv và Yolo:

Đây là dự án được Gilbert Tanner công bố vào 25/05/2020 . Dự án nói về cách dùng Pre-Trained Yolo để nhận diện vật thể trong ảnh và video . Cùng với đó là hướng dẫn code chi tiết từng phần trong dự án . Cùng với đó là một và demo nhỏ nhỏ của dự án trên video chứa con người



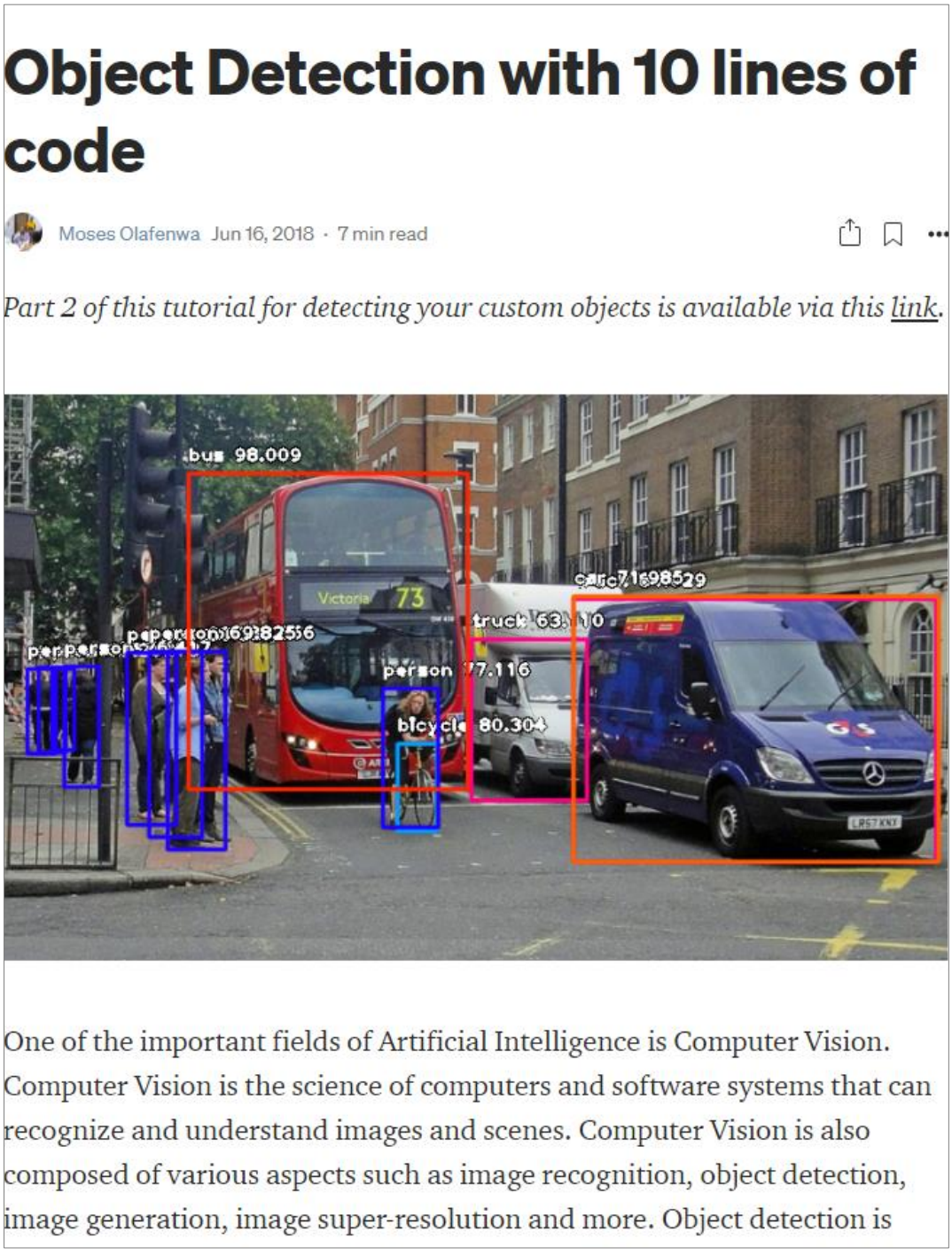
hình 2- Bài báo nhận diện vật thể qua video

Link kham khảo: https://gilberttanner.com/blog/yolo-object-detection-with-opencv

Project nhận diện vật thể với 10 dòng code với thư viện ImageAI:

Trong bài viết này đã sử dụng đến một thư viện do Moses Olafenwa phát triển với mục đích nhận diện vật thể trong hình ảnh và video . Bài viết đã sử dụng thư viện để nhận diện vật thể trong hình ảnh chỉ với 10 dòng code .ImageAI là một thư viện tuyệt vời , bài đã được rất nhiều lượt vỗ tay cũng như bình luận . Tuy nhiên điểm yếu của thư viện này là không có khả năng áp dụng cho thực tế vì tốc độ xử

lý rất chậm.



hình 3- Bài báo nhận diện vật thể với imageAI

Link kham khảo: https://towardsdatascience.com/object-detection-with-10-lines-of-code-d6cb4d86f606

1.3. Kết luận

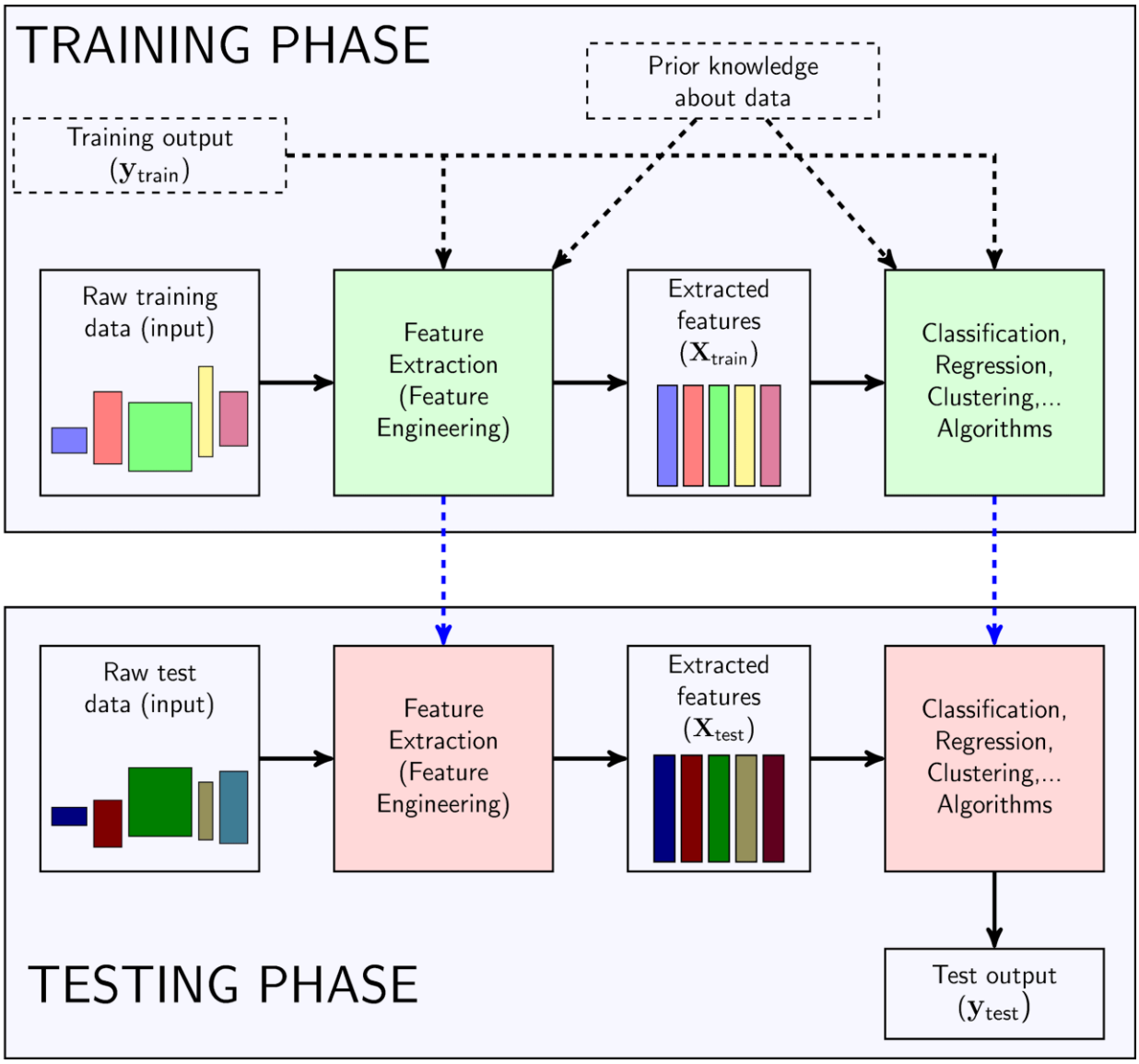
Các đề tài cũng đã đề cập đến Yolo và bài toán nhân diện và phân loại vật thể

cũng như đã nói rõ về các thuật toán hiện nay có thể áp dụng đề tài . Cùng với đó là có giới thiệu những thư viện có sẵn có thể áp dụng cho bài toán cho người mới tiếp cận . Hạn chế duy nhất của các đề tài này là chưa nói về cách cho người ít kiến thức về công nghệ thông tin có thể sử dụng ví dụ áp dụng vào 1 trang web để mọi người sử dụng hay làm ứng dụng điện thoại …

2. Cơ sở lý thuyết

2.1. Giới thiệu học máy (ML)

2.1.1. Học Máy - Machine Learning



hình 4-Sơ đồ chung của mô hình học máy

Học máy là kỹ thuật phổ biến nhất để dự đoán tương lai hoặc phân loại thông tin để giúp chúng ta đưa ra quyết định cần thiết. Các thuật toán Học Máy được đào tạo dựa trên các trường hợp hoặc ví dụ mà qua đó máy tính sẽ học hỏi từ kinh nghiệm trong quá khứ và cũng phân tích dữ liệu lịch sử. Do đó, khi luyện tập lặp đi lặp lại các ví dụ, nó có thể xác định các mẫu để đưa ra dự đoán về tương lai .

2.1.2. Vì Sao Lại Sử Dụng Học Máy - Machine Learning

Thế giới ngày nay ngày càng phát triển và nhu cầu, yêu cầu của con người cũng vậy. Hơn nữa, chúng ta đang chứng kiến cuộc cách mạng công nghiệp 4.0 về dữ liệu. Để có được những thông tin chi tiết , có ý nghĩa từ dữ liệu này và học hỏi từ cách mà mọi người và hệ thống giao tiếp với dữ liệu, chúng ta cần các thuật toán tính toán có thể xáo trộn dữ liệu và cung cấp cho chúng ta kết quả có lợi theo nhiều cách khác nhau. Học máy đã được áp dụng trong các ngành như y học, chăm sóc sức khỏe, sản xuất, ngân hàng và một số ngành khác , làm cho ngành này có tiến bộ vượt bậc . Do đó, Học máy đã trở thành một phần thiết yếu của nền công nghiệp hiện đại. Dữ liệu đang mở rộng theo cấp số nhân và để khai thác sức mạnh của dữ liệu này,chúng ta cần một thứ có thể tính toán thông tin, Học máy đã thêm một khía cạnh khác vào cách chúng ta nhận thức thông tin. Học máy đang được sử dụng ở mọi nơi. Các thiết bị điện tử bạn sử dụng, các ứng dụng là một phần trong cuộc sống hàng ngày của bạn đều được hỗ trợ bởi các thuật toán học máy mạnh mẽ.

Ví dụ về Học máy - Google có thể cung cấp cho bạn các kết quả tìm kiếm thích hợp dựa trên thói quen duyệt web. Tương tự, Netflix có khả năng đề xuất các bộ phim hoặc chương trình mà bạn muốn xem dựa trên các thuật toán máy học thực hiện các dự đoán dựa trên lịch sử xem của bạn. Hơn nữa, học máy đã tạo điều kiện thuận lợi cho việc tự động hóa các tác vụ dư thừa đã làm mất đi nhu cầu lao động chân tay. Tất cả những điều này đều có thể thực hiện được do lượng dữ liệu khổng lồ mà bạn tạo ra hàng ngày. Học máy tạo điều kiện thuận lợi cho một số phương pháp luận để hiểu dữ liệu này và cung cấp cho bạn kết quả ổn định và chính xác.

2.1.3. Các Loại Học Máy ?

2.1.3.1. Học tập có giám sát

Trong Học tập có giám sát, tập dữ liệu mà chúng tôi đào tạo mô hình của mình được gắn nhãn. Có một bản đồ đầu vào và đầu ra rõ ràng và khác biệt. Dựa trên các đầu vào ví dụ, mô hình có thể được đào tạo trong các trường hợp. Một ví dụ về học có giám sát là lọc thư rác. Dựa trên dữ liệu được gắn nhãn, mô hình có thể xác định xem dữ liệu đó có phải là thư rác hay thịt nguội hay không. Đây là một hình thức đào tạo dễ dàng hơn. Lọc thư rác là một ví dụ của loại thuật toán học máy này.

2.1.3.2. Học tập không giám sát

Trong Học tập không giám sát, không có dữ liệu được gắn nhãn. Thuật toán xác định các mẫu trong tập dữ liệu và học chúng. Thuật toán nhóm dữ liệu thành các cụm khác nhau dựa trên mật độ của chúng. Sử dụng nó, người ta có thể thực hiện trực quan hóa trên dữ liệu chiều cao. Một ví dụ về loại thuật toán học máy này là Phân tích thành phần nguyên tắc. Hơn nữa, K-Means Clustering là một loại hình học không giám sát khác trong đó dữ liệu được nhóm lại thành các nhóm có thứ tự giống nhau. Quá trình học tập trong Học tập không giám sát chỉ dựa trên cơ sở tìm kiếm các mẫu trong dữ liệu. Sau khi tìm hiểu các mô hình, mô hình sẽ đưa ra kết luận.

2.1.3.3. Học tăng cường

Học tăng cường là một loại Thuật toán học máy mới nổi và phổ biến nhất. Nó được sử dụng trong các hệ thống tự hành khác nhau như ô tô và robot công nghiệp. Mục đích của thuật toán này là đạt được mục tiêu trong một môi trường năng động. Nó có thể đạt được mục tiêu này dựa trên một số phần thưởng được cung cấp bởi hệ thống. Nó được sử dụng nhiều nhất trong việc lập trình robot để thực hiện các hành động tự động. Nó cũng được sử dụng để tạo ra những chiếc xe tự lái thông minh. Chúng ta hãy xem xét trường hợp điều hướng bằng robot. Hơn nữa, hiệu quả

có thể được cải thiện khi thử nghiệm thêm với tác nhân trong môi trường của nó. Đây là nguyên tắc chính đằng sau việc học tăng cường. Có các chuỗi hành động tương tự trong một mô hình học tăng cường.

2.2. Phân loại ML và DL

Học Sâu (Deep Learning) và Học Máy (Machine Learning) là hai công nghệ thịnh hành nhất trên thế giới hiện nay. Các công nghệ này thường được sử dụng thay thế cho nhau. Mặc dù Deep Learning là một tập hợp con của học máy, nhưng nhiều người bị nhầm lẫn giữa hai thuật ngữ này. Vì cậy đề tài sẽ so sánh hai công nghệ này để làm rõ sự khác biệt giữa chúng .

2.2.1. Định nghĩa nhanh về hai công nghệ :

Học máy là một tập hợp con của trí tuệ nhân tạo liên quan đến việc tạo ra các thuật toán có thể tự thay đổi mà không cần sự can thiệp của con người để có được kết quả mong muốn - bằng cách tự cung cấp thông tin thông qua dữ liệu có cấu trúc.

Học sâu là một tập hợp con của học máy trong đó các thuật toán được tạo ra và hoạt động tương tự như học máy, nhưng có nhiều cấp độ của các thuật toán này, mỗi cấp độ cung cấp một cách diễn giải khác nhau về dữ liệu mà nó truyền tải. Mạng thuật toán này được gọi là mạng nơron nhân tạo. Nói một cách dễ hiểu, nó giống với các kết nối thần kinh tồn tại trong não người.

2.2.2. Sự khác nhau của hai công nghệ

Sự khác biệt chính giữa Học sâu và Học máy là do cách dữ liệu được trình bày trong hệ thống. Các thuật toán Học máy hầu như luôn yêu cầu dữ liệu có cấu trúc, trong khi mạng Học sâu dựa trên các lớp ANN (mạng nơ-ron nhân tạo).

Các thuật toán học máy được thiết kế để "học" hành động bằng cách hiểu dữ liệu được gắn nhãn và sau đó sử dụng nó để tạo ra kết quả mới với nhiều bộ dữ liệu

hơn. Tuy nhiên, khi kết quả không chính xác, cần phải dạy lại cho máy tính hiểu.

Mạng Học sâu không yêu cầu sự can thiệp của con người, vì các lớp đa cấp trong mạng thần kinh đặt dữ liệu trong một hệ thống phân cấp của các khái niệm khác nhau, cuối cùng chúng sẽ học hỏi từ những sai lầm của chính chúng. Tuy nhiên, ngay cả chúng cũng có thể sai nếu chất lượng dữ liệu không đủ tốt.

Dữ liệu quyết định mọi thứ. Đó là chất lượng của dữ liệu cuối cùng quyết định chất lượng của kết quả.

2.3. Tìm hiểu thư viện OpenCv và Tầm Nhìn Máy Tính (Computer Vision)

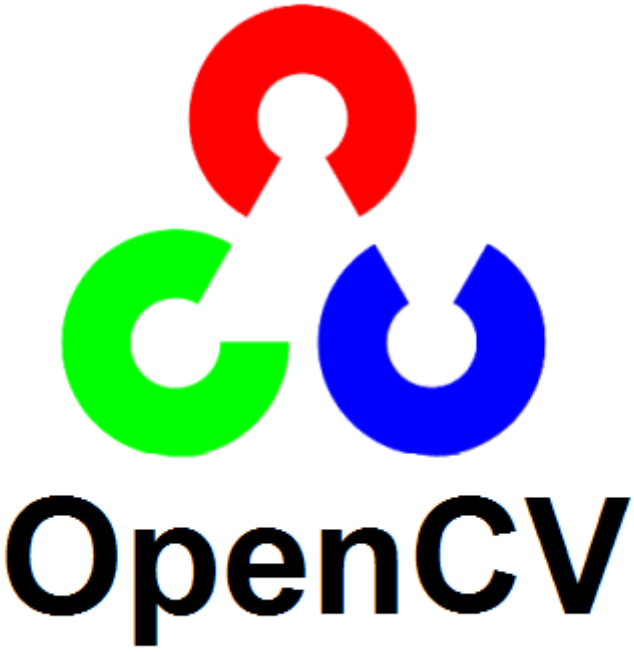
2.3.1. OpenCV

OpenCV được bắt đầu tại Intel vào năm 1999 bởi Gary Bradsky, và bản phát hành đầu tiên ra mắt vào năm 2000. Vadim Pisarevsky cùng với Gary Bradsky quản lý nhóm OpenCV phần mềm của Intel. Năm 2005, OpenCV được sử dụng trên Stanley, chiếc xe đã giành chiến thắng trong cuộc thi DARPA Grand Challenge 2005. Sau đó, sự phát triển tích cực của nó tiếp tục dưới sự hỗ trợ của Willow Garage với Gary Bradsky và Vadim Pisarevsky dẫn đầu dự án. OpenCV hiện hỗ trợ vô số thuật toán liên quan đến Thị giác máy tính và Học máy và đang mở rộng từng ngày. OpenCV hỗ trợ nhiều ngôn ngữ lập trình như C, Python, Java, v.v. và có sẵn trên các nền tảng khác nhau bao gồm Windows, Linux, OS X, Android và iOS. Các giao diện cho hoạt động GPU tốc độ cao dựa trên CUDA và OpenCL cũng đang được phát triển tích cực. OpenCV-Python là API Python cho OpenCV, kết hợp những phẩm chất tốt nhất của OpenCV C API và ngôn ngữ Python.

2.3.1.1. OpenCV-Python

OpenCV-Python là một thư viện liên kết Python được thiết kế để giải quyết các vấn đề về thị giác máy tính.

Python là một ngôn ngữ lập trình có mục đích chung do Guido van Rossum bắt đầu và trở nên rất phổ biến nhanh chóng, chủ yếu là vì tính đơn giản và dễ đọc mã của nó. Nó cho phép lập trình viên thể hiện ý tưởng trong ít dòng code hơn mà không làm giảm khả năng đọc.



hình 5- Logo OpenCV

So với các ngôn ngữ như C / C ++, Python chậm hơn. Điều đó nói rằng, Python có thể được mở rộng dễ dàng với C / C ++, cho phép chúng ta viết mã tính toán chuyên sâu bằng C / C ++ và tạo trình bao bọc Python có thể được sử dụng như các mô-đun Python. Điều này mang lại cho chúng ta hai lợi thế: thứ nhất, chạy nhanh như C / C ++ gốc (vì nó là mã C ++ thực tế hoạt động trong nền) và thứ hai, việc code bằng Python dễ dàng hơn C / C ++. OpenCV-Python là một trình bao bọc Python để triển khai OpenCV C ++ ban đầu.

OpenCV-Python sử dụng Numpy, là một thư viện được tối ưu hóa cao cho các phép toán số với cú pháp kiểu MATLAB. Tất cả các cấu trúc mảng OpenCV được chuyển đổi thành và từ mảng Numpy. Điều này cũng giúp dễ dàng tích hợp với các thư viện khác sử dụng Numpy như SciPy và Matplotlib.

2.3.2. Tầm Nhìn Máy Tính

Computer Vision hay Tầm Nhìn Máy Tính là một lĩnh vực liên ngành liên quan đến cách thức máy tính có thể tạo ra để đạt sự hiểu biết cao cấp từ hình ảnh hoặc video kỹ thuật số . Từ quan điểm của kỹ thuật , nó tìm cách tự động hóa các nhiệm vụ mà hệ hệ thống thị giác của người có thể thực hiện . Tầm nhìn máy tính liên quan đến việc tự động trích xuất , phân tích và hiểu thông tin hữu ích từ hình ảnh hoặc frame hình video . Nó liên quan đến việc phát triển cơ sở lý thuyết và thuật toán để đạt được trực quan tự động . Là một môn khoa học , Tầm nhìn máy tính liên quan đến lý thuyết đằng sau các hệ thống nhân tạo trích xuất thông tin từ hình ảnh .

2.4. Phân loại Detect Object và Image Classification

Nhận diện đối tượng và phân loại đối tượng là hai giải pháp rất quan trọng trong lĩnh vực thị giác máy tính (computer vision). Những kỹ thuật này giúp máy tính giúp máy tính hay các loại máy móc khác có thể xác định đối tượng và môi trường thời gian thực (real-time) qua các thiết bị có thể ghi hình .

Hai kỹ thuật này rất hay gây ra dễ hiểu lầm vì tính chất xoay quanh việc xác định các đối tượng trong ảnh hay frame hình . Vậy thì 2 kỹ thuật này ruốc cuộc là gì ? Và chúng khác nhau như thế nào ?

2.4.1. Phân Loại Đối Tượng

Có thể hiểu đơn giản , Phân Loại đối tượng là một kỹ thuật dùng để phân loại hoặc dự đoán lớp của một đối tượng của trong ảnh . Mục tiêu của kỹ thuật này là trích rút chính xác đặc trưng trong hình ảnh.

2.4.1.1. Cách Thức Hoạt Động Của Kỹ Thuật Phân Loại :

hình 6- Hình ảnh về kỹ thuật phân loại

Nói chung, các kỹ thuật phân loại ảnh có thể được phân loại thành các bộ phân loại tham số và không tham số hoặc có giám sát và không giám sát cũng như phân loại cứng và mềm. Đối với phân loại có giám sát, kỹ thuật này mang lại kết quả dựa trên ranh giới quyết định được tạo ra, chủ yếu dựa vào đầu vào và đầu ra được cung cấp trong khi đào tạo mô hình. Tuy nhiên, trong trường hợp phân loại không được giám sát, kỹ thuật cung cấp kết quả dựa trên việc phân tích tập dữ liệu đầu vào của chính nó; các tính năng không được cung cấp trực tiếp cho các mô hình. Các bước chính liên quan đến kỹ thuật phân loại ảnh là xác định một hệ thống phân loại phù hợp, trích xuất tính năng, chọn mẫu huấn luyện tốt, xử lý trước ảnh và lựa chọn phương pháp phân loại thích hợp, xử lý sau phân loại, và cuối cùng là đánh giá độ chính xác tổng thể. Trong kỹ thuật này, các đầu vào thường là hình ảnh của một đối tượng cụ thể, chẳng hạn như con thỏ trong hình trên và đầu ra là các lớp

được dự đoán xác định và khớp với các đối tượng đầu vào. Mạng nơ-ron hợp hiến (CNN) là mô hình mạng nơ-ron phổ biến nhất được sử dụng cho bài toán phân loại hình ảnh.

2.4.1.2. Các Loại Kỹ Thuật Phân Loại Ảnh Có Giám Sát

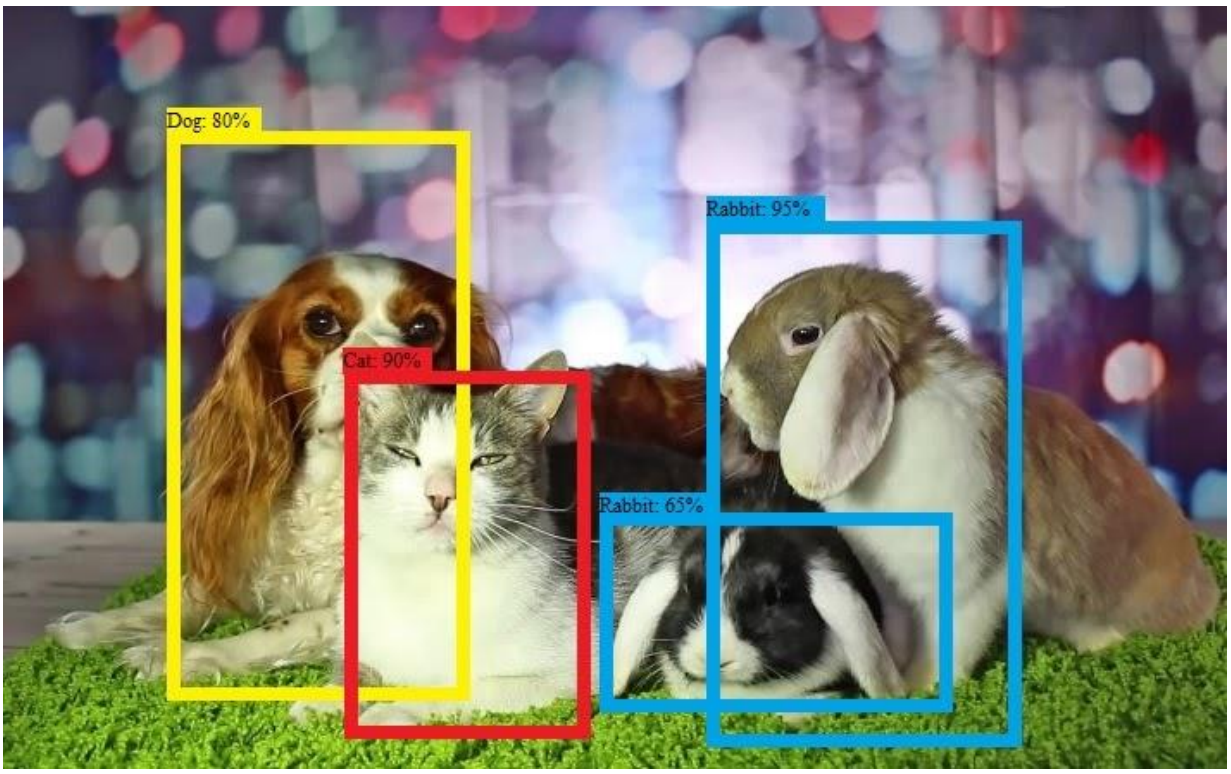
Các kỹ thuật phân loại ảnh có giám sát bao gồm kỹ thuật song song, bộ phân loại khoảng cách tối thiểu, bộ phân loại khả năng xảy ra tối đa, và các loại các. Trong một bài báo nghiên cứu, các nhà nghiên cứu đã đề cập đến một số loại kỹ thuật phân loại hình ảnh như đã đề cập đến dưới đây:

- Phân loại hình ảnh dựa trên thông tin thu được từ các cảm biến khác nhau
- Phân loại hình ảnh dựa trên bản chất của mẫu đào tạo được sử dụng trong phân loại
- Phân loại hình ảnh dựa trên cơ sở của các tham số khác nhau được sử dụng trên dữ liệu
- Phân loại hình ảnh dựa trên bản chất của thông tin pixel được sử dụng trên dữ liệu
- Phân loại hình ảnh dựa trên số lượng đầu ra được tạo cho mỗi phần tử dữ liệu không gian
- Phân loại hình ảnh dựa trên bản chất của thông tin không gian

2.4.1.3. Nhược Điểm Của Kỹ Thuật Phân Loại

Trong kỹ thuật phân loại ảnh có giám sát hay không thì nhược điểm là thời lượng đào tạo mô hình lớn và không phù hợp để xử lý big data.

2.4.2. Nhận Diện Đối Tượng



hình 7-Hình Ảnh Nhận Diện Đối Tượng

Kỹ Thuật Nhận Diện Đối Tượng là xác định vị trí của đối tượng trong hình ảnh nhất định hay frame hình từ video , camera . Chẳng hạn như khoanh vùng đối tượng và cho biết chúng thuộc lớp nào . Có thể nói đơn giản , nhận diện đối tượng là kỹ thuật phân loại ảnh , bên cạnh việc phân loại , kỹ thuật này còn cho biết vị trí chính xác từng đối tượng trong bức ảnh .

Kỹ thuật này có khả năng tìm kiếm một lớp đối tượng cụ thể , chẳng hạn như người , bàn ,ghế … và được sử dụng thành công trong hệ thống xử lý ảnh hay gần đây nó đã được áp dụng thành công trong video .

Các kỹ thuật phát hiện đối tượng có thể áp dụng để nhận diện khuôn mặt , phát hiện phương tiện giao thông , phát hiện biển báo giao thông …

2.4.2.1. Các Thức Hoạt Động Của Kỹ Thuật Nhận Diện Đối Tượng

Luồng hoạt động của các mô hình phát hiện đối tượng truyền thống chủ yếu có

thể được chia thành ba giai đoạn, đó là lựa chọn khu vực thông tin, trích xuất tính năng và phân loại. Có một số mô hình dựa trên học sâu phổ biến để phát hiện đối tượng, đã được các tổ chức và học viện sử dụng để đạt được hiệu quả cũng như kết quả chính xác trong việc phát hiện đối tượng từ hình ảnh. Các mô hình phổ biến bao gồm MobileNet, You Only Live Once (YOLO), Fast-RCNN, RetinaNet, …

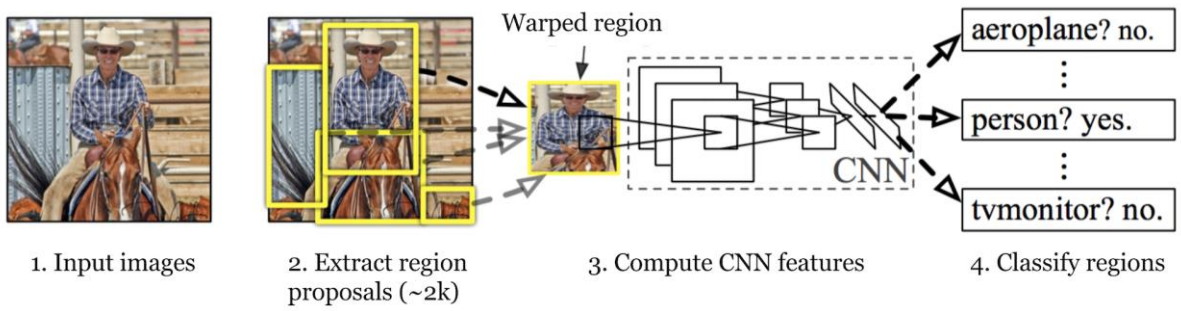
2.4.2.2. Nhược Điểm

Trong vài năm qua, đã đạt được thành công lớn trong một môi trường tối ưu cho vấn đề phát hiện đối tượng. Tuy nhiên, vấn đề vẫn chưa được giải quyết ở những nơi không tối ưu, đặc biệt, khi các đối tượng được đặt ở các tư thế tùy ý trong một môi trường lộn xộn và kín kẽ.

2.5. Các mô hình nhận diện vật thể hiện nay

2.5.1. R-CNN và các biến thể của chúng, bao gồm R-CNN, Fast R-CNN và Faster R-CNN

Với những bước nhảy lớn trong những bài toán về thị giác máy tính với mạng CNNs, một mô hình mạng mang tên R-CNNs đã được công bố để giải quyết bài toán Object Detection, Localization và Classification. Nhìn chung, R-CNN là một loại CNN đặc biệt có khả năng định vị và nhận biết được các đối tượng trong ảnh: đầu ra của mạng thường là một tập những bounding box bao quanh những đối tượng được nhận biết, cũng như nhãn của đối tượng được phân vào.



hình 8-Mô Hình Kiến Trúc RCNN

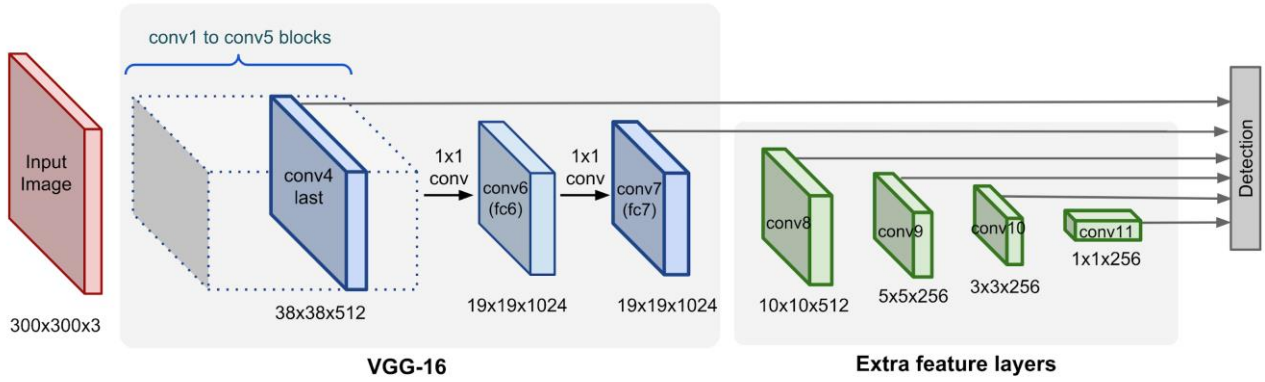
(Nguồn: Girshick et al., 2014)

Sau R-CNN, một vài mô hình mạng khác cũng được phát triển (Fast-RCNN, Faster-RCNN) dựa trên nó với mục đích chính là cải thiện thời gian huấn luyện và độ chính xác của mô hình với mong muốn áp dụng được với những bài toán chạy với thời gian thực. Tuy nhiên chúng vẫn tồn tại một số hạn chế :

- Việc huấn luyện mô hình vẫn quá cồng kềnh và tiêu tốn nhiều thời gian.
- Quá trình huấn luyện xảy ra trên nhiều phase.
- Mô hình mạng làm việc chậm so với thời gian thực.

2.5.2. SSD: Single Shot MultIBox Detector

Single Shot MultIBox Detector là một trong những nỗ lực đầu tiên trong việc sử dụng hệ thống phân cấp tính năng hình chóp của mạng nơ-ron phức hợp để phát hiện hiệu quả các đối tượng có kích thước khác nhau.



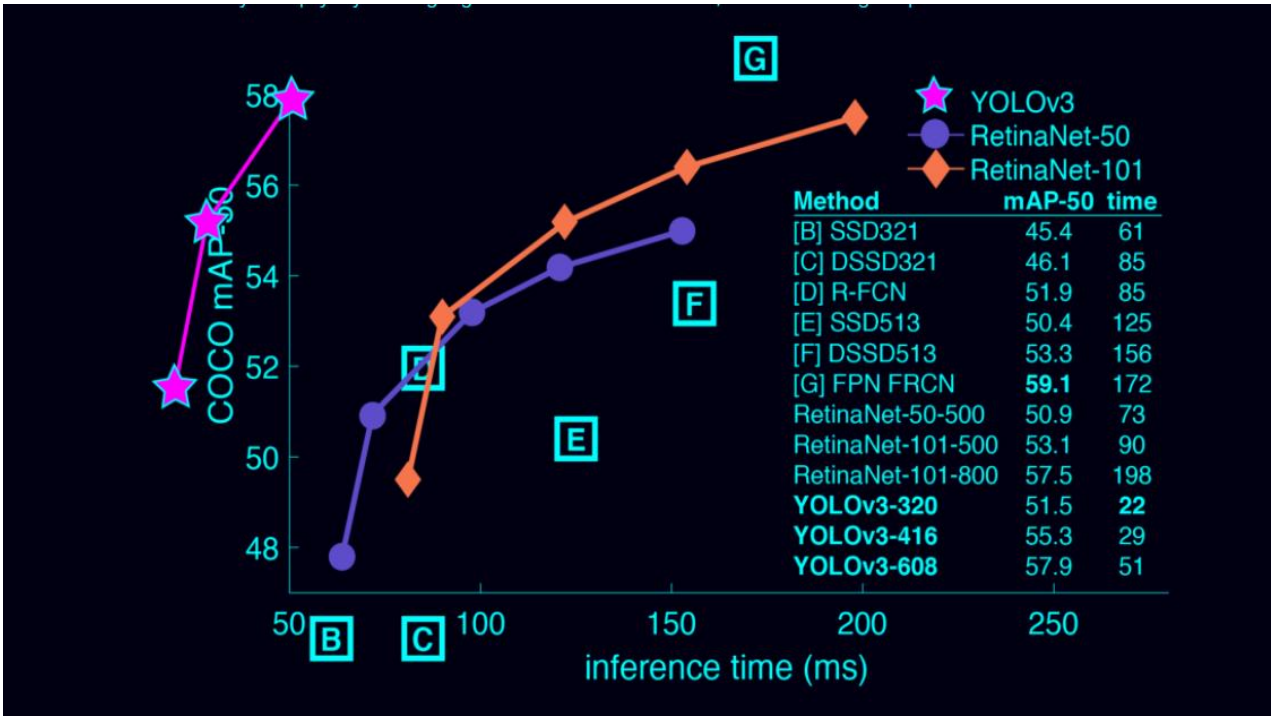
hình 9-Mô Hình SSD

(Nguồn : lilianweng)

SSD sử dụng mô hình VGG-16 được đào tạo trước trên ImageNet làm mô hình cơ sở để trích xuất các tính năng hình ảnh hữu ích. Trên đầu VGG16, SSD bổ sung thêm một số lớp tính năng chuyển đổi với kích thước giảm dần. Chúng có thể được xem như là một đại diện hình kim tự tháp của hình ảnh ở các tỷ lệ khác nhau. Bản đồ tính năng chi tiết lớn trực quan ở các cấp trước đó có khả năng nắm bắt tốt các

vật thể nhỏ và bản đồ tính năng chi tiết thô nhỏ có thể phát hiện tốt các vật thể lớn. Trong SSD, việc phát hiện xảy ra ở mọi lớp hình chóp, nhắm mục tiêu vào các đối tượng có kích thước khác nhau.

2.5.3. Yolo



Method	mAP-50	time
[B] SSD321	45.4	61
[C] DSSD321	46.1	85
[D] R-FCN	51.9	85
[E] SSD513	50.4	125
[F] DSSD513	53.3	156
[G] FPN FRCN	**59.1**	172
RetinaNet-50-500	50.9	73
RetinaNet-101-500	53.1	90
RetinaNet-101-800	57.5	198
YOLOv3-320	51.5	**22**
YOLOv3-416	55.3	29
YOLOv3-608	57.9	51

hình 10-Hình ảnh về tốc độ của từng mô hình.

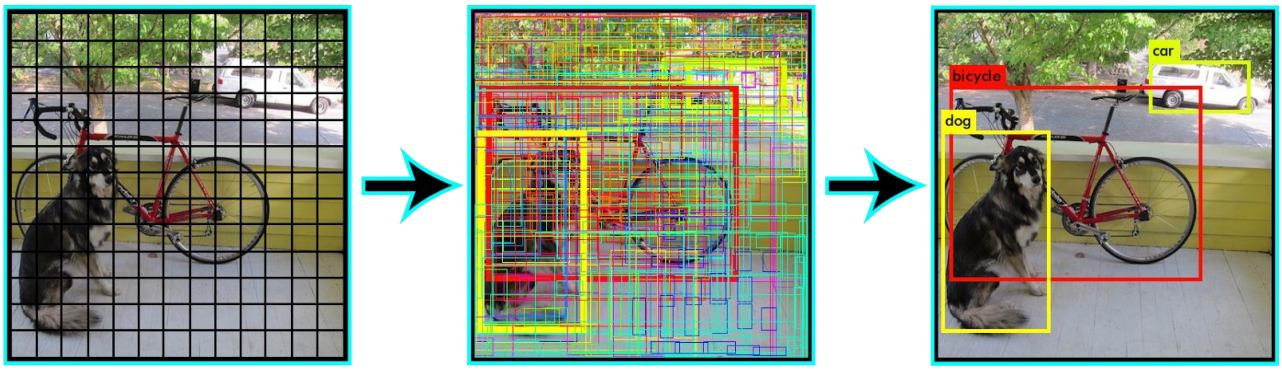
Mô hình có tốc độ có độ chính xác và tốc độ vượt trội , phổ biến và được áp dụng nhiều trên các thiết bị IoT .

2.6. Yolo

2.6.1. Yolo là gì

You only look once (YOLO) là một trong những mạng nơ-ron tích hợp phát hiện đối tượng phổ biến nhất (CNN). Sau khi Joseph Redmon et al. xuất bản bài báo

YOLO đầu tiên của họ vào năm 2015[1], các phiên bản tiếp theo được họ xuất bản vào năm 2016[2], 2017[3] và bởi Alexey Bochkovskiy vào năm 2020. , Yolo có nhiều ưu điểm cũng như sự tiến bộ hơn các mô hình detector trước , trong đó đáng chú ý là tốc độ nhanh chóng hơn nhiều trong dự đoán . Thậm chí có thể chạy tốt trên những IOT device như raspberry pi .



hình 11-Mô hình nhận diện của Yolo V3

Sự phát triển của mạng neural đang dần làm hiện thực hoá khái niệm chúng ta vẫn thường gọi là Computer Vision - Thị giác máy tính. Tuy nhiên, tại thời điểm ban đầu, việc sử dụng Neural Network vẫn còn gặp nhiều khó khăn. Khi bạn muốn phát hiện ra object trong một bức ảnh, sau đó đánh nhãn cho object đó, các phương pháp lúc bấy giờ quá chậm để phục vụ trong real-time, hoặc đòi hỏi thiết bị mạnh mẽ, đắt đỏ..... cho đến khi YOLO ra đời. YOLO và sau này là YOLOv2 có khả năng gán nhãn cho toàn bộ object trong khung hình với chỉ duy nhất một operation. Có thể nói YOLO đã xây dựng một hướng tiếp cận đầu tiên giúp đưa Object detection thực sự khả thi trong cuộc sống.

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

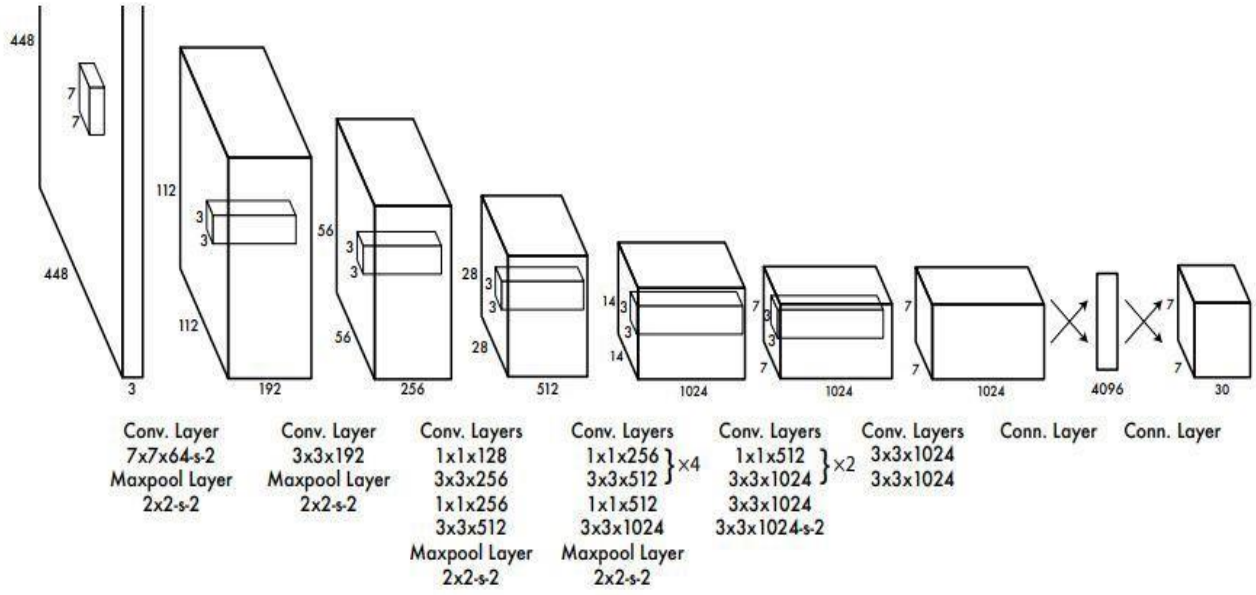
[1] 2015: https://arxiv.org/pdf/1506.02640.pdf
[2] 2016: https://arxiv.org/pdf/1612.08242.pdf
[3] 2017: https://arxiv.org/pdf/1804.02767.pdf

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

2.6.2. Các phiên bản

2.6.2.1. Yolo V1

Yolo v1 sử dụng framework Darknet được train trên tập ImageNet-1000. Nó không thể tìm thấy các object nhỏ nếu chúng xuất hiện dưới dạng một cụm. Phiên bản này gặp khó khăn trong việc phát hiện các đối tượng nếu hình ảnh có kích thước khác với hình ảnh được train.



hình 12-Hình ảnh cấu trúc Yolo V1

Yolo V1 CNN được mô tả trong hình 2. Nó có 24 lớp tích chập hoạt động như một bộ giải nén tính năng. Theo sau chúng là 2 lớp được kết nối đầy đủ có nhiệm vụ phân loại các đối tượng và hồi quy các hộp giới hạn. Đầu ra cuối cùng là tensor

7 x 7 x 30. YOLO CNN là một CNN đường dẫn đơn đơn giản tương tự như VGG19 . YOLO sử dụng các chập 1x1, sau đó là các chập 3x3 với nguồn cảm hứng từ CNN phiên bản 1 của Google. Kích hoạt ReLU rò rỉ được sử dụng cho tất cả các lớp ngoại trừ lớp cuối cùng (Lớp cuối cùng là lớp tuyến tính).

2.6.2.2. YoloV2 và Yolo9000

Yolov2 đặt tên là YOLO9000 đã được Joseph Redmon và Ali Farhadi công bố vào cuối năm 2016 và có mặt trong 2017 CVPR. Cải tiến chính của phiên bản này tốt hơn, nhanh hơn, tiên tiến hơn để bắt kịp faster R-CNN (phương pháp sử dụng Region Proposal Network), xử lý được những vấn đề gặp phải của Yolo V1.

Yolo v2 sử dụng vài trick để cải thiện quá trình huấn luyện và nâng cao performance. YOLOv2 cũng sử dụng Full-convolutional model như SSD, tuy nhiên vẫn train toàn bộ các bức ảnh thay vì chỉ hard negative. Theo hướng của Faster R-CNN, YOLOv2 sẽ điều chỉnh thông số bounding box trước, thay vì predict ngay w và h, tuy nhiên toạ độ (x, y) vẫn được dự đoán trực tiếp.

Cải thiện độ chính xác và chuẩn hóa hàng loạt, thêm chuẩn hóa hàng loạt trong các lớp chập. Điều này loại bỏ sự cần thiết phải bỏ (dropouts and pushes) học và đẩy mAP lên 2%. Phân loại độ phân giải cao.

Việc đào tạo Yolo bao gồm 2 giai đoạn. Đầu tiên, Yolo đào tạo một mạng phân loại như VGG16. Sau đó, Yolo thay thế các lớp được kết nối đầy đủ bằng một lớp chập và giữ lại từ đầu đến cuối để phát hiện đối tượng. Yolo huấn luyện bộ phân loại với các hình ảnh có kích thước 224 × 224, sau đó là các hình ảnh có kích thước 448 × 448 để phát hiện đối tượng. Yolo v2 bắt đầu với các hình ảnh 224 × 224 cho đào tạo phân loại nhưng sau đó thử lại bộ phân loại một lần nữa với các hình ảnh 448 × 448 sử dụng ít kỷ nguyên hơn nhiều. Điều này làm cho việc đào tạo máy dò dễ dàng hơn và di chuyển mAP lên 4%.

Sự thay đổi của YoloV2 so với Yolo V1

Batch Normalization : Giảm sự thay đổi giá trị unit trong hidden layer , do đó sẽ

cải thiện được tính ổn định của neural network .

Higher Resolution Classifier : Kích thước ảnh đầu vào trong Yolo V2 tăng từ 224*244 lên 448*448 .

Fine-Grained Features : Yolo V2 chia thành 13*13 grid cells , do đó có thể phát hiện được những vật thể nhỏ hơn, đồng thời cũng hiệu quả hơn với các vật thể lớn .

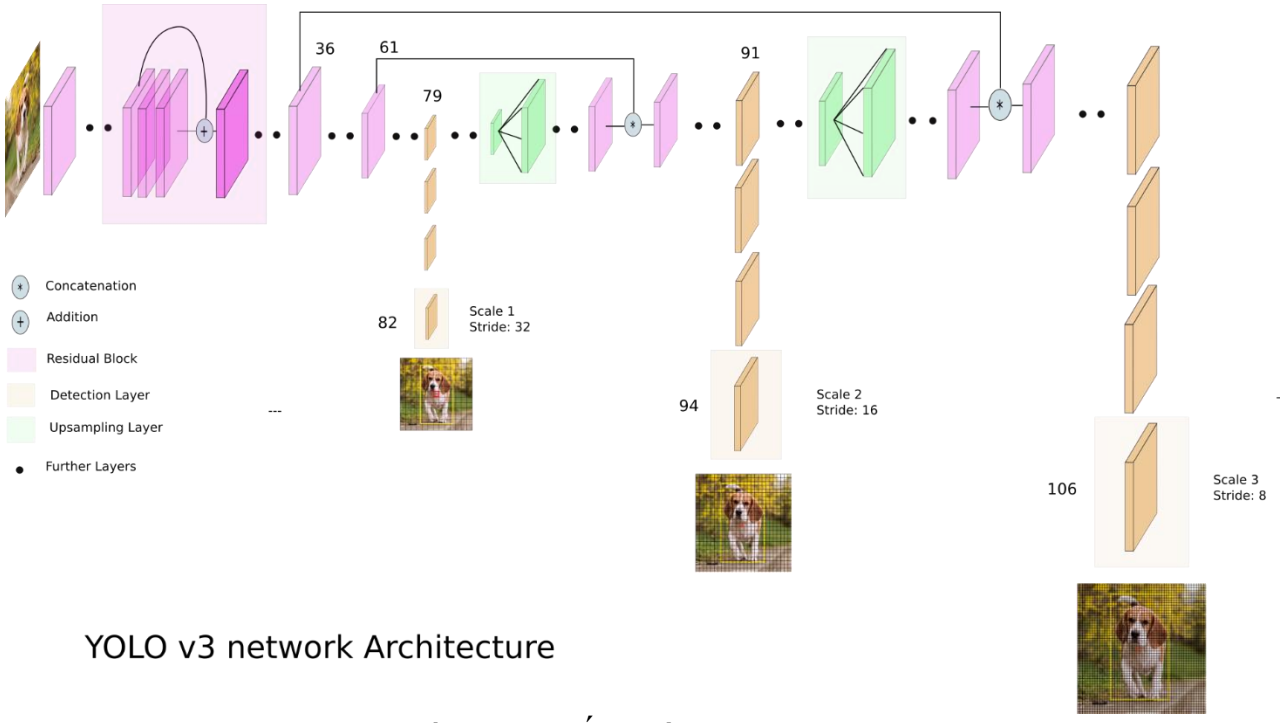
Darknet19 : YoloV2 sử dụng Darknet 19 với 19 Convolutional layers , 5 max pooling layers và sorfmax layers .

Multi-Scale Training : Điểm yếu của Yolo V1 là không phát hiện đối tượng có kích cỡ đầu vào khác nhau . Vì thế để giải quyết vấn đề này Yolo V2 đã train với kích thước ảnh ngẫu nhiên từ 320*320 đến 608*608.

2.6.2.3. YoloV3

Sau một thời gian cho ra mắt Yolo V2 , Yolo V2 được xem là mô hình nhanh nhất , mạnh nhất và chính xác nhất . Tuy nhiên các mô hình SSD ,RetinaNet phát triển làm cho YoloV2 không còn là mô hình chính xác nữa . Tuy nhiên nó vẫn là một trong những mô hình nhanh nhất.

Nhóm tác giả đã quyết định đánh đổi tốc độ này lấy sự chính xác ở bản Yolo V3 bằng cách gia tăng sự phức tạp của kiến trúc cơ bản Darknet.



hình 13-Kiến Trúc Yolo V3

Sự thay đổi của Yolo V3 so với Yolo V2

Logistic regression cho confidence score : Yolo V3 cung cấp Score cho bounding box (có chứa vật hay không) sử dụng logistic regression .

Thay softmax bằng các logistic classifier rời rạc: Yolo v3 sử dụng các logistic classifier thay vì softmax cho việc classify đối tượng.

Darknet 53: YOLO v3 sử dụng một biến thể của Darknet - Darknet 53 , ban đầu có mạng 53 lớp được đào tạo trên Imagenet. Đối với nhiệm vụ phát hiện, 53 lớp khác được xếp chồng lên nó, mang lại cho chúng ta một kiến trúc cơ bản hoàn toàn phức hợp 106 lớp cho YOLO v3. Đây cũng là lý do vì sao Yolo V3 chậm hơn Yolo V2

Feature Pyramid Networks(FPN) : Tăng để độ chính xác của mô hình bằng cách dò tìm được các vật thể nhỏ .

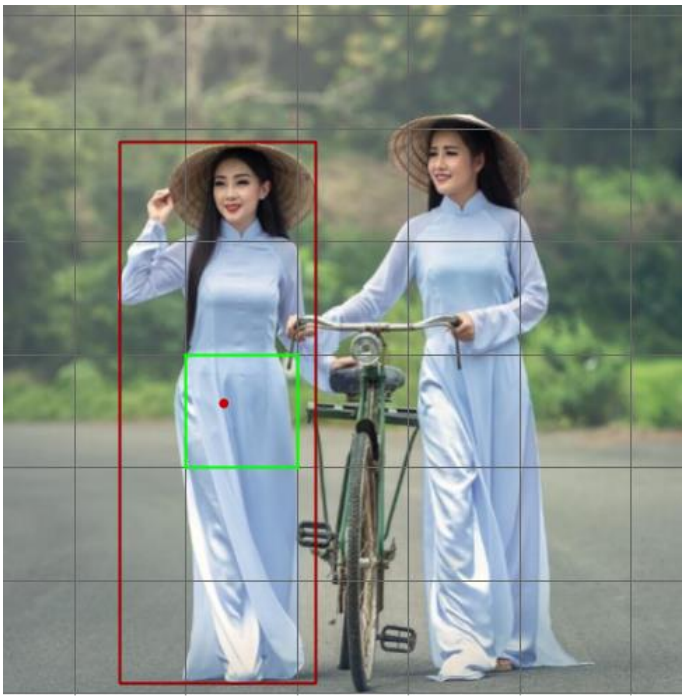
2.6.3. Cấu trúc Yolo

Một trong những ưu điểm mà Yolo đem lại đó là chỉ sử dụng thông tin toàn bộ

bức ảnh một lần và dự đoán toàn bộ object box chứa các đối tượng, mô hình được xây dựng theo kiểu end-to-end nên được huấn luyện hoàn toàn bằng gradient descent. Sau đây, đề tài sẽ trình bày chi tiết về mô hình YOLO:

2.6.3.1. Grid System

Ảnh sẽ được phân chia thành một ma trận ô vuông 7x7, mỗi ô vuông bao gồm một tập các thông tin mà mô hình phải dự đoán. Như hình dưới:

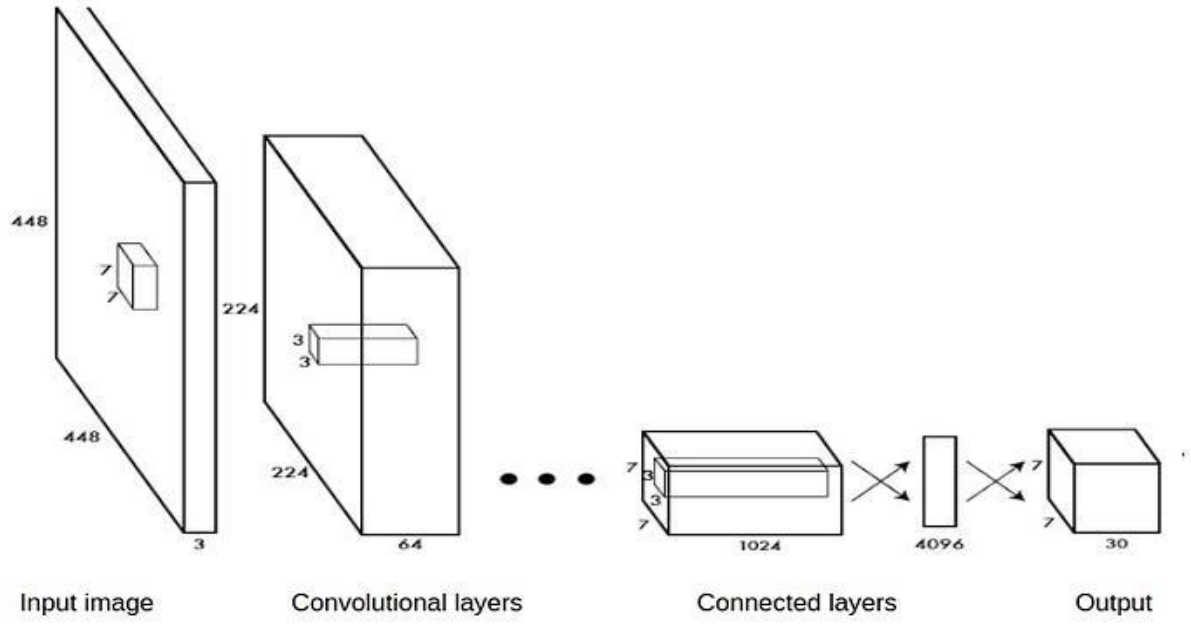


hình 14-Hình mô tả Grid của Yolo

Mỗi ô vuông sẽ chỉ chứa một đối tượng duy nhất. Tâm của đối tượng cần xác định nằm trong ô vuông nào thì ô vuông đó chứa đối tượng đó. Ví dụ tâm của cô gái nằm trong ô vuông màu xanh, do đó bắt buộc mô hình sẽ phải dự đoán được nhãn của ô vuông đó là cô gái. Lưu ý, cho dù phần ảnh cô gái có nằm ở ô vuông khác mà tâm không thuộc ô vuông đó thì vẫn không tính là chứa cô gái, ngoài ra, nếu có nhiều tâm nằm trong một ô vuông thì chúng ta vẫn chỉ gán một nhãn cho ô vuông đó thôi.Vì mỗi ô vuông chỉ chứa một đối tượng nên nếu có nhiều hơn một đối tượng thì mô hình cũng sẽ chỉ gán nhãn được một đối tượng. Nó làm cho chúng ta không thể nhận diện những object có tâm nằm cùng một ô vuông. Để khắc phục vấn đề này những mô hình Yolo sau này đã tăng kích cỡ ma trận lên để nhận diện

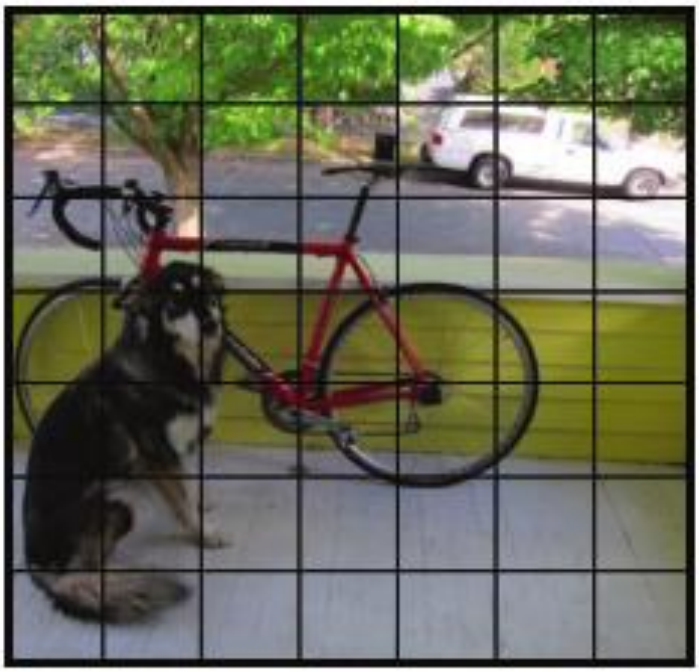
được nhiều vật thể hơn. Ngoài ra, khi nhận diện thì ảnh đầu vào phải là bội số của grid size.

Mỗi ô vuông chịu trách nhiệm dự đoán 2 boundary box của đối tượng. Mỗi boundary box dữ đoán có chứa object hay không và thông tin vị trí của boundary box gồm trung tâm boundary box của đối tượng và chiều dài, rộng của boundary box đó. Ví dụ ô vuông màu xanh cần dự đoán 2 boundary box chứa cô gái như hình



hình 15-Grid System

minh họa ở dưới. Một điều cần lưu ý, lúc cài đặt chúng ta không dự đoán giá trị pixel mà cần phải chuẩn hóa kích thước ảnh về đoạn từ [0-1] và dự đoán độ lệch của tâm đối tượng đến box chứa đối tượng đó. Ví dụ, chúng ta thay vì dự đoán vị trí pixel của điểm màu đỏ, thì cần dự đoán độ lệch a,b trong ô vuông chứa tâm object.

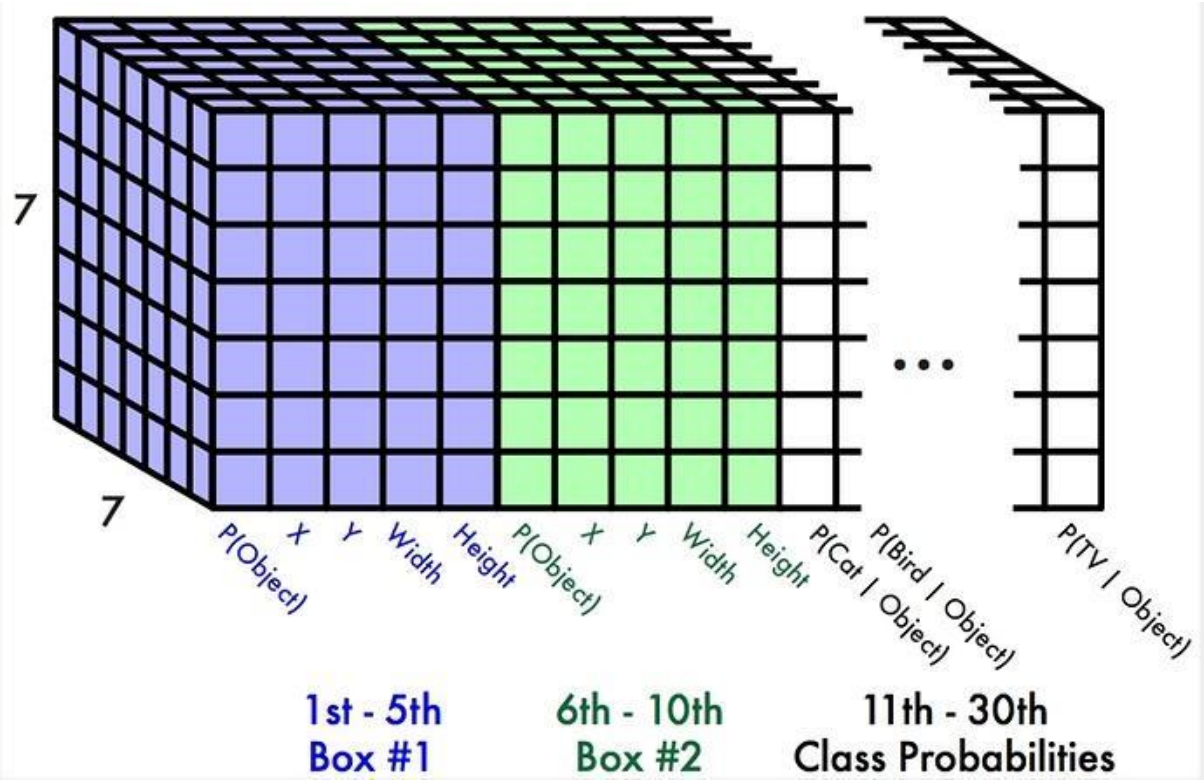


hình 16-Các box được chia nhỏ trong ảnh

Tổng hợp lại, với mỗi ô vuông chúng ta cần dự đoán các thông tin sau :

- Ô vuông có chứa đối tượng nào hay không?
- Dự đoán độ lệch 2 box chứa object so với ô vuông hiện tại
- Lớp của object đó

Như vậy ta có với mỗi ô vuông thì chúng ta cần dự đoán một vectơ sẽ có (nbox+4*nbox+nclass) chiều. Ví dụ, chúng ta cần dự đoán 2 box, và 3 lớp đối với mỗi ô vuông thì chúng sẽ có một ma trận 3 chiều 7x7x30 chứa toàn bộ thông tin cần thiết.



hình 17-Cách xử lý ảnh

2.6.3.2. CNN for YOLO Object Detection

Chúng ta đã cần biết phải dự đoán những thông tin nào đối với mỗi ô vuông, điều quan trọng tiếp theo là xây dựng một mô hình CNN có cho ra output với shape phù hợp theo yêu cầu của chúng ta, tức là gridsize x gridsize x (nbox+4*nbox+nclass). Ví dụ với gridsize là 7x7 là mỗi ô vuông dự đoán 2 boxes, và có 3 loại object tất cả thì chúng ta phải cần output có shape 7x7x13 từ mô hình CNN.

object 1?	object 2?	offset x1	offset y1	width 1	height 1	offset x2	offset y2	width 2	height 2	0	0	1

hình 18-CNN for YOLO Object Detection

Yolo sử dụng Linear Regression để dự đoán các thông tin ở mỗi ô vuông. Do đó, ở layer cuối cùng chúng ta sẽ không sử dụng bất kì hàm kích hoạt nào cả. Với hình ảnh đầu vào là 448x448, mô hình CNN có 6 tầng max pooling với size 2x2 sẽ giảm

64 lần kích thước ảnh xuống còn 7x7 ở output đầu ra. Đồng thời thay vì sử dụng tầng full connected ở các tầng cuối cùng, chúng ta có thể thay thế bằng tầng 1x1 convolution với 13 feature maps để output shape dễ dàng cho ra 7x7x13.

2.6.3.3. Loss function

Đã định nghĩa được những thông tin mà mô hình cần phải dự đoán, và kiến trúc của mô hình CNN. Bây giờ là lúc mà chúng ta sẽ định nghĩa hàm lỗi.

YOLO sử dụng hàm độ lỗi bình phương giữ dự đoán và nhãn để tính độ lỗi cho mô hình. Cụ thể, độ lỗi tổng của chúng ta sẽ là tổng của 3 độ lỗi con sau:

- Độ lỗi của việc dự đoán loại nhãn của object - Classification loss.
- Độ lỗi của dự đoán tạo độ cũng như chiều dài, rộng của boundary box - Localization loss.
- Độ lỗi của ô vuông có chứa object nào hay không - Confidence loss.

Chúng ta mong muốn hàm lỗi có những chức năng sau. Trong quá trình huấn luyện, mô hình sẽ nhìn vào những ô vuông có chứa object. Tăng classification score lớp đúng của object đó lên. Sau đó, cũng nhìn vào ô vuông đó, tìm boundary box tốt nhất trong 2 boxes được dự đoán. Tăng localization score của boundary box đó lên, thay đổi thông tin boundary box để gần đúng với nhãn. Đối với những ô vuông không chứa object, giảm confidence score và chúng ta sẽ không quan tâm đến classification score và localization score của những ô vuông này.

Tiếp theo, chúng ta sẽ đi lần lượt vào chi tiết ý nghĩa của các độ lỗi trên.

Các khái niệm

- $1_{i\,j}^{obj} = 1$ nếu box thứ j của ô thứ i có chứa object. Vì huấn luyện cần các image với ground-truth (vị trí của các objects) nên YOLO biết điểm trung tâm của từng object rơi vào ô nào trong grid 7x7.
- $1_{i\,j}^{noobj} = 1$ nếu box thứ j của ô thứ i không chứa object.
- $1_{i}^{obj} = 1$ nếu ô thứ i có chứa object
- $S^2 = 7 \times 7$, B = boxes_number = số box mỗi ô sẽ dự đoán, được cố định = 2
- $\lambda_{coord} = 5.0$, $\lambda_{noobj} = 0.5$
- $classes$: các lớp đối tượng cần được nhận dạng, ví dụ chó, mèo, oto...

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \quad (1)$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \quad (2)$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \quad (3)$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \quad (4)$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \quad (5)$$

- (1) Tính toán loss (tổn thất) của điểm trung tâm (x, y) cho hộp j của ô i nơi object tồn tại. Chú ý là $\hat{x}_i$ là tham số của tensor output của neural net còn x_i là của ground-truth. Tương tự cho tất cả các biến khác.
- (2) Tính toán tổn thất width và height của hộp j của ô i nơi object tồn tại.
- (3) Đối với các hộp j của ô i nơi object tồn tại, tính tổn thất của xác xuất object tồn tại. Chú ý C_i luôn = 1.
- (4) Đối với hộp j của ô i và nơi không có object, tính tổn thất của xác xuất này. Chú ý C_i luôn = 0.
- (5) Tính tổn thất của xác suất có điều kiện cho ô i nơi object tồn tại. Chú ý $p_i(c)$ luôn = 1 nếu đúng lớp c với ground-truth, ngược lại thì $p_i(c)$ luôn = 0.
- $\lambda_{coord} = 5.0$ thông số cân bằng để cân bằng tổn thất tọa độ (x, y, w, h) với các tổn thất khác.
- $\lambda_{noobj} = 0.5$ thông số cân bằng để cân bằng giữa hộp có và không có object. (Nói chung, đa số các ô trong image không có object, rất ít ô có object)

2.6.3.4. Classification Loss

Chúng ta chỉ tính classification loss cho những ô vuông được đánh nhãn là có object. Classification loss tại những ô vuông đó được tính bằng đỗ lỗi bình phương giữa nhãn được dự đoán và nhãn đúng của nó.

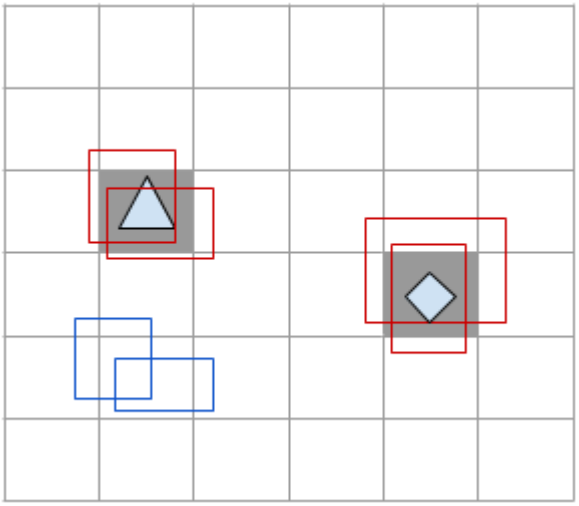
$$L_{classification} = \sum_{i=0}^{S^2} \mathbb{I}_i^{obj} \sum_{c \in class} (p_i(c) - \hat{p}_i(c))^2$$

Trong đó:

$\mathbb{I}_i^{obj}$: bằng 1 nếu ô vuông đang xét có object ngược lại bằng 0

$\hat{p}_i(c)$: là xác xuất có điều của lớp c tại ô vuông tương ứng mà mô hình dự đoán

hình 19- Công Thức classification Loss



hình 20- Hình minh họa Công Thức classification Loss

Ví dụ, trong hình minh họa ở trên, chúng ta có 2 object tại ô vuông (dòng,cột) là (2,1) và (3,4), chứa object là hình tam giác và hình tức giác đều. Độ lỗi classification loss chỉ tính cho 2 object này mà ko quan tâm đến những ô vuông khác. Lúc cài đặt chúng ta cần lưu ý phải nhân với một mask để triệt tiêu giá trị lỗi tại những ô vuông ko quan tâm.

2.6.3.5. Localization Loss

Localization loss dùng để tính giá trị lỗi cho boundary box được dự đoán bao gồm offset x,y và chiều dài, rộng so với nhãn chính xác của chúng ta. Các bạn nên lưu ý rằng, chúng ta không tính toán trực tiếp giá trị lỗi này trên kích thước của ảnh mà cần chuẩn dưới kính thước ảnh về đoạn [0-1] đối với tọa độ điểm tâm, và không dự đoán trực tiếp điểm tâm mà phải dự đoán giá trị lệch offset x,y so với ô vuông tương ứng. Việc chuẩn hóa kích thước ảnh và dự đoán offset làm cho mô hình nhanh hội tụ hơn so với việc dự đoán giá trị mặc định.

$$L_{localization} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{ij}^{obj} [(offsetx_i - \hat{offsetx_i})^2 + (offsety_i - \hat{offsety_i})^2 + (width_i - \hat{width_i})^2 + (height_i - \hat{height_i})^2]$$

Độ lỗi localization loss được tính bằng tổng đỗ lỗi bình phương của offsetx, offsetx và chiều dài, rộng trên tất cả các ô vuông có chứa object. Tại mỗi ô vuông đúng,ta chọn 1 boundary box có IOU (Intersection over union) tốt nhất, rồi sau đó tính độ lỗi theo các boundary box này. Theo hình minh họa trên chúng ta có 4 boundary box tại ô vuông đúng có viền màu đỏ, chúng ta chọn 1 box tại mỗi ô vuông để tính độ lỗi. Còn box xanh được bỏ qua.

Localization loss là độ lỗi quan trọng nhất trong 3 loại độ lỗi trên. Do đó, ta cần đặt trọng số cao hơn cho độ lỗi này.

2.6.3.6. Confidence Loss

Confidence loss thể hiện độ lỗi giữa dự đoán boundary box đó chứa object so với nhãn thực tế tại ô vuông đó. Độ lỗi này tính nên cả những ô vuông chứa object và không chứa object.

$$L_{confidence} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobject} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{ij}^{noobj} (C_i - \hat{C}_i)^2$$

Độ lỗi này là độ lỗi bình phường của dự đoán boundary đó chứa object với nhãn thực tế của ô vuông tại vị trí tương ứng, chúng ta lưu ý rằng, độ lỗi tại ô vuông mà

nhãn chứa object quan trọng hơn là độ lỗi tại ô vuông không chứa object, do đó chúng ta cần sử dụng hệ số lambda để cân bằng điều này.

Tổng kết lại, tổng lỗi của chúng ta sẽ bằng tổng của 3 loại độ lỗi trên.

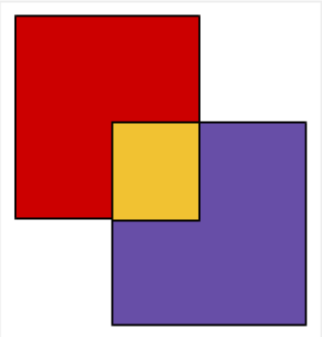
$$L_{total} = L_{classification} + L_{localization} + L_{confidence}$$

2.6.3.7. Dự đoán lớp và tạo độ boundary box sau quá trình huấn luyện – Inference

Chúng ta chỉ giữ lại những boundary box mà có chứa object nào đó. Để làm điều này, chúng ta cần tính tích của xác suất có điều kiện ô vuông thuộc về lớp i nhân với xác suất ô vuông đó chứa object, chỉ giữ lại những boundary box có giá trị này lớn hơn ngưỡng nhất định.

Mỗi object lại có thể có nhiều boundary box khác nhau do mô hình dự đoán. Để tìm boundary box tốt nhất các object, chúng ta có thể dùng thuật toán non-maximal suppression để loại những boundary box giao nhau nhiều, tức là có IOU giữ 2 boundary box lớn.

Để tính IOU giữ 2 box chúng ta cần tính diện tích giao nhau giữa 2 box chia cho tổng diện tích của 2 box đó.



$$iou = \frac{S_{vàng}}{S_{đỏ} + S_{tím} + S_{vàng}}$$

hình 21- Công thức IOU

2.6.3.8. Hạn chế của YOLO

YOLO áp đặt các ràng buộc về không gian trên những bounding box, mỗi grid cell chỉ có thể predict rất ít bounding box và duy nhất một class. Các ràng buộc này hạn chế khả năng nhận biết số object nằm gần nhau, cũng như đối với các object có kích thước nhỏ.

YOLO sử dụng các feature tương đối thô để predict bounding box, do model sử dụng nhiều lớp downsampling từ ảnh đầu vào. Bởi các hạn chế này của model khi huấn luyện để predict bounding box từ data, dẫn đến YOLO không thực sự tốt trong việc nhận diện các object với tỉ lệ hình khối mới hoặc bất thường so với tập data. YOLOv2 đã khắc phục phần nào vấn đề này, nhưng vẫn thua kém nhiều so với FRCNN.

Ngoài ra, trong quá trình training, loss function không có sự đánh giá riêng biệt giữa error của bounding box kích thước nhỏ so với error của bounding box kích thước lớn. Việc coi chúng như cùng loại và tổng hợp lại làm ảnh hưởng đến độ chính xác toàn cục của mạng. Error nhỏ trên box lớn nhìn chung ít tác hại, nhưng error nhỏ với box rất nhỏ sẽ đặc biệt ảnh hưởng đến giá trị IOU.

2.7. Web Server

Web server là một máy tính chạy dùng để chạy các trang web. Đó là một chương trình máy tính phân phối các trang web khi chúng được truy cập. Mục tiêu cơ bản của máy chủ web là lưu trữ, xử lý và cung cấp các trang web cho người dùng. Thông tin liên lạc này được thực hiện bằng cách sử dụng Giao thức truyền siêu văn bản (HTTP). Các trang web này chủ yếu là nội dung tĩnh bao gồm tài liệu HTML, hình ảnh, biểu định kiểu, thử nghiệm, v.v. Ngoài HTTP, máy chủ web còn hỗ trợ giao thức SMTP (Giao thức truyền thư đơn giản) và giao thức FTP (Giao thức truyền tệp) để gửi email và truyền tệp và lưu trữ.

Công việc chính của web server là hiển thị nội dung trang web. Nếu một máy chủ web không được công khai và được sử dụng trong nội bộ, thì nó được gọi là

Intranet Server. Khi bất kỳ ai yêu cầu một trang web bằng cách thêm URL hoặc địa chỉ web trên thanh địa chỉ của trình duyệt web (như Chrome hoặc Firefox), trình duyệt sẽ gửi yêu cầu đến Internet để xem trang web tương ứng cho địa chỉ người dùng nhập. Máy chủ tên miền (DNS) chuyển đổi URL này thành Địa chỉ IP (Ví dụ: 192.168.216.345), đến lượt nó trỏ đến Máy chủ Web. Máy chủ Web được yêu cầu hiển thị trang web nội dung cho trình duyệt của người dùng. Tất cả các trang web trên Internet đều có một mã định danh duy nhất về địa chỉ IP. Địa chỉ Giao thức Internet này được sử dụng để giao tiếp giữa các máy chủ khác nhau trên Internet. Ngày nay, máy chủ Apache là máy chủ web phổ biến nhất hiện có trên thị trường. Apache là một phần mềm mã nguồn mở xử lý gần 70% tất cả các trang web hiện có. Hầu hết các ứng dụng dựa trên web sử dụng Apache làm môi trường Web Server mặc định của chúng. Một máy chủ web khác thường có sẵn là Dịch vụ Thông tin Internet (IIS). IIS thuộc sở hữu của Microsoft.

2.8. Flask và ứng dụng

Flask là một API của Python cho phép chúng ta xây dựng các ứng dụng web. Nó được phát triển bởi Armin Ronacher. Khung của Flask rõ ràng hơn khung của Django và cũng dễ học hơn vì nó có ít mã cơ sở hơn để triển khai một Ứng dụng web đơn giản. Khung ứng dụng web hoặc Khung công tác web là tập hợp các mô-đun và thư viện giúp nhà phát triển viết ứng dụng mà không cần viết mã cấp thấp như giao thức, quản lý luồng, v.v. Flask dựa trên bộ công cụ WSGI (Web Server Gateway Interface) và công cụ mẫu Jinja2.

Vì sao chọn Flask :

- **Flask là một micro web framework :** Flask là một micro web framework viết bằng Python, không cầu kì trong việc chọn tool hay thư viện cụ thể nào. “Micro” không có nghĩa là thiếu chức năng mà “micro” theo triết lý thiết kế là cung cấp một lõi chức năng “súc tích” nhất cho ứng dụng web nhưng người dùng có thể mở rộng bất cứ lúc nào. Flask luôn hỗ trợ các thành phần tiện ích mở rộng

cho ứng dụng như tích hợp cơ sở dữ liệu, xác thực biểu mẫu, xử lý upload, các công nghệ xác thực, template, email, RESTful..., chỉ khác là khi nào bạn muốn thì bạn mới đưa vào thôi. Người dùng có thể tập trung xây dựng web application ngay từ đầu trong một khoảng thời gian rất ngắn và có thể phát triển quy mô của ứng dụng tùy theo yêu cầu.

- **Flask dễ cài đặt và triển khai**
  - Quá trình cài đặt rất dễ dàng : *pip install flask*
  - Dễ dàng triển khai một ứng dụng web chào người dùng bằng flask với vài dòng code .

chao.py

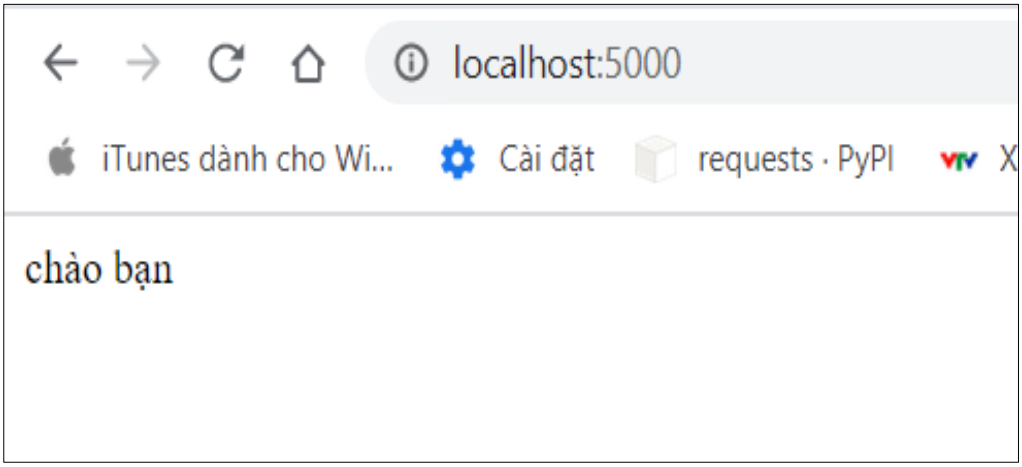
```python
from flask import Flask

app = Flask(__name__)

@app.route('/')
def chao():
    return "chào bạn"

if __name__ == "__main__":
    app.run()
```

hình 23-Hình Ảnh Code Ứng Dụng Flask



hình 22-Hình Ảnh Kết Quả Chương Trình Flask

● **Flask là một lựa chọn thông minh :** Flask thật sự phù hợp cho việc xây dựng các web application có quy mô vừa và nhỏ, các API và web services :

○ Xây dựng web application rất giống với việc viết các module Python chuẩn, cấu trúc gọn gàng và rõ ràng.

○ Thay vì cung cấp hết tất cả mọi thứ, Flask cung cấp cho người dùng các thành phần cốt lõi thường được sử dụng nhất của khung ứng dụng web như URL routing, request & response object, template…

○ Với Flask, việc chọn component nào cho ứng dụng là việc của chúng ta. Điều này thật tuyệt, vì mỗi web application có những đặc điểm và tính năng riêng, nó không phần phải chứa các component mà nó không dùng.

● **Dễ áp dụng machine learning vào server :**

○ Vì tính tối giản nên bạn không cần tập trung quá nhiều cho flask thay vào đó bạn có thể tập trung cho mục tiêu của đề tài , ở đây là machine learning.

○ Hơn nữa, việc viết bằng python nên thể áp dụng các framework của machine learning để làm 1 server xử lý các vấn đề liên quan tới machine learning.

● **Cộng đồng của flask khá lớn :**

○ Có thể dễ dàng tìm kiếm hay các giải pháp cho vấn đề gặp phải qua cộng đồng người sử dụng Flask một cách dễ dàng từ việc cài đặt , gỡ cài đặt , các lỗi …

○ Một số link của cộng đồng Flask :

https://www.python.org/community/

https://stackoverflow.com/questions/tagged/flask

https://github.com/pallets/flask/issues

2.9. Anaconda

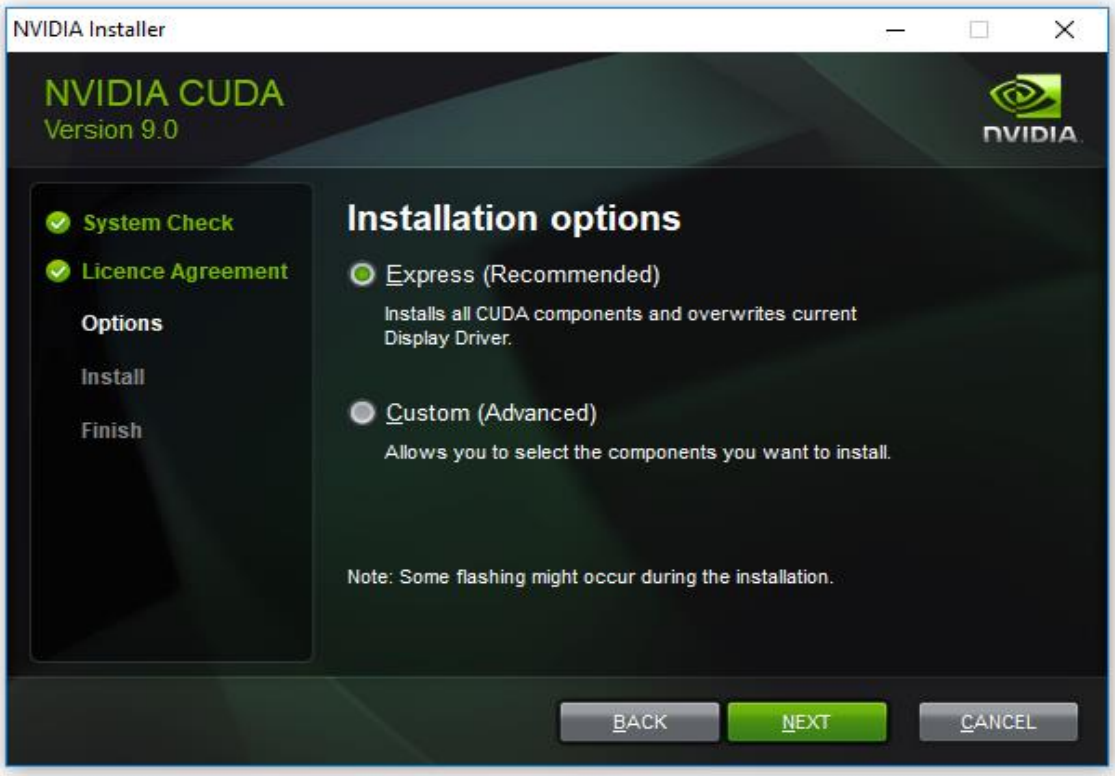
2.9.1. Anaconda là gì ?

Anaconda là một mã nguồn mở phân phối các gói được xây dựng cho khoa học dữ liệu. Nó đi kèm với conda, một gói và trình quản lý môi trường. Conda thường

sử dụng để tạo môi trường độc lập cho các dự án sử dụng các phiên bản Python khác nhau hoặc phiên bản gói khác nhau. Anaconda cũng được sử dụng để cài đặt, gỡ cài đặt và cập nhật các gói trong môi trường dự án. Khi tải xuống Anaconda lần đầu tiên, nó đi kèm với conda, Python và hơn 150 gói khoa học và các gói phụ thuộc của chúng. Anaconda là một bản tải xuống khá lớn (~ 500 MB) vì nó đi kèm với các gói khoa học dữ liệu phổ biến nhất bằng Python, đối với những người thận trọng về dung lượng đĩa, còn có Miniconda, một bản phân phối nhỏ hơn chỉ bao gồm conda và Python. Bạn vẫn có thể cài đặt bất kỳ gói nào có sẵn với conda, theo mặc định với phiên bản tiêu chuẩn.

2.10. CUDA và CuDNN

2.10.1. CUDA là gì ?



hình 24-Hình Cuda

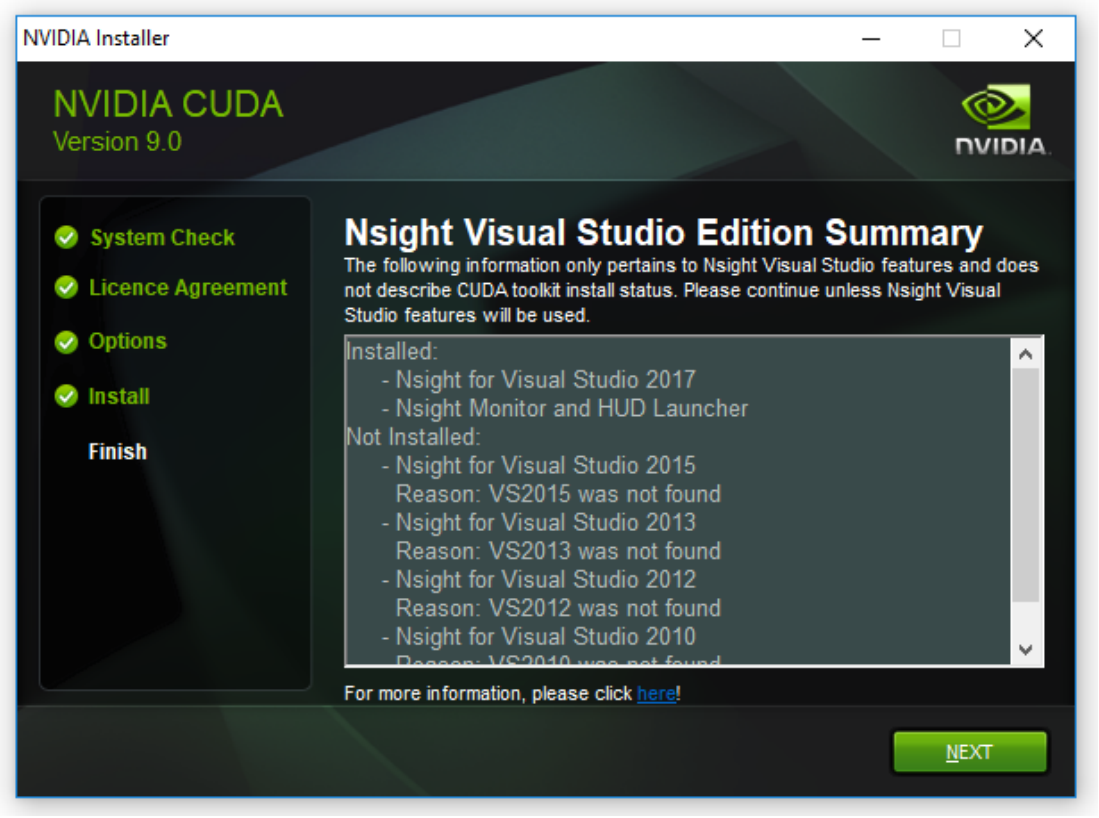
CUDA (Compute Unified Device Architecture - Kiến trúc thiết bị tính toán hợp nhất) là một kiến trúc tính toán song song do NVIDIA phát triển. Nói một cách ngắn gọn, CUDA là động cơ tính toán trong các GPU (Graphics Processing Unit -

Đơn vị xử lý đồ họa) của NVIDIA, nhưng lập trình viên có thể sử dụng nó thông qua các ngôn ngữ lập trình phổ biến. Lập trình viên dùng ngôn ngữ C for CUDA, dùng trình biên dịch PathScale Open64 C[1], để cài đặt các thuật toán chạy trên GPU. Kiến trúc CUDA hỗ trợ mọi chức năng tính toán thông qua ngôn ngữ C. Các bên thứ ba cũng đã phát triển để hỗ trợ CUDA trong Python, Fortran, Java và MATLAB.

CUDA cho phép các nhà phát triển truy nhập vào tập các chỉ lệnh ảo và bộ nhớ của các phần tử tính toán song song trong đơn vị xử lý đồ họa của CUDA (CUDA GPU). Sử dụng CUDA, các GPU mới nhất do NVIDIA sản xuất có thể dễ dàng thực hiện các tính toán như những CPU. Tuy nhiên, không giống như các CPU, các GPU có kiến trúc song song trên toàn bộ giúp cho sự tập trung vào khả năng thực thi một cách chậm rãi nhiều luồng dữ liệu một lúc, hơn là thực thi rất nhanh một luồng dữ liệu. Cách tiếp cận giải quyết các vấn đề có mục đích tổng quát này trên các GPU được gọi là GPGPU.

2.10.2. CuDNN là gì ?

CuDNN – NVidia CUDA® Deep Neural Network: Là một thư viện nền tảng cho các deep neural network được tăng tốc bởi GPU. cuDNN cung cấp các thiết lập được tinh chỉnh cho các thủ tục được chuẩn hóa như forward and backward convolution, pooling, normalization và các lớp kích hoạt. cuDNN là một phần của Deep Learning SDK do NVidia cung cấp.



hình 25-CuDNN install

3. Trình bày việc lấy dữ liệu, phân tích đưa ra mô hình

3.1. Phương pháp thu thập dữ liệu

Để tiến hành thu thập dữ liệu thì đầu tiên , chúng ta cần xác định mô hình cần loại dữ liệu gì để có thể tìm kiếm được dữ liệu cần thiết cho mô hình training .Với các bài toán classification có thể dựa vào tên các classes để tạo thành các keywords và sử dụng các công cụ crawling data từ Internet cho việc tìm ảnh. Hoặc có thể tìm ảnh, videos từ các trang mạng xã hội, ảnh vệ tinh trên Google, free collected data từ camera công cộng hay xe hơi (Waymo, Tesla), thậm chí có thể mua dữ liệu từ bên thứ 3 (lưu ý tính chính xác của dữ liệu).

3.2. Phương pháp đánh giá mô hình train

Trong quá trình xây dựng một mô hình machine learning, một phần không thể thiếu để biết được chất lượng của mô hình như thế nào đó chính là đánh giá mô

hình.

Đánh giá mô hình giúp chúng ta lựa chọn được mô hình phù hợp nhất đối với bài toán của mình. Tuy nhiên để tìm được thước đo đánh giá mô hình phù hợp thì chúng ta cần phải hiểu về ý nghĩa, bản chất và trường hợp áp dụng của từng thước đo.

Chính vì vậy bài viết này sẽ cung cấp cho các bạn kiến thức về các thước đo cơ bản nhất, thường được áp dụng trong các mô hình phân loại trong machine learning nhưng chúng ta đôi khi còn chưa nắm vững hoặc chưa biết cách áp dụng những thước đo này sao cho phù hợp với từng bộ dữ liệu cụ thể.

3.2.1. Overfitting là gì ?

Overfitting đề cập đến tình huống trong đó mô hình học máy không thể tổng quát hóa hoặc không phù hợp tốt trên tập dữ liệu chưa thấy. Một dấu hiệu rõ ràng của việc trang bị quá mức cho máy học là nếu lỗi của nó trên tập dữ liệu kiểm tra hoặc xác thực lớn hơn nhiều so với lỗi trên tập dữ liệu đào tạo.

Overfitting là một thuật ngữ được sử dụng trong thống kê đề cập đến lỗi mô hình hóa xảy ra khi một hàm tương ứng quá gần với tập dữ liệu. Do đó, việc trang bị quá nhiều có thể không phù hợp với dữ liệu bổ sung và điều này có thể ảnh hưởng đến độ chính xác của việc dự đoán các quan sát trong tương lai.

Việc trang bị quá mức xảy ra khi một mô hình tìm hiểu chi tiết và nhiễu trong tập dữ liệu huấn luyện đến mức nó tác động tiêu cực đến hiệu suất của mô hình trên tập dữ liệu mới. Điều này có nghĩa là nhiễu hoặc dao động ngẫu nhiên trong tập dữ liệu huấn luyện được mô hình thu thập và học dưới dạng các khái niệm. Vấn đề là những khái niệm này không áp dụng cho tập dữ liệu mới và tác động tiêu cực đến khả năng tổng quát hóa của mô hình.

3.2.2. Underfitting là gì ?

Overfitting đề cập đến một mô hình không thể mô hình hóa tập dữ liệu đào tạo

cũng như không tổng quát hóa thành tập dữ liệu mới. Một mô hình học máy không phù hợp không phải là một mô hình phù hợp và sẽ hiển nhiên vì nó sẽ có hiệu suất kém trên tập dữ liệu đào tạo.

Việc trang bị dưới trang bị thường không được thảo luận vì rất dễ phát hiện nếu có chỉ số hiệu suất tốt.

3.2.3. Một số phương pháp đánh giá mô hình?

3.2.3.1. Phương pháp Hold-Out

Phương pháp này chia làm dữ liệu thành 3 tập dữ liệu lớn :

Training set : là một tập con của tập dữ liệu được sử dụng để xây dựng các mô hình dự đoán .

Validation set : là một tập con của tập dữ liệu được sử dụng để đánh giá hiệu suất của mô hình được xây dựng trong giai đoạn huấn luyện. Nó cung cấp một nền tảng thử nghiệm để tinh chỉnh các thông số của mô hình và chọn mô hình hoạt động tốt nhất. Không phải tất cả các thuật toán mô hình hóa đều cần một bộ xác nhận.

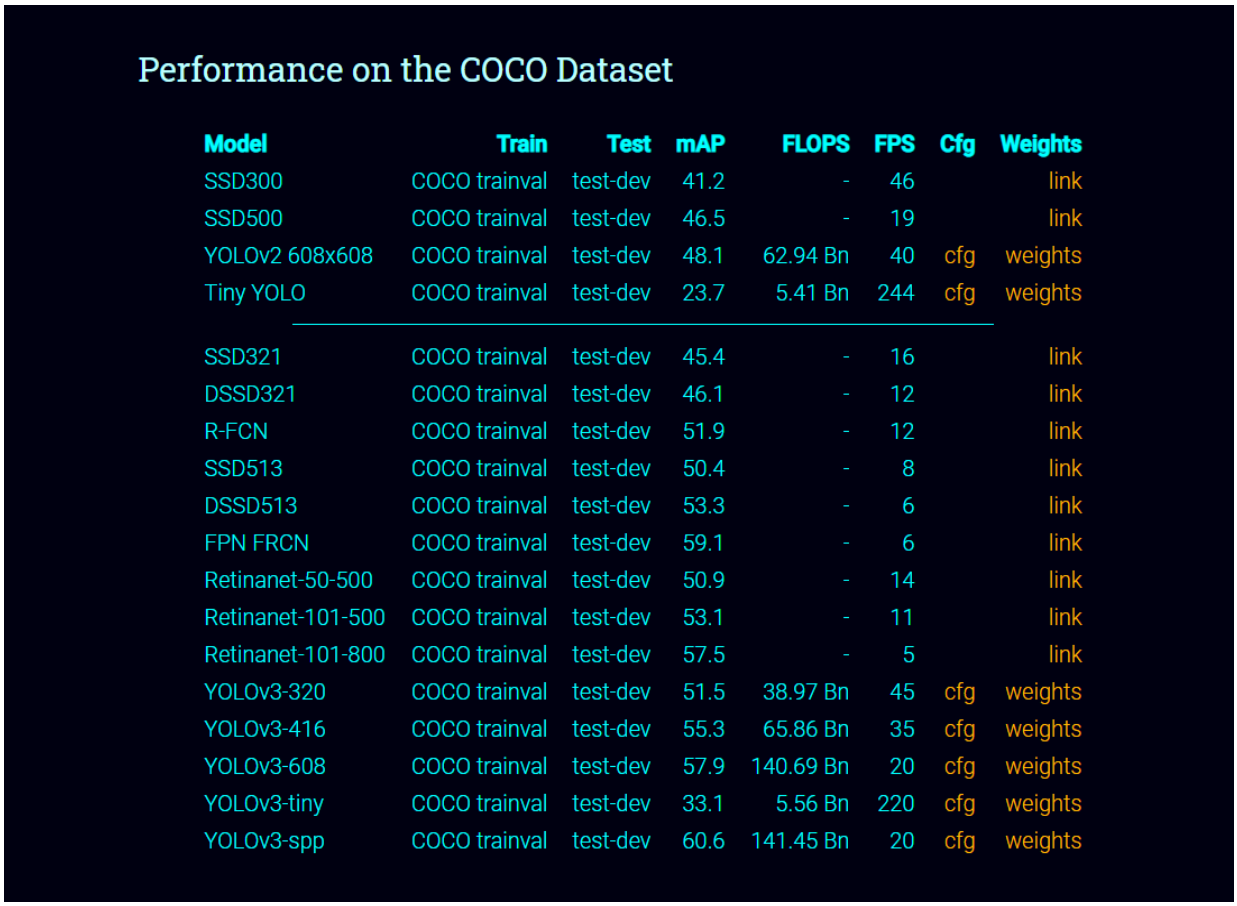
Test set : là một tập hợp con của tập dữ liệu để đánh giá khả năng hoạt động trong tương lai của một mô hình. Nếu một mô hình phù hợp với bộ đào tạo tốt hơn nhiều so với bộ thử nghiệm, thì việc train quá nhiều có thể là nguyên nhân .

3.2.3.2. Phương pháp Cross-Validation

Khi chỉ có sẵn một lượng dữ liệu hạn chế, để đạt được ước tính không thiên vị về hiệu suất mô hình, chúng tôi sử dụng xác nhận chéo N lần. Trong xác nhận chéo N lần , chia dữ liệu thành N tập con có kích thước bằng nhau. Xây dựng các mô hình N lần, mỗi lần loại bỏ một trong các tập con khỏi quá trình huấn luyện và sử dụng nó làm tập thử nghiệm. Nếu N bằng với kích thước mẫu, điều này được gọi là "bỏ một lần".

3.3. Mô hình train

Vì chi phí cho việc training cũng như vấn đề về dữ liệu nên đề tài đã thu thập model từ tác giả của Yolo V3 . Nó được công khai tại trang chủ của Yolo V3.



Performance on the COCO Dataset

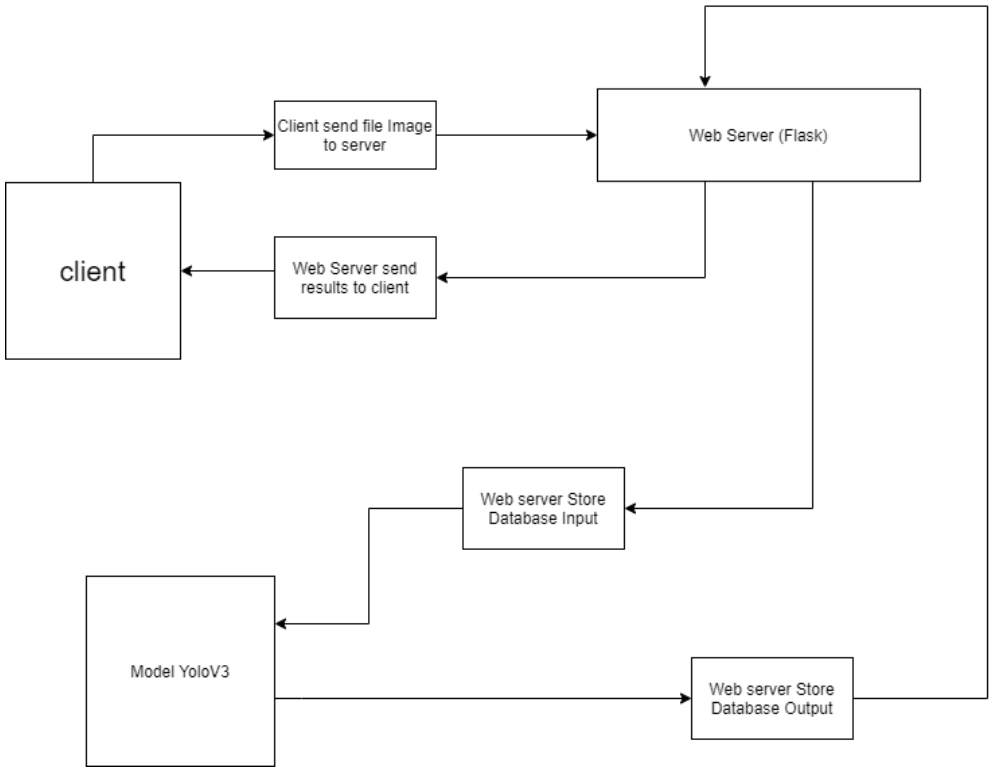
Model	Train	Test	mAP	FLOPS	FPS	Cfg	Weights
SSD300	COCO trainval	test-dev	41.2	-	46		link
SSD500	COCO trainval	test-dev	46.5	-	19		link
YOLOv2 608x608	COCO trainval	test-dev	48.1	62.94 Bn	40	cfg	weights
Tiny YOLO	COCO trainval	test-dev	23.7	5.41 Bn	244	cfg	weights
SSD321	COCO trainval	test-dev	45.4	-	16		link
DSSD321	COCO trainval	test-dev	46.1	-	12		link
R-FCN	COCO trainval	test-dev	51.9	-	12		link
SSD513	COCO trainval	test-dev	50.4	-	8		link
DSSD513	COCO trainval	test-dev	53.3	-	6		link
FPN FRCN	COCO trainval	test-dev	59.1	-	6		link
Retinanet-50-500	COCO trainval	test-dev	50.9	-	14		link
Retinanet-101-500	COCO trainval	test-dev	53.1	-	11		link
Retinanet-101-800	COCO trainval	test-dev	57.5	-	5		link
YOLOv3-320	COCO trainval	test-dev	51.5	38.97 Bn	45	cfg	weights
YOLOv3-416	COCO trainval	test-dev	55.3	65.86 Bn	35	cfg	weights
YOLOv3-608	COCO trainval	test-dev	57.9	140.69 Bn	20	cfg	weights
YOLOv3-tiny	COCO trainval	test-dev	33.1	5.56 Bn	220	cfg	weights
YOLOv3-spp	COCO trainval	test-dev	60.6	141.45 Bn	20	cfg	weights

hình 26-So sánh các mô hình

Vì tính chất chính xác cũng như cho phép người dùng có thể tải ảnh có nhiều vật thể hơn nên đề tài đã chọn Yolo V3 608 .

4. Thiết kế hệ thống

4.1. Backend



hình 27-Sơ đồ Backend

4.1.1. Công cụ xây dựng Server Backend

Với Web Server Machine Learning thường có yêu cầu về phần cứng cao hơn các Web Server bình thường vì yêu cầu xử lý các tác vụ nặng chứ không chỉ đơn giản lưu là lưu trữ như những Web Server khác . Vì hiện nay vì vấn đề kinh phí nên đề tài quyết định chạy trên laptop cá nhân với một số công cụ sau :

- Cấu hình máy tính cá nhân : Asus, Cpu: Intel Core i7 8th , gpu nvidia 1050Ti, SSD 258Gb, Ram 8gb.
- Thư viện OpenCV-Python
- Framework Flask
- Mô hình Yolo

- Thư Viện Numpy

4.1.2. Luồng xử lý

Sau khi client gửi 1 một file ảnh lên web server , web server sẽ tiến hành lưu lại vào database của server , sau đó hệ thống lấy ảnh vừa tải lên trong database để tiến hành detect vật thể . Quá trình này kết thúc sẽ cho ra 1 file ảnh mới đã vẽ khung , đánh tên vật thể và list danh sách các vật trong ảnh sau đó sẽ tự lưu vào database Output của server . Server sẽ lấy đường dẫn ảnh của file output vừa trả về và gửi về client để hiển thị . Cùng với server còn truyền thêm list danh sách các vật thể để cho server xử lý list và hiện thị cho người dùng .

4.1.3. Cách Xây Dựng Server Backend Với Yolov3 và Flask

Để xây dựng một Backend Server Machine Learning chúng ta cần thực hiện rất nhiều công việc phức tạp , với cách xây dựng một Server trên laptop đề tài thực hiện những bước sau :

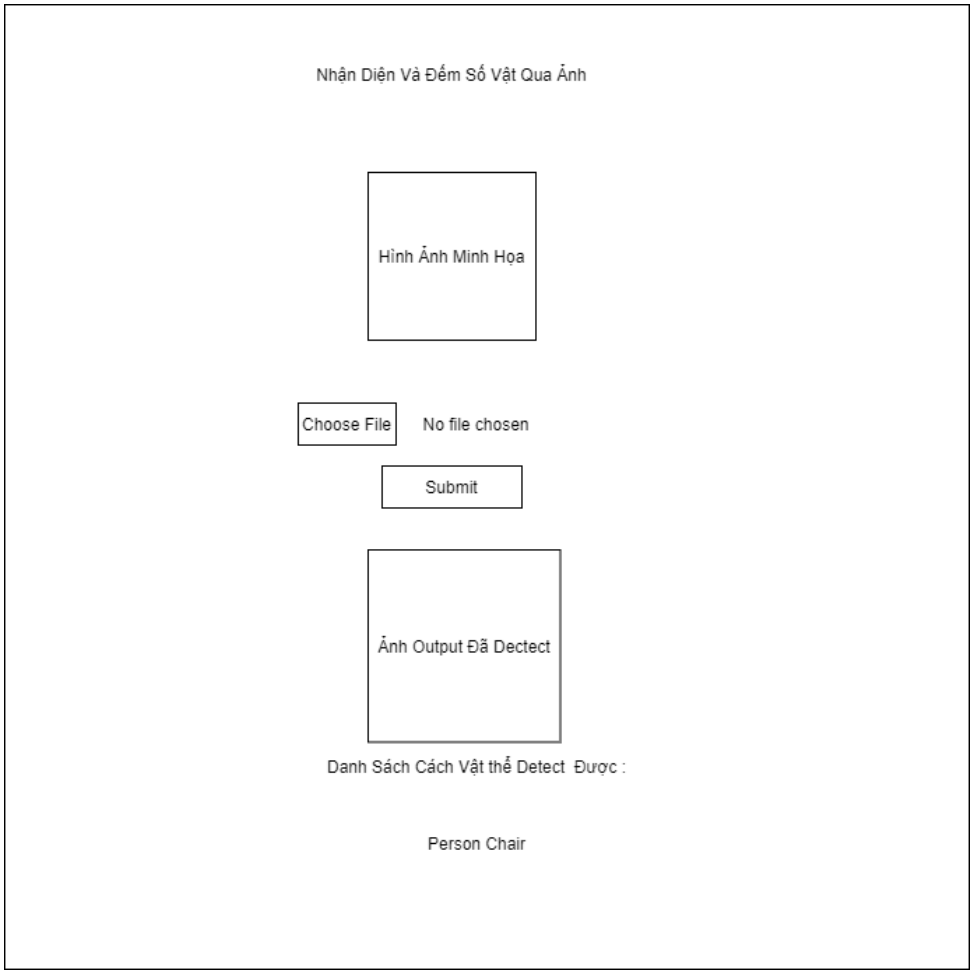
- Tải và cài đặt môi trường làm việc Anaconda .
- Tải file pre-trained bằng dataset CoCo 2017 của Yolo tại link :https://pjreddie.com/media/files/yolov3.weights
- Tải file cfg(yolov3-608) (Nơi chứa cấu hình của hình ảnh đầu vào cũng như layers của model yolo) của yolo tại link :https://raw.githubusercontent.com/pjreddie/darknet/master/cfg/yolov3.cfg và lưu lại(Ctrl +s) file **yolo.cfg**
- Tải file(**coco.name**) chứa tên các vật thể mà Model có thể detect được tại link : https://raw.githubusercontent.com/pjreddie/darknet/master/data/coco.names
- Thiết lập cổng giao tiếp là localhost:5000 bằng flask framework
- Dùng flask framework để xây dựng web api để nhận file ảnh từ client gửi về theo giao thức POST và lưu file vào 1 thư mục.

- Dùng Yolo Pre-Trained và OpenCv để detect vật thể qua file ảnh nhận được .Sau đó viết lại 1 file ảnh mới đã vẽ khung các vật detect được trong một thư mục mới và 1 biến các chứa danh sách các vật detect được.
- Dùng flask framework để trả về client đường dẫn của ảnh mới tạo ra và biến chứa danh sách các vật thể .

4.2. Frontend - Giao diện website cho người dùng

Để tạo ra một web hoàn chỉnh thì ngoài Server Backend chúng ta cần một giao diện web cho người dùng tương tác với Server . Đề tài đã cố gắng thực hiện làm một giao diện đơn giản nhất bằng kiến thức về thiết kế web cơ bản . Để xây dựng một giao diện cho người dùng trên laptop đề tài đã thực hiện những bước bao gồm :

- Phác thảo bản vẽ giao diện tương tác người dùng
- Xây dựng giao diện web bằng html cho cổng giao tiếp localhost:5000
- Xây dựng form để chọn file từ máy tính và gửi file ảnh lên server bằng giao thức POST .
- Cài đặt các thành phần trong file html để hiển thị các thành phần mà server gửi về



hình 28-Hình Ảnh Phác Thảo Giao Diện Frontend

Bằng những lệnh cơ bản của html , đề tài đã thực hiện được quá một giao diện có chức năng nhận ảnh và gửi ảnh , cũng như nhận các loại data khác từ server .

5. Kết luận đánh giá kết quả đạt được

5.1. Thử nghiệm

Để chạy chương trình thì chúng ta cần một môi trường Anaconda và cài thêm package cần thiết bao gồm :

- Opencv - Python
- Flask
- CuDNN
- Cuda Toolkit

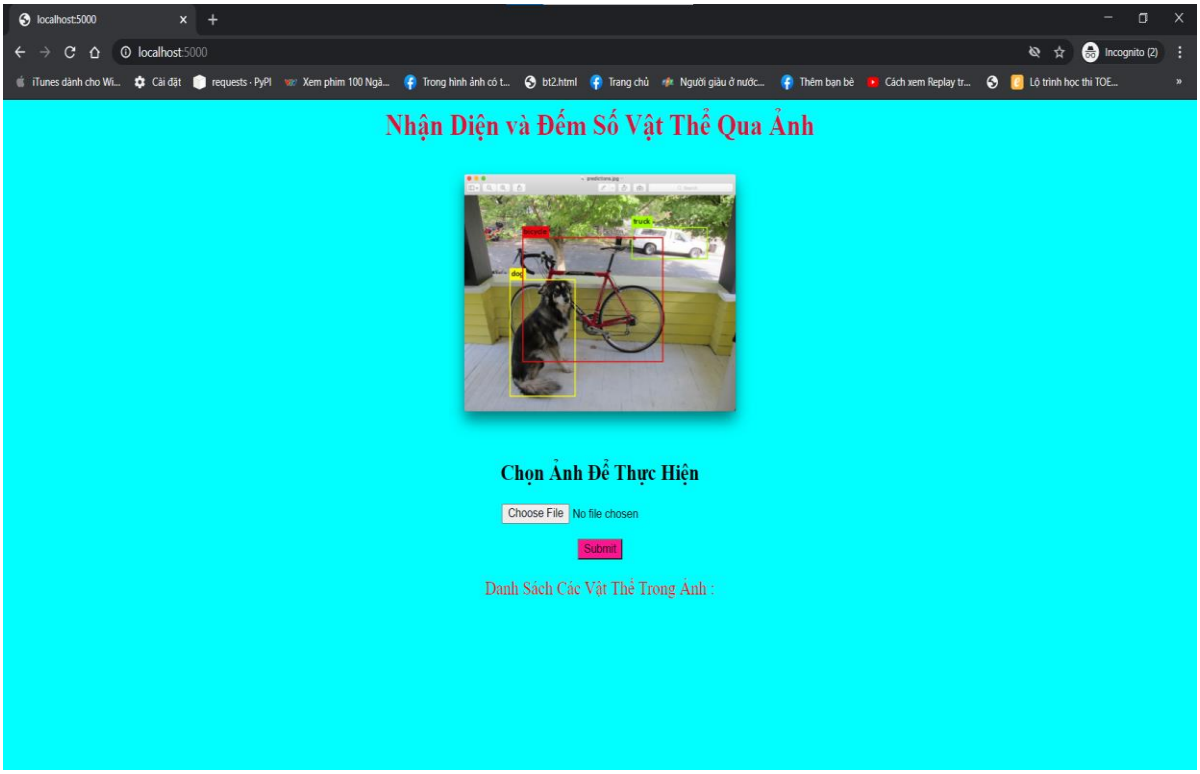
Tiến Hành chạy chương trình bằng việc sử dụng dòng lệnh để kích server :

python main.py

```
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

hình 29-Ảnh Chạy Chương Trình

Tiến hành sử dụng bằng cách truy cập vào trình duyệt bất kỳ và gõ đường dẫn: *Localhost : 5000*

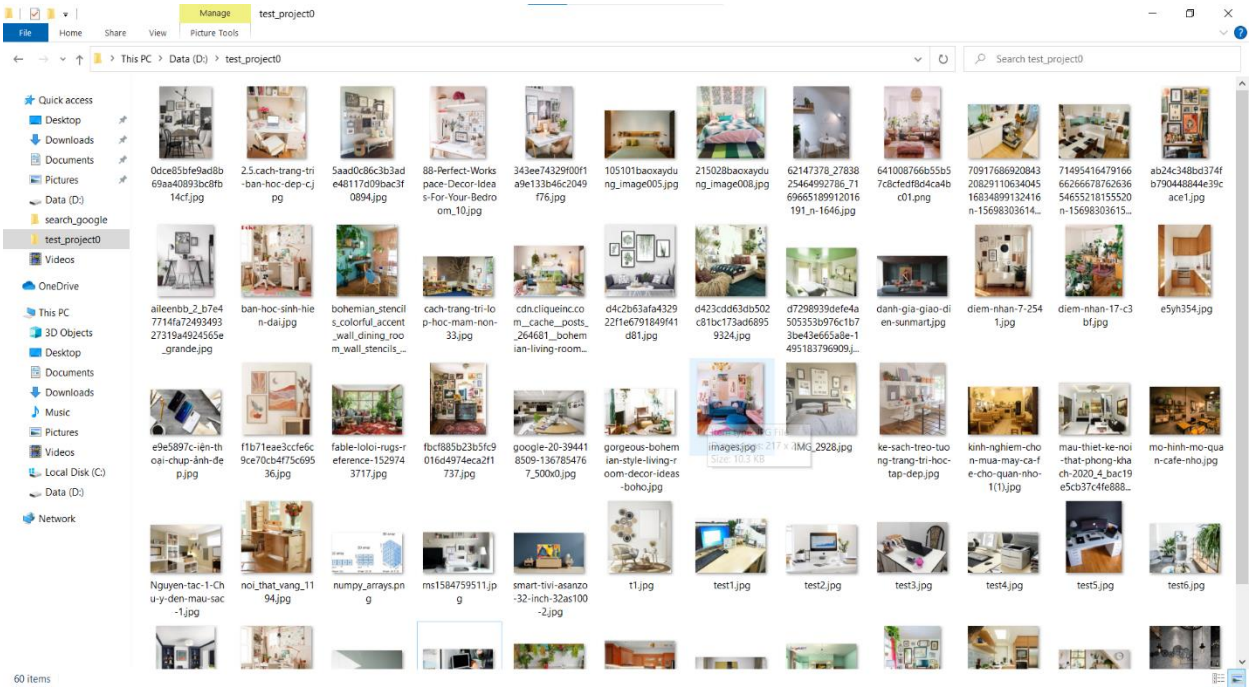


hình 30-Ảnh chạy thử chương trình

Để kiểm thử đề tài đã chuẩn bị các hình ảnh chứa đồ vật thu thập trên mạng và kiểm thử bao gồm 60 hình ảnh . Đề tài sẽ chọn ngẫu nhiên 10 ảnh để tiến hành kiểm tra và đánh giá mô hình. Tất cả những hình ảnh được đề tài thử nghiệm cũng như ảnh chứa kết quả thu được đã được đề tài công khai tại :

BẢNG 1: Kết quả thu được từ quá trình kiểm tra chương trình

Tên Hình	Vật thể có trong hình	Vật thể Chương trình detect được
h1.jpg	Người : 1 , laptop : 1 , chậu cây : 2, chuột máy tính : 1 , đèn : 1 , bàn phím :1 , ghế : 1 ,bàn làm việc : 1	Người : 1 , ghế : 1, chậu cây : 1, laptop : 1 , chuột : 1
h2.jpg	Màn hình máy tính : 1, Chậu : 1, hoa hồng : 5 , ly :1 , chuột :1 , bàn phím : 1	Màn hình : 1 , chậu : 1 , chuột: 1,bàn phím :1 .
h3.jpg	Người : 4 , laptop :4, giường: 1 , đồ ăn : 1	Người: 4 , Laptop: 4 , giường 1.
h4.jpg	Ghế : 1, màn hình máy tính : 1 , chậu cây : 1,chuột : 2 , bàn phím :1 , bàn làm việc : 1, sách : 1, bút : 1	Ghế :1 , màn hình máy tính : 1 , chuột : 2 , bàn phím 1 , chậu cây : 1.
h5.jpg	Màn hình : 1 , bàn phím : 1 , chuột : 1 , ghế : 1, chậu : 1	Màn hình máy tính : 1, chuột : 1 , ghế : 1 , chậu: 1
h6.jpg	Bàn : 1, sách : 1, bút : 1,chậu cây : 1,laptop :1, tranh :1	Bàn : 1,Sách : 1,chậu cây : 1 , laptop : 1
h7.jpg	Chậu cây : 1 , bàn : 1, laptop : 1, điện thoại : 1, cốc : 1, gấu bông : 1, rổ : 1.	Chậu cây : 1, laptop : 1, điện thoại: 1, gấu bông : 1
h8.jpg	Bàn : 1, ghế 1 , laptop : 1, tranh : 1, đèn:1 , chậu cây : 1.	Ghế : 1, laptop: 1, chậu cây : 1
h9.jpg	Ghế : 1, bút : 2 , điện thoại : 1 , laptop : 1,chuột : 1, bàn : 1 , cốc : 1	Ghế : 1, chuột : 1 , laptop:1, điện thoại : 1
h10.jpg	Bàn : 1 , laptop : 1 , chậu cây : 1 , máy tính bảng : 1, vở : 1, bút : 1, chuột 1, cốc : 1, điện thoại : 1, ghế 1	Chậu cây : 1 , chuột : 1,cốc : 1,chậu hoa : 1,điện thoại : 1,laptop : 1



hình 31- Hình ảnh thư mục test

5.2. Đánh giá

Qua quá trình kiểm tra , đánh giá mô hình thì đề tài nhận thấy rằng : Với mô hình đã được train sẵn bởi tác giả trên tập dữ liệu COCO 2017 (là dataset bao gồm rất nhiều ảnh của các đồ vật chia thành 80 loại khác nhau) có tỷ lệ dự đoán chính xác 80% trên những hình ảnh đã kiểm tra .Tuy nhiên, mô hình này còn khuyết điểm khi các vật chất chồng lên nhau khiến mô hình dự đoán sai hoặc thiếu kết quả.

Tài Liệu Tham Khảo

https://medium.com/@venkatakrishna.jonnalagadda/object-detection-yolo-v1-v2-v3-c3d5eca2312a

https://aicurious.io/posts/tim-hieu-yolo-cho-phat-hien-vat-tu-v1-den-v5

https://lilianweng.github.io/lil-log/2018/12/27/object-detection-part-4.html