**TRƯỜNG ĐẠI HỌC THỦY LỢI**

**KHOA CÔNG NGHỆ THÔNG TIN**



ĐỒ ÁN TỐT NGHIỆP

**XÂY DỰNG HỆ THỐNG GÁN NHÃN VÀ NHẬN DIỆN THUỐC LÁ ĐIỆN TỬ**

Sinh viên thực hiện: Đào Duy Đán

Giảng viên hướng dẫn: TS. Nguyễn Thọ Thông

HÀ NỘI, NĂM 2023

# MỤC LỤC

[MỤC LỤC 2](#_Toc154727882)

[DANH MỤC HÌNH ẢNH 4](#_Toc154727883)

[DANH MỤC BẢNG BIỂU 4](#_Toc154727884)

[DANH MỤC CÁC TỪ VIẾT TẮT VÀ GIẢI THÍCH CÁC THUẬT NGỮ 4](#_Toc154727885)

[**I.** **TỔNG QUAN ĐỒ ÁN TỐT NGHIỆP.** 5](#_Toc154727886)

[**1.1.** **Giới thiệu đề tài.** 5](#_Toc154727887)

[**1.2.** **Cơ sở lý thuyết.** 6](#_Toc154727888)

[**1.3.** **Các công nghệ sử dụng.** 28](#_Toc154727889)

[**II.** **XÂY DỰNG MÔ HÌNH HỌC SÂU.** 33](#_Toc154727890)

[**2.1.** **Thu thập dữ liệu.** 33](#_Toc154727891)

[**2.2.** **Tiền xử lý dữ liệu.** 34](#_Toc154727892)

[**2.3.** **Gán nhãn dữ liệu.** 36](#_Toc154727893)

[**2.4.** **Lựa chọn mô hình.** 37](#_Toc154727894)

[**2.5.** **Xây dựng mô hình.** 38](#_Toc154727895)

[**2.6.** **Huấn luyện mô hình.** 41](#_Toc154727896)

[2.7. Đánh giá mô hình. 42](#_Toc154727897)

[2.8. Triển khai mô hình. 42](#_Toc154727898)

[**III.** **PHÂN TÍCH THIẾT KẾ HỆ THỐNG.** 42](#_Toc154727899)

[3.1. Phân tích yêu cầu hệ thống. 42](#_Toc154727900)

[3.2. Đặc tả yêu cầu hệ thống. 42](#_Toc154727901)

[3.3. Thiết kế hệ thống gán nhãn. 42](#_Toc154727902)

[**IV.** **THỰC NGHIỆM, SO SÁNH VÀ ĐÁNH GIÁ.** 42](#_Toc154727903)

[4.1. Môi trường thực nghiệm – cấu hình. 42](#_Toc154727904)

[4.2. Kết quả thực nghiệm. 42](#_Toc154727905)

[4.3. So sánh với các phương pháp khác. 42](#_Toc154727906)

[4.3.1. Thuật toán Fast R-CNN. 42](#_Toc154727907)

[4.4. Đánh giá hiệu suất mô hình và phân tích kết quả. 42](#_Toc154727908)

[4.5. Kết luận. 42](#_Toc154727909)

# DANH MỤC HÌNH ẢNH

# DANH MỤC BẢNG BIỂU

# DANH MỤC CÁC TỪ VIẾT TẮT VÀ GIẢI THÍCH CÁC THUẬT NGỮ

1. **TỔNG QUAN ĐỒ ÁN TỐT NGHIỆP.**
   1. **Giới thiệu đề tài.**

Với sự phát triển không ngừng của lĩnh vực thị giác máy tính trong thập kỷ gần đây đã mở ra những cánh cửa mới cho ứng dụng thực tế như phân loại, trích xuất đặc trưng và phát hiện đối tượng trong hình ảnh. Cùng với đó là sự phát triển nhanh chóng của học sâu đã giúp chúng ta tiến xa hơn với việc tiếp cận vấn đề trên. Học sâu cho phép mô hình hóa tự động các đặc trưng phức tạp từ dữ liệu giúp chúng ta giải quyết các thách thức đặt ra trong việc nhận diện đối tượng trong các hình ảnh đa dạng.

Trong thời đại công nghệ hiện đại ngày nay, người tiêu dùng thường xuyên tiếp cận với những loại quảng cáo trực tuyến trên website, mạng xã hội và ứng dụng di động. Một trong những loại quảng cáo rất phổ biến và đã trở thành một thách thức lớn với sức khỏe người tiêu dùng, nhất là với những người trẻ tuổi đó là thuốc lá điện tử (e-cigarettle). Tại Việt Nam hiện nay xuất hiện tràn lan các sản phẩn thuốc lá điện tử không rõ nguồn gốc có giá thành rất rẻ, chủ yếu là hàng kém chất lượng, rất độc hại và đã gây ra rất nhiều ca ngộ độc. Vì vậy hiện nay nước ta và rất nhiều quốc gia trên thế giới đã thực hiện các biện pháp nhằm kiểm soát, hạn chế quảng cáo và buôn bán loại thuốc lá này.

Từ sự phát triển của công nghệ và tình hình thực tiễn đã nêu ở trên, đồ án này sẽ tập trung vào việc xây dựng một hệ thống gán nhãn và dùng thuật toán Faster R-CNN để xây dựng mô hình nhận diện đối tượng thuốc lá điện tử. Hệ thống gán nhãn dùng để gán nhãn những đối tượng thuốc lá điện tử thu thập được từ thực tế và từ đó trích xuất ra dữ liệu đúng định dạng dùng cho việc huấn luyện. Thuật toán Faster R-CNN là một trong những thuật toán tiên tiến nhất trong lĩnh vực nhận diện đối tượng, với khả năng tự động tạo ra các vùng đề xuất (Region Proposals) thông qua RPN và sau đó phân loại đối tượng trong các vùng này.

Các nội dung chủ yếu sẽ hướng tới việc phân tích thiết kế, xây dựng hệ thống gán nhãn và nêu các ý tưởng, phạm vi và cài đặt thuật toán Faster R-CNN trong việc nhận diện thuốc lá điện tử. Mục tiêu của đề tài không chỉ cung cấp giải pháp hiệu quả mà còn mang lại những hiệu quả tích cực cho xã hội đặc biệt trong lĩnh vực nghiên cứu y tế trong việc có thể kiểm soát và hạn chế việc buôn bán thuốc lá điện tử.

* 1. **Cơ sở lý thuyết.**
     1. **Tổng quan về học sâu.**
        1. **Lý thuyết về bài toán nhận diện đối tượng trong thị giác máy tính.**

Thị giác máy tính (Computer Vision) là một lĩnh vực bao gồm các phương pháp thu nhận, xử lý ảnh kỹ thuật số, phân tích và nhận dạng các hình ảnh, phát hiện các đối tượng, tạo ảnh, siêu phân giải hình ảnh và còn nhiều hơn nữa. Object Detection – Nhận diện đối tượng có lẽ là khía cạnh sâu sắc nhất của thị giác máy tính do số lần sử dụng trong thực tế.

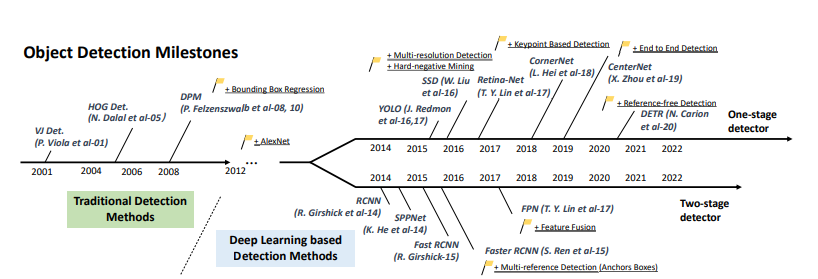
Object Detection là một nhiệm vụ trong thị giác máy tính, trong đó mục tiêu là phát hiện và định vị các đối tượng được quan tâm trong hình ảnh hoặc video. Nhiệm vụ này liên quan đến việc xác định vị trí và ranh giới của các đối tượng trong ảnh và phân loại các đối tượng thành các loại khác nhau. Nó tạo thành một phần quan trọng trong nhận dạng thị giác, bên cạnh việc phân loại và truy xuất hình ảnh.

Các phương pháp nhận diện đối tượng được bắt đầu sử dụng và triển khai sớm liên quan đến việc sử dụng các thuật toán cổ điển. Tuy nhiên, các thuật toán cổ điển này không thể đạt hiệu suất đủ để làm việc trong các điều kiện khác nhau.

Việc áp dụng đột phát và nhanh chóng của Deep Learning – Học sâu vào năm 2012 đã đưa vào sự tồn tại các thuật toán và phương pháp phát hiện đối tượng hiện đại và chính xác cao như R-CNN, Fast R-CNN, Faster R-CNN, RetinaNet,… và nhanh hơn nhưng rất chính xác như SSD và YOLO.

* + - 1. **Các cột mốc phát triển của lĩnh vực phát hiện đối tượng.**

Có thể chia cột mốc phát triển của lĩnh vực phát hiện đối tượng thành 2 giai đoạn chính đó là: Giai đoạn Truyền thống (Trước Deep Learning) và Giai đoạn Deep Leaning.



1. **Giai đoạn Truyền thống.**

Trong thời kỳ đầu của lĩnh vực thị giác máy tính, hầu hết các thuật toán phát hiện đối tượng ban đầu đều được xây dựng dựa trên các tính năng thủ công.

**Viola Jones Detectors** – Năm 2001, P. Viola và M. Jones lần đầu tiên đã đạt được khả năng phát hiện khuôn mặt người theo thời gian thực mà không có bất kỳ ràng buộc nào (ví dụ: màu da) đó là. Thuật toán này thực hiện theo cách đơn giản nhất là dùng một cửa sổ trượt đi qua tất cả các vị trí và tỷ lệ có thể có trong hình ảnh để xem liệu có cửa sổ nào chứa đối tượng không.

**Histogram of Oriented Gradients** – Năm 2005, N. Dalal và B. Triggs đã đề xuất bộ mô tả đặc trưng Histogram of Oriented Gradients (HOG). HOG là một cải tiến quan trọng của phép biến đổi tính năng bất biến tỷ lệ và định hình bối cảnh lúc đó. HOG hiệu quả trong việc mô tả hình dạng và biểu diễn đối tượng.

**Deformable Part Models** (DPM) – DPM được Felzenszmalb đè xuất vào năm 2008 như một phần mở rộng của HOG detector. Nó tuân theo triết lý detection “phân chia và chinh phục”, trong quá trình training có thể hiểu đơn giản là phân tách đối tượng và suy luận có thể coi là tập hợp các phát hiện trên các phần đối tượng khác nhau. Ví dụ như Oto có thể tách thành việc phát hiện cửa sổ, thân xe và bánh xe.

1. **Giai đoạn Deep Learning.**

Năm 2012, thế giới chứng kiến sự hồi sinh của Mạng lưới thần kinh tích chập – Convolutional Neural Networks (CNN). Năm 2014, R. Girshick và các cộng sự đã đề xuất Regions with CNN features (RCNN). Kể từ đó, lĩnh vực object detection phát triển với tốc độ chưa từng thấy. Có hai nhóm Detectors trong kỷ nguyên học sâu đó là: “two-stage detectors” and “one-stage detectors”. Trong khi “two-stage-detectors” coi việc phát hiện đối tượng là một quá trình “từ thô đến tinh”, thì “one-stage detectors” được đóng khung là “hoàn thành trong một bước”

1. **CNN based Two-stage Detectors.**

**RCNN (Region-based Convolutional Neural Network)** – RCNN sử dụng thuật toán selective search để đề xuất đối tượng và CNN(ví dụ AlexNet) để trích xuất đặc trưng. Được đào tạo với SVM và dự đoán đối tượng. RCNN mang lại hiệu suất đáng kể trên tập dữ liệu VOCO7, với sự tăng cường độ chính xác trung bình (mAP) từ 33.7% (DPM-v5) lên 58.5.

**SPPNet (Spatial Pyramid Pooling Networks)** – Năm 2014, K. He và đồng nghiệp đề xuất SPPNet. SPP cho phép tạo đặc trưng cố định mà không cần thay đổi kích thước ảnh. Điều này giúp tránh việc tính toán lặp đi lặp lại khi đào tạo đối tượng.

**Fast-RCNN (Fast Region-based Convolutional Neural Network)** –Fast-RCNN tích hợp thành công những ưu điểm của R-CNN và SPPNet. Tuy nhiên, tốc độ phát hiện của nó vẫn còn hạn chế bởi quá trình đề xuất.

**Faster RCNN (Faster Region-based Convolutional Neural Network)** –Năm 2015, S. Ren và đồng nghiệp đề xuất Faster RCNN là mô hình học sâu gần thời gian thực đầu tiên. Faster RCNN giới thiệu Region Proposal Network (RPN) cho đề xuất các vùng một cách nhanh chóng. Faster RCNN được tích hợp hầu hết các khối chức năng vào một framework học end-to-end.

**FPN (Feature Pyramid Networks)** – Năm 2017, T.-Y Lin và đồng nghiệp đề xuất FPN. FPN sử dụng kiến trúc từ trên xuống dưới để xây dựng ngữ cảnh ở mọi quy mô. Đạt được tiến bộ đáng kể trong việc phát hiện đối tượng ở nhiều quy mô. Trở thành khối cơ bản cho nhiều bộ phát hiện hiện đại.

1. **CNN based One-stage Detectors.**

**YOLO (You Only Look Once)** – Được đề xuất vào năm 2015 bởi R. Joseph và đồng nghiệp. YOLO chia ảnh thành một lưới các ô nhỏ và áp dụng một mô hình CNN một lần. Mô hình dự đoán bounding boxes và xác suất cho các đối tượng trực tiếp. Tuy tính toán nhanh, nhưng có độ chính xác thấp cho đối tượng nhỏ.

**Single Shot MultiBox Detector (SSD)** – SSD được đề xuất vào năm 2015 bởi W. Liu và đồng nghiệp. SSD giới thiệu kỹ thuật phát hiện đa tham chiếu và đa độ phân giải, cải thiện độ chính xác đối tượng và đối tượng nhỏ. Ưu điểm của SSD vừa nhanh và vừa chính xác.

**RetinaNet** – Được ra mắt vào năm 2017 bởi T.-Y Lin và đồng nghiệp, giúp giải quyết vấn đề độ chính xác của “Two-stage detectors”. RestinaNet giới hiệu hàm mất mát “focal loss” giúp tập trung hơn các ví dụ khó, giúp phát hiện được các đối tượng khó mà vẫn duy trì được tốc độ cao.

**CorrerNet** – Được H. Law và các đồng nghiệp đề xuất vào 2019, CorrerNet đảm bảo độ chính xác bằng cách dự đoán các điểm chính (góc của bounding box).

**CenterNet** – Đề xuất bởi X. Zhou và đồng nghiệp vào năm 2019, xem xét đối tượng là một điểm duy nhất (tâm của đối tượng) và hồi quy tất cả các thuộc tính dựa trên điểm tâm tham chiếu. Nó loại bỏ các bước xử lý sau cùng, giúp cho mô hình đơn giản và linh hoạt.

**DETR (Detection Transformer)** – DETR được ra mắt vào năm 2020, sử dụng kiến trúc Transformer để dự đoán tập hợp đối tượng mà không cần sử dụng anchor boxes. Deformable DETR sau đó giải quyết vấn đề thời gian hội tụ và hiệu suất giới hạn đối với đối tượng nhỏ.

* + - 1. **Cơ sở lý thuyết về học sâu.**

Học sâu (Deep Learning) là một phân nhánh rộng thuộc học máy dựa trên mạng nơ ron nhân tạo kết hợp với việc học biểu diễn đặc trưng (representation learning). Học sâu được sử dụng để dạy máy tính xử lý dữ liệu theo cách được lấy cảm hứng từ bộ não con người. Mô hình học sâu có thể nhận diện nhiều hình mẫu phức tạp trong hình ảnh, văn bản, âm thanh và các dữ liệu khác để tạo ra thông tin chuyên sâu và dự đoán chính xác.

Các thuật toán học sâu là các mạng nơ ron được lập mô hình theo bộ não con người. Một bộ não con người chứa hàng triệu nơ ron được kết nối với nhau, làm việc cùng nhau để tìm hiểu và xử lý thông tin. Tương tự, các mạng nơ ron học sâu, hay mạng nơ ron nhân tạo, được tạo thành từ nhiều lớp nơ ron nhân tạo hoạt động cùng nhau trong máy tính. Các nơ ron nhân tạo là những module phần mềm được gọi là nút (node), sử dụng các phép toán để xử lý dữ liệu. Các mạng nơ ron nhân tạo là những thuật toán học sâu sử dụng các nút này để giải quyết các vấn đề phức tạp.



*Hình 4: Cấu trúc cơ bản của một mô hình học sâu.*

Một mạng nơ ron chuyên sâu sẽ có các thành phần sau.

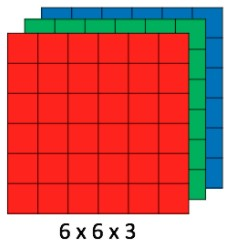
* Lớp đầu vào (Input Layer): Một mạng nơ ron nhân tạo sẽ có một số nút để nhập dữ liệu đầu vào.
* Lớp ẩn (Hidden Layer): Lớp đầu vào xử lý và chuyển dữ liệu đến các lớp sâu hơn trong mạng nơ ron. Các lớp ẩn này xử lý thông tin ở các cấp độ khác nhau, thích ứng với hành vi của mình khi nhận được thông tin mới. Các mạng học sâu có hàng trăm lớp ẩn để có thể phân tích một vấn đề từ nhiều góc độ khác nhau.
* Lớp đầu ra (Output Layer): Lớp đầu ra bao gồm các nút xuất dữ liệu. Các mô hình học sâu xuất ra đáp án “có” hoặc “không” nếu chỉ có hai nút trong lớp đầu ra. Mặt khác, các mô hình xuất ra nhiều đáp án hơn sẽ có nhiều nút hơn.

Học sâu thường được thực hiện thông qua học có giám sát, trong đó mô hình cố gắng dự đoán đầu ra dựa trên đầu vào và điều chỉnh trọng số sao cho sự chênh lệch giữa dự đoán và thực tế là nhỏ nhất.

Tuy nhiên, học sâu không chỉ giới hạn trong phạm vi học có giám sát. Có cả phương pháp học không giám sát và bán giám sát, nơi mô hình tự học cấu trúc của dữ liệu mà không cần nhãn.

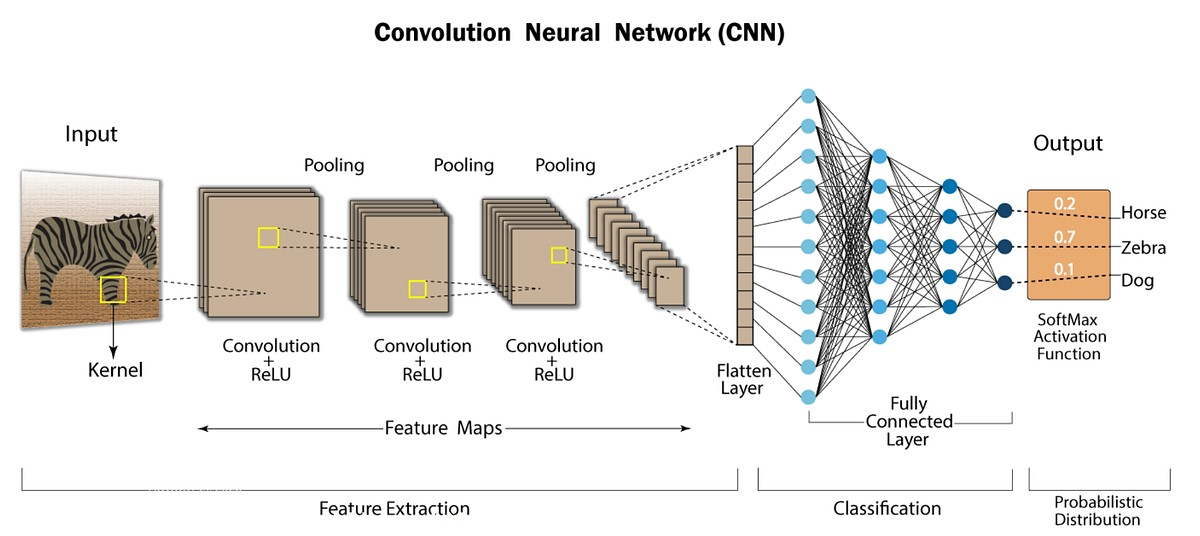
* + - 1. **Mạng nơ ron tích chập – CNN (Convolutional Neural Networks).**

CNN là một loại mạng nơ ron thị giác máy tính được thiết kế đặc biệt cho việc xử lý và nhận diện dữ liệu không gian như ảnh. CNN phân loại hình ảnh bằng cách lấy một hình ảnh đầu vào, xử lý và phân loại nó theo các hạng mục nhất định. Máy tính coi hình ảnh đầu vào là một mảng pixel và nó phụ thuộc và độ phân giải của hình ảnh. Dựa trên độ phân giải của hình ảnh, máy tính sẽ thấy H x W x D (H: Chiều cao, W: Chiều rộng, D: Độ dày).



*Hình 5: Hình ảnh là mảng ma trận RGB 6x6x3 (3 ở đây là giá trị RGB).*

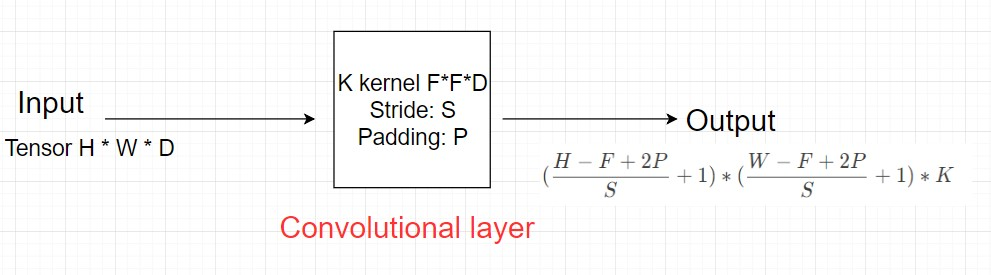
Về kỹ thuật, mô hình CNN để huấn luyện và kiểm tra, mỗi hình ảnh đầu vào sẽ chuyển nó qua một loạt các lớp tích chập (Convolutional) và các bộ lọc (Kernels) được áp dụng để trích xuất đặc trưng từ hình ảnh đầu vào, với hàm kích hoạt giữ lại thông tin quan trọng. Tiếp theo, lớp pooling giảm kích thước không gian của biểu diễn đặc trưng bằng cách chọn giá trị lớn nhất hoặc giá trị trung bình từ các vùng đặc trưng. Các lớp Fully Connected sau đó tạo ra biểu diễn phẳng để áp dụng softmax activation, chuyển đổi đầu ra thành xác suất dự đoán cho từng lớp. Mô hình này được huấn luyện thông qua việc điều chỉnh trọng số dựa trên hàm mất mát và kiểm tra dữ liệu mới để đánh giá hiệu suất.



*Hình 6: Cấu trúc mạng nơ ron tích chập – CNN.*

1. **Lớp tích chập – Convolutional Layer**

Là lớp đầu tiên để trích xuất các tính năng từ hình ảnh đầu vào. Tích chập duy trì mối quan hệ giữa các pixel bằng cách tìm hiểu các tính năng hình ảnh bằng cách sử dụng các ô vuông nhỏ của dữ liệu đầu vào.



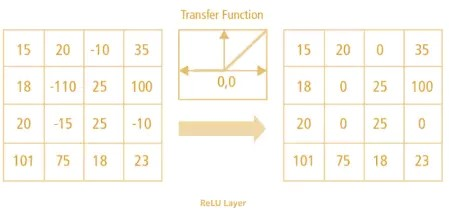
Một số khái niệm chính trong lớp kích hoạt là: Filter Map, Stride, Padding, Feature Map.

* Filter Map: CNN sử dụng Filter để áp dụng vào các vùng của hình ảnh. Những Filter này có thể xem là một ma trận 3 chiều gồm các con số là tham số đầu vào.
* Stride: Stride được hiểu là khi chúng ta dịch chuyển Fiter Map theo Pixel và dựa vào giá trị từ trái sang phải. Stride đơn giản biểu thị cho sự dịch chuyển này.
* Padding: Chính là những giá trị 0 được thêm vào lớp Input.
* Feature Map: Đây là kết quả được hiển thị sau mỗi lần Filter Map quét qua Input. Cứ mỗi lần quét qua sẽ thấy được sự xuất hiện của quá trình tính toán.

1. **Hàm phi tuyến – ReLU.**

ReLu là một hàm phi tuyến với đầu ra là

ReLu quan trọng bởi vì dữ liệu trong các vấn đề mà chúng ta tìm hiểu là các giá trị tuyến tính không âm.

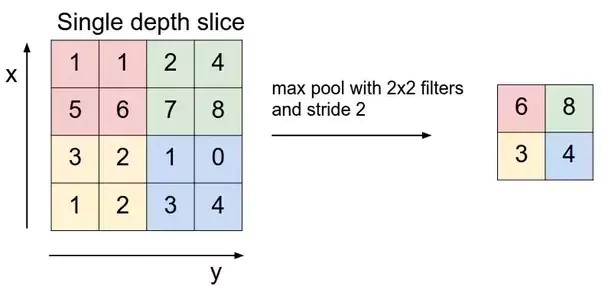


Có một số hàm phi tuyến khác như tanh, sigmoid cũng có thể thay thế cho ReLu. Nhưng hầu hết người ta dùng ReLu bởi vì nó có hiệu suất tốt.

1. **Lớp gộp – Pooling Layer.**

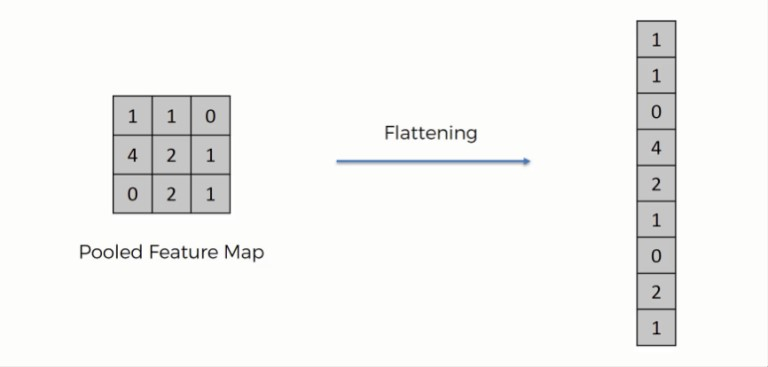
Lớp pooling sẽ giảm bớt số lượng tham số khi hình ảnh quá lớn. Không gian pooling còn được gọi là lấy mẫu con hoặc lấy mẫu suống làm giảm kích thước của mỗi map nhưng vẫn giữ lại thông tin quan trọng. Một số loại pooling:

* Max Pooling: Lấy lấy tử lớn nhất từ ma trận đối tượng.
* Average Pooling: Lấy tổng trung bình từ ma trận đối tượng.
* Sum Pooling: Tổng tất cả phần tử trong ma trận đối tượng.



1. **Fully connected layer.**

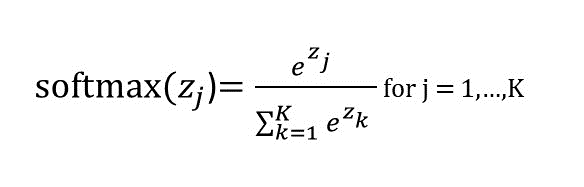
Sau khi ảnh được truyền qua nhiều Convolutional Layer và Pooling Layer thì model đã học được tương đối các đặc điểm của ảnh (ví dụ: đầu tank thuốc lá điện tử, khung thuốc lá điện tử,…) thì tensor của output của layer cuối cùng, kích thước H\*W\*D sẽ được chuyển về một vector kích thước HxWxD



Sau đó vector này được kết nối với một số lớp được kết nối đầy đủ giống như Mạng nơ ron nhân tạo và thực hiện các phép toán tương tự.

1. **Hàm kích hoạt softmax – softmax activation function.**

Sau khi đi qua các lớp được kết nối đầy đủ, lớp cuối cùng sử dụng chức năng kích hoạt softmax (thay vì ReLu) để nhận xác suất của đầu vào thuộc một lớp cụ thể (phân loại). Và cuối cùng, chúng ta có xác suất của đối tượng thuộc các lớp khác nhau.

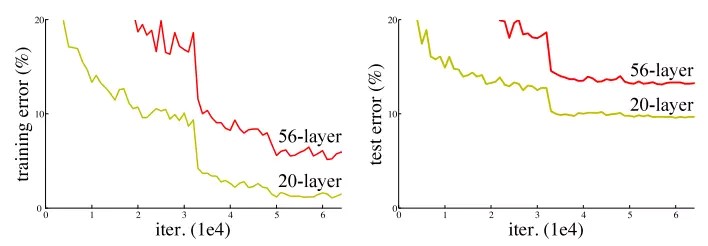


* + - 1. **Mạng Residual – ResNet.**

Một vấn đề xảy ra khi xây dựng mạng CNN với nhiều lớp tích chập sẽ xảy ra hiện tượng Vanishing Gradient dẫn tới quá trình học tập không tốt. Mạng ResNet là một mạng CNN được thiết kế để làm việc với hàng trăm lớp và gải quyết vấn đề trên.

1. **Vanishing Gradient.**

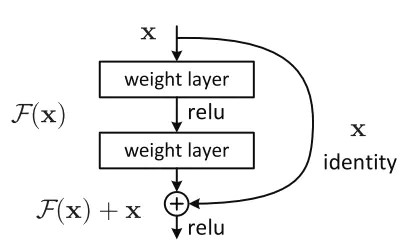
Vanishing Gradient là một vấn đề trong quá trình huấn luyện mạng nơ ron và đặc biệt là trong các mạng nơ ron nhiều lớp. Hiện tượng này xảy ra khi gradient của hàm mất mát giảm đáng kể khi lan truyền ngược (backpropagation) qua các lớp của mạng. Khi gradient giảm quá mức, các trọng số không được cập nhật đúng cách dẫn đến mô hình học chậm hoặc thậm chí không học được gì từ dữ liệu. Và mạng ResNet ra đời để giải quyết vấn đề này.



So sánh kiến trúc 20 lớp và 56 lớp

1. **Residual Block của ResNet.**

Giải pháp mà ResNet đưa ra là sử dụng kết nối “tắt” đồng nhất để xuyên qua một hay nhiều lớp. Một khối như vậy được gọi là một Residual Block như hình dưới đây.



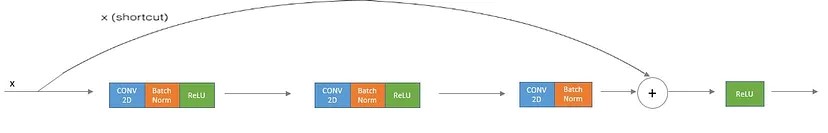
Hình () biểu diễn một khối dư (Residual Block) được sử dụng trong mạng ResNet. Với đầu vào là x là kết quả đầu ra từ lớp trước đó trong mạng. Xuất hiện một mũi tên cong xuất phát từ đầu và kết thúc tại cuối khối dư. Việc này sẽ bổ sung x vào đầu ra của layer và sẽ tránh được việc đạo hàm bằng 0 do Vanishing Gradient gây ra. Ở đây hàm H(x) là giá trị dự đoán, F(x) là giá trị thật của nhãn. Và giá trị mong muốn trong thực tế là H(x) bằng hoặc xấp xỉ F(x).

F(x) thu được từ x đầu vào:

1. **Các thành phần chính trong ResNet**.

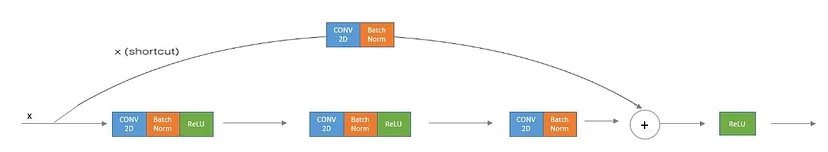
Có hai loại block chính được sử dụng trong ResNet, chủ yếu phụ thuộc vào kích thước đầu vào và đầu ra giống hay khác nhau.

* **Identity block:** là block tiêu chuẩn được sử dụng trong ResNet tương ứng với trường hợp đầu vào của hàm kích hoạt cùng kích thước với đầu ra của hàm kích hoạt.



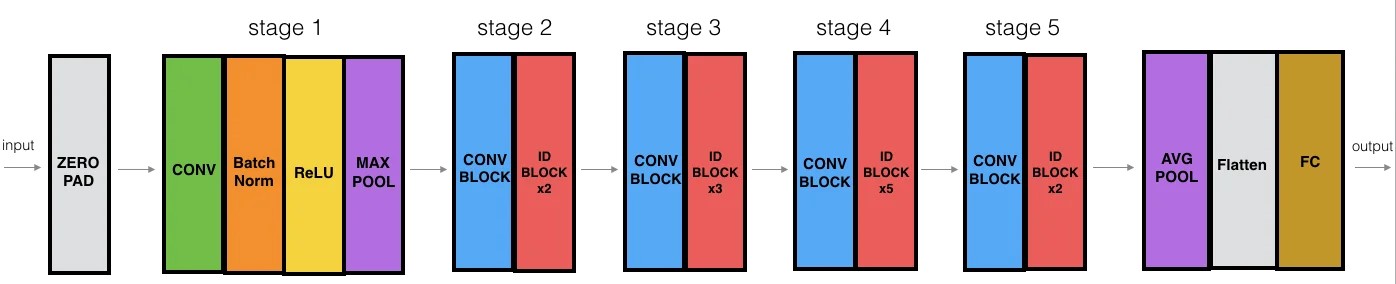
Hình. Ví dụ về Identity block.

* **Convolutional Block:** Khi kích thước đầu vào và đầu ra không khớp nhau thì một lớp convolutional sẽ được thêm vào đường dẫn shortcut. Việc này giúp đảm bảo giữ nguyên thông tin chiều sâu x, điều chỉnh kích thước của x phù hợp để có thể thêm vào đầu ra phần còn lại của Convolutional Block.



Hình. Ví dụ về Convolutional Block.

1. **Kiến trúc ResNet-50.**



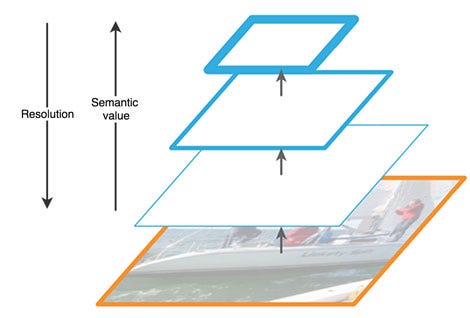
Hình. Kiến trúc mạng ResNet-50.

* + - 1. **Mạng Pyramid đặc trưng – Feature Pyramid Network (FPN).**

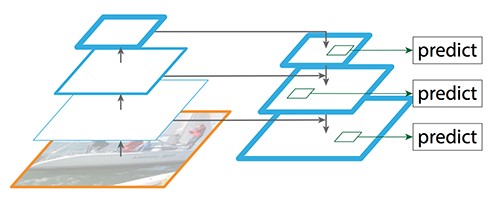
1. **Feature Pyramid Network – FPN.**

Feature Pyramid Network (FPN) được thiết kế để xử lý vấn đề của việc trích xuất đặc trưng ở nhiều tỉ lệ độ phân giải khác nhau trong hình ảnh. Mục tiêu chính của FPN là tạo ra một pyramid đặc trưng, nơi mỗi cấp độ của pyramid tương ứng với một tỉ lệ độ phân giải khác nhau. Điều này giúp cho mô hình có khả năng phát hiện đối tượng ở các kích thước khác nhau. FPN có 2 phần chính đó là **Bottom-up** và **Top-down**.

Bottom-up pathway là một mạng CNN thông thường để thực hiện feature extraction. Càng lên cao, độ phân giải càng giảm và kích thước của feature maps cũng giảm theo vì vậy nên giá trị thông tin về ngữ cảnh càng cao (semantic value).



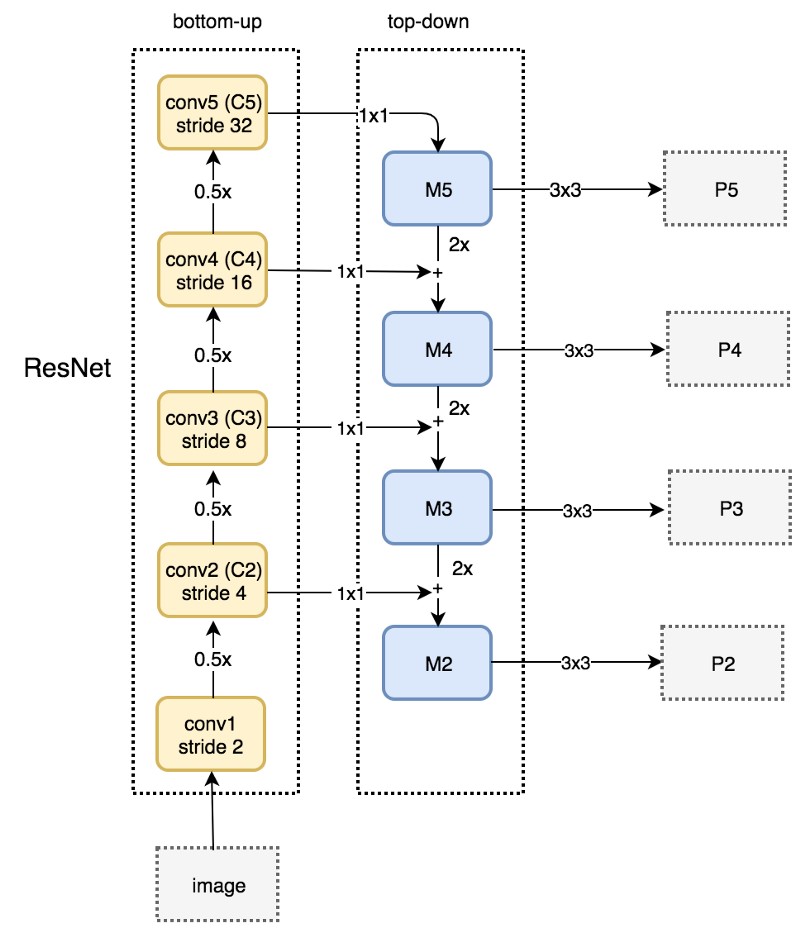
FPN sử dụng thêm Top-down pathway để tránh nhược điểm khó phát hiện các vật thể nhỏ của các thuật toán chỉ sử dụng Bottom-up pathway. Top-down pathway được sử dụng nhằm xây dựng các layer có độ phân giải cao từ các layer có ngữ nghĩa cao.



Có một vấn đề nghiêm trọng trong quá trình xây dựng các layer từ top xuống bottom đó là mất mát thông tin của các đối tượng. Ví dụ như khi một đối tượng nhỏ lên top sẽ không thấy nó, và từ top đi ngược lại sẽ không thể tái tạo được. Để khắc phục vấn đề này, FPN sẽ tạo các kết nối skip connection giữa các reconstruction layer và các feature map để giúp quá trình dự đoán vị trí đối tượng tốt hơn và hạn chế tối đa việc mất mát thông tin.

1. **Cách hoạt động của Bottom-up pathway và Top-down pathway.**

**Bottom-up pathway.**



*Hình 6: Mô hình diễn giải chi tiết đường đi theo bottom-up và top-down. Trong đó P2, P3, P4 là các pyramid của các feature map.*

Theo như mô hình trên, Bottom-up pathway chứa rất nhiều convolution modules, mỗi modules có một vài conv layers. Theo chiều mũi tên đi lên thì spatial resolution giảm một nửa (stride tăng lên gấp đôi). Đầu ra của mỗi convolution module được đánh số là Ci và đươc sử dụng trong Top-down pathway.

**Top-down pathway.**

Còn ở Top-down pathway, đầu tiên chúng ta dùng convolution layer 1x1 để giảm số channel của C5 xuống 256-d để tạo ra M5. M5 là feature map đầu tiên được sử dụng cho object detection và cũng là P5.

Theo mũi tên chiều đi xuống của top-down pathway, chúng ta upsample layer phía trước M5 lên 2 lần bằng cách sử dụng phương pháp nearest neightbors upsampling. Sau đó lại áp dụng convolution layer 1x1 cho feature map C4 rồi cộng chúng lại với nhau để nhận được feature map M4. Sau đó áp dụng convolution layer 3x3 cho M4 để nhận được P4 nhằm giúp giảm hiệu ứng răng cưa – aliasing effect khi kết hợp với upsample layer từ M5. Quy trình này được lặp lại cho P3 và P2. Bởi vì classifier cùng box regressor được chia sẻ cho cho mỗi output feature maps, do đó tất cả các pyramid feature maps (P5, P4, P3, P2) đều có 256-d channels.

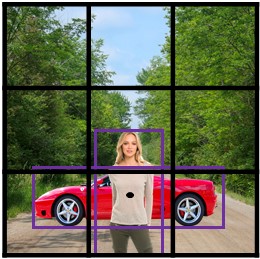
* + - 1. **Mạng đề xuất khu vực – Region Proposal Network (RPN).**

Mạng đề xuất khu vực – RPN là một loại mạng nơ ron dự đoán cả giới hạn (bounds) và khả năng xuất hiện của một đối tượng trong hình ảnh. Về cơ bản, RPN cố gắng xác định vị trí của các đối tượng trong hình ảnh bằng cách đề xuất một vùng của hình ảnh tương ứng với một đối tượng.

Dưới đây là mô tả cách RPN hoạt động.

1. **Feature Map.**

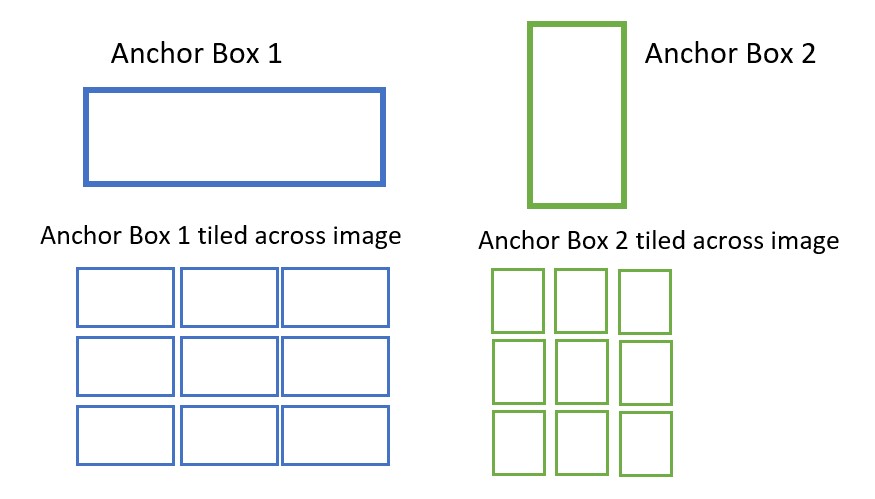
Đầu vào của RPN là một hình ảnh được biểu diễn dưới dạng mảng đa chiều Height x Width x Depth.



Sau đó mảng đa chiều này được truyền qua Convolutional Neural Network (CNN). Layer cuối cùng của CNN trả về convolutional feature map được sử dụng ở bước sau.

1. **Anchor Boxes (Anchor Proposals).**

Anchor box là một hộp bao quanh đối tượng. Trong tập dữ liệu huấn luyện, các Anchor box được xác định và gán nhãn theo lớp của đối tượng đó. Các box có chiều cao và chiều rộng nhất định được xác định dựa trên chiều cao và chiều rộng của các đối tượng trong tập dữ liệu huấn luyện.



Các anchor box được xác định theo từng vị trí trong Convolutional Feature Map được tạo từ ảnh đầu vào. Nó sẽ trả về các anchor box được xếp chồng lên nhau trên toàn bộ hình ảnh. Mỗi anchor box sẽ tạo ra dự đoán xem nó có chứa đối tượng cần tìm không. Dự đoán sẽ ở dạng nhị phân. Giả sử nếu anchor box có chứa đối tượng thì sẽ được phân loại là “foreground” còn ngược lại sẽ là “background”.

1. **Phân loại Anchor Box.**

Để phân loại từng anchor box, FPN sử dụng thước đo Intersection over Union (IoU). Nó được sử dụng để mô tả mức độ chồng chéo của anchor box thực tế với đối tượng mong muốn. Vùng chồng chéo càng lớn thì IoU càng lớn. Nếu anchor box có IoU lớn hơn 0.5 nó sẽ được coi là “foreground”.

1. **Bounding Box Regression Layer.**

Regression Layer cố gắng làm cho tọa độ của anchor box xung quanh đối tượng được chính xác hơn. Tọa độ của đối tượng là chiều rộng, chiều cao và chiều sâu (hay còn gọi là trung tâm).

Loss function của Region Proposal Network là tổng của classification (cls) loss và regression (reg) loss.

Classification Loss là đánh giá xem mỗi anchor box có chứa đối tượng không và nếu có thì đối tượng đó thuộc lớp nào. Đối với mỗi anchor box, FPN dự đoán xác xuất thuộc về lớp foreground hay lớp background. Classification Loss sử dụng hàm entropy loss để so sánh giữa xác suất dự đoán và xác suất thực tế.

Regression Loss đo lường sự chênh lệch giữa vị trí (đặc biệt là anchor box) được dự đoán cho lớp foreground và vị trí thực tế của anchor box chứa đối tượng trong dữ liệu đào tạo. RPN dự đoán các thay đổi (dịch chuyển và tỷ lệ) cần áp dụng cho anchor box để chính xác hơn trong việc dự đoán đối tượng.

Loss Function được tính toán bằng công thức sau.

Trong đó:

* i – vị trí của anchor box.
* p – xác xuất anchor box có chứa đối tượng hay không.
* t – vector chứa 4 tọa độ tham số của anchor box được dự đoán.
* \* – đại diện cho box chứa đối tượng thực tế trong tập dữ liệu. L đại diện cho log loss của 2 lớp cls và reg. Ncls và Nreg đã được chuẩn hóa.
* (mặc định) và nó chia tỉ lệ cho classifier và regressor ở cùng cấp độ.
  + - 1. **Một số thuật toán trong lĩnh vực nhận phát hiện đối tượng trong thị giác máy tính.**

1. **RCNN.**

Ý tưởng của thuật toán RCNN có thể gói gọn trong 2 bước.

* Bước 1: Dùng thuật toán Selective Search để lấy ra khoảng 2000 bounding box trong ảnh đầu vào mà có khả năng chứa đối tượng.
* Bước 2: Với mỗi bounding box, RCNN sẽ xác định xem đó là đối tượng nào.

**Selective Search algorithm.**

Đầu vào của thuật toán là một ảnh màu, đầu ra là khoảng 2000 region proposal (bounding box) mà có khả năng chứa đối tượng.

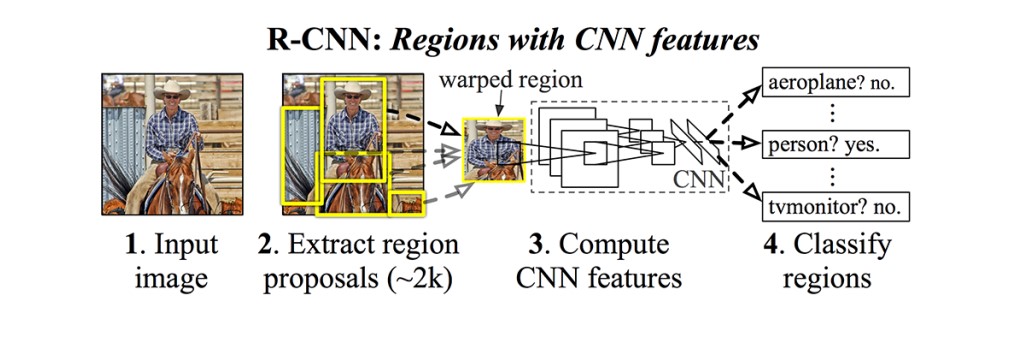
Đầu tiên, ảnh được segment qua thuật toán Graph Based Image Segmentation. Tiếp theo các vùng màu được nhóm với nhau dựa trên độ tương đồng về màu sắc, hướng gradient, kích thước,…



Vì mỗi đối tượng có thể chứa nhiều hơn một màu và các đối tượng có thể bị che một phần. Vì vậy cần nhóm các vùng màu với nhau để làm region proposal. Cuối cùng các region proposal được xác định dựa trên nhóm vùng màu.

**Phân loại Region Proposal.**

Sau khi thu được 2000 region proposal từ thuật toán Selective Search thì bài toán trở thành phân loại ảnh cho các region proposal đó. Trong đó có thể có rất nhiều region proposal không chứa đối tượng nào. Vì vậy chúng ta cần thêm một lớp background (không chứa đối tượng nào).



Sau đó các region proposal được resize lại về đúng kích thước và thực hiện transfer learning với feature extractor, sau đó các đặc trưng được trích xuất được cho vào thuật toán SVM để phân loại ảnh.

**Vấn đề của RCNN.**

RCNN mất nhiều thời gian để training do phải phân loại 2000 region proposal và nó không thể áp dụng cho real-time.

1. **Fast RCNN.**

Tương tự như RCNN, Fast RCNN vẫn dùng selective search để lấy ra các region proposal. Tuy nhiên nó không tách 2000 region proposal ra khỏi ảnh mà nó thực hiện image classification cho mỗi ảnh. Fast RCNN cho cả bức ảnh vào ConvNet (chứa một vài convolutional layer + max pooling layer) để tạo ra convolutional feature map.

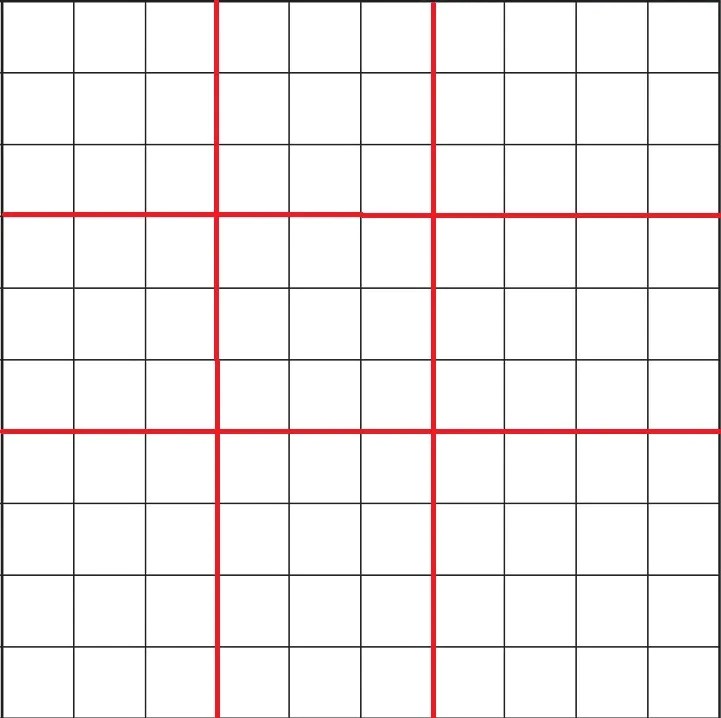
Sau đó các vùng region proposal được lấy ra tương ứng từ convolutional feature map. Tiếp theo đó được Flatten và thêm 2 Fully connected layer (FCs) để dự đoán lớp của region proposal và giá trị offset values của bounding box.

Vì kích thước của các region proposal khác nhau nên khi Flatten sẽ ra các vector có kích thước khác nhau. Điều này khác với RCNN vì nó đã resize các region proposal về cùng một kích thước. Để khắc phục vấn đề này, Fast RCNN đã sử dụng Region of Interest (RoI) pooling.

**Region of Interest (RoI) pooling.**

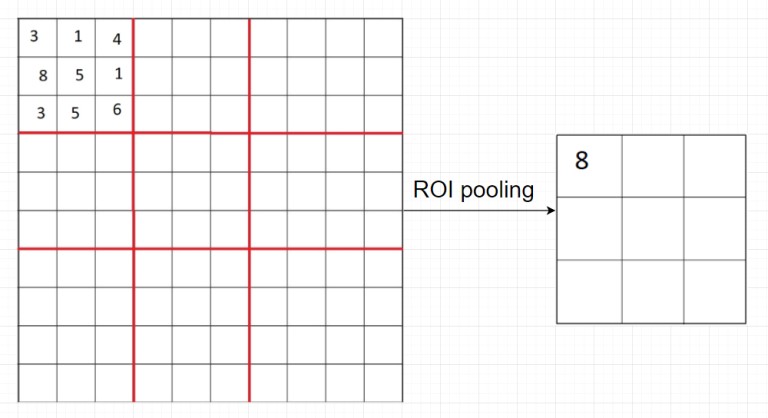
RoI pooling là một dạng của pooling layer. Điểm khác so với max pooling hay average pooling là dùng tensor input có kích thước như nào thì ROI pooling luôn cho ra output có kích thước cố định được định nghĩa trước. Gọi đầu vào của RoI pooling kích thước m\*n và đầu ra có kích thước h\*k (thông thường h và k có kích thước nhỏ).

* Ta chia chiều rộng thành h phần, h-1 phần có kích thước m/h, phần cuối có kích thước m/h + m%h.
* Tương tự ta chia chiều dài thành k phần, k-1 phần đầu tiên có kích thước n/k, phần cuối có kích thước n/k + n%k.



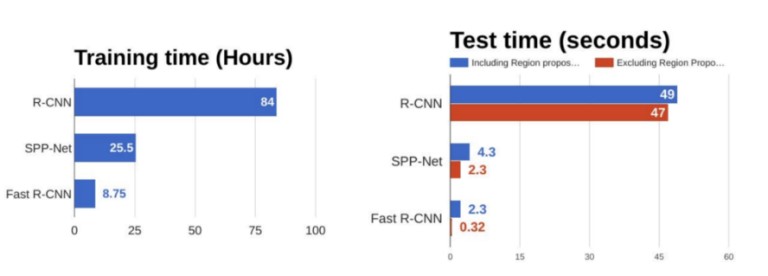
Ví dụ đầu vào có kích thước m=n=10, h=k=3.

Với mỗi khối được tạo ra bởi các đường phân cách, ta thực hiện max pooling lấy ra một giá trị.



Thực hiện RoI pooling.

**Đánh giá Fast RCNN.**

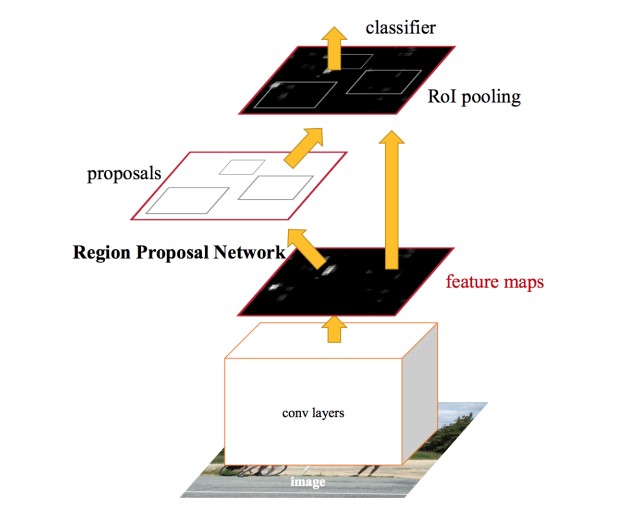


So sánh hiệu suất của Fast RCNN với RCNN và SPPNet.

Tuy Fast RCNN đã có cải thiện hơn RCNN về tốc độ do thực hiện feature map với cả ảnh và sau đó lấy region proposal ra từ feature map. Tuy nhiên Fast RCNN vẫn rất chậm ở test time như trên ảnh vì nó sử dụng thuật toán Selective Search → Cần có giải pháp thay thế thuật toán này.

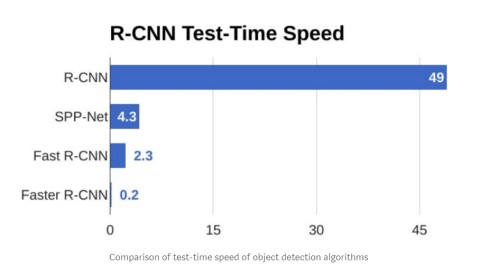
1. **Faster RCNN.**

Faster RCNN không dùng thuật toán selective search để lấy ra các region proposal, mà nó thêm một mạng CNN mới gọi là Region Proposal Network (RPN)[1.2.2.7] để tìm ra các region proposal.



Mô hình Faster RCNN.

Đầu tiên cả bước ảnh sẽ được cho qua pre-trained model để lấy feature map. Sau đó feature map được dùng cho Region Proposal Network để lấy các regon proposal. Sau khi lấy được các vị trí các region proposal thì thực hiện tương tự như Fast RCNN.



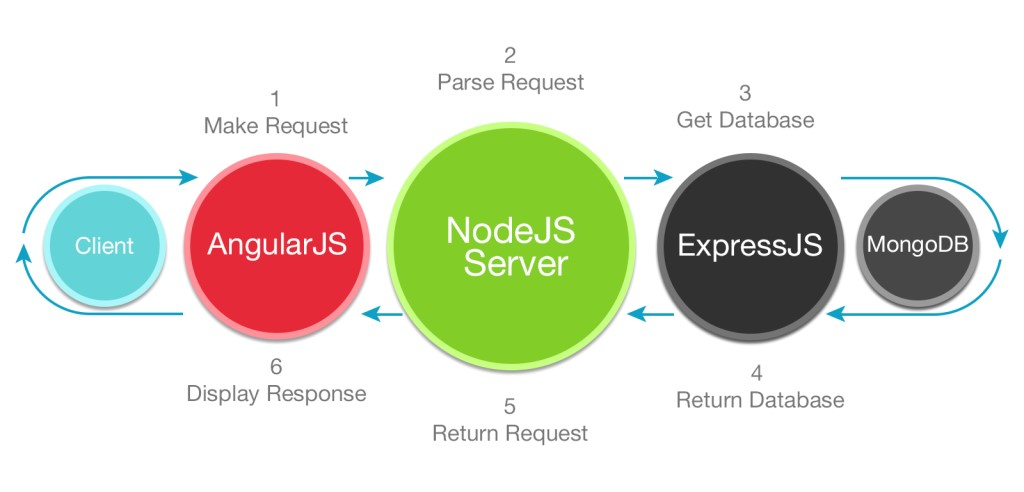
So sánh tốc độ giữa RCNN, SPPNet, Fast RCNN và Faster RCNN.

Theo biểu đồ so sánh tốc độ giữa các thuật toán trên, thuật toán Faster RCNN mang lại tốc độ nhanh nhất và có thể tiệm cận với real-time object detection.

* 1. **Các công nghệ sử dụng.**
     1. **MEAN.js stack.**

Thuật ngữ MEAN.js là một giải pháp mã nguồn mở JavaScript đầy đủ được sử dụng để xây dựng các trang web và ứng dụng web động. MEAN là từ viết tắt của MongoDB, ExpressJS, NodeJS và AngularJS là các thành phần chính của MEAN.js stack.

Về cơ bản nó được phát triển để giải quyết các vấn đề phổ biến khi kết nối các khung đó (MongoDB, ExpressJS, NodeJS, AngularJS. Stack nghĩa là sử dụng cơ sở dữ liệu, máy chủ web ở back end, ở giữa sẽ có logic và khả năng kiểm soát ứng dụng cũng như sự tương tác với người dùng ở front end.



*Hình 1: Cấu trúc MEAN.js stack và các luồng hoạt động.*

Lựa chọn MEAN.js cho hệ thống gán nhãn được thực hiện với những lý do cụ thể, đặc biệt tập trung vào sự linh hoạt và hiệu suất trong xử lý dữ liệu đa dạng để phát triển hệ thống gán nhãn. Điều đó được thể hệ qua các thành phần bên trong nó:

1. **MongoDB.**

MongoDB là một database hướng tài liệu, một dạng NoSQL database. Vì vậy, MongoDB sẽ tránh cấu trúc table-based của relational database để thích ứng với các tài liệu như JSON có một schema rất linh hoạt gọi là BSON.

Dữ liệu trong MongoDB được lưu trữ phi cấu trúc, không có tính ràng buộc, toàn vẹn nên tính sẵn sàng cao. Điều này tạo điều kiện lưu trữ các thông tin về ảnh cũng như thông tin các đối tượng được gán nhãn trong ảnh vào JSON file một cách nhanh chóng và khi truy vấn có thêm thông tin về gán ảnh vào đối tượng một cách dễ dàng với định dạng JSON.

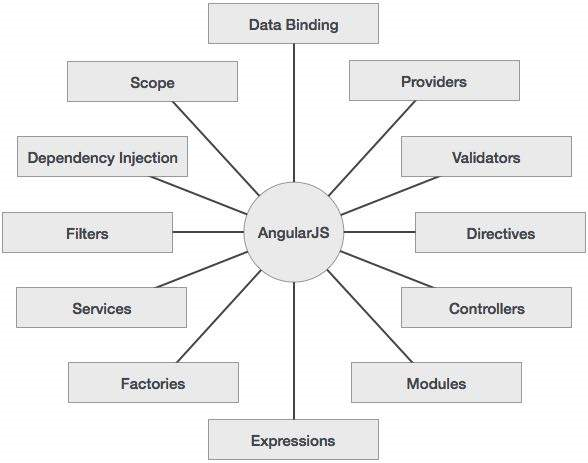
Ngoài ra với đặc tính hiệu suất cao, dễ dàng mở rộng lưu trữ và dữ liệu được ghi đệm lên RAM giúp hạn chế việc truy cập vào ổ cứng của MongoDB có thể cho phép hệ thống lưu trữ rất nhiều ảnh mà vẫn đảm bảo được tốc độ truy vấn dữ liệu nhanh.

1. **ExpressJS.**

ExpressJS là một khung ứng dụng web NodeJS được tối giản và linh hoạt, cung cấp một bộ tính năng mạnh mẽ để phát triển các ứng dụng web và di động. Nó tạo điều kiện cho sự phát triển nhanh chóng của các ứng dụng Web dựa trên NodeJS. Rất dễ dàng để cấu hình và tùy chỉnh, cho phép xây dựng các ứng dụng, module một cách dễ dàng nhanh chóng. Nó chỉ định các routes của một ứng dụng tùy thuộc và HTTP method và URLs. Và nó cũng cho phép thiết lập middlewares để phản hồi HTTP requests.

1. **AngularJS.**

AngularJS là một framework phát triển web mã nguồn mở được phát triển bởi Google, được sử dụng để phát triển ứng dụng web đơn trang (SPAs). Được viết bằng Javascript, Angular cung cấp một cơ sở hạ tầng mạnh mẽ để phát triển ứng dụng web phức tạp và linh hoạt. Angular được tổ chức theo mô hình MVC giúp tách rõ ràng các thành phần của ứng dụng, tạo điều kiện thuận lợi cho quản lý mã nguồn và bảo trì.

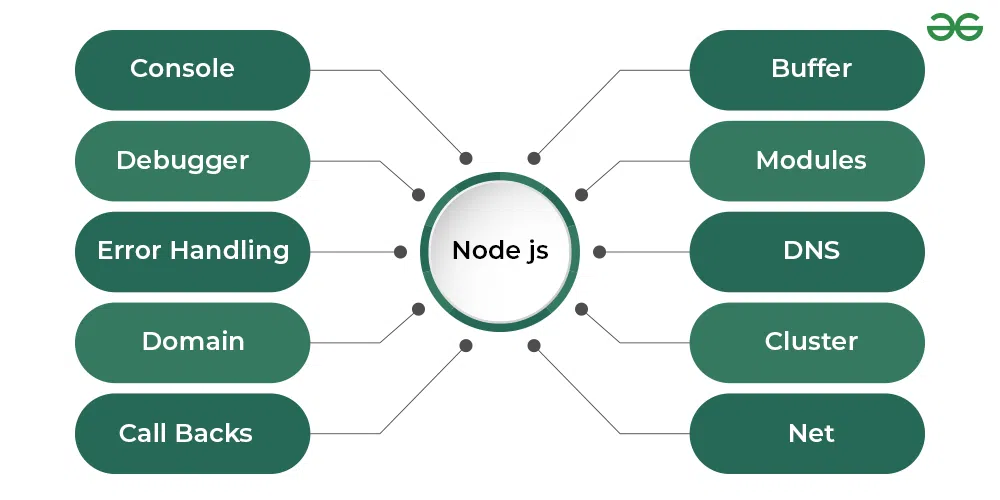


*Hình 2: Các thành phần cốt lõi của AngularJS.*

1. **NodeJS.**

NodeJS là một nền tảng phía máy chủ được sử dụng để phát triển các ứng dụng web như các trang video clip, forum và đặc biệt là các trang mạng xã hội phạm vi hẹp. Nó cung cấp một thư viện phong phú gồm nhiều JavaScript module khác nhau giúp đơn giản hóa việc phát triển ứng dụng web bằng NodeJS ở mức độ lớn. Nó được xây dựng trên công cụ JavaScript V8 của Google Chrome vì vậy nó thực thi mã rất nhanh.

NodeJS = Runtime Environment + JavaScript Library.



*Hình 3: Một số thành phần quan trọng trong NodeJS.*

* + 1. **Python.**

Python là một ngôn ngữ lập trình thông dịch, hướng đối tượng và cấp cao với ngữ nghĩa động được phát triển bởi Guido van Rossum. Ban đầu được ra mắt vào năm 1991. Python nổi tiếng với việc là ngôn ngữ thân thiện với người mới học, thay thế Java để trở thành ngôn ngữ giới thiệu phổ biến nhất bởi vì nó giảm bớt nhiều phức tạp cho người dùng, giúp người mới có thể tập trung hoàn toàn vào việc nắm vững các khái niệm lập trình thay vì chi tiết nhỏ.

Python có rất nhiều ưu điểm như: Đơn giản và dễ đọc, Đa năng, Thư viện và Framework phong phú, Hỗ trợ Học Máy và Trí Tuệ Nhân Tạo rất nhiều, Hỗ trợ đa nền tảng,…

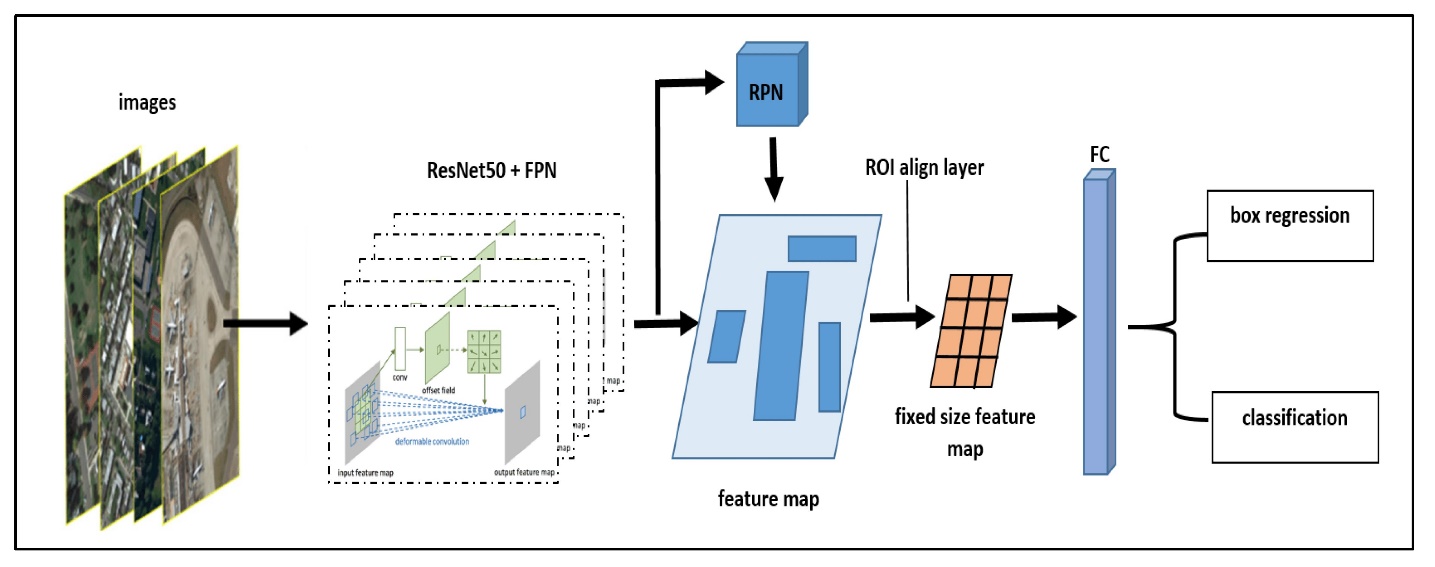
* + 1. **Thư viện Pytorch.**

Pytorch là một Framework học máy mã nguồn mở dựa trên ngôn ngữ lập trình Python và thư viện Torch. Torch là một thư viện máy học mã nguồn mở được sử dụng để tạo ra các Deep Neural Network và được viết bằng ngôn ngữ kịch bản Lua. Đây là một trong những nền tảng được ưa chuộng cho nghiên cứu học sâu. Framework này được xây dựng để tăng tốc quá trình từ việc nghiên cứu mô hình đến triển khai. Pytorch nổi tiếng với cú pháp đơn giản và trực quan, giúp người sử dụng dễ dàng hiểu và điều chỉnh mô hình của mình. Được phát triển bởi Facebook AI Research (FAIR), Pytorch ngày càng trở thành một trong những công cụ ưu tiên cho cộng đồng nghiên cứu và phát triển học máy.

* + 1. **Mô hình Faster RCNN with a ResNet50–FPN backbone.**

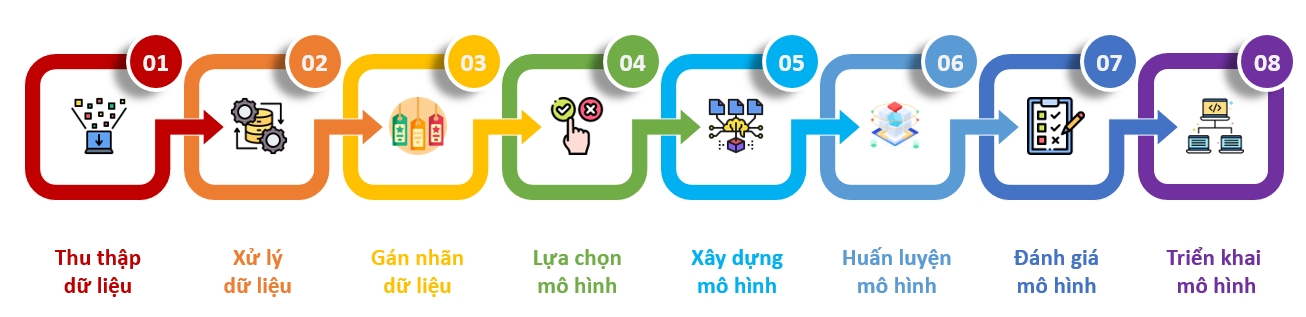
Faster RCNN with a ResNet50–FPN backbone là một kiến trúc mạng nơ ron được thiết kế để thực hiện nhiệm vụ phát hiện đối tượng trong hình ảnh. Kiến trúc này tích hợp các thành phần chính bao gồm ResNet50, một mạng nơ ron sâu với khối Residua để trích xuất đặc trưng và FPN – một cơ chế tạo ra pyramid đặc trưng để xử lý đối tượng ở nhiều tỷ lệ khác nhau.

Các thành phần này hợp nhất với nhau để tạo ra một mô hình phát hiện đối tượng hiệu quả và linh hoạt, phù hợp cho nhiều ứng dụng trong lĩnh vực thị giác máy tính.



Cấu trúc mô hình Faster RCNN with a ResNet50–FPN backbone.

1. **XÂY DỰNG MÔ HÌNH HỌC SÂU.**

****

Hình. Các bước trong quá trình xây dựng mô hình học sâu.

* 1. **Thu thập dữ liệu.**

Thu thập dữ liệu là một bước quan trọng trong quá trình xây dựng mô hình học sâu. Việc này giúp đảm bảo mô hình có đủ dữ liệu đa dạng để học tập và nhận diện chính xác các đối tượng thuốc lá điện tử trong nhiều tình huống khác nhau. Bằng cách sử dụng các nguồn dữ liệu hình ảnh đáng tin cậy và đa dạng trên mạng, từ những hình ảnh quảng cáo thuốc lá điện tử, cho đến những hình ảnh từ video thực tế trong việc hút thuốc lá điện tử giúp cho mô hình học sâu được xây dựng một cách tốt nhất.

1. **Thu thập ảnh trên mạng.**

Công cụ được sử dụng để tìm kiếm hình ảnh thuốc lá điện tử trên mạng là Google Images.

Các từ khóa như “electronic cigarette”, “vape”, “thuốc lá điện tử” được sử dụng để tìm kiếm những hình ảnh đa dạng và bao quát về thuốc lá điện tử. Ngoài ra tên của các loại thuốc lá điện tử hiện đang được bày bán tại Việt Nam cũng như quốc tế trên các website thương mại điện tử cũng được dùng để tìm kiếm nhằm đa dạng loại hình sản phẩm.

Việc này giúp mô hình học được cách nhận diện các đặc trưng của từng loại sản phẩm, từ thiết kế đến màu sắc, làm tăng khả năng nhận diện của mô hình.

1. **Thu thập hình ảnh từ những video thực tế.**

Những video thực tế chủ yếu được tìm kiếm và sử dụng để thu thập ảnh từ Youtube và Facebook. Hai nền tảng này chứa rất nhiều những video chất lượng cao và chứa những ảnh sử dụng thuốc lá điện tử trong thực tế.

Thu thập hình ảnh từ những video đó bằng cách tua đến những phân cảnh người dùng sử dụng thuốc lá điện tử và sử dụng phần mềm Snip & Sketch để chụp lại từ màn hình.

Việc lấy các hình ảnh từ các khung hình có góc chụp và môi trường khác nhau đảm bảo cho tập dữ liệu được sử dụng để xây dựng mô hình có chất lượng cao, phong phú và đa dạng. Điều này giúp mô hình học sâu hiệu quả trong việc nhận diện thuốc lá điện tử từ nhiều góc cạnh khác nhau.

* 1. **Tiền xử lý dữ liệu.**

1. **Chuyển đổi định dạng và tối ưu hóa ảnh.**

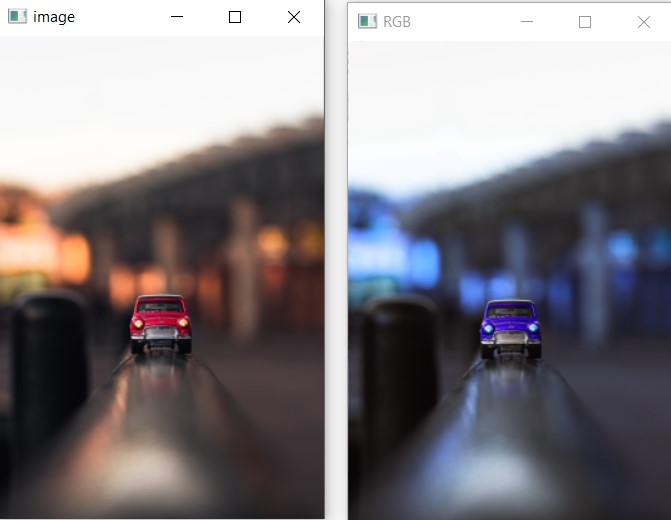
Công cụ **ImageMagick** được sử dụng để chuyển đổi định dạng ảnh. Vì quá trình thu thập hình ảnh từ rất nhiều nguồn trên mạng với nhiều định dạng khác nhau (jpg, png, webp, jpeg,…) nên phải chuyển về một kiểu định dạng để có thể dễ dàng xử lý và tối ưu hóa ảnh hơn. Định dạng được chọn là png. Quá trình này giúp cho các ảnh sau khi chuyển đổi có thể giữ nguyên tất cả các tùy chọn của ảnh gốc, bao gồm cả chất lượng và kích thước.

Sau khi chuyển đổi định dạng, công cụ **PNGCRUSH** được sử dụng để tối ưu hóa hình ảnh. Ở bước này, **PNGCRUSH** thực hiện loại bỏ rất cả các phần phụ trợ ngoại trừ tRNS và gAMA (loại bỏ một phần của thông tin về màu sắc và bảng màu không cần thiết) và giúp giảm kích thước bằng cách loại bỏ các thông tin không cần thiết.

1. **Chuyển đổi màu trong hình ảnh từ BGR sang RGB.**

Khi làm việc với hình ảnh trong thư viện OpenCV, thư viện thường sử dụng định dạng màu BGR (Blue – Green – Red) mặc định. Tuy nhiên trong thư viện Pytorch, định dạng màu chung được sử dụng là RGB (Red – Green – Blue).

Việc chuyển đổi định dạng màu từ BGR sang RGB là quy trình cần thiết để đảm bảo tính nhất quán trong quá trình xây dựng mô hình. Việc chuyển đổi diễn ra bằng cách hoán đổi vị trí của thành phần màu, tức là chuyển đổi từ (B, G, R) sang (R, G, B).



Hình: Hình ảnh được chuyển đổi từ định dạng BGR sang RGB.

1. **Chuẩn hóa giá trị pixel.**

Sau khi chuyển đổi định dạng màu sang RGB, hình ảnh được biểu diễn bởi ba kênh màu tương ứng với Red, Green và Blue, mỗi kênh màu sẽ có giá trị pixel riêng cho từng điểm ảnh và giá trị này nằm trong khoảng [0, 255]. Mỗi một pixel trong ảnh có thể được biểu diễn dưới dạng vector (R, G, B).

Nhằm giúp tối ưu hóa quá trình huấn luyến mô hình, giá trị của pixel sẽ được chuyển về miền [0, 1]. Việc này làm cho giá trị đầu vào ổn định và tăng tính nhất quán trong quá trình huấn luyện. Đồng thời, chuẩn hóa giá trị pixel cũng đảm bảo tính tương thích của dữ liệu với các mô hình đã được huấn luyện sẵn và giúp tránh được các vấn đề liên quan đến gradient hay activation function.

1. **Tăng cường dữ liệu.**

Phương pháp Horizontal Flip là một cách thức giúp lật hình ảnh theo chiều ngang, được sử dụng nhằm làm tăng sự đa dạng về dữ liệu hình ảnh. Việc lật ảnh như vậy giúp mô hình học được các biểu diễn, đặc trưng chung của hình ảnh và tránh được việc overfitting từ tập dữ liệu huấn luyện.

Horizontal Flip làm thay đổi vị trí của đối tượng trong ảnh một cách tự nhiên, giúp mô hình trở nên robust hơn với các biến đổi không gian của đối tượng trong dữ liệu thực tế.



Hình: Hình ảnh trước và sau khi sử dụng Horizontal Flip.

* 1. **Gán nhãn dữ liệu.**

Gán nhãn dữ liệu là một bước đóng vai trò rất quan trọng trong việc xây dựng và huấn luyện mô hình. Gán nhãn dữ liệu về cơ bản là quá trình xác định các đối tượng trong hình ảnh sau đó khoanh vùng đối tượng bằng một bounding box bằng cách thủ công và kèm theo lớp mà đối tượng đó thuộc về. Mục đích của việc gán nhãn là giúp mô hình học hỏi từ dữ liệu được gán nhãn, tối ưu hóa các tham số để thực hiện nhiệm vụ và đưa ra dự đoán chính xác.

1. **Chọn nhãn lớp cho đối tượng cần gán nhãn.**

Khi xác định được vị trí của đối tượng trong ảnh, chúng ta sẽ quyết định đối tượng đó thuộc nhãn lớp nào.

1. **Tạo Bounding Boxes (Hộp giới hạn).**

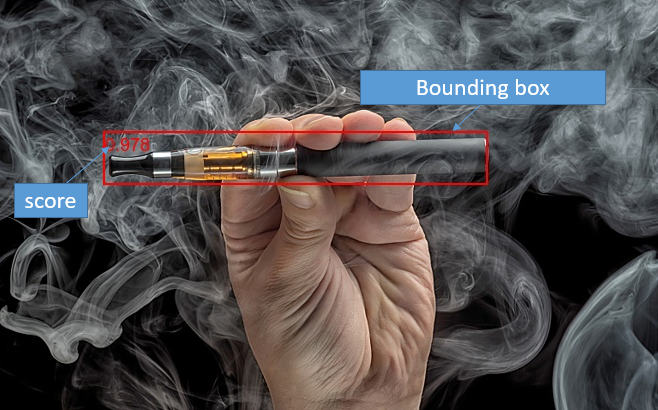
Bounding boxes được tạo ra để bao quanh đối tượng trong hình ảnh. Mỗi bounding box được đặc trưng bởi bốn điểm x\_min, y\_min, x\_max, y\_max. Bốn điểm này có thể giúp mô hình xác định được vị trí của đối tượng cũng như chiều dài và chiều rộng của đối tượng trong ảnh.



Hình: Tạo các bounding box và gán nhãn dữ liệu cho các đối tượng.

* 1. **Lựa chọn mô hình.**

Bài toán nhận diện thuốc lá điện tử yêu cầu mô hình nhận diện một cách chính xác nhiều đối tượng khác nhau trong mỗi hình ảnh. Để giải quyết vấn đề này chúng ta cần một mô hình học sâu có khả năng nhận diện nhiều đối tượng đồng thời với độ chính xác cao. Vì vậy mô hình Faster RCNN with ResNet50-FPN backbone đã giới thiệu ở [1.2.1.8.c] sẽ được sử dụng để huấn luyện trên tập dữ liệu được gán nhãn và dự đoán, nhận diện đối tượng thuốc lá điện tử trong thực tế.



Hình. Kết quả thu được sau khi dự đoán bằng mô hình Faster RCNN.

* 1. **Xây dựng mô hình.**

Sau khi đã có tập dữ liệu bao gồm các hình ảnh và bounding boxes chứa đối tượng thu được từ những bước trước đó thì ở bước này sẽ triển khai xây dựng mô hình để có thể huấn luyện trên tập dữ liệu này.

1. **Chuẩn bị dữ liệu.**

Một hình ảnh trong tập dữ liệu sẽ có các thành phần sau khi được chuẩn bị dữ liệu cho mô hình:

* Hình được được tiền xử lý dữ liệu và được lưu dưới định dạng Tensor.
* Các bounding box sẽ được lấy ra từ một file XML (Ngôn ngữ đánh dấu mở rộng) được lưu ở bước gán nhãn. Từng bounding box chứa x\_min, x\_max, y\_min, y\_max và nhãn (label) của đối tượng đó. Từ 4 điểm trên có thể tính được area (diện tích khu vực) của đối tượng đó.
* Mỗi bounding box sẽ có một giá trị iscrowd để xác định xem liệu các đối tượng trong ảnh có chồng lấn lên nhau không. Và image\_id xác định định danh của hình ảnh đó trong tập dữ liệu.

Dựa vào đó sẽ xây dựng được tập dữ liệu được sử dụng cho huấn luyện và kiểm thử. Tập dữ liệu dùng để kiểm thử khác huấn luyện là sẽ không có Horizontal Flip ở bước tiền xử lý dữ liệu.

Tập dữ liệu sẽ được chia làm 2 phần.

* Tập dữ liệu huấn luyện chiếm 80%. Tập này sẽ chia làm nhiều batch để đưa vào huấn luyện trong mô hình. Mỗi batch có 6 hình ảnh và thứ tự các hình ảnh được xáo trộn.
* Tập dữ liệu kiểm thử chiếm 20%. Tập này cũng chia thành nhiều batch và mỗi batch cũng có 6 hình ảnh nhưng thứ tự những hình ảnh không được xáo trộn.
* Mỗi batch sẽ được tổ chức dưới dạng tuple và chứa hai thông tin sau.
* Thứ nhất chứa thông tin các hình ảnh được tiền xử lý dưới dạng Tensor.
* Thứ hai chứa thông tin đi kèm của mỗi ảnh bao gồm: labels, area, iscrowd, image\_id và boxes.

1. **Xây dựng mô hình Faster RCNN with ResNet50-FPN backbone.**

Mô hình Faster RCNN with ResNet50-FPN backbone sẽ được xây dựng theo mô hình có sẵn của thư viện Pytorch đã được giới thiệu ở phần.

Đầu tiên, mô hình sẽ được sử dụng một phiên bản được huấn luyện sẵn của mô hình. Mô hình này đã được huấn luyện trên tập dữ liệu COCO, nơi nó học được khả năng nhận diện nhiều loại đối tượng khác nhau. Mô hình sẽ trích xuất các đặc trưng học được trên tập dữ liệu này để sử dụng thay vì khởi tạo một cách ngẫu nhiên ban đầu. Mô hình sẽ được chỉnh thành 2 lớp đầu ra cho các đối tượng đó là “Thuốc lá điện tử” hoặc “Background” (Không chứa đối tượng nào).



Hình. Tập dữ liệu COCO.

1. **Điều chỉnh tham số.**

**Khởi tạo Optimizer.**

Optimizer (tối ưu hóa) được sử dụng để cập nhật trọng số của mô hình dựa trên gradient của hàm mất mát. Mô hình sẽ sử dụng SGD (Stochastic Gradient Descent) được giới thiệu ở phần … với để cập nhật gradient cho hàm mất mát của mô hình với các tham số sau.

* Params được lấy từ mô hình vừa được xây dựng ở bước trên
* Learning rate – 0.005
* Momentum – 0.
* Weight Decay – 0.0005
* Dampening – 0.00005

**Khởi tạo Learning Rate Scheduler.**

Learning rate scheduler sẽ quyết định cách learning rate thay đổi theo thời gian. Hàm STEPLR của Pytorch sẽ giúp mô hình giảm tốc độ học theo gamma – 0.5 sau mỗi 3 epochs huấn luyện.

* 1. **Huấn luyện mô hình.**

Quá trình huấn luyện mô hình sẽ trải qua 10 epochs. Mỗi epoch sẽ được huấn luyện và kiểm thử trên toàn bộ tập dữ liệu. Như ở phần chuẩn bị dữ liệu thì tập dữ liệu huấn luyện và kiểm thử chia thành các batch mỗi batch chứa 6 hình ảnh.

**Bước 1**: Bắt đầu một epoch, tất cả các dữ liệu đã được chuẩn bị trước đó sẽ tiến hành đưa vào mô hình để huấn luyện.

**Bước 2**: Mỗi epoch, tập dữ liệu huấn luyện và kiểm thử sẽ chia thành các batch để tối ưu hóa hiệu suất tính toán. Mỗi batch sẽ được chuẩn hóa thành một kiểu dữ liệu có cùng định dạng để đưa vào huấn luyện.

**Bước 3**: Mô hình thực hiện **Feedforward** – lan truyền tiến để tính toán các dự đoán bounding box trên mỗi batch. Sau đó sẽ thu được các thông số loss sau mỗi lần training một batch bao gồm:

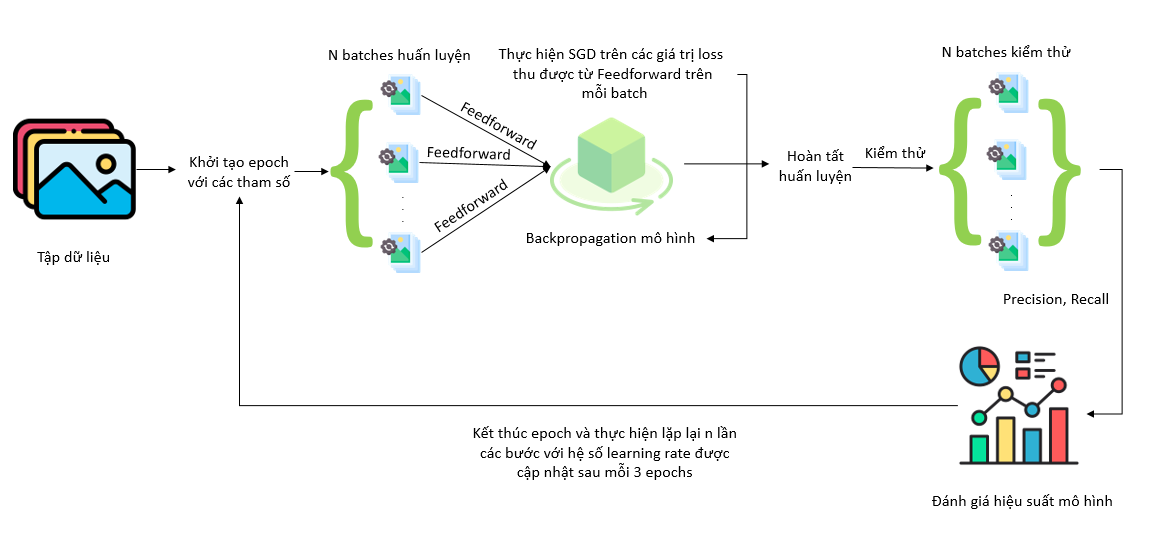
* **Loss**: giá trị hàm mất mát.
* **Loss Classifier**: giá trị loss khi dự đoán các lớp đối tượng của bounding boxes.
* **Loss Box Reg**: tính toán sự chênh lệch giữa các bounding box dự đoán so với các bounding box thực tế.
* **Loss Objectness**: giá trị loss liên quan đến việc dự đoán xác suất tồn tại của đối tượng trong các bounding box.
* **Loss RPN Box Reg**: tính toán sự chênh lệch giữa bounding box được chọn từ RPN với các bounding box thực tế.

**Bước 4**: Sau khi tính toán được các loss thì sẽ thực hiện bước tính toán gradient của các loss và Backpropagation – lan truyền ngược qua mô hình để cập nhật những trọng số trong mô hình bằng phương pháp SGD – Stochastic Gradient Descent.

**Bước 5**: Lặp lại bước 3 và 4 để mô hình có thể được huấn luyện qua tất cả các batch của tập dữ liệu huấn luyện.

**Bước 6**: Đánh giá hiệu suất của mô hình dựa trên tập dữ liệu test.

**Bước 7**: Lặp lại quy trình từ bước 1 thực hiện qua 10 epochs. Sau mỗi 3 epochs thì learning rate được giảm đi một nửa như trong quá trình thiết lập tham số của Learning Rate Scheduler.



Hình. Mô tả các bước huấn luyện mô hình Faster RCNN.

* 1. Đánh giá mô hình.
  2. Triển khai mô hình.

1. **PHÂN TÍCH THIẾT KẾ HỆ THỐNG.**
   1. Phân tích yêu cầu hệ thống.
   2. Đặc tả yêu cầu hệ thống.
   3. Thiết kế hệ thống gán nhãn.
2. **THỰC NGHIỆM, SO SÁNH VÀ ĐÁNH GIÁ.**
   1. Môi trường thực nghiệm – cấu hình.
   2. Kết quả thực nghiệm.
   3. So sánh với các phương pháp khác.
      1. Thuật toán Fast R-CNN.
   4. Đánh giá hiệu suất mô hình và phân tích kết quả.
   5. Kết luận.