



BÁO CÁO CUỐI KỲ

Nghiên cứu và ứng dụng phương pháp Mask R-CNN để nhận diện và phân loại đối tượng trong hình ảnh

Nhóm 3

Ngày 10 tháng 6 năm 2024



Sinh viên thực hiện

106200284 - Hồ Đức Vũ - 20KTMT2

106200241 - Nguyễn Minh Phương - 20KTMT1

106200240 - Huỳnh Vũ Đình Phước - 20KTMT1

Giáo viên hướng dẫnTS. Hồ Phước Tiến

Môn họcPBL 4: Trí tuệ nhân tạo

Đề tàiNghiên cứu và ứng dụng phương pháp Mask R-CNN để nhận diện và phân loại đối tượng trong hình ảnh

Xuất bản

Đà Nẵng, Ngày 10 tháng 6 năm 2024

Số trang23

Mục lục

Nội dung báo cáo	3
1 Giới thiệu	3
2 Nội dung nghiên cứu	4
2.1 Faster R-CNN	5
2.2 Instance Segmentation	9
3 Ứng dụng Mask R-CNN nhận diện vật thể trên tập dữ liệu COCO	13
3.1 Tiền xử lý	13
3.2 Huấn luyện mô hình	16
3.3 Sử dụng phương pháp đánh giá IoU để đánh giá mô hình	19
4 Demo	20
4.1 Nhận diện và phân loại đối tượng trên ảnh tĩnh	20
4.2 Nhận diện và phân loại đối tượng trong video	21
5 Kết luận	21
Tài liệu tham khảo	22
Phụ lục	23

Nội dung báo cáo

1 Giới thiệu

Nhận diện (detect) và phân loại (classification) đối tượng trong hình ảnh và video là một lĩnh vực nghiên cứu quan trọng trong thị giác máy tính và trí tuệ nhân tạo. Đây là một bước cực kỳ quan trọng trong quá trình hiểu và xử lý thông tin từ hình ảnh và video, đóng vai trò quan trọng trong nhiều ứng dụng thực tế như xe tự lái, giám sát an ninh, nhận diện khuôn mặt hay hình dáng cấu trúc đối tượng, nhận dạng biển số xe...

Nhận diện đối tượng là quá trình tự động xác định và nhận biết các vật thể trong hình ảnh hoặc video. Trong khi đó, phân loại đối tượng là quá trình gán nhãn cho các đối tượng đã được nhận diện và cho vào các danh mục được quy định trước đó. Cả hai nhiệm vụ này đều đặt ra nhiều thách thức do sự đa dạng và phức tạp của các đối tượng, khung cảnh xung quanh trong thế giới thực, cũng như sự biến đổi của điều kiện ánh sáng, góc nhìn và quy mô hình ảnh.

Trong những năm gần đây, các phương pháp dựa trên deep learning đã đạt được nhiều thành tựu đáng kể trong lĩnh vực nhận diện và phân loại đối tượng trong hình ảnh. Các mô hình deep learning như Convolutional Neural Network (CNN) [1] đã giúp cải thiện hiệu suất của các hệ thống nhận diện và phân loại đối tượng, với độ chính xác và tốc độ xử lý ngày càng cao.

Trong báo cáo này, chúng tôi sẽ trình bày một trong những phương pháp tiêu biểu trong lĩnh vực nhận diện và phân loại đối tượng dựa trên mô hình CNN là Mask R-CNN [5].

Như đã đề cập ở trên, Mask R-CNN là một mô hình deep learning được phát triển dựa trên CNN cho các nhiệm vụ nhận diện đối tượng và tạo ra các lớp mặt nạ (mask layer) chi tiết bao quát đối tượng trong hình ảnh. Mô hình này được giới thiệu vào năm 2017 và nhanh chóng trở thành một công cụ quan trọng trong lĩnh vực này.

Mask R-CNN được mở rộng từ Faster R-CNN [4] - mô hình nhận diện và phân loại đối tượng có đầu ra là các nhãn (label layer) và bounding box của đối tượng đó. Mask R-CNN cung cấp khả năng tạo ra các mask dựa theo điều chỉnh pixel-to-pixel [5] cho từng đối tượng được nhận diện trong hình ảnh - đây là phần bổ xung so với Faster R-CNN - giúp định vị và phân loại chính xác hơn các vùng chứa đối tượng, thậm chí khi các đối tượng trong hình ảnh chùng chéo (overlapping) lên nhau.

Điểm mạnh của Mask R-CNN là khả năng tích hợp với các ứng dụng thực tế như nhận dạng cấu trúc vật thể hay phân tích hành vi của đối tượng trong video thông qua phân tích mask layer của đối tượng đó.

Hiện nay, có nhiều nghiên cứu và ứng dụng trong lĩnh vực thị giác máy tính dựa trên Mask R-CNN với mục đích cải thiện hiệu suất và độ chính xác của hệ thống nhận diện và phân loại đối tượng trong hình ảnh và video.

Mục tiêu chung của báo cáo này là nghiên cứu và hiểu sâu về kiến trúc nền tảng, nguyên

lý hoạt động và các cách thức ứng dụng phương pháp Mask R-CNN để giải quyết các bài toán nhận diện vật thể - điển hình là nhận dạng con người - trong hình ảnh và video.

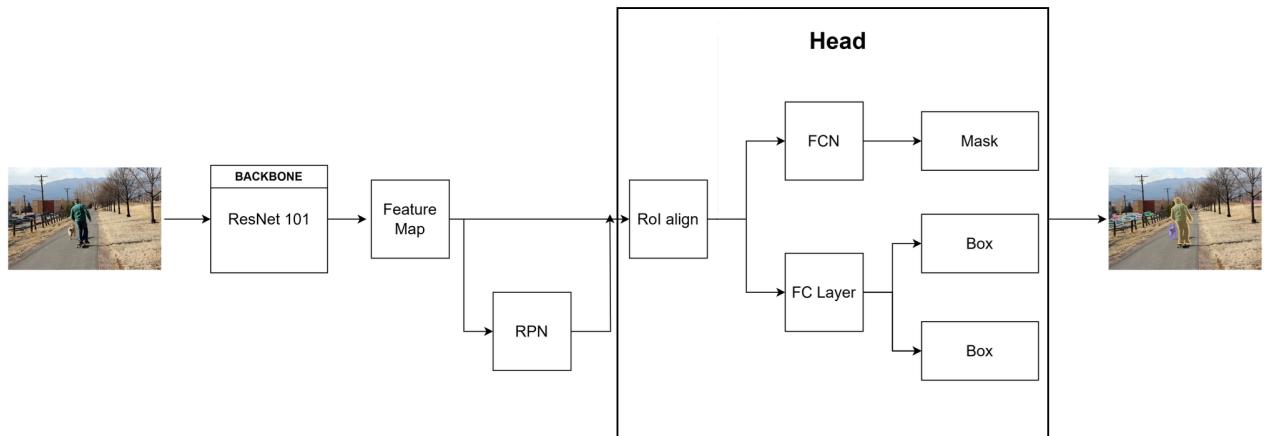
Mục tiêu kỹ thuật là triển khai ứng dụng Mask R-CNN để nhận diện đối tượng trên tập dữ liệu COCO (Common Objects in Context), một tập dữ liệu nổi tiếng được sử dụng rộng rãi trong lĩnh vực nhận diện và định vị đối tượng trong hình ảnh.

Tại nội dung §2, chúng ta sẽ đi sâu vào việc nghiên cứu kiến trúc và các nền tảng mà Mask R-CNN kế thừa từ những phương pháp trước đó, từ đó đưa ra những nhận định ưu và nhược điểm của phương pháp này và cũng như khả năng ứng dụng của nó vào thực tiễn.

2 Nội dung nghiên cứu

Mask R-CNN là một phương pháp mở rộng của phương pháp Faster R-CNN, nó thêm vào một nhánh dự đoán lớp mặt nạ cho đối tượng (object mask) - một phương pháp instance segmentation - được triển khai song song với nhánh nhận diện và bounding box đối tượng còn được gọi là bounding box recognition, dựa trên phương pháp Faster R-CNN.

Có thể thấy rằng kiến trúc Mask R-CNN được chia ra với hai phần chính: Instance segmen-

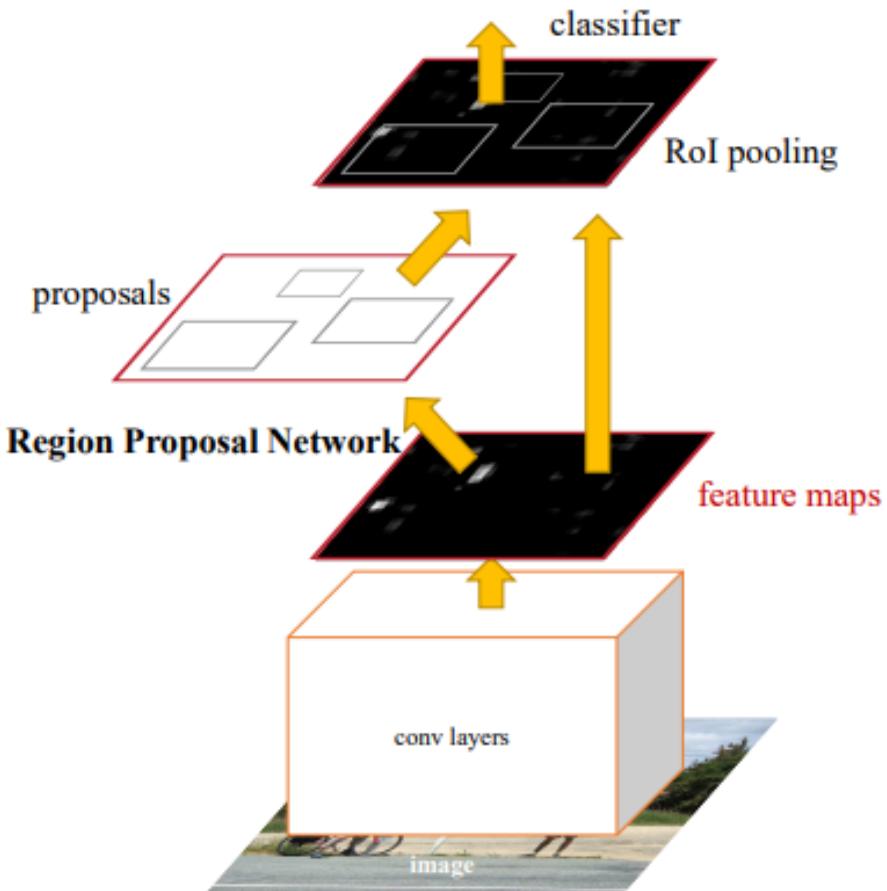


Hình 2.1: Kiến trúc tổng thể của Mask-RCNN

tion và Faster R-CNN. Instance segmentation tạo ra một lớp mặt nạ để phân vùng đối tượng trong hình ảnh, trong khi Faster R-CNN được dùng để nhận diện, phân loại đối tượng và bounding box cho chúng, hình 2.1 mô tả trực quan về kiến trúc của Mask R-CNN. Faster R-CNN là phương pháp nền tảng quan trọng và được làm trọng tâm cho Mask R-CNN, vì vậy trước khi tìm hiểu sâu về Mask Object - một phần mới và là ưu điểm của Mask R-CNN - thì ta cần có kiến thức và hiểu biết nền tảng về Faster R-CNN trước tiên, từ đó có được cái nhìn tổng quát và hiểu rõ nua các đặc tính của phương pháp Mask R-CNN.

2.1 Faster R-CNN

Faster R-CNN [4] bao gồm 2 phần chính, phần thứ nhất được gọi là Region Proposal Network (RPN) được dùng để đưa ra các đề xuất về các vùng có tiềm năng chứa đối tượng trong ảnh. Phần thứ hai là nhận diện đối tượng dựa trên một phương pháp nền tảng trước đó là Fast R-CNN [3]. Fast R-CNN sử dụng các vùng đề xuất do RPN cung cấp để nhận diện đối tượng trong các vùng đó, trong Mask R-CNN, thay vì sử dụng ROI Pooling như các phương pháp trước đó thì nó được thay thế bằng RoI Align, nhằm tối ưu cho việc điều chỉnh và phân vùng mask cho đối tượng theo điều chỉnh pixel-to-pixel được đề cập tại §2.2. Hai phần trong Faster R-CNN được liên kết với nhau thông qua chia sẻ đặc trưng tích hợp của chúng.



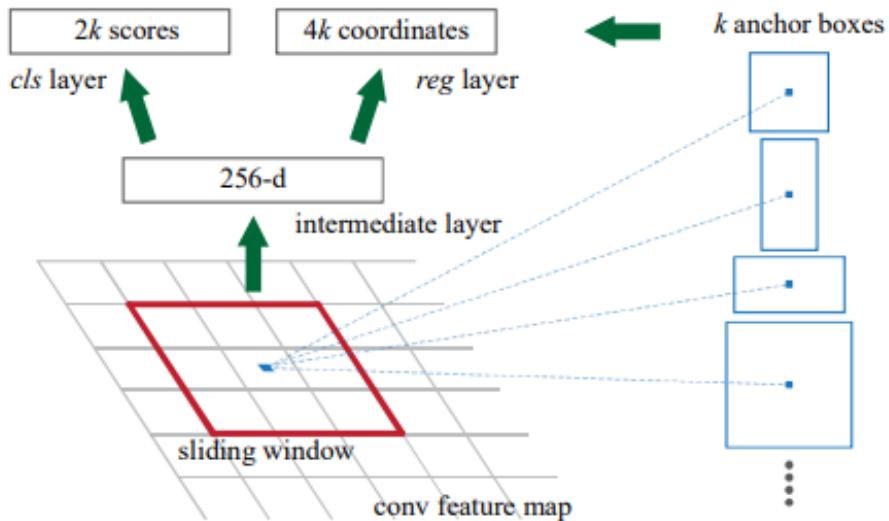
Hình 2.2: Cấu trúc tổng quát Faster R-CNN.

Hình 2.2 mô tả quá trình tổng quát của mô hình Faster R-CNN như đã đề cập ở trên. Để cụ thể hơn, ta sẽ nghiên cứu từng phần cấu tạo nên Faster R-CNN. Trước tiên ta cần phải hiểu thuật ngữ Region Proposal Network (RPN) là gì và nó đóng vai trò như thế nào bên trong phương pháp Faster R-CNN nói riêng và Mask R-CNN nói chung.

Trong phương pháp Faster R-CNN, RPN là một mạng tích chập toàn phần (fully convolutional network - FCN) được sử dụng để sinh ra các dự đoán về khu vực tiềm năng chứa đối

tương trong hình ảnh. RPN với đầu vào là một hình ảnh với bất kỳ kích cỡ và đầu ra của nó là một tập hợp các vùng đề xuất đối tượng có khung chữ nhật với tỉ lệ cho đối tượng có trong khung chữ nhật đó. RPN sử dụng mạng tích chập toàn phần (FCN) với mục đích chia sẻ các tính toán (share computation) với Fast R-CNN để nhận diện đối tượng trong vùng đề xuất, giúp giảm thời gian huấn luyện và tăng hiệu suất của mô hình.

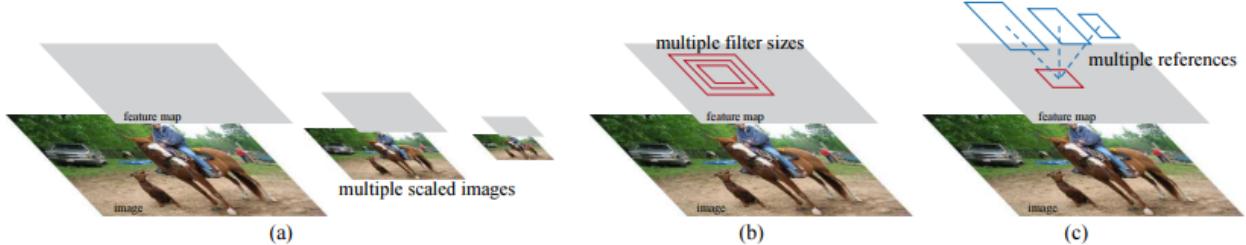
Để tạo các vùng đề xuất, chúng ta trượt một khung cửa sổ (sliding window) trên một lớp đặc trưng tích chập (convolutinal feature map), lớp đặc trưng tích chập này đã được tạo ra trước đó bằng cách nhúng đầu vào là các tập hợp hình ảnh qua một mô hình mạng nền tảng là CNN [1]. Khung cửa sổ này có đầu vào là $n \times n$ trên lớp đặc trưng tích chập. Với mỗi sliding window được trích ra, các giá trị sẽ đi qua một lớp trung gian được gọi là 256-d, tại đây sẽ tạo ra 2 lớp, lớp thứ nhất là box-regression (reg) và lớp thứ hai là box-classification (cls). Trên mỗi sliding window, RPN sẽ dự đoán một số lượng tối đa các vùng đề xuất khả thi trên một điểm trung tâm (maximum possible proposals), phương pháp này được gọi là *anchors*. Có thể hiểu là với mỗi điểm trên feature mapping layer có thể được sử dụng để làm trung tâm cho một khung cửa sổ có kích thước $n \times n$, thì ta sẽ có tối đa k lần vùng đề xuất khả thi, điều này được miêu tả một cách trực quan tại hình 2.3. Như đã đề cập trước đó, các sliding window được trích ra sẽ đi qua lớp trung gian 256-d, với mỗi sliding window cho ra tối đa k lần vùng đề xuất khả thi - còn được gọi là k anchor boxes - thì ta cũng sẽ có k lần 2 lớp reg và cls, với reg có 4 tham số tọa độ bao gồm chiều dài, chiều rộng, chiều cao và độ lớn của đối tượng trong hình ảnh, cls có 2 tham số là tỉ lệ đối tượng và background, khi trích ra k anchor boxes, đầu ra của RPN chính là $2k$ scores của cls layer và $4k$ coordinates của reg layer được dùng để nhận diện và bounding box theo thứ tự tương ứng cho giai đoạn tiếp theo sử dụng Fast R-CNN.



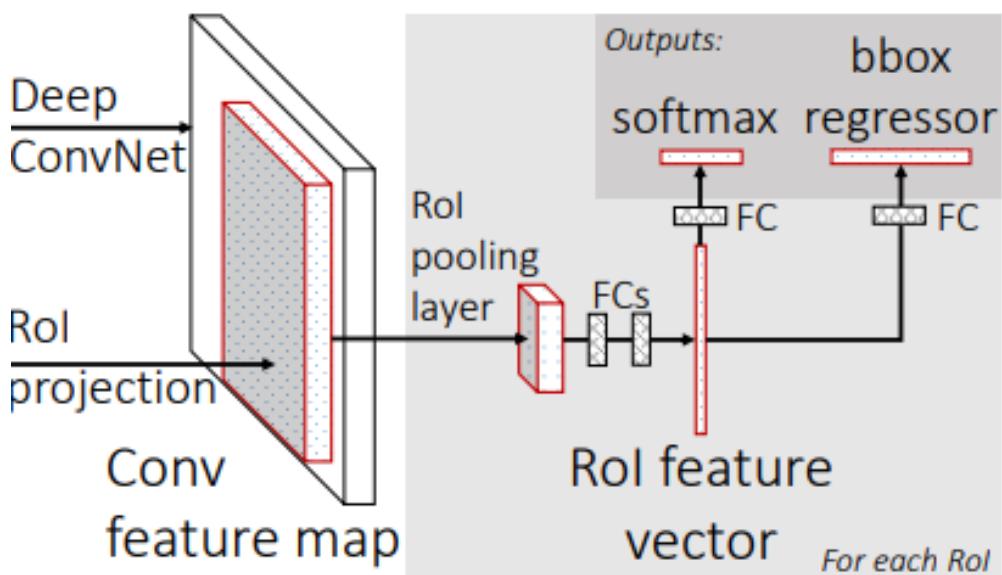
Hình 2.3: Region Proposal Network [4]

Có thể thấy thành phần đóng vai trò quan trọng trong cấu trúc RPN là các điểm anchors, vậy thì khi ta đã xác định được các điểm anchors này trên feature map layer, làm thế nào để tạo ra được các anchor boxes như đã đề cập ở trên? Để tạo ra các anchor boxes, chúng ta có thể

thực hiện bằng các phương pháp được mô tả tại hình 2.4. Tại 2.4(a), ta sẽ scaling kích thước của ảnh và feature map theo chiều kim tự tháp và chạy mô hình trên toàn bộ các scaling này. Đối với phương pháp ở hình 2.4(b), thay vì scaling các dữ liệu đầu vào, ta có thể scaling bộ lọc (filter) thành nhiều kích thước với trung tâm là điểm anchor. Phương pháp thứ 3 được mô tả tại hình 2.4(c), sử dụng các hàm hồi quy tạo ra các anchor boxes. Phương pháp mà RPN sử dụng để tạo ra các anchor boxes được đề cập trong báo cáo này như hình 2.3 sử dụng các hàm hồi quy để đưa ra các vùng đề xuất khả thi trên một điểm anchor.



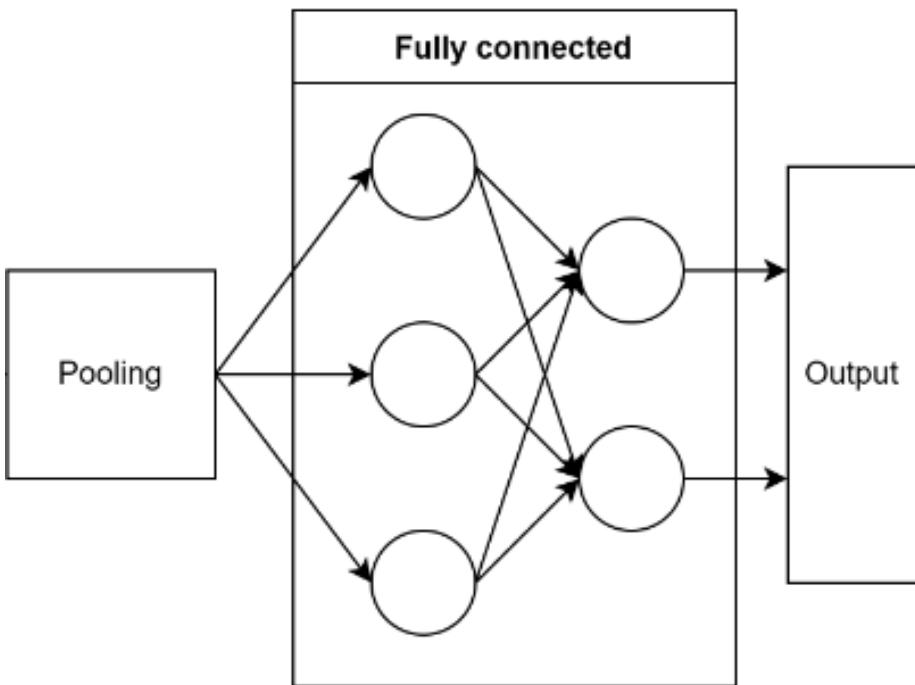
Hình 2.4: Phương pháp tạo các anchor boxes



Hình 2.5: Cấu trúc giai đoạn 2 trong Faster R-CNN (Fast R-CNN) [3]

Giai đoạn tiếp theo chính là nhận diện vật thể dựa trên Fast R-CNN. Sau khi đã có được các vùng đề xuất từ phương pháp RPN, ta sẽ sử dụng nó như là đầu vào cho Fast R-CNN, các vùng đề xuất này đi qua một lớp được gọi là RoIAlign và chuyển các vùng đề xuất này thành các fully connected với các tham số bao hàm reg layer và cls layer tại đầu ra RPN -

trong phương pháp Faster R-CNN thì lớp lọc các vùng đề xuất thành các fully connected là RoIPooling, nhưng trong Mask R-CNN thì ta sử dụng RoIAlign (§2.2). Cấu trúc của giai đoạn này được mô tả tại hình 2.5. Phương pháp biến đổi các vùng đề xuất thành các fully connected (FCs) được mô tả tại hình 2.6, Pooling layer được dùng nhằm mục đích giảm kích thước ma trận đặc trưng của các vùng đề xuất và cầu nối với lớp fullyconnected. Trong Mask R-CNN, có 2 cách để thực hiện pooling đó là chọn ra giá trị lớn nhất (max-pooling) hoặc lấy giá trị trung bình (average-pooling) trong vùng pooling đó. Sau khi trích đặc trưng và làm phẳng thành vector đặc trưng sẽ đưa vào lớp fully connected, trong lớp này các nơ-ron được kết nối với nhau và tính toán các trọng số của mô hình. Sau khi có được các FCs, các FCs này được sử dụng làm đầu vào cho 3 lớp, như trên hình 2.5, ta thấy rằng có 2 lớp có đầu vào FC là softmax và bbox regressor, ngoài ra còn có mask layer được triển khai một cách độc lập như được đề cập tại §2.2.



Hình 2.6: Mô hình cấu trúc Fully connected layer (FCs)

Softmax được dùng để đưa ra dự đoán tỉ lệ và phân loại đối tượng có trong khung hình, vớ đầu ra là class và score của đối tượng. Bbox regressor được dùng để tạo các bounding box từ các tham số reg layer của FC, đầu ra của bbox regressor là một khung hình chữ nhật bao quát đối tượng xuất hiện trong khung hình, hay còn gọi là bounding box.

Để mô hình hoạt động hiệu quả và giảm thời gian huấn luyện, Faster R-CNN sử dụng sharing convolutional layers để chia sẻ các đặc trưng tích hợp (sharing convolutional features) giữa RPN và Fast R-CNN. Với Fast R-CNN, đầu vào của nó là các tham số được lấy từ đầu ra của RPN, trong khi đó, trong quá trình huấn luyện cho mô hình RPN, ta sẽ liên tục cải thiện các tham số được sử dụng cho mô hình này thông qua đánh giá mức hiệu quả đầu ra bởi Fast

R-CNN.

2.2 Instance Segmentation

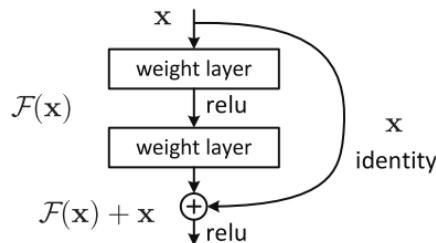
RoIPool là một phép toán tiêu chuẩn để trích xuất một bản đồ đặc trưng nhỏ (ví dụ, 7×7) từ mỗi vùng quan tâm (RoI). RoIPool trước tiên lượng tử hóa một RoI có giá trị số thực thành độ rộng rời rạc của bản đồ đặc trưng, sau đó RoI đã được lượng tử hóa này được chia thành các vùng không gian mà chúng lại được lượng tử hóa, và cuối cùng giá trị đặc trưng được che phủ bởi mỗi vùng được tổng hợp (thường là bằng cách tối đa hóa). Lượng tử hóa được thực hiện, ví dụ, trên một tọa độ liên tục x bằng cách tính toán $[x/16]$, trong đó 16 là bước đi của bản đồ đặc trưng và $\lfloor \cdot \rfloor$ là làm tròn; tương tự, lượng tử hóa được thực hiện khi chia thành các vùng (ví dụ, 7×7). Những lượng tử hóa này tạo ra sự không phù hợp giữa RoI và các đặc trưng được trích xuất. Mặc dù điều này có thể không ảnh hưởng đến việc phân loại, nhưng nó ảnh hưởng lớn đến việc dự đoán các mask chính xác từng pixel. Để giải quyết vấn đề này, lớp ROIAlign đã được đề xuất để loại bỏ việc lượng tử hóa nghiêm ngặt của RoIPool, cẩn chỉnh đúng các đặc trưng được trích xuất với đầu vào. Sự thay đổi đề xuất là đơn giản: tránh bất kỳ lượng tử hóa nào của biên độ hoặc các vùng của RoI (tức là, sử dụng $x/16$ thay vì $\lfloor x/16 \rfloor$). Sử dụng nội suy hai chiều để tính toán các giá trị chính xác của các đặc trưng đầu vào tại bốn vị trí được lấy mẫu đều đặn trong mỗi vùng RoI, và tổng hợp kết quả (sử dụng tối đa hoặc trung bình).

Trong Mask R-CNN, biểu diễn mask là một cách để mã hóa cấu trúc không gian của đối tượng đầu vào. Khác với các nhãn lớp hoặc offset hộp được gộp vào các vector đầu ra ngắn thông qua các lớp FCs, việc trích xuất cấu trúc không gian của các mask có thể được giải quyết một cách tự nhiên thông qua sự tương quan từ pixel này sang pixel khác được cung cấp bởi các phép tích chập. Cụ thể, chúng ta dự đoán một mask kích thước $m \times m$ từ mỗi vùng quan tâm (RoI) bằng cách sử dụng một mạng Fully Convolutional Network (FCN). Điều này cho phép mỗi lớp trong nhánh mask duy trì cấu trúc không gian rõ ràng $m \times m$ của đối tượng mà không gộp nó vào một biểu diễn vector thiếu các chiều không gian.

Feature extraction network

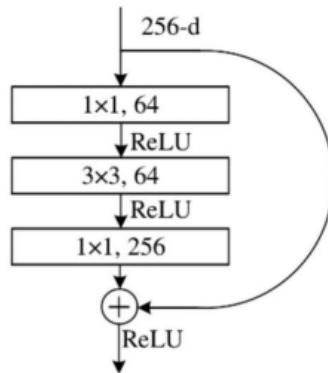
ResNet101:

Hình 2.7 biểu thị cấu trúc của khối dư trong mạng resnet. Xuất hiện một mũi tên cong xuất



Hình 2.7: Khối dư 2 lớp

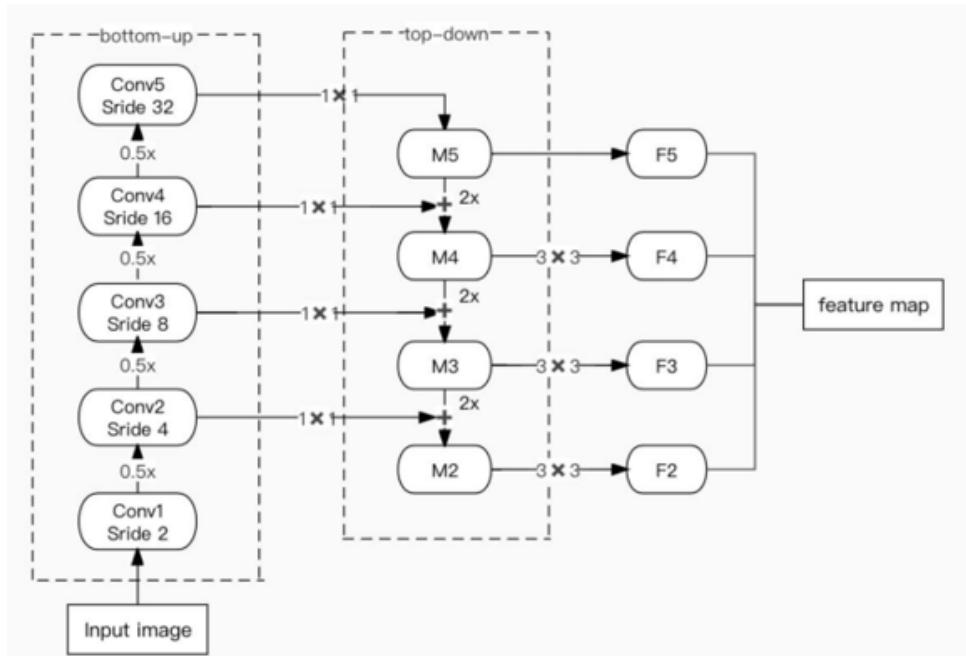
phát từ đầu và kết thúc tại cuối khối là dư sử dụng kết nối "tắt" đồng nhất để xuyên qua một hay nhiều lớp. Hay nói cách khác là sẽ bổ sung Input X vào đầu ra của layer, việc làm này sẽ chống lại việc đạo hàm bằng 0, do vẫn còn cộng thêm X. weight layer đại diện cho các tham



Hình 2.8: Cấu trúc nút cổ chai

số học tương ứng với lớp mạng. Trong cấu trúc này, kích thước của lớp chập thường là 3×3 . Trong thiết kế cấu trúc mạng, để giảm bớt khó khăn trong việc tối ưu hóa mô hình, ResNet áp dụng cấu trúc khối dư với ba lớp rất phù hợp với mạng sâu. Nó dựa trên cấu trúc hai lớp, còn được gọi là cấu trúc cổ chai, và được thể hiện trong Hình 2.8. Trong cấu trúc này, phần tích chập trong khối dư chủ yếu bao gồm các lớp có kích thước 1×1 pixel và 3×3 pixel. Tích chập 1×1 có thể được sử dụng để giảm kích thước của feature map đầu vào và số lượng kênh của feature map để tránh tiêu tốn tính toán không cần thiết. Mặt khác, nó có thể mở rộng số lượng kênh của feature map đầu ra theo kích thước của feature map đầu vào. So với cấu trúc dư hai lớp, cấu trúc nút cổ chai ba lớp có thể giảm đáng kể số lượng tham số mô hình và cải thiện tốc độ huấn luyện mạng.

Feature Pyramid Network (FPN):

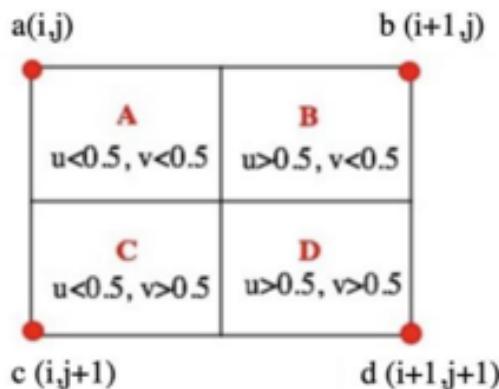


Hình 2.9: FEN dựa trên FPN

Cấu trúc FPN chủ yếu bao gồm ba phần: : bottom-up, top-down and lateral connection.

Bottom-up: là quá trình đưa hình ảnh vào các lớp ConvNet. Kích thước đầu ra của bản đồ đặc trưng sẽ không đổi hoặc giảm 2 lần. Ví dụ, đầu ra của các khối tích chập Conv2, Conv3, Conv4, Conv5 lần lượt là $\frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{16}$, $\frac{1}{32}$ lần so với bản đồ gốc

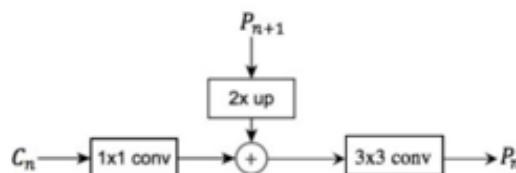
Top-down: Sử dụng phương pháp lấy mẫu lân cận gần nhất (nội suy), làm cho bản đồ đặc trưng mở rộng hai lần. Mục đích của việc lấy mẫu lân cận gần nhất là phóng to hình ảnh. Dựa trên các pixel của ảnh gốc, thuật toán nội suy phù hợp được sử dụng để chèn các pixel mới vào giữa các pixel. Đây là phương pháp nội suy đơn giản nhất, không cần tính toán. Trong số bốn pixel liền kề của pixel cần tính, nó được gán giá trị của pixel liền kề gần nhất cần tính. Giả sử



Hình 2.10: Lấy mẫu lân cận gần nhất

(hình 2.10) tọa độ của điểm mới được đặt là $(i + u, j + v)$, i, u là các số nguyên dương và j, v là số thập phân lớn hơn 0 và nhỏ hơn 1. Nếu $(i + u, j + v)$ thuộc vùng A, tức là $(u < 0,5, v < 0,5)$ thì giá trị pixel của điểm A được gán cho pixel cần tính. Tương tự, nếu rơi vào vùng B thì giá trị pixel của điểm B được gán cho pixel cần tính.

Lateral connection: Kết nối bên Như được hiển thị trong Hình 2.11, kết nối bên chủ yếu bao gồm ba bước:



Hình 2.11: Kết nối bên

Giảm Kích Thước: Bản đồ đặc trưng P_n thu được từ mỗi giai đoạn trải qua một phép tích chập 1×1 để giảm kích thước của nó. Bước này giúp nén bản đồ đặc trưng trong khi vẫn giữ

lại các đặc điểm quan trọng.

Kết Hợp Bản Đồ Đặc Trưng: Tiếp theo, các đặc trưng thu được từ phép tích chập 1×1 được kết hợp với bản đồ đặc trưng P_{n+1} lấy mẫu từ tầng trên. Quá trình kết hợp này được thực hiện thông qua phép cộng trực tiếp. Vì mối quan hệ giữa các bản đồ đặc trưng phát ra từ mỗi giai đoạn được coi là trực tiếp (hoặc tỷ lệ), kích thước của bản đồ đặc trưng lấy mẫu từ tầng trước là giống như tầng hiện tại. Do đó, các phần tử tương ứng từ cả hai bản đồ đặc trưng có thể được cộng trực tiếp.

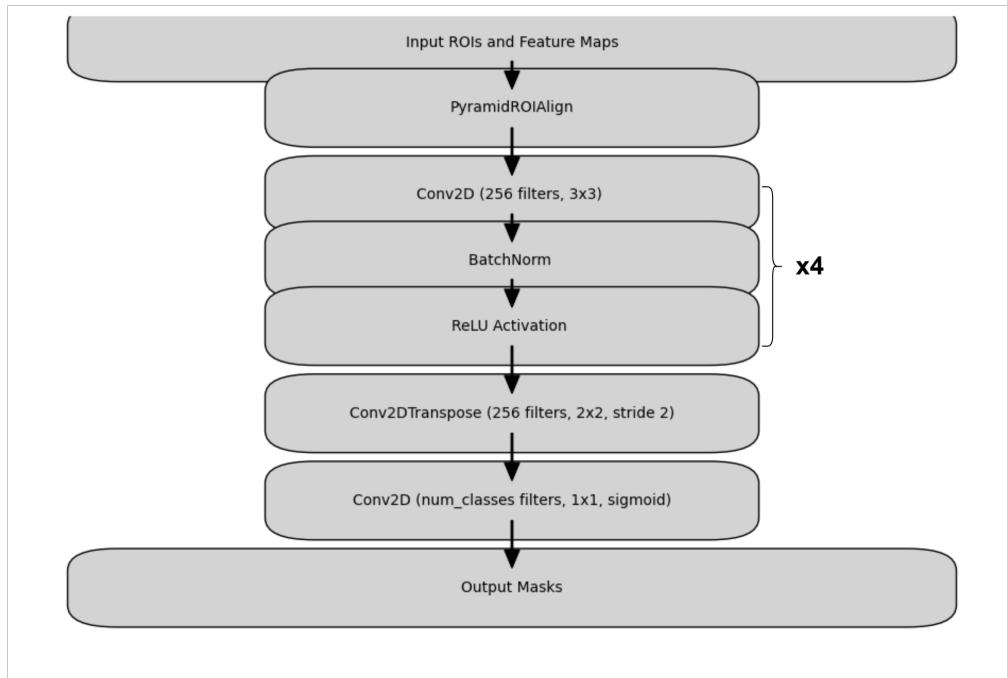
Xử Lý Sau Phép Tích Chập: Sau khi thực hiện phép cộng, một phép tích chập 3×3 được áp dụng để thu được đầu ra đặc trưng cuối cùng P_n của tầng này. Bước tích chập này nhằm loại bỏ hiệu ứng nhòe gây ra bởi việc phóng to, giúp giảm thiểu vấn đề về hình dạng gồ ghề trong kết quả đầu ra. Bằng cách áp dụng phép tích chập, các đặc trưng được làm mịn và cải thiện, nâng cao chất lượng tổng thể của bản đồ đặc trưng đầu ra.

Hình 2.1 minh họa cấu trúc của Mask R-CNN, bao gồm ba thành phần chính: kiến trúc backbone, mạng đề xuất vùng (RPN), và kiến trúc head, trong đó 2 phần đầu đã được trình bày chi tiết ở §2 nên trong phần này ta tập trung chủ yếu vào phần head. Phần backbone có nhiệm vụ trích xuất các đặc trưng và feature map M2, M3, M4, M5, từ các feature map đó thông qua quá trình upsample và kết hợp tích chập tạo ra các feature map P2, P3, P4, P5, P6 như hình trong hình 2.9. Các feature map này cung cấp thông tin chi tiết về các đặc điểm của đối tượng ở các độ phân giải khác nhau, từ độ phân giải thấp (P5) đến độ phân giải cao (P2) và chúng sẽ được sử dụng làm đầu vào cho mạng RPN và head để phân đoạn đối tượng.

Sau khi trích xuất đặc trưng và tạo đề xuất đối tượng, các đề xuất đối tượng xếp hạng cao nhất (top-N) sẽ được đưa vào mạng Head theo điểm số phân loại của chúng. Trong đó chúng được đưa vào lớp RoI Align để căn chỉnh các feature map sau đó các lớp fully connected được sử dụng để phân loại và điều chỉnh hộp giới hạn với độ chính xác cao hơn tương tự như với cấu trúc giai đoạn 2 của Faster-RCNN. Đồng thời, mạng con FCN (nhánh mặt nạ) sử dụng các đặc trưng này để tạo ra mặt nạ phân đoạn cho từng đề xuất, đảm bảo rằng các mặt nạ này chính xác và chi tiết.

Hình 2.12 mô tả chi tiết cấu trúc nhánh mask trong mạng Head, trong đó đầu vào và lớp RoI Align đã được đề cập ở trên. Tiếp theo là lớp tích chập với 4 khối của các lớp Conv2D, BatchNorm và ReLU Activation; nghiên cứu trong [5] cho thấy việc sử dụng 4 khối và với các tham số này mang lại kết quả tốt trong việc nhận diện và phân đoạn đối tượng. Lớp Transposed Convolutional thực hiện giải nén thông tin từ các feature map đã được trích xuất để tạo ra các feature map với kích thước lớn hơn. Lớp tích chập cuối cùng với hàm kích hoạt Sigmoid tạo ra mặt nạ (mask) cho từng class. Hàm sigmoid chuyển đổi giá trị đầu ra của lớp tích chập về một phạm vi giữa 0 và 1. Điều này làm cho giá trị đầu ra của mỗi pixel trên mask được coi là xác suất. Mỗi giá trị trên mask đại diện cho xác suất mà pixel tương ứng thuộc về class cụ thể mà mô hình đang dự đoán. Từ đó đầu ra của nhánh này là một tensor có kích thước là [batch_size, num_rois, MASK_POOL_SIZE, MASK_POOL_SIZE, num_classes], mỗi phần tử trong tensor này đại diện cho một mask cho mỗi class trong mỗi vùng quan tâm của mỗi ảnh trong batch.

Trong quá trình huấn luyện, một hàm mất mát đa nhiệm (multi-task loss)[3][5], L là tổng



Hình 2.12: Nhánh Mask trong mạng Head

của các hàm mất mát, bao gồm mất mát phân loại (Lcls), mất mát hồi quy hộp giới hạn (Lbox), và mất mát mặt nạ (Lmask). Các giá trị của hàm mất mát, L, được sử dụng như một chỉ số quan trọng để quyết định quá trình huấn luyện của mô hình Mask R-CNN có hoàn thành hay chưa. Khi hàm mất mát hội tụ, quá trình huấn luyện có thể được xem là hoàn tất.

3 Ứng dụng Mask R-CNN nhận diện vật thể trên tập dữ liệu COCO

Trong phần này, nhóm sẽ trình bày về ứng dụng mô hình Mask R-CNN trên tập dữ liệu COCO để thực hiện nhiệm vụ nhận diện vật thể và tạo ra các mask cho từng vật thể trong hình ảnh.

3.1 Tiết xử lý

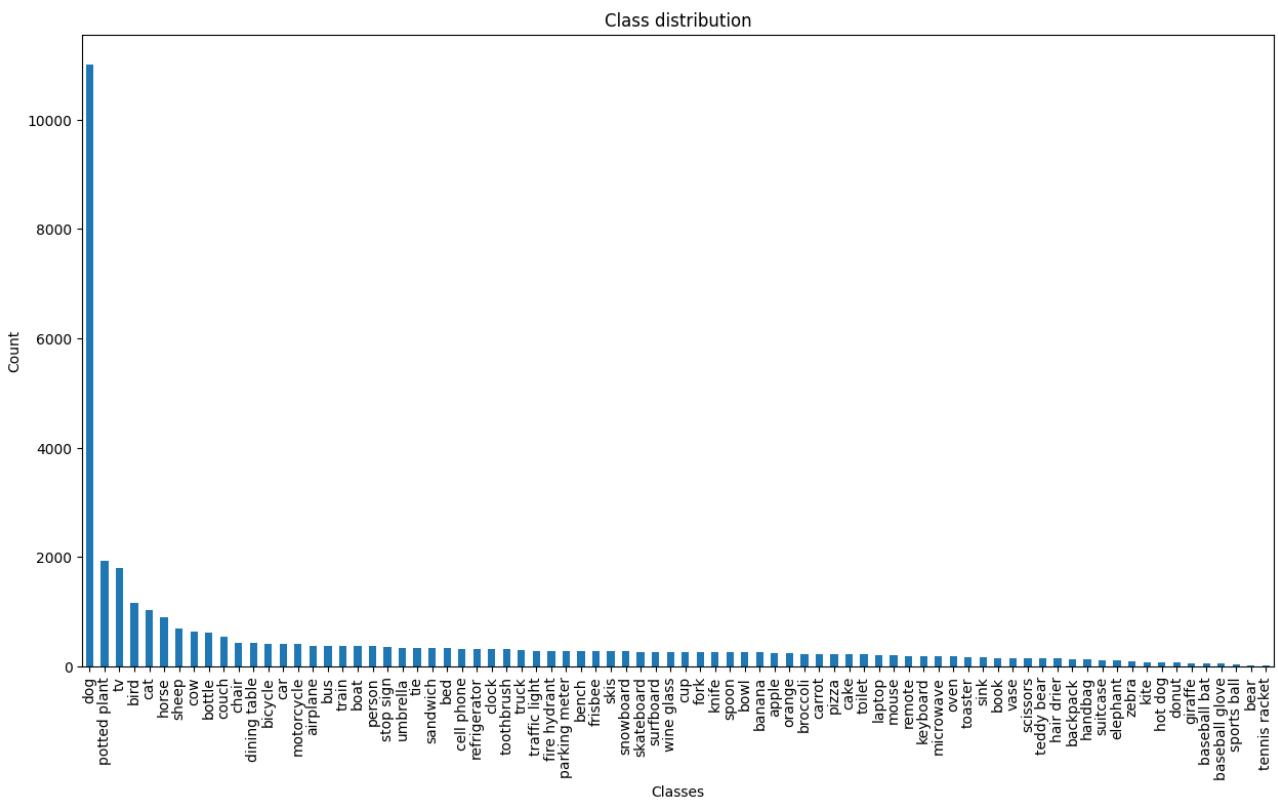
Việc tiền xử lý dữ liệu là một bước quan trọng trong quy trình huấn luyện mô hình Mask R-CNN trên tập dữ liệu COCO. Trong phần này, chúng ta sẽ đi vào chi tiết từng bước trong quá trình chuẩn bị dữ liệu trước khi đưa vào mô hình, bao gồm cấu hình mô hình, chuẩn bị dữ liệu COCO và xử lý annotations.

Đầu tiên, cần cấu hình cho mô hình Mask R-CNN. Mô hình sẽ được cấu hình với 80 lớp trong đó có 79 lớp đối tượng và 1 lớp nền. Bộ dữ liệu được nhóm sử dụng là bộ COCO validate 2017 với đầu vào có xấp xỉ 5000 ảnh.

Sau khi tải dữ liệu, chúng ta cần chuyển đổi các chú thích từ định dạng COCO thành các

mặt nạ nhị phân cho từng đối tượng trong ảnh. Điều này giúp mô hình Mask R-CNN có thể học và dự đoán các vùng đối tượng trong ảnh. Để thực hiện ta cần lấy thông tin như các lớp đối tượng và các mặt nạ của chúng trong file annotations của COCO. Tiếp theo, chúng ta phải chuyển đổi các chú thích từ các định dạng ban đầu (như polygons hoặc RLE) thành các mặt nạ nhị phân. Quá trình này bao gồm việc ánh xạ các điểm và vùng của đối tượng từ các chú thích sang các điểm và vùng trên ảnh. Các mặt nạ này cung cấp thông tin cần thiết về vị trí và hình dáng của các đối tượng trong ảnh, giúp mô hình học được cách phân biệt và dự đoán các đối tượng một cách chính xác.

Hình 3.1 biếu diễn số lượng các đối tượng thuộc mỗi class, có thể thấy một số class có lượng đối tượng lớn hơn rất nhiều so với các class khác, điều này có thể dẫn đến sự mất cân bằng trong tập dữ liệu huấn luyện, tuy vậy những class nổi bật được nhóm chú ý thì vẫn có số lượng đối tượng đủ lớn từ 1000 đến 2000, nên sự mất đối xứng này có thể không ảnh hưởng quá nhiều đến quá trình huấn luyện mô hình.



Hình 3.1: Biểu đồ thống kê số lượng đối tượng thuộc các classes

Hình 3.2 mô tả cấu trúc annotation các đối tượng có trong tập dữ liệu COCO, trong đó có các thuộc tính quan trọng ảnh hưởng trực tiếp đến kết quả huấn luyện mô hình như *annotation/segmentation*, *annotation/bbox* và *categories*.

Chúng tôi đã dựa vào cấu trúc tại hình 3.2 và trích xuất ra các thuộc tính quan trọng được sử dụng làm tham số cho quá trình huấn luyện, kết quả trích xuất các thuộc tính này được mô tả

```

{
  "info"          : info,
  "images"        : [image],
  "annotations"   : [annotation],
  "licenses"      : [license],
}

info{
  "year"          : int,
  "version"       : str,
  "description"   : str,
  "contributor"   : str,
  "url"           : str,
  "date_created" : datetime,
}

image{
  "id"            : int,
  "width"         : int,
  "height"        : int,
  "file_name"     : str,
  "license"       : int,
  "flickr_url"   : str,
  "coco_url"      : str,
  "date_captured" : datetime,
}

license{
  "id"            : int,
  "name"          : str,
  "url"           : str,
}

```

```

annotation{
  "id"            : int,
  "image_id"      : int,
  "category_id"   : int,
  "segmentation"  : RLE or [polygon],
  "area"          : float,
  "bbox"          : [x,y,width,height],
  "iscrowd"       : 0 or 1,
}

categories[{
  "id"            : int,
  "name"          : str,
  "supercategory" : str,
}]

```

Hình 3.2: Cấu trúc annotation của bộ dữ liệu COCO

tại hình 3.3.

Bộ dữ liệu sau khi đã được xử lý sẽ được chia thành 2 phần bao gồm *Training Samples* và *Validation Samples* với 80% và 20% tương ứng. Ảnh đầu vào đều được resize với cùng một kích thước 512x512. Điều này đảm bảo rằng tất cả các ảnh được đồng nhất về kích thước, việc này rất cần thiết cho quá trình huấn luyện. Các kỹ thuật như padding, crop được áp dụng giúp duy trì tỷ lệ và các chi tiết quan trọng trong ảnh. Ngoài việc resize ảnh thì kích thước của lớp mặt nạ (mask) và boundingbox cũng được resize tương ứng.

Theo như hình 3.3, mỗi bức ảnh đều có chứa thông tin chi tiết về "Annotation" hay còn gọi là chú thích của bức ảnh, chứa dữ liệu về các tọa độ mặt nạ và boundingbox. Các thông tin này

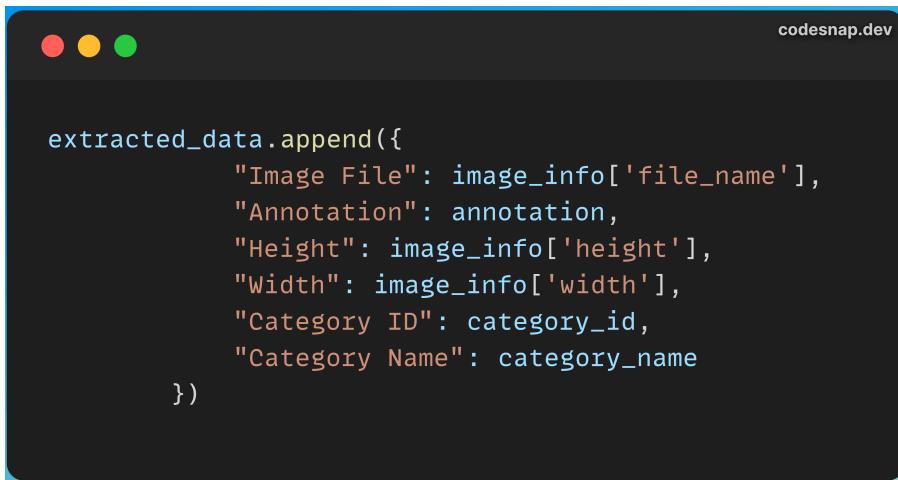
	Image File	Annotation	Height	Width	Category ID	Category Name
index						
000000394559	000000394559.jpg	{"segmentation": [[264.63, 509.22, 230.11, 512.1, 214.29, 504.9, 235.87, 489.08, 228.67, 443.06, 214.29, 407.11, 224.36, 374.03, 224.36, 362.52, 221.48, 325.13, 221.48, 297.8, 191.28, 297.8, 178.34, 289.17, 165.39, 286.3, 169.71, 280.55, 192.72, 279.11, 181.21, 258.97, 168.27, 250.34, 172.58, 233.08, 186.97, 235.96, 189.84, 244.59, 195.6, 257.53, 217.17, 280.55, 217.17, 256.1, 221.48, 243.15, 222.92, 231.65, 214.29, 230.21, 208.54, 211.51, 209.98, 185.62, 227.24, 175.56, 247.37, 178.43, 268.94, 200.01, 290.52, 211.51, 279.01, 238.84, 273.26, 246.03, 276.13, 273.35, 287.64, 306.43, 290.52, 333.76, 290.52, 363.96, 279.01, 381.22, 270.38, 398.48, 274.7, 424.37, 296.27, 454.57, 319.28, 454.57, 323.6, 473.26, 302.02, 514.97, 290.52, 514.97, 279.01, 274.7, 464.64, 257.44, 448.81, 254.56, 473.26, ...]], 'area': 22386.591749999996, 'iscrowd': 0, 'image_id': 394559, 'bbox': [165.39, 175.56, 158.21, 339.41], 'category_id': 1, 'id': 656579}	640	426	1	person
000000394559	000000394559.jpg	{"segmentation": [[177.55, 238.9, 180.48, 212.56, 181.45, 197.93, 179.5, 183.3, 174.63, 168.18, 169.75, 148.18, 176.09, 126.23, 186.33, 112.08, 202.92, 110.13, 218.53, 114.04, 224.38, 129.16, 225.35, 152.08, 212.18, 175.98, 205.36, 185.25, 195.11, 200.86, 191.21, 217.93, 188.28, 236.47], [179.02, 243.29, 179.5, 250.61, 174.14, 253.54, 183.41, 254.03, 183.89, 241.83]], 'area': 4069.647650000001, 'iscrowd': 0, 'image_id': 394559, 'bbox': [169.75, 110.13, 55.6, 143.9], 'category_id': 43, 'id': 656579}	640	426	43	tennis racket
000000394677	000000394677.jpg	{"segmentation": [[425.81, 335.48, 287.1, 319.35, 278.23, 283.06, 316.94, 281.29, 337.9, 260.48, 341.94, 251.61, 358.87, 233.87, 352.42, 215.32, 346.77, 191.94, 334.68, 180.65, 333.87, 160.48, 424.19, 152.42, 398.39, 191.13, 381.45, 233.87, 364.52, 262.1, 387.1, 263.71, 433.06, 279.03], [156.45, 304.03, 83.87, 294.35, 90.32, 264.52, 107.26, 266.13, 140.32, 249.19, 164.52, 253.23, 156.45, 270.97], [27.42, 286.29, 2.42, 281.45, 0.81, 267.74, 11.29, 255.65, 13.71, 256.87, 36.29, 262.9], 'area': 17501.850150000002, 'iscrowd': 0, 'image_id': 394677, 'bbox': [10.81, 152.42, 432.25, 183.06], 'category_id': 63, 'id': 114176]}	375	500	63	couch
000000394677	000000394677.jpg	{"segmentation": [[217.91, 95.86, 239.86, 101.77, 251.69, 110.22, 258.45, 118.67, 263.51, 118.67, 282.94, 111.91, 293.07, 108.53, 304.05, 113.6, 330.24, 150.76, 338.68, 180.32, 327.7, 185.39, 309.12, 188.77, 281.25, 191.3, 272.8, 196.37, 267.74, 203.13, 255.91, 214.1, 253.38, 214.1, 238.18, 203.13, 226.35, 198.9, 220.44, 198.9, 214.53, 202.28, 211.99, 212.42, 211.99, 220.86, 216.22, 226.77, 237.33, 231.0, 330.24, 234.38, 343.75, 246.2, 336.99, 259.71, 320.95, 269.85, 284.63, 279.14, 280.41, 325.59, 298.99, 366.98, 243.24, 368.67, 222.13, 284.21, 220.44, 270.69, 204.39, 277.45, 201.86, 313.77, 190.88, 367.82, 152.03, 362.75, 157.09, 296.03, 160.47, 244.51, 174.83, 231.84, 184.97, 220.02, 191.72, 193.83, 211.15, 171.88, 218.75, 164.27, 208.61, 160.05, 198.48, 163.43, 193.41, 161.74, 181.59, 133.87, ...]], 'area': 29351.495599999996, 'iscrowd': 0, 'image_id': 394677, 'bbox': [152.03, 95.86, 191.72, 272.81], 'category_id': 1, 'id': 488119}	375	500	1	person

Hình 3.3: Mô tả kết quả trích xuất annotation từ bộ dữ liệu COCO.

rất cần thiết cho việc huấn luyện. Cho nên để đảm bảo dữ liệu được rõ ràng, chúng tôi đã lọc và loại bỏ các file ảnh không đảm bảo yêu cầu.

3.2 Huấn luyện mô hình

Trong báo cáo này, chúng tôi sử dụng framework Pytorch để huấn luyện mô hình Mask R-CNN. Trước tiên ta cần cài đặt các công cụ và thư viện hỗ trợ Pytorch và CUDA (dùng cho GPU), đây là bước cực kỳ quan trọng. Sau khi đã cài đặt các công cụ cần thiết, bước tiếp theo ta cần trích xuất các annotation trong bộ dữ liệu COCO như đã đề cập tại mục 3.1, sử dụng framework Pandas để xử lý annotation cho dữ liệu đầu vào, hình 3.4 mô tả đoạn code được sử dụng để trích các thông tin mong muốn cho mô hình huấn luyện.



```

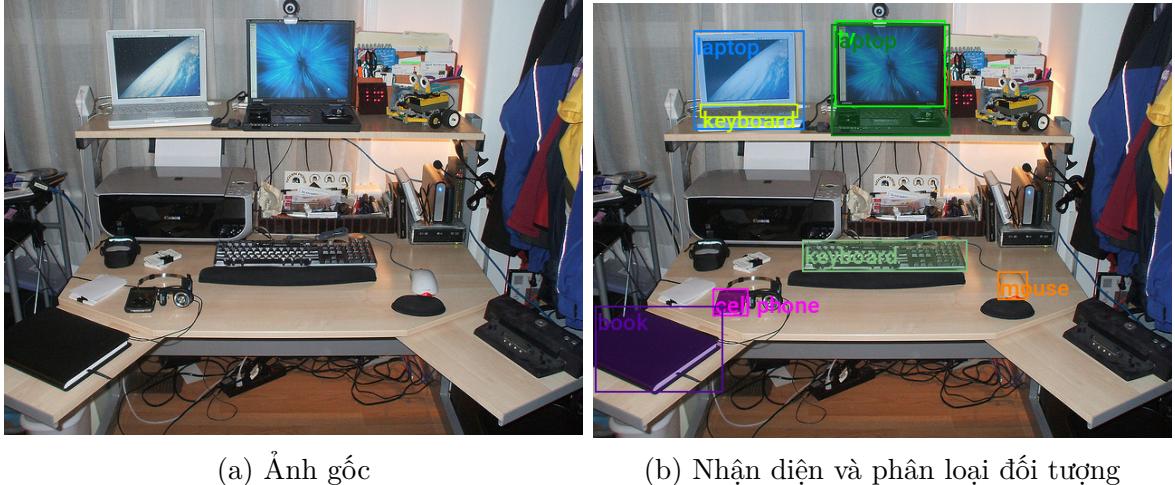
extracted_data.append({
    "Image File": image_info['file_name'],
    "Annotation": annotation,
    "Height": image_info['height'],
    "Width": image_info['width'],
    "Category ID": category_id,
    "Category Name": category_name
})

```

Hình 3.4: Trích xuất tham số huấn luyện mô hình.

Sau khi trích xuất các thông tin cần thiết được dùng làm tham số cho mô hình, ta sẽ kiểm tra các tham số này ngay trên tập dữ liệu. Trong COCO database, đối với 1 bức ảnh bất kỳ có thể sẽ có nhiều đối tượng bên trong, vì vậy tại mỗi bức ảnh sẽ cho ra một danh sách các annotation, tuy nhiên đôi khi cũng có một số bức ảnh chỉ trả về duy nhất một annotation, vì

vậy ta phải kiểm tra xem liệu annotation được trích ra từ bức ảnh là một danh sách hay chỉ là một đối tượng đơn lẻ từ đó điều chỉnh cho phù hợp với từng đối tượng trong bức ảnh. Tại tham số *segmentation*, ta cần chuyển đổi tham số này thành mask, sau đó chuyển lớp mask này sang dạng nhị phân. Hình 3.5 mô tả kết quả tại bước kiểm tra các tham số này, một bức ảnh bất kỳ trong tập dữ liệu 3.5(a) sẽ tích hợp với các tham số trên và cho ra kết quả là các đối tượng trong ảnh sẽ được bouding box và phủ một lớp mask 3.5(b).



Hình 3.5: Kiểm tra nhận diện đối tượng trong ảnh Testing

Bước tiếp theo là chia bộ dữ liệu thành 2 phần, gồm *Training Samples* với 4000 ảnh và *Validation Samples* với 1000 ảnh. Ta sẽ điều chỉnh kích thước dữ liệu đầu vào về cùng một kích thước 512x512, điều này giúp giảm kích thước đầu vào cho mạng huấn luyện và đồng thời tránh sự không đồng bộ kích thước trong bộ dữ liệu đầu vào. Tại bước này còn có một yếu tố quan trọng, khi ta resize ảnh đầu vào, các tham số annotation cũng cần được resize phù hợp với dữ liệu, hình 3.6 mô tả một đoạn code đại diện cho việc điều chỉnh kích thước và match mask với ảnh đã được resize, trong đó ta sử dụng hàm *Mask()* để match mask đã được hiệu chỉnh kích thước phù hợp với đối tượng trong ảnh. Kết quả ảnh được resize 512x512 với mask và bbox tương ứng được biểu diễn tại hình 3.7.

```

js untitled
codesnap.dev

targets = {
    'masks': Mask(masks),
    'boxes': bboxes,
    'labels': torch.Tensor([class_names.index(label) for label in labels])
}
targets['boxes'] = BoundingBoxes(data=targets['boxes'], format='xyxy',
                                 canvas_size=sample_img.size[::-1])
targets['masks'] = transforms.functional.resize(targets['masks'].unsqueeze(0),
                                                size=sanitized_img.size[::-1]).squeeze(0)

```

Hình 3.6: Resize và match masks với dữ liệu đầu vào.

Trong tập dữ liệu COCO mà nhóm sử dụng để huấn luyện mô hình có một số ảnh có annotation bị lỗi, vì vậy tại bước cấu hình cho train model thì ta cần loại bỏ các file dữ liệu lỗi. Cấu



Hình 3.7: Resize ảnh đầu vào và mask cho các đối tượng trong ảnh.

hình bộ dữ liệu cho quá trình huấn luyện được xây dựng tại class COCOPDataset. Đầu tiên ta sẽ đọc file ảnh từ tham số img_keys, tham số này chứa tên ảnh (không bao gồm phần mở rộng .jpg), như đã đề cập đến trước đó, trong tập dữ liệu có những ảnh sẽ trả về một list đối tượng nhưng cũng có những ảnh chỉ trả về một đối tượng đơn, để xử lý được sự khác biệt kiểu trả về này ta sẽ kiểm tra annotation là kiểu *pb.Serial* hay không trước khi thực hiện trích xuất các tham số cho mô hình.

Đối với tham số segmentation thì nó sẽ trả về một danh sách các tọa độ pixel (kiểu float), nhưng trong quá trình câu hình, đôi lúc giá trị trả về của tham số này có lẫn một vài cặp ký tự, dẫn đến lỗi trong quá trình tạo mask cho đối tượng, để khắc phục điều này nhóm đã bỏ qua các cặp giá trị ký tự xuất hiện trong quá trình xử lý tham số segmentation, tuy nhiên chính vì bỏ qua một số cặp giá trị đó mà tại bboxes, xuất hiện trường hợp số lượng bbox nhỏ hơn so với số lượng đối tượng trong ảnh, dẫn đến lỗi và mất mát thông tin cho đối tượng, nhóm đã quyết định giảm số lượng đối tượng đầu vào bị lỗi trong ảnh tương ứng với mask để khắc phục lỗi này. Kết quả của class COCOPDataset này trả về ảnh gốc và ảnh được nhận diện.

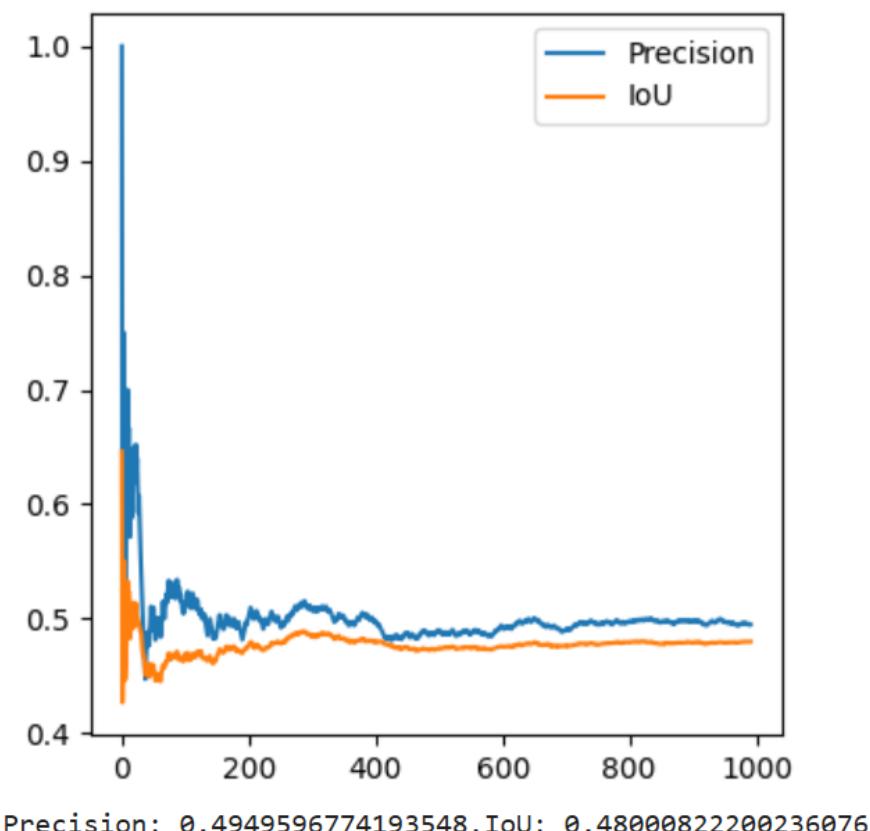
Sau khi cấu hình COCOPDataset, ta sẽ bắt đầu huấn luyện mô hình. Nhóm quyết định chia bộ dữ liệu thành 4 batch, với việc đã loại bỏ các ảnh lỗi thì bây giờ trong mỗi batch sẽ có tập train với 990 ảnh và tập validate với 249 ảnh. Với learn rate có giá trị $5 * 10^{-4}$, và epochs = 40, mô hình được lưu và checkpoint_path sau mỗi epoch. Nhóm đã train xấp xỉ 5000 bức ảnh trong mỗi epoch, sau epoch thứ 11 lúc này GPU trên colab bị ngắt nên nhóm có thể huấn luyện mô hình maximum đến epoch thứ 10.

3.3 Sử dụng phương pháp đánh giá IoU để đánh giá mô hình

Intersection over Union (IoU) là một phép đo thường được sử dụng để đánh giá hiệu suất của các mô hình phát hiện vật thể trong lĩnh vực thị giác máy tính và trí tuệ nhân tạo. IoU đo lường độ chồng chéo giữa kết quả phát hiện của mô hình và đối tượng thực tế trong hình ảnh. IoU có công thức như sau:

$$IoU = \frac{\text{Area of Intersection}}{\text{Area of Union}} \quad (3.1)$$

Trong đó "Area of Intersection" là diện tích phần giao giữa kết quả phát hiện của mô hình và vật thể thực tế, "Area of Union" là diện tích phần hợp của toàn bộ kết quả phát hiện của mô hình và vật thể thực tế. Giá trị của IoU dao động từ 0 đến 1. Giá trị 0 nếu không có sự chồng chéo nào giữa kết quả phát hiện và đối tượng thực tế, trong khi giá trị 1 chỉ ra rằng kết quả phát hiện hoàn toàn trùng khớp với đối tượng thực tế.



Hình 3.8: Kết quả đánh giá trên tập test

Hình 3.8 là kết quả đánh giá của mô hình MaskRCNN thực hiện trên tập dữ liệu kiểm tra với ngưỡng IoU được xác định là 0.5, True Positives (TP) - những trường hợp mà giá trị IoU lớn hơn ngưỡng, False Positives (FP) - những trường hợp mà giá trị IoU nhỏ hơn hoặc bằng ngưỡng, False Negatives (FN) là tổng số lượng mục tiêu trù đi số lượng True Positives. Đối với Precision ta có giá trị trung bình là 0.49, IoU có giá trị trung bình là 0.48. Thông thường, một mô hình tốt sẽ có IoU từ 0.5 trở lên. Vì vậy, IoU = 0.48 có thể xem là chấp nhận được, nhưng vẫn cần cải thiện để đạt được kết quả tốt hơn. Precision ở mức 0.49 cho thấy gần một nửa số

dự đoán của mô hình là chưa chính xác.

Nguyên nhân có thể dẫn đến IoU có giá trị thấp là do việc mất cân bằng giữa các lớp, một số lớp có số lượng đối tượng lớn hơn rất nhiều so với các lớp khác như lớp 'dog' 'potted plant' so với các lớp 'bear' 'tennis racket', gây ra sự mất cân bằng trong tập dữ liệu. Điều này có thể dẫn đến việc mô hình ưu tiên học các lớp phổ biến hơn và bỏ qua các lớp ít phổ biến.

Tổng quan, tuy mô hình nhận diện được tốt những đối tượng có số lượng lớn trong tập dữ liệu như 'dog' 'potted plant' 'person'... sẽ được trình bày ở 4. Nhưng mô hình có mức độ chính xác không cao, vẫn cần tiếp tục cải thiện để đảm bảo hiệu suất tốt hơn trên mọi trường hợp.

4 Demo

4.1 Nhận diện và phân loại đối tượng trên ảnh tĩnh



Hình 4.1: Kết quả nhận diện và phân loại đối tượng trên ảnh tĩnh bất kỳ.

Để sử dụng mô hình Mask-RCNN đã được nhóm training để dự đoán đối tượng trong bất kỳ ảnh tĩnh nào, ta sẽ thực hiện theo các bước như sau:

1. Định nghĩa classes: trong mô hình có 80 class, ta cần gán giá trị cho classes với 80 "name" class đó.
2. Load model: model được lưu vào `./pytorch-mask-r-cnn-instance-segmentation-custom-data/2024-06-09_05-23-18/maskrcnn_resnet50_fpn_v2.pth`, tại bước này ta cần cấu hình và load model

Mask-RCNN trong đường dẫn trên với dòng lệnh:

```
model.load_state_dict(torch.load(model_path, map_location=device))
```

3. Khởi tạo font cho các bbox và mask tương ứng với 80 class.
4. Bước cuối cùng là load ảnh cần dự đoán và sử dụng model để xác định bbox và mask cho các đối tượng trong ảnh. Kết quả được biểu diễn tại hình [4.1](#).

4.2 Nhận diện và phân loại đối tượng trong video

Nhóm thực hiện nhận diện và phân loại đối tượng trong video với ý tưởng theo cách bước như sau:

1. Lọc các frame từ video gốc.
2. Sau khi có được các frame, ta sẽ tiến hành xử lý các frame này như mô tả tại mục [4.1](#) cho toàn bộ các frame được trích ra từ video.
3. Cuối cùng ta sẽ ghép các frame đã được nhận diện này thành một output_video.

Với đầu vào là video có thời lượng 2 giây, kết quả được trình bày tại Link [drive](#).

5 Kết luận

Trong đồ án này, nhóm đã tìm hiểu, triển khai và huấn luyện mô hình Mask R-CNN trên tập dữ liệu COCO nhằm nhận diện và phân loại đối tượng trong ảnh và video. Quá trình thực hiện bao gồm nhiều bước từ xử lý dữ liệu, huấn luyện mô hình đến đánh giá hiệu suất và thử nghiệm thực tế. Tuy mô hình đã đạt được những kết quả tốt trong việc nhận diện và phân loại một số đối tượng, nhưng do mô hình được train trên một lượng lớn class nên có nhiều class có độ nhận diện chính thấp, lý do là vì bộ dữ liệu mà nhóm sử dụng có bất cân bằng trong các class.

Qua quá trình này, nhóm nhận thấy rằng việc huấn luyện và triển khai mô hình Mask R-CNN đòi hỏi sự chuẩn bị kỹ lưỡng về dữ liệu cũng như sự điều chỉnh liên tục các thông số và kỹ thuật. Tuy nhiên, với những kết quả đạt được, nhóm tin tưởng rằng mô hình có thể được cải thiện và ứng dụng rộng rãi trong các bài toán nhận diện và phân loại đối tượng trong ảnh, video và mở ra nhiều hướng nghiên cứu mới nhằm nâng cao độ chính xác và hiệu quả của mô hình trong các ứng dụng thực tế.

Tài liệu tham khảo

- [1] O'shea, Keiron, and Ryan Nash. "An introduction to convolutional neural networks." arXiv preprint arXiv:1511.08458 (2015).
- [2] Girshick, Ross, et al. *Rich feature hierarchies for accurate object detection and semantic segmentation*. Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.
- [3] Girshick, Ross. "Fast r-cnn." Proceedings of the IEEE international conference on computer vision. 2015.
- [4] Ren, Shaoqing, et al. "Faster R-CNN: Towards real-time object detection with region proposal networks." IEEE transactions on pattern analysis and machine intelligence 39.6 (2016): 1137-1149.
- [5] He, Kaiming, et al. "Mask r-cnn." Proceedings of the IEEE international conference on computer vision. 2017.

Nhóm đã tách ra file training và file demo riêng, tại file train thì có thể demo trực tiếp sau khi train, nhưng để khách quan thì nhóm thực hiện demo mô hình riêng biệt với training.

Phụ lục

- [1] Training-Pytorch-mask-RCNN-with-COCO-Dataset
- [2] Demo-Pytorch-MaskRCNN-on-Image
- [3] Demo-Pytorch-MaskRCNN-on-Video
- [4] COCO Dataset
- [5] Model được train với 10 epochs
- [6] Output video: Nhận diện đối tượng trong video.
- [7] Drive nhòm dùng để làm việc
- [8] Đánh giá model bằng IoU