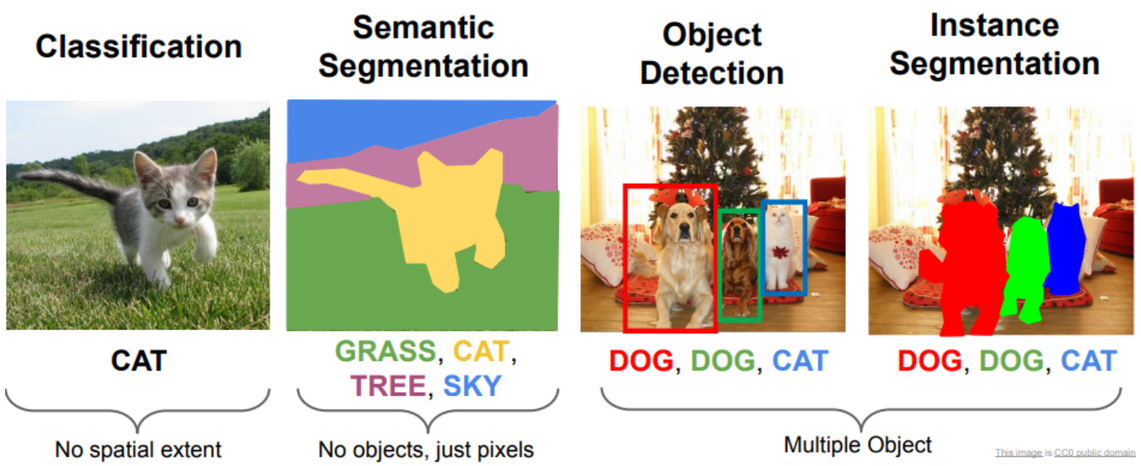
# Object detection

## 1. Giới thiệu chung bài toán object detection

Bài toán object detection là một bài toán rất phổ biến trong computer vision và được coi là một trong số các bài toán machine learning kinh điển.

Tính ứng dụng của bài toán object detection trong thực tiễn là rất lớn trong nhiều ngành nghề khác nhau. Object detection được sử dụng trong y tế giúp xác định vị trí bị bệnh trong cơ thể, trong bảo mật giúp định vị vị trí của con người trong những khu vực cấm, trong nông nghiệp giúp xác định số lượng nông sản, trong hệ thống xe tự hành ...




Bài toán object detection là sự tổng hợp của hai bài toán con: object localization và image classification.

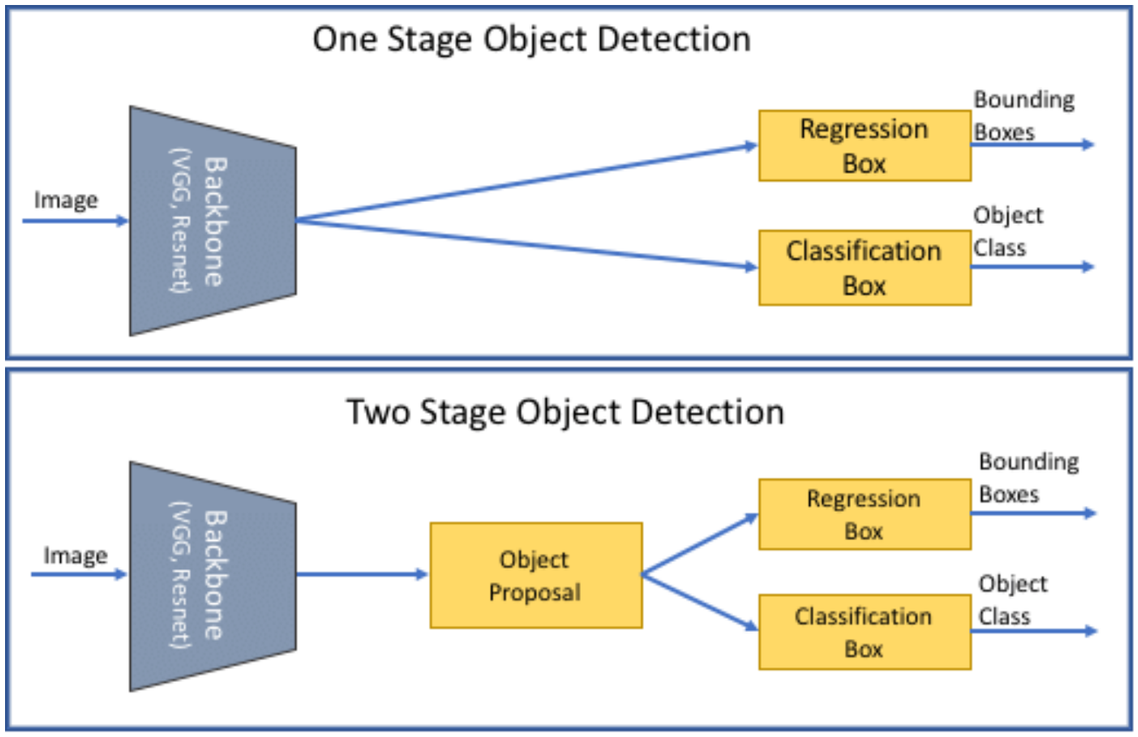
- Object localization là bài toán định vị vị trí của object trong ảnh: nhận đầu vào là một ảnh và trả đầu ra là một hoặc nhiều bbox của từng đối tượng.
- Image classification là bài toán phân lớp ảnh: nhận đầu vào là một ảnh và trả đầu ra là lớp của đối tượng đó. Bài toán object detection kết hợp cả hai bài toán trên, yêu cầu mô hình vừa định vị vị trí của một hoặc nhiều đối tượng trong ảnh vừa xác định lớp của từng đối tượng đó.

## 2. Khái quát các mô hình giải quyết bài toán object detection

### 2.1. Nhóm các mô hình two-stage

Các mô hình thuộc nhóm two-stage ra đời khá sớm từ năm 2014 đến 2017. Nhóm này có đặc điểm chung về kiến trúc gồm hai phần:

- Region proposals module: module nhận đầu vào là ảnh ban đầu và trả đầu ra là các khu vực trên ảnh mà có khả năng chứa đối tượng.
- Feature extraction module: module nhận đầu vào là các region từ Region proposals module giúp xác định chính xác đối tượng trong khu vực đó là đối tượng nào và tinh chỉnh toạ độ của khu vực chính xác hơn.




### 2.2. Nhóm các mô hình single-stage

Các mô hình thuộc nhóm single-stage ra đời muộn hơn từ năm 2016 đến nay, tuy nhiên lại đang nhận được sự quan tâm rất lớn của giới nghiên cứu trong thời gian trở lại đây vì tính ứng dụng trong thực tiễn cao của chúng.

Các mô hình single-stage đều dựa vào động lực trong việc loại bỏ Region proposals module nhằm giảm khối lượng tính toán, qua đó tăng tốc độ và đưa mô hình đến gần hơn với khả năng chạy real-time.

## 3. Nhóm các mô hình R-CNN, Fast R-CNN và Faster RCNN

### 3.1. Mô hình R-CNN

Một trong các mô hình đầu tiên ứng dụng deep learning giải quyết bài toán object detection là Regions with CNN features (gọi tắt là R-CNN).

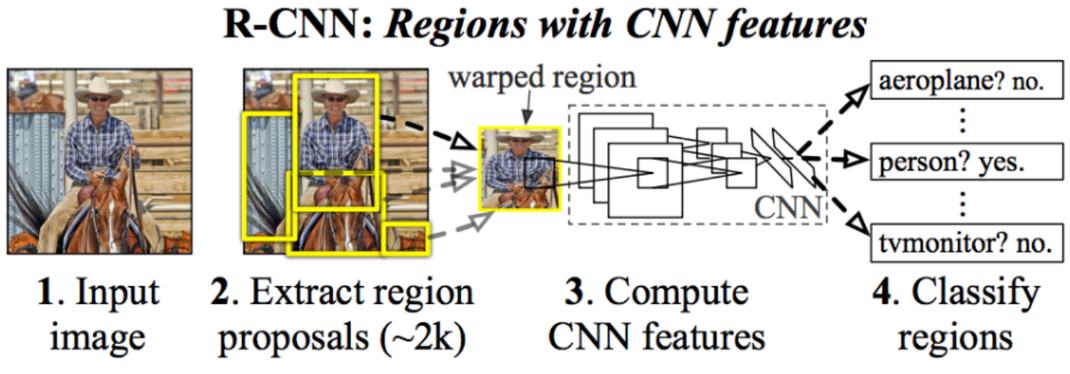
Tuy nhiên, ở thời điểm mà R-CNN ra đời, do các deep learning chưa thật sự phát triển nên R-CNN không hoàn toàn sử dụng deep learning mà vẫn dựa trên kết quả của thuật toán xử lý ảnh như Graph-Based Image Segmentation và Selective Search.

#### 3.1.1. Kiến trúc mô hình R-CNN

Là một trong số các mô hình two-stage, R-CNN bao gồm hai thành phần:

- Region proposals module của R-CNN là thuật toán Selective Search: nhận đầu vào là ảnh, Selective Search trả đầu ra là khoảng 2000 khu vực có khả năng có chứa đối tượng.

- Feature extraction module của R-CNN là một mô hình phân lớp ảnh, cụ thể theo nghiên cứu là AlexNet: nhận đầu vào là ảnh, AlexNet đánh giá xem ảnh đó chứa đối tượng hay không và nếu có thì khu vực đó chứa đối tượng nào.




#### 3.1.2. Vấn đề tồn đọng của mô hình R-CNN

Vấn đề lớn nhất của R-CNN là thời gian cần cho quá trình train và quá trình test là rất lớn. Trong quá trình test, R-CNN mất tới 47 giây để hoàn thành việc xử lý một ảnh. Kết quả này khiến cho R-CNN gần như không có giá trị ứng dụng thực tiễn.

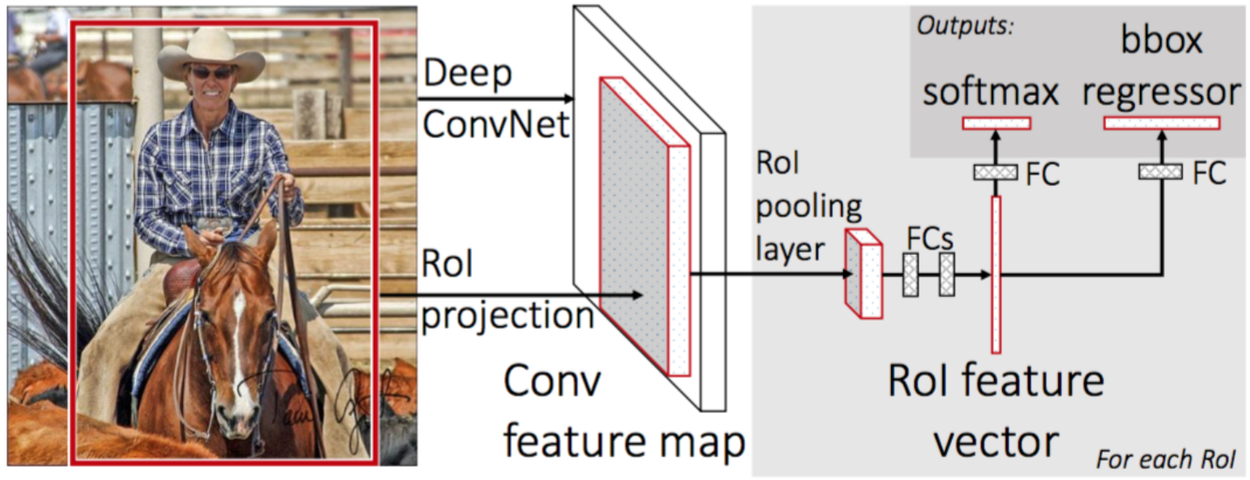
### 3.2. Mô hình Fast R-CNN

Fast R-CNN là một phiên bản nâng cấp hơn so với R-CNN giúp phần nào giải quyết được một phần điểm yếu về tốc độ.

#### 3.2.1. Kiến trúc mô hình Fast R-CNN

Là một phiên bản nâng cấp của R-CNN, nên Fast R-CNN cũng bao gồm hai thành phần:

- Region proposals module của Fast R-CNN vẫn là thuật toán Selective Search tương tự như R-CNN.
- Feature extraction module của Fast R-CNN là một mô hình phân lớp ảnh, cụ thể là VGG16. Các thành phần của Fast R-CNN không có thay đổi gì quá nổi bật so với R-CNN, tuy nhiên, điểm khác biệt mang lại giá trị của Fast R-CNN nằm ở cách mà nó kết hợp hai thành phần trên.




Khác với R-CNN, Fast R-CNN đưa toàn bộ ảnh ban đầu qua các lớp conv và pooling của Feature extraction module để tạo ra được đặc trưng của toàn bộ ảnh.

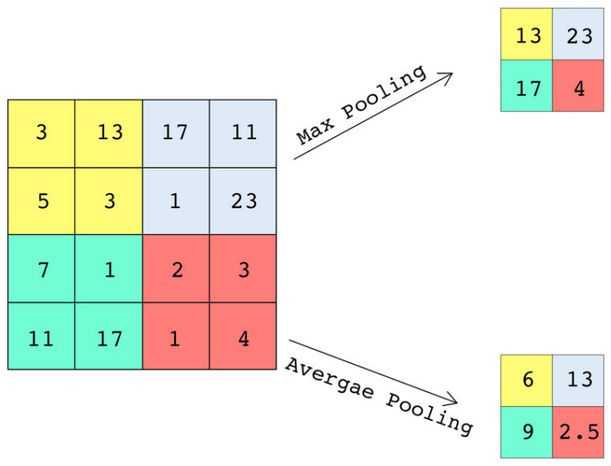
Tiếp theo, với mỗi khu vực mà thuật toán Selective Search đề xuất (regions of interest hay RoIs), Fast R-CNN crop từ đặc trưng của toàn bộ ảnh ra đặc trưng đại diện cho region proposal đó.

Cuối cùng, mỗi đặc trưng đại diện cho mỗi region proposal được đưa qua các lớp fully-connected và trả hai đầu ra gồm giá trị xác suất khu vực đó là đối tượng nào và giá trị độ lệch của bbox.

Tuy nhiên, mỗi region proposal từ thuật toán Selective Search có kích thước khác nhau, nên kích thước của đặc trưng đại diện cho mỗi region proposal cũng khác nhau. Tuy nhiên, ta lại cần các đặc trưng này có cùng kích thước để có thể đưa vào cùng chung các lớp fully-connected. Đây là lý do ra đời của lớp RoI pooling

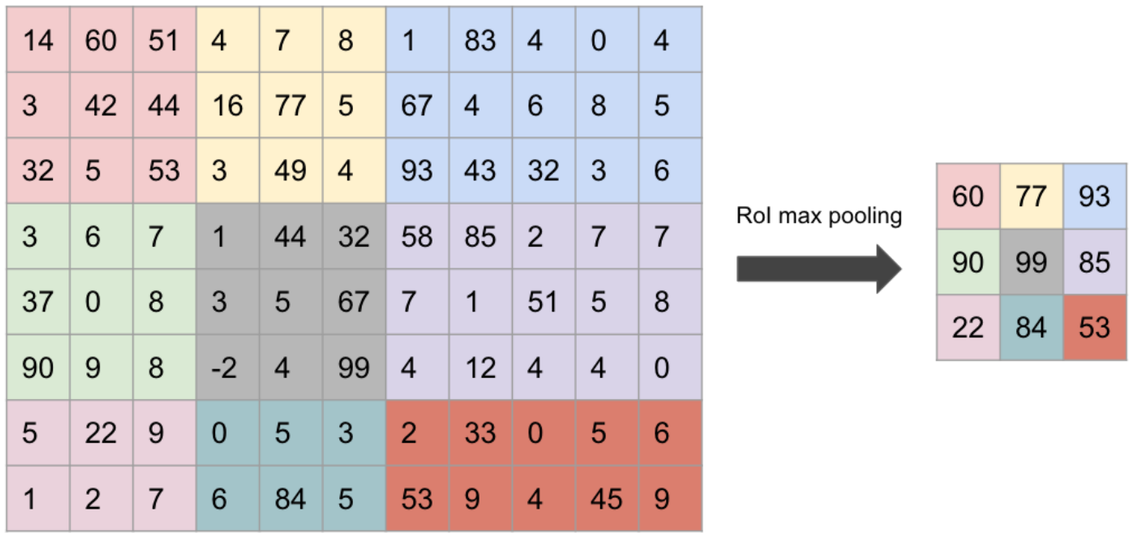
### 3.2.2. Lớp RoI pooling

Có hai phương pháp pooling phổ biến là maxpooling và average pooling.




Trong khi đó, RoIs pooling được giới thiệu không hoạt động giống như Max Pooling hay Average Pooling thông thường.

Thay vì yêu cầu ta phải định nghĩa kernel và stride của lớp pooling, RoI pooling yêu cầu ta phải định nghĩa kích thước của đặc trưng đầu ra, từ đó, RoI pooling sẽ tính toán và chia đặc trưng đầu vào thành các vùng trước khi thực hiện phép max pooling.




#### 3.2.3 Vấn đề tồn đọng của mô hình Fast R-CNN

Những kết quả vượt bậc về mặt tốc độ của mô hình Fast R-CNN đã giải quyết được vấn đề tồn đọng của R-CNN trong khi vẫn duy trì được độ chính xác cao.

Tuy nhiên, kiến trúc của mô hình Fast R-CNN vẫn phụ thuộc vào một thuật toán Selective Search và điều này tạo động lực để xây dựng mô hình deep learning thay thế cho các thuật toán này.

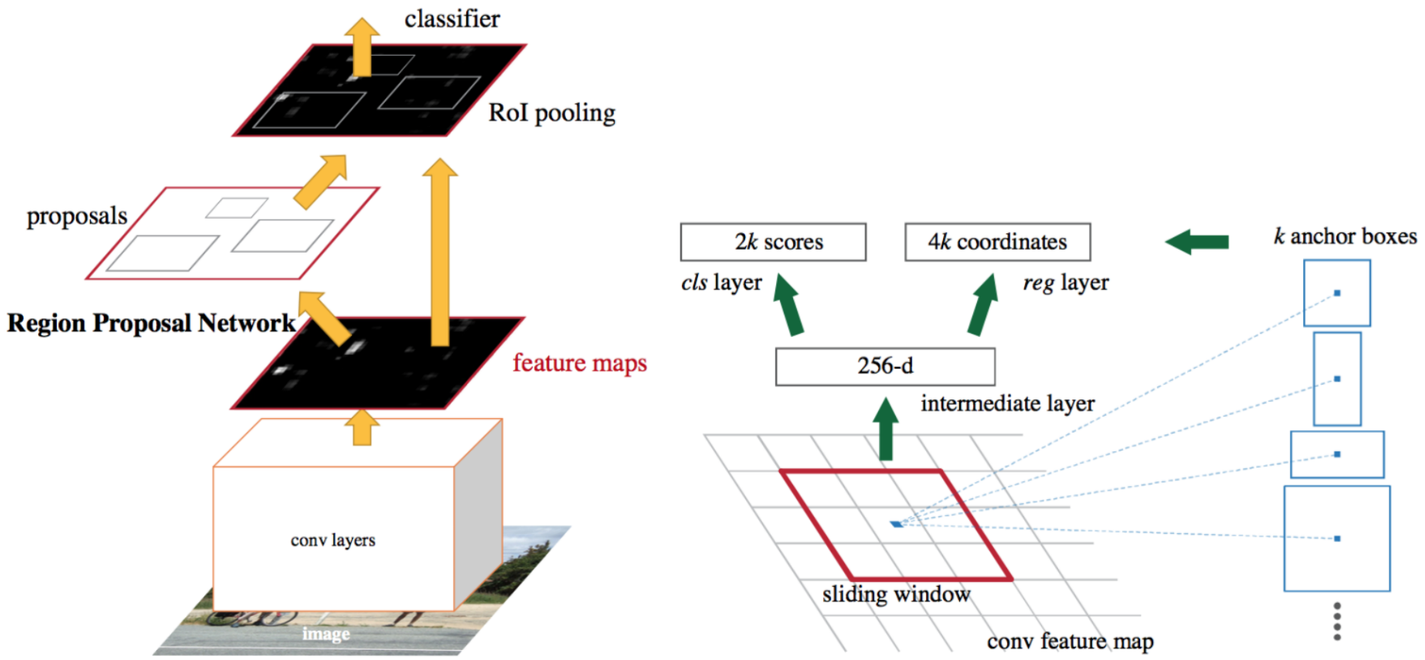
### 3.3. Mô hình Faster R-CNN

Mô hình Faster R-CNN được phát triển với trung tâm là kiến trúc mô hình Region Proposal Network (gọi tắt là RPN).

Mô hình RPN được kỳ vọng sẽ thay thế hoàn toàn các thuật toán như Selective Search trong thành phần Region proposals module của các mô hình two-stage. Việc thay thế các thuật toán bằng một kiến trúc deep learning hướng đến việc cải thiện không chỉ tốc độ của mô hình mà còn cải thiện về độ chính xác.

#### 3.3.1. Kiến trúc mô hình RPN và khái niệm Anchor

Mô hình RPN nhận đầu vào là ảnh với kích thước bất kỳ và trả đầu ra là toạ độ của các khu vực và xác suất khu vực đó là đối tượng nào trong các lớp đối tượng. Nhằm tiết kiệm chi phí tính toán, mô hình RPN dùng chung phần Feature extraction module với Fast R-CNN.




Mô hình RPN nhận đầu vào là feature maps từ Feature extraction module và trả đầu ra là các region proposal gọi là các anchor. Cụ thể:

- RPN đưa feature maps qua một lớp Conv và thu được feature maps mới có kích thước WxH.
- Với mỗi pixel trên feature maps kích thước WxH, tác giả lấy ra 9 khu vực gọi là 9 anchor. Từ đó, ta có tổng cộng (WxHx9) anchor.
- Các feature maps đại diện cho các anchor này được tiếp tục đưa qua các lớp Conv để biến đổi về các feature maps mới
  - có dạng (WxHx9)x1 đại diện cho xác suất anchor đó là object
  - có dạng (W xHx9)x4 đại diện cho 4 toạ độ x của góc trái trên, y của góc trái trên, chiều dài và chiều rộng của bbox.

Một điểm mạnh của RPN so với các mô hình object detection thời bấy giờ đó chính là khả năng dự đoán được các object có kích thước khác nhau và tỷ lệ giữa chiều dài và chiều rộng khác nhau nhờ vào cách cấu hình của anchor.

#### 3.3.2. Hàm loss và cách train mô hình RPN

Để train được RPN, nhóm tác giả gán cho mỗi anchor một lớp groundtruth và thiết lập hàm loss đối với từng anchor. Nhóm tác giả gán lớp groundtruth positive cho anchor dựa theo hai cách sau:

- Những anchor có chỉ số IoU lớn nhất đối với một groundtruth bbox được gán là anchor positive.
- Những anchor có chỉ số IoU lớn hơn 0.7 đối với một groundtruth bbox được gán là anchor positive.

Với hai cách như trên, một groundtruth bbox có thể gán được cho nhiều anchor khác nhau.

Ngoài ra, nhóm tác giả cũng gán lớp groundtruth negative cho các anchor không phải là positive và có chỉ số IoU nhỏ hơn 0.3 đối với một groundtruth bbox.

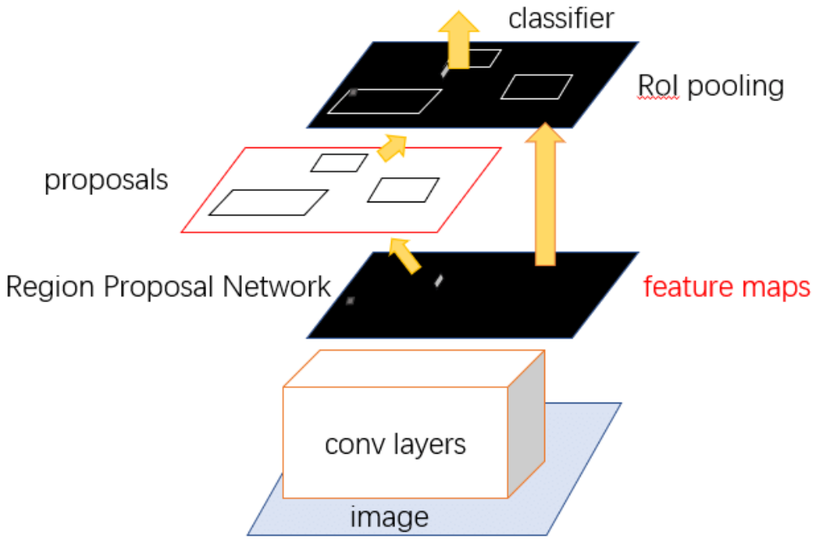
RPN được thiết kế để có thể train cùng với quá trình train object detection từ đó giúp kết quả đề xuất khu vực trở nên chính xác hơn.

Tuy nhiên, RPN sẽ đề xuất ra nhiều các anchor negative hơn rất nhiều so với số anchor positive từ đó gây mất cân bằng dữ liệu. Ngoài ra, việc train mô hình với toàn bộ số anchor được đề xuất ra cũng sẽ khiến cho khối lượng tính toán lớn và thời gian kéo dài quá trình train.

Từ đó, nhóm tác giả đề xuất việc lựa chọn ngẫu nhiên 256 anchor trên mỗi ảnh để thực hiện việc tính loss. Việc lựa chọn này giúp tỷ lệ anchor positive và negative trở nên cân bằng hơn và giảm thiểu bởi những phần khối lượng tính toán dư thừa.

#### 3.3.3. Sự kết hợp giữa Region Proposal Network và Fast R-CNN

Nhóm tác giả cho rằng, việc train mô hình RPN và Fast R-CNN cần phải diễn ra đồng thời, vì từ đó, việc chia sẻ chung thành phần backbone Conv mới trở nên hiệu quả.




Nhóm tác giả nêu ra ba phương án để train RPN kết hợp với Fast R-CNN:

- Cách 1: Alternating training:
    - Nhóm tác giả train RPN trước sử dụng những hàm loss của RPN nói trên.
    - Sau khi train xong RPN, tác giả sử dụng những khu vực được đề xuất bởi RPN để train Fast R-CNN.
    - Backbone sau khi được train bởi Fast R-CNN tiếp tục được sử dụng để train RPN mới và vòng lặp này tiếp tục diễn ra cho đến khi kết quả hội tụ.
- Cách 2: Approximate joint training:
    - Phương pháp này kết hợp RPN và Fast R-CNN thành một mô hình duy nhất trong quá trình train.
    - Các khu vực được đề xuất bởi RPN được coi như là tất định đối với nhánh Fast RCNN và khiến cho phương pháp train này được gọi là approximate bởi vì những thông tin từ nhánh Fast R-CNN sẽ không được cập nhật cho nhánh RPN.
    - Quá trình backprop được thực hiện độc lập giữa RPN và Fast R-CNN, riêng phần backbone chung được cập nhật theo giá trị hàm loss chung.
    - Phương pháp này đạt hiệu quả thấp hơn chút so với Alternating training tuy nhiên thời gian train được giảm 25 - 50%.
- Cách 3: Non-approximate joint training:
    - Phương pháp này cải thiện được vấn đề tồn đọng của Approximate joint training.
    - Tuy nhiên, để làm được điều này, nhóm tác giả cần tinh chỉnh lại lớp RoI pooling trong Fast R-CNN để có thể update cho cả các thành phần của Fast RCNN và RPN.

Tóm lại, nhóm tác giả dựa vào phương pháp Alternating training và thực hiện quá trình train gồm 4 bước như sau:

- Bước 1: Nhóm tác giả khởi tạo RPN với pretrained ImageNet và train RPN.
- Bước 2: Nhóm tác giả khởi tạo Fast R-CNN với pretrained ImageNet và train Fast R-CNN với các khu vực được đề xuất bởi RPN.
- Bước 3: Nhóm tác giả khởi tạo lại RPN nhưng sử dụng phần backbone đã được train từ bước 2. Nhóm tác giả chỉ train những lớp riêng của RPN và không cập nhật cho phần backbone.
- Bước 4: Nhóm tác giả finetune lại những lớp riêng của Fast RCNN với các khu vực được đề xuất bởi RPN và thu được Faster R-CNN cuối cùng.

Nhóm tác giả cũng đã lặp lại 4 bước trên vài lần nhưng kết quả không thay đổi quá nhiều.

#### 3.3.4. Vấn đề tồn đọng của mô hình Faster R-CNN

Kết quả của Faster R-CNN và tâm điểm là kiến trúc RPN giúp thay thế thuật toán Selective Search đã giúp cho Faster R-CNN đạt độ chính xác cao hơn so với Fast R-CNN sử dụng Selective Search.

Hơn nữa, RPN giúp cho Faster R-CNN nhanh hơn tới 10 lần so với cấu hình tương tự Fast R-CNN sử dụng Selective Search.

Điều này giúp cho Faster R-CNN cho đến nay vẫn là một mô hình tốt để giải quyết bài toán object detection, vừa đạt độ chính xác cao, vừa có tốc độ tương đối tốt.