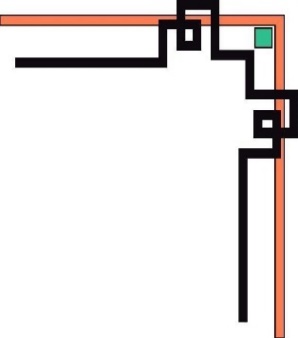
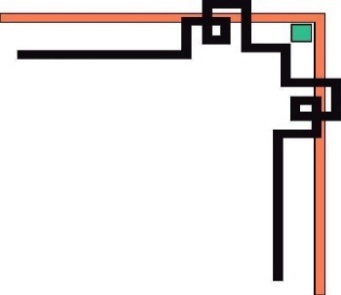
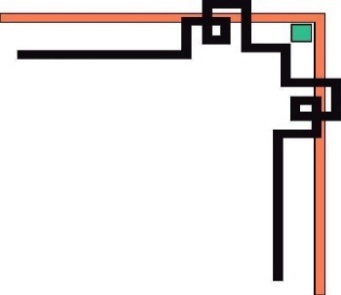
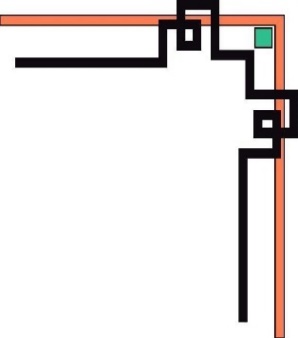
**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT**



**TP. HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**NGÀNH CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO ĐỒ ÁN 3**

**Đề tài: Phần mềm nhận dạng trái cây**

**tr mô hình Mask-RCNN**

**GVHD: ThS.Trần Tiến Đức**

|  |  |
| --- | --- |
| MSSV | Họ và tên |
| 17110246 | Nguyễn Nhật Trường |
| 17110149 | Nguyễn Hứa Huy |
| 17110101 | Phạm Dĩ Bằng |
| 17110253 | Nguyễn Quốc Việt |

**TP.HCM, ngày 13 tháng 1 năm 2021**

# **LỜI CẢM ƠN**

Lời đầu tiên, cho phép nhóm em được gửi lời cảm ơn chân thành đến thầy Trần Tiến Đức , người đã phụ trách hướng dẫn nhóm chúng em trong việc thực hiện và hoàn thành bài báo cáo một cách tốt và hiệu quả nhất. Những sự chỉ dạy, nhật xét, đóng góp ý kiến từ thầy đã góp phần rất quan trọng trong việc giúp chúng em hoàn hành đề tài một cách hoàn thiện nhất.

Cảm ơn các thầy cô trong khoa Công Nghệ Thông Tin đã luôn nhiệt tình, cống hiến, tận tình giải đáp các thắc mắc của chúng em. Bên cạnh đó, chúng em xin được cảm ơn các bạn cùng khóa, các anh chị khóa trên đã không ngần ngại chia sẻ các thông tin, kinh nghiệm bổ ích giúp chúng em hoàn thiện hơn đề tài của mình.

Báo cáo được thực hiện trong một khoảng thời gian có hạn, cùng với những hạn chế về mặt kiến thức nên việc sai sót là điều không thể tránh khỏi trong quá trình thực hiện. Chúng em kính mong nhận được những lời nhận xét, ý kiến đóng góp quý báo từ quý Thầy Cô để chúng em có thêm kinh nghiệm và hoàn thành tốt hơn những đề tài sau này.Chúng em xin chân thành cảm ơn!

**MỤC LỤC**

[LỜI CẢM ƠN 2](#_Toc62406393)

[MỤC LỤC 1](#_Toc62406394)

[DANH MỤC BẢNG 2](#_Toc62406395)

[DANH MỤC HÌNH ẢNH 3](#_Toc62406396)

[CHƯƠNG 1: TỔNG QUAN 1](#_Toc62406397)

[1.1 Tính cấp thiết của đề tài 1](#_Toc62406398)

[1.2 Mục đích của đề tài 1](#_Toc62406399)

[1.3 Các thành phần chính hệ thống phát hiện và phân vùng đối tượng 2](#_Toc62406400)

[1.4 Một số nghiên cứu liên quan về phát hiện và phân loại các loại trái cây trong ảnh 3](#_Toc62406401)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 5](#_Toc62406402)

[2.1 Giới thiệu chung về mạng nơ ron tích chập 5](#_Toc62406403)

[2.1.1 Mạng nơ ron 5](#_Toc62406404)

[2.1.2 Mạng nơ ron tích chập 6](#_Toc62406405)

[2.1.3 Kiến trúc của CNN cho bài toán nhận dạng và phân vùng đối tượng R-CNN 8](#_Toc62406406)

[2.2 Mạng Mask R-CNN 12](#_Toc62406407)

[CHƯƠNG 3: TRIỂN KHAI MÔ ĐUN PHÁT HIỆN VÀ PHÂN LOẠI TRÁI CÂY 18](#_Toc62406408)

[3.1 Mô hình đề xuất nghiên cứu 18](#_Toc62406409)

[3.1.1. Quá trình huấn luyện Mask R-CNN 20](#_Toc62406410)

[3.1.2. Cài đặt module và huấn luyện Mask R-CNN 21](#_Toc62406411)

[3.2. Kết quả thí nghiệm 22](#_Toc62406412)

[3.2.1. Kết quả huấn luyện với ảnh resize kích cỡ 256x256 23](#_Toc62406413)

[3.2.2. Kết quả huấn luyện với ảnh crop kích cỡ 292x261 25](#_Toc62406414)

[CHƯƠNG 4: KẾT LUẬN 27](#_Toc62406415)

[4.1 Kết quả đạt được 27](#_Toc62406416)

[4.2 Nhược điểm và hướng phát triển 27](#_Toc62406417)

[4.2.1 Nhược điểm 27](#_Toc62406418)

[4.2.2 Hướng phát triển 27](#_Toc62406419)

[TÀI LIỆU THAM KHẢO 29](#_Toc62406420)

**DANH MỤC BẢNG**

[Table 1: Kết quả thử nghiệm trên ảnh gốc 23](#_Toc62406291)

[Table 2: Kết quả thử nghiệm với ảnh resize (292x261) 23](#_Toc62406292)

**DANH MỤC HÌNH ẢNH**

[Hình 1: Sơ đồ tổng thể hệ thống áp dụng học máy (https:machinlearningcoban.com) 3](#_Toc62406325)

[Hình 2: Minh họa mạng nơ ron của người (từ Rob Fergus) 5](#_Toc62406326)

[Hình 3: Mạng nơ ron nhiều tầng 6](#_Toc62406327)

[Hình 4:Kết nối giữa các tầng trong mạng nơ ron truyền thống 7](#_Toc62406328)

[Hình 5:Kết nối giữa các tầng trong mạng nơ ron tích chập 7](#_Toc62406329)

[Hình 6:Các bước cơ bản trong mạng neuron tích chập 8](#_Toc62406330)

[Hình 7:Cấu trúc mạng R-CNN 8](#_Toc62406331)

[Hình 8:Kiến trúc Fast R-CNN 10](#_Toc62406332)

[Hình 9: Hoạt động của Spatial pyramid pooling 11](#_Toc62406333)

[Hình 10: Kiến trúc Faster R-CNN 12](#_Toc62406334)

[Hình 11: Kiến trúc Mask R-CNN 13](#_Toc62406335)

[Hình 12: Giải thích kiến trúc Mask R-CNN (Medium.com) 13](#_Toc62406336)

[Hình 13: Cách hoạt động của mạng tích chập đầy đủ 14](#_Toc62406337)

[Hình 14: Mô tả cách hoạt động của khối deconvolution và unpooling 14](#_Toc62406338)

[Hình 15: Max Pooling 15](#_Toc62406339)

[Hình 16: Mô tả phương pháp RoIAlign 16](file:///C:\Users\TranNguyen\Desktop\NguyenQuocViet-17110253-CNPMM-1.docx#_Toc62406340)

[Hình 17: Kiến trúc backbone Mask R-CNN 16](#_Toc62406341)

[Hình 18: Mạng kim tự tháp 17](#_Toc62406342)

[Hình 19: Giá trị hàm mất mát trong quá trình huấn luyện 21](file:///C:\Users\TranNguyen\Desktop\NguyenQuocViet-17110253-CNPMM-1.docx#_Toc62406343)

[Hình 20: Kết quả nhận dạng ảnh đã được resize 292x261 25](#_Toc62406344)

**CHƯƠNG 1: TỔNG QUAN**

## **1.1 Tính cấp thiết của đề tài**

Phần mềm nhận diện trái cây người là một bài toán cơ bản trong lĩnh vực thị giác máy tính. Ứng dụng của nó rất đa dạng, bao gồm hệ thống nhận diện các loại trái cây , nhận dạng tên và hình các loại trái cây thông qua hình ảnh . Để các hệ thống trên hoạt động một cách nhanh chóng và chính xác thì phát hiện các trái cây khác nhau là một pha hết sức quan trọng. Đối tượng là trái cây trong môi trường bình thường có rất nhiều các tư thế, góc nhìn khác nhau như to , nhỏ , các hình dạng khác nhau , các mẫu hình khác nhau. Vì vậy bài toán phát hiện và phân loại các trái cây vẫn còn là một thách thức.  
Trong khuôn khổ của Đồ Án 3, em hướng đến giải quyết bài toán phát hiện và nhận diện các loại trái cây khác nhau người trong một ứng dụng nhận diện thông minh (AI).Môi trường thử nghiệm là các vườn trái cây bao gồm các loại trái cây khác nhau và sau đó chụp các loại trái cây vào app nhận diện và qua hình ảnh đó chúng ta nhận diện được các loại trái cây và hiển thị thông tin tên trái cây . Tuy nhiên, nghiên cứu mới này tập trung đánh giá độ bền vững của các giải thuật đề xuất dựa khi có sư thay đổi của góc nhìn. Với bài toán này, các thách thức cơ bản đặt ra đó chính là sự thay đổi góc nhìn khi cùng quan sát một loại trái cây.

Nhận diện và phân loại các loại trái cây là một bài toán có nhiều thách thức đối với các nhà khoa học do nhiều nguyên nhân , góc nhìn thay đổi , đa dạng mẫu hình, kích thước trong từng loại trái cây.Để việc phát hiện được chính xác cần có các phương pháp biểu diễn tốt cũng như cần có một bộ dữ kiệu đủ lớn và đa dạng để việc học có hiệu quả cao. Trong đồ án 3 này nhóm em tìm hiểu các nghiên cứu liên quan , từ đó đề xuất giải pháp cho bài toán phát hiện nhận dạng các loại trái cây khác nhau thông qua hình dáng trái cây và tên trái cây

## **1.2 Mục đích của đề tài**

Đề tài “XÂY DỰNG PHẦN MỀM NHẬN DIỆN TRÁI CÂY” đặt ra 2 vấn đề trọng tâm về mục tiêu mà sinh viên nghiên cứu cần thực hiện được bao gồm:

Vấn đề đầu tiên đưa ra về yếu tố lý thuyết, sinh viên thực hiện cần có kết quả nghiên cứu chuyên môn và hiểu được tổng quan và khả năng ứng dụng của từng thành phần trong mô hình nói chung. Với từng thành phần xác định được các đặc trưng về nguyên tắc hoạt động, thành phần hoặc mô hình. Thư viện được xây dựng dựa trên ngôn ngữ lập trình Python đạt được sự phổ biến đến lập trình viên. Song song với việc hiểu về thành phần tiêu biểu của Python sinh viên cũng cần phải có thêm tìm hiểu về các thư viện, mã nguồn module giúp ích trong quá trình xây dựng phần mềm thực tế.

Vấn đề thứ hai của đề tài đặt ra về ứng dụng Python vào xây dựng một sản phẩm cụ thể, ở đây là “Phần mềm nhận diện trái cây”. Sản phẩm hoàn thành phải là một ứng dụng AI được xây dựng có đầy đủ các thành phần và đáp ứng thêm các yêu cầu về nhận diện và phân loại trái cây

## **1.3 Các thành phần chính hệ thống phát hiện và phân vùng đối tượng**

Một hệ thống phát hiện và phân loại đối tượng ảnh dựa trên các kỹ thuật học có giám sát (để phân lớp đối tượng quan tâm với các đối tượng ảnh còn lại) thông thường được thực hiện thông qua 2 pha sau:

- Pha huấn luyện: Sử dụng bộ dữ liệu học đưa vào huấn luyện để đưa ra mô hình nhận dạng.

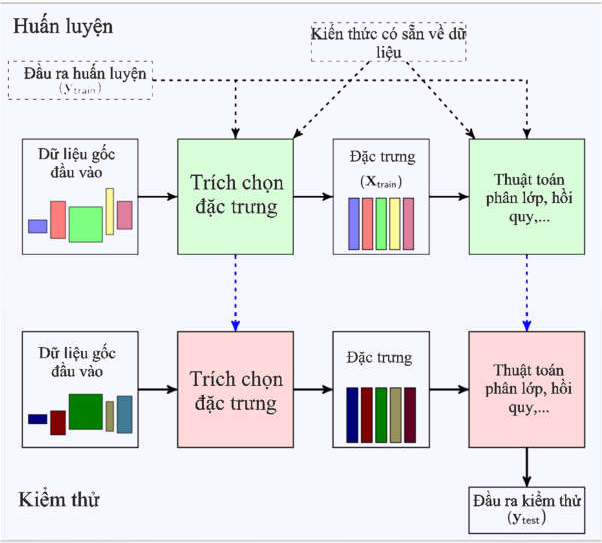
- Pha thử nghiệm: Sử dụng mô hình vừa huấn luyện được ở trên để thực hiện phân lớp trên dữ liệu mới.

Hình 1 mô tả mô hình chung cho một bài toán áp dụng kĩ thuật học máy.

Áp dụng cho bài toán phát hiện, phân vùng đối tượng mỗi pha đều có các khối xử lý cơ bản như sau:

* Tiền xử lý dữ liệu: Chuyển đổi dữ liệu đầu vào sang định dạng chuẩn với mô hình.
* Trích chọn đặc trưng: Từ dữ liệu đã được tiền xử lý, thực hiện trích rút các đặc trưng biểu diễn đối tượng quan tâm.
* Huấn luyện: sử dụng các đặc trưng được trích chọn để làm dữ liệu đầu vào cho việc huấn luyện các khối trong hệ thống có khả năng huấn luyện được như khối trích trọn đặc trưng, khối phân lớp,...
* Dự đoán bao đóng đối tượng: dữ liệu cần phát hiện được chuyển qua các bước tiền xử lý, trích chọn đặc trưng, sau đó dùng bộ phân lớp đã được huấn luyện để dự đoán bao đóng có chứa đối tượng trên ảnh
* Phân vùng : Từ ánh xạ đặc trưng trích đưa các vùng quan tâm vào mạng để phân lớp từng pixel ảnh để dự đoán mặt nạ trên ảnh. Phân vùng và dự đoán bao đóng đối tượng có thể tách rời hoặc nằm chúng trong một khối tùy hệ thống

Hình 1 minh họa sơ đồ tổng thể hệ thống với các thành phần như phân tích ở trên.



Hình 1: Sơ đồ tổng thể hệ thống áp dụng học máy (https:machinlearningcoban.com)

* 1. **Một số nghiên cứu liên quan về phát hiện và phân loại các loại trái cây trong ảnh**

Có nhiều phương pháp đã được đề xuất cho bài toán phát hiện và phân loại trái cây trong ảnh. Các phương pháp này có thể được chia thành hai loại: nhóm các phương pháo dựa trên các đặc trưng trích chọn biểu diễn các hình dáng trái cây và nhóm các phương pháp dựa trên đặc trưng học được từ dữ liệu. Phần dưới sẽ trình bày tóm lược những tìm hiểu của em về một số phương pháp thuộc từng nhóm tiếp cận này.

*Hướng tiếp cận phát hiện các loại trái cây dựa trên các đặc trưng của mỗi loại trái cây*

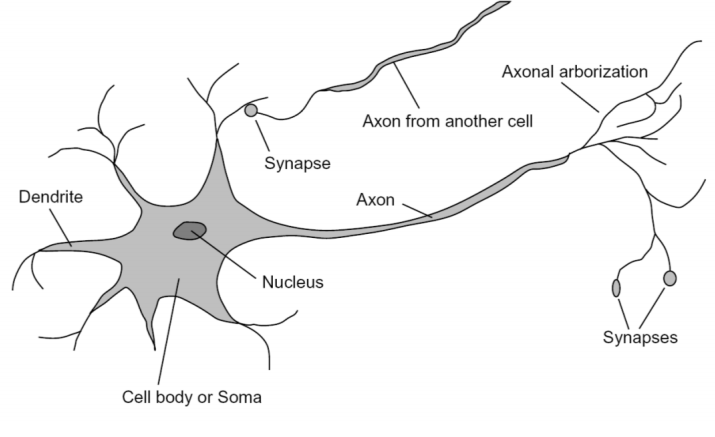
Đặc trưng hình dáng của trái cây là các đặc trưng hình dáng của từng loại trái cây, nhằm đưa ra các cấu trúc đặc trưng mới phù hợp nhất cho từng trái cây .

**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT**

**2.1 Giới thiệu chung về mạng nơ ron tích chập**

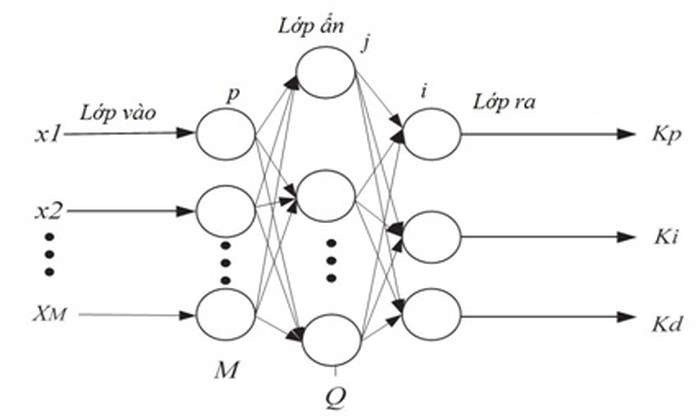
### **2.1.1 Mạng nơ ron**

Mạng nơ ron nhân tạo được thiết kế nhằm mô phỏng mạng neuron của bộ não người.Mạng neuron nhân tạo được cấu thành từ một tập các phần tử xử lý đơn giản được kết nối với nhau. Mỗi phần tử xử lý này chỉ có thể thực hiện được một thao tác tính toán nhỏ, nhưng một mạng lưới các phần tử như vậy có một khả năng tính toán lớn hơn rất nhiều. Phần tử tính toán cơ bản của mạng nơ ron là một perceptron hay một nơ ron.



Hình 2: Minh họa mạng nơ ron của người (từ Rob Fergus)

Một nơ ron mô phỏng quá trình tính toán của bộ não con người. Dữ liệu được đưa tới các nơ ron thông qua các Dendrite vào Nucleus để tính toán. Tín hiệu ra được xuất ra ở dây Axon. Các nơ ron được liên kết với nhau thông qua các dây Synapse. Mạng nơ ron bao gồm rất nhiều phần tử như vậy liên kết với nhau



Hình 3: Mạng nơ ron nhiều tầng

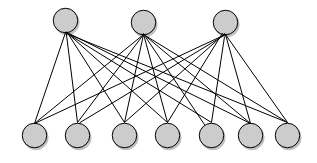
Hình minh họa một mạng nơ ron truyền thẳng với 3 loại node sau:  
- Input nodes (Node đầu vào): Chứa dữ liệu đầu vào từ bên ngoài và đưa trực tiếp vào các Hidden nodes.  
- Hidden nodes (Node ẩn): Nó không chứa kết nối trực tiếp đến dữ liệu từ bên ngoài. Nó thực hiện tính toán các dữ liệu nhận được từ các input nodes, thực hiện tính toán và đưa ra các output nodes. Tập hợp các node ẩn trong mạng tạo thành tầng ẩn. Một mạng nơ ron truyền thẳng có thể có hoặc không có tầng ẩn.  
- Output nodes (Node đầu ra): Có nhiệm vụ tính toán và đưa dữ liệu từ trong mạng ra bên ngoài.

**2.1.2 Mạng nơ ron tích chập**

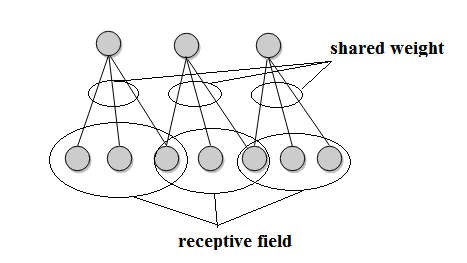
Mạng nơ ron tích chập là một mạng neuron nhân tạo với các toán tử tích chập. Nó có khả năng học một lượng lớn các dữ liệu trong khoảng thời gian ngắn hơn nhiều so với mạng nơ ron thông thường. Lý do là nó sử dụng ít trọng số hơn trong khi độ chính xác chỉ kém hơn một phần nhỏ so với kiến trúc truyền thống. Mô hình này sử dụng trong [16] và đã đạt kết quả khá tốt trong bài toán phân loại ảnh.

Trong mạng nơ ron truyền thống, các node ở các tầng phía sau sẽ liên kết với toàn bộ các node ở layer phía dưới thông qua một tập các trọng số. Với mỗi nơ ron khác nhau, chúng ta cần một tập trọng số hoàn toàn độc lập để liên kết với các nơ ron ở tầng trước đó.

Điểm khác biệt của mạng nơ ron tích chập so với mạng nơ ron truyền thống đó là trong liên kết giữa 2 tầng liên tiếp nhau việc các node ở các tầng phía sau chỉ liên kết với một bộ phận các node ở tầng phía trước đó gọi là receptive field thông qua một tập các trọng số. Hơn nữa tập trọng số này là như nhau đối với mỗi nơ ron ở tầng sau. Do đó số lượng tham số cần huấn luyện ít hơn trong khi vẫn giữ được lượng thông tin cần thiết.



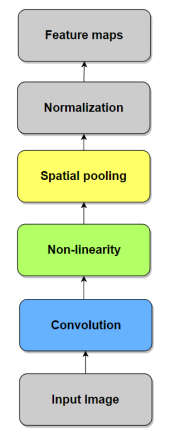
Hình 4:Kết nối giữa các tầng trong mạng nơ ron truyền thống



Hình 5:Kết nối giữa các tầng trong mạng nơ ron tích chập

**Một mạng nơ ron tích chập thường được thực hiện thông qua các bước sau :**

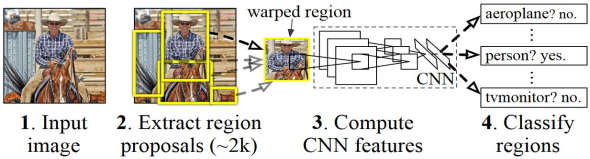
* Convolutional layer
* Pooling layer
* Non-linearity layer
* Fully-connected layer



Hình 6:Các bước cơ bản trong mạng neuron tích chập

Trên đây là các khối cơ bản trong một mạng neuron tích chập, mạng neuron tích chập được cấu thành bằng việc xếp trồng nhiều các lớp thành một cấu trúc chặt chẽ. Trong nội dung đồ án này em sẽ không đề cập đến việc thiết kế mạng chỉ áp dụng những cấu trúc đã được nghiên cứu và cho hiệu quả tốt

### **2.1.3 Kiến trúc của CNN cho bài toán nhận dạng và phân vùng đối tượng R-CNN**



Hình 7:Cấu trúc mạng R-CNN

Ở phần trên em đã giải thích về các kĩ thuật đã sử dụng để tạo ra các mạng nơ ron tích chập. Trong phần này em sẽ trình bày kiến trúc cụ thể sử dụng cho bài toán phát hiện và phân vùng đối tượng trái cây em sử dụng trong đồ án này.

Để hiểu được kiến trúc Mask R-CNN em đi theo trật tự như sau:

Fast R-CNN

R-CNN

Fast R-CNN

R-CNN

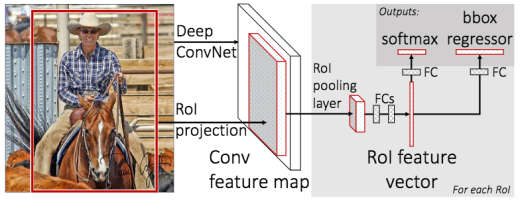
R-CNN được đề xuất từ năm 2013 thể hiện 3 bước để nhận dạng đối tượng:

* Tạo ra các vùng bao đóng giới hạn trên ảnh bằng một thuật toán đề xuất vùng bên ngoài mạng như selective search
* Đóng gói dữ liệu từ vùng đề xuất (các bước tiền xử lí) để đưa vào mạng đã được học sẵn để trích dẫn ra các đặc trưng
* Truyền đặc trưng trích dẫn được từ bước trên đưa qua bộ phân lớp tách rời có thể là một mạng kết nối đầy đủ hoặc SVM để chọn xem bao đóng có thực sự chứa đối tượng quan tâm hay không.

Kiến trúc R-CNN đạt được hiệu quả độ chính xác khoảng 40% trên tập dữ liệu Pascal VOC (2010).Hình 11 từ bài báo [17] mô tả chi tiết các khối của kiến trúc R-CNN. Tuy vậy mạng R-CNN vẫn gặp một số vấn đề sau Tăng độ chính xác của R-CNN chỉ là một phần của R-CNN, vẫn đề chủ yếu đặt ra với R-CNN đó là nó chạy khá chậm và qua một số thí nghiệm cho thấy vấn đề tốc độ bị giới hạn là do R-CNN chạy CNN độc lập cho mỗi bao đóng và việc sinh ra hộp bao bởi các thuật toán bên ngoài là rất nhiều. Cụ thể R-CNN chạy mất khoảng 40s cho mỗi ảnh trên GPU. Mô hình này rất khó để có thể học do có 2 phần mô hình học máy (CNN và SVM) cần học độc lập. Thêm nữa R-CNN yêu cầu một thuật toán bên ngoài như selective search

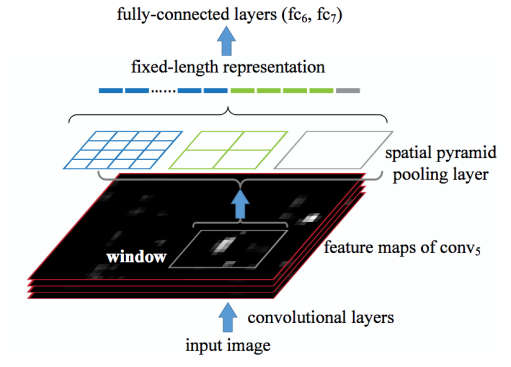
Fast R-CNN

Để khắc phục những hạn chế của R-CNN, một thuật toán mới cải tiến từ R-CNN đã được đề xuất với tên gọi Fast R-CNN. Cải tiến chính của thuật toán này so với R-CNN là cải tiến về tốc độ. Thời gian phát hiện đối tượng trên một ảnh giảm từ 40s xuống còn 0.2s tức là nhanh hơn khoảng 200 lần so với R-CNN. Ý tưởng chính của Fast R-CNN đó là sử dụng một mạng nơ ron duy nhất cho trích xuất đặc trưng và phân lớp thay thế cho mạng SVM độc lập. Kiến trúc cụ thể được mô tả trong hình dưới đây:



Hình 8:Kiến trúc Fast R-CNN

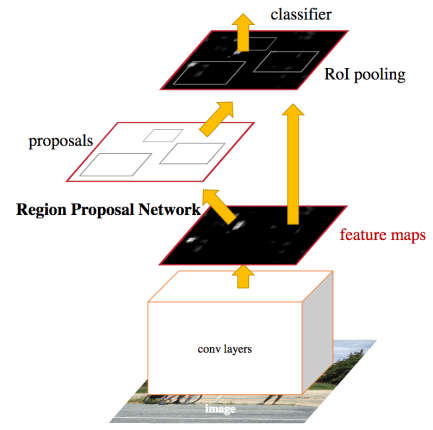
Để phát hiện đối tượng một cách độc lập với kích cỡ ảnh, Fast R-CNN sử dụng một tầng Spatial Pyramid Pooling được giới thiệu trong [18]. Ý tưởng là thay vì cắt ảnh đầu vào thành nhiều các phần khác nhau, Fast R-CNN tính toán đặc trưng một lần qua mạng trích xuất đặc trưng và ánh xạ vùng quan tâm trên ảnh (được sinh ra nhờ thuật toán đề xuất selective search), mỗi vùng quan tâm trên đặc trưng được pooling theo các kích cỡ khác nhau rồi ghép lại thành một đặc trưng có kích cỡ cố định và đưa vào tầng tiếp theo, nhờ vậy mà đặc trưng tạo ra độc lập với kích cỡ vùng quan tâm. Bên cạnh đó dễ thấy kiến trúc này có thể tái sử dụng những lớp CNN ở tầng trên thay vì phải tính hàng nghìn lần như R-CNN. Spatial Pyramid Pooling được sử dụng để chuẩn hóa đầu vào cho tầng kết nối đầy đủ khi mà kích cỡ RoI là biến đổi. Hình mô tả chi tiết cách Spatial Pyramid Pooling hoạt động:



Hình 9: Hoạt động của Spatial pyramid pooling

Mặc dù nhanh hơn khoảng 200 lần so với R-CNN tuy nhiên Fast R-CNN vẫn chưa đủ nhanh  
để hoạt động trong các hệ thống yêu cầu thời gian thực

**Faster R-CNN**Nhanh hơn Fast R-CNN một mô hình có tên là Faster R-CNN được đề xuất trong [19].Faster R-CNN cải tiến hơn so với Fast R-CNN bằng việc thay thế thuật toán sinh vùng độc lập với mô hình. Mạng Faster R-CNN chỉ cần sử dụng một mạng nơ ron duy nhất để thực hiện nhiệm vụ sinh vùng đặc trưng bằng việc thêm vào mạng một số tầng mới làm nhiệm vụ đề xuất vùng đặc trưng khối này có tên Region Proposal Network (RPN). Vì vậy kiến trúc  
Faster R-CNN được mô tả như hình 14:

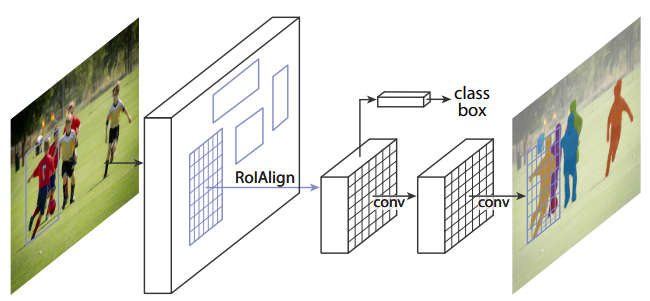


Hình 10: Kiến trúc Faster R-CNN

RPN là một mạng tích chập đầy đủ (Fully convolution network) không có các tầng kết nối đầy đủ do đó chi phí tính toán không đáng kể khoảng 10ms cho mỗi ảnh. Ngoài ra Faster R-CNN còn thay thế các Spatial Pyramid bằng các Anchor với các tỉ lệ khung khác nhau.

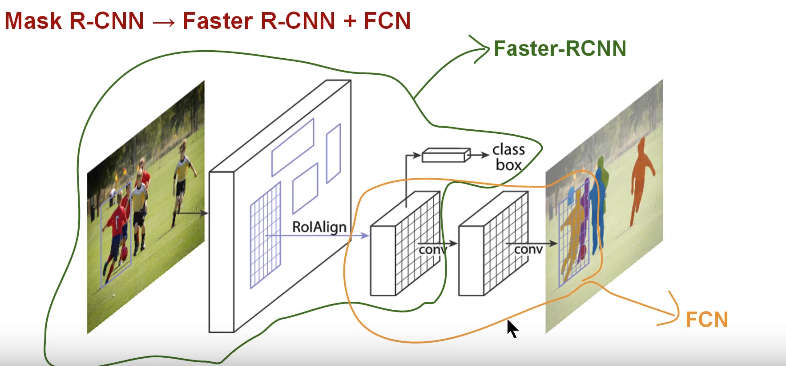
**2.2 Mạng Mask R-CNN**

Mask R-CNN được phát triển bởi Kaiming He và đồng nghiệp. Mask R-CNN kế thừa từ kiến trúc Faster R-CNN bằng cách thêm một nhánh làm nhiệm vụ dự đoán phân vùng đối tượng song song với nhánh dự đoán lớp đối tượng và hộp bao. Tốc độ Mask RCNN đạt được khá nhanh khoảng 5 frame trên giây. Mask R-CNN là mô hình tốt nhất trên tập dữ liệu COCO năm 2016. Bên cạnh đó việc cài đặt và sử dụng cũng tương đối đơn giản. Vì vậy em đã chọn Mask R-CNN là để nghiên cứu cho bài toán cụ thể là phát hiện và phân vùng trái cây trong ảnh và video, một pha tiền xử lý cho hệ thống nhận dạng cử chỉ động của trái cây trong ứng dụng điều khiển thiết bị trong phòng thông minh.  
**Kiến trúc mạng Mask R-CNN**Mask R-CNN mở rộng Faster R-CNN bằng cách thêm nhánh dự đoán mặt nạ đối tượng trên mỗi vùng đề xuất từ RPN. Kiến trúc các phần còn lại tương đối giống với Faster RCNN. Hình 15 mô tả kiến trúc Faster R-CNN:



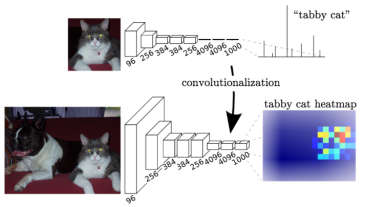
Hình 11: Kiến trúc Mask R-CNN

Nhánh phía trên dùng dự đoán hộp bao và lớp đối tượng trên ảnh, và nhánh dưới là nhánh làm nhiệm vụ đánh nahnx cho mỗi pixel trong vùng quan tâm (RoI) để xây dựng mặt nạ đối tượng. Hình 16 giải thích lại kiến trúc Mask R-CNN như sau:



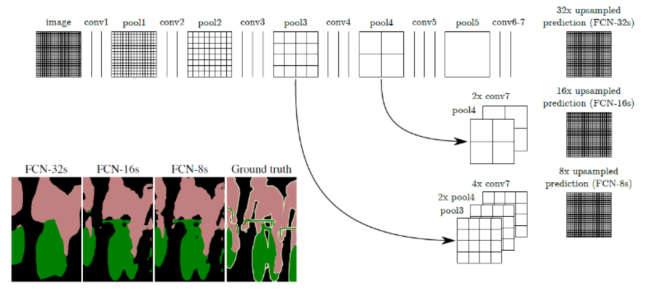
Hình 12: Giải thích kiến trúc Mask R-CNN (Medium.com)

Như đã đề cập ở trên FCN được sử dụng ở cả 2 nhiệm vụ phát hiện đối tượng và phân vùng đối tượng. Một mạng nơ ron bình thường được sử dụng cho phát hiện và nhận dạng đối tượng ở tầng cuối thường là một vector có cùng kích thước với số lớp và cho biết điểm số dự đoán của mỗi lớp. Nếu chúng ta dừng lại ở một số lớp trung gian của mạng và thay thế vector bởi một số bước tích chập, và thay vì một vector có kích thước bằng số lớp ta sẽ có số lượng đặc trưng cùng kích thước với số lớp. Sau quá trình học thích hợp chúng ta có được điểm số dự đoán lớp cho tất cả các điểm ảnh của lớp cuối cùng. Mỗi lớp sẽ nhận được một “bản đồ nhiệt”. Đây chính là cách hoạt động của bộ phân loại trên top và tích chập đầy đủ bên dưới hoạt động



Hình 13: Cách hoạt động của mạng tích chập đầy đủ

Phần phân đoạn của Mask R-CNN sẽ sử dụng kết quả “bản đồ nhiệt” như trên và từ đây ta deconvolution và unpooling để thu được mặt nạ trên ảnh gốc. Cụ thể cách thức hoạt động của 2 bước này như sau:  
- Deconvolution (giải tích chập): thực chất chỉ là tích chập với ma trận chuyển vị  
- Unpooling: để hiểu về unpooling ta cùng xem lại về pooling. Với max pooling ta lấy giá trị max của khối vì vậy thông tin sẽ bị mất mát đi trong quá trình pooling. Unpooling là quá trình ta xây dựng lại ma trận bằng cách ghi nhớ tọa độ của điểm max và điền lại chính xác điểm max còn lại các điểm khác trong khối sẽ được sấp xỉ từ điểm đã có giá trị. Thông tin mất mát nhưng trong trường hợp này chúng ta có thể chấp nhận được.

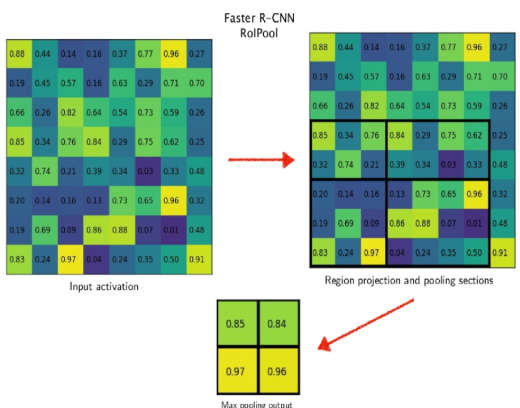


Hình 14: Mô tả cách hoạt động của khối deconvolution và unpooling

Qua việc deconvolution và unpooling, chúng ta có thể xây dựng một phân vùng dự đoán trên ảnh gốc cho tất cả các lớp đối tượng. Đây cũng chính là đầu ra cho khối phân vùng đối tượng.

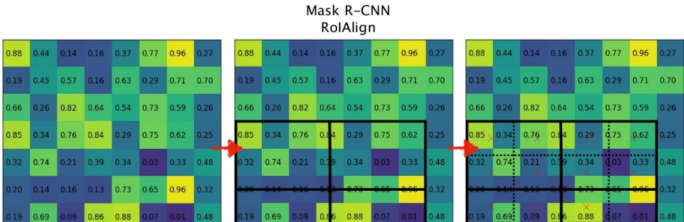
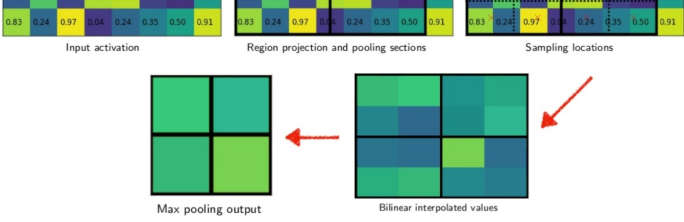
**RoIAlign**

Ngoài ra so với Faster R-CNN Mask RCNN có một cải tiến đó là thay thế việc sử dụng khôi RoI Pooling bằng một khối có tên là RoIAlign. Theo tác giả đây là phần rất quan trọng để cải thiện độ chính xác cho Mask RCNN. RoIPooling hay RoIAlign đều có nhiệm vụ chính là chuẩn hóa kích cỡ của vùng quan tâm đưa vào tầng kế tiếp. Ví dụ sau đây sẽ cho thấy RoIPooling làm cho mô hình mất đi độ chính xác:



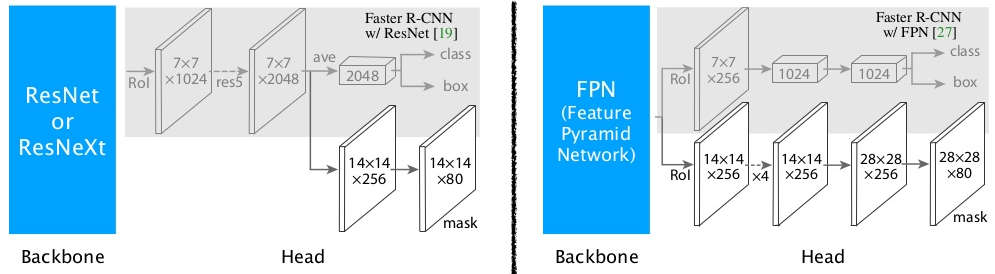
Hình 15: Max Pooling

Khá đơn giản và dễ sử dụng tuy nhiên một phần thông tin cục bộ đã bị mất đi khá nhiều.  
Sau đây là cách RoIAlign cải tiến so với RoIPooling được minh họa trong hình 19. Thay vì việc tính mỗi điểm trên đặc trưng thông qua việc lượng tử hóa từng khổi nhỏ trên ma trận để thu được kích cỡ cố đihj thì RoIAlign thực hiện một phép nội suy phi tuyến để tính ra từ đặc trưng của mỗi vùng như hình dưới. Theo tác giả việc áp dụng RoIAlign cải thiện khá nhiều độ chính xác cho mạng



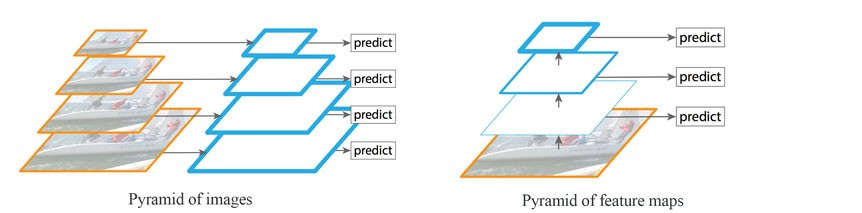
Hình 16: Mô tả phương pháp RoIAlign

**Kiến trúc Mask R-CNN được dùng trong các thí nghiệm**Mask R-CNN được chia làm 2 phần: phần đầu và phần trích xuất đặc trưng. Tác giả đề xuất 2 kiến trúc mạng cho Mask R-CNN dự vào cách chọn mạng cho phần trích chọn đặc trưng (Backbone) tương ứng với phần đầu mạng tương ứng như hình 21 dưới đây:



Hình 17: Kiến trúc backbone Mask R-CNN

Kiến trúc mạng bên trái phần backbone sử dụng mạng ResNet hoặc ResNeXt với số lớp convolution là 50 hoặc 101 lớp được đề xuất trong [20]. Đầu ra đặc trưng được trích dẫn ở tầng thứ 4 của mạng gọi là C4. Bên phải là áp dụng Mask RCNN với phần Backbone là FPN ( Feature Pyramid Network) được nghiên cứu và đề xuất bởi Li và đồng nghiệp tại [21]. FPN là một bộ trích chọn đặc trưng được thiết kế có cấu trúc kim tự tháp cho tốc độ tính toán và độ chính xác cao với đối tượng ở nhiều kích cỡ khác nhau



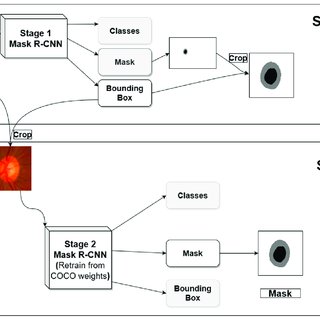
Hình 18: Mạng kim tự tháp

Kiến trúc đầu mạng được mở rộng từ Faster R-CNN với phần thêm vào là mạng tích chập đầy đủ cho dự đoán phân vùng đối tượng. Phần đầu với mạng ResNet-C4 được thêm vào tầng convolution thứ 5 – res5 (mất thời gian tính toán), với FPN backbone kiến trúc đã bao gồm res5 vì vậy mà mạng ít bộ lọc và hiệu quả hơn. Trên đây em đã trình bày kiến trúc chung của Mask R-CNN tham khảo từ bài báo chính của tác giả Kaiming He tại [1]. Trong nội dung đồ án em sử dụng Mask R-CNN với kiến trúc Backbone là FPN như hình 22 bên phải cho việc phát hiện và phân vùng vùng trái cây

**CHƯƠNG 3: TRIỂN KHAI MÔ ĐUN PHÁT HIỆN VÀ PHÂN LOẠI TRÁI CÂY**

**3.1 Mô hình đề xuất nghiên cứu**

Mô hình đề xuất của em là sử dụng Mask R-CNN với kiến trúc backbone là FPN để phát hiện vùng trái cây trên ảnh.Sở đồ khối hệ thống như sau:



Giải thích các khối:  
1- Phát hiện đối tượng trái cây bằng mô hình (4)  
2- Khởi tạo cửa sổ theo vết dựa vào kết quả của Mask R-CNN hoặc từ ảnh trước đó đã phát hiện được đối tượng  
3- Mở rộng cửa sổ theo vết và phát hiện bằng mô hình (5)  
4- Mô hình Mask R-CNN được huấn luyện lại từ bộ trọng số tập dữ liệu COCO với đối tượng trái cây trên ảnh kích cỡ gốc (720x960)  
5- Mô hình Mask R-CNN được huấn luyện lại từ bộ trọng số tập dữ liệu COCO với đối tượng trái cây trên ảnh kích cỡ resize (292x261)  
**Giải thích mô hình**: Với những công cụ được trình bày tại các phần trên em xin đề xuất một mô hình kết hợp giữa Mask R-CNN và thuật toán theo vết đối tượng Mean shift để giải quyết bài toán phát hiện và phân vùng đối tượng trái cây trên ảnh

* Em chuẩn bị 2 mô hình Mask R-CNN đã được huấn luyện với đối tượng trái cây: -Một mô hình được coi là mô hình phụ em huấn luyện bằng cách crop vùng trái cây lại kích cỡ 292x261 trên tập huấn luyện và đưa vào mô hình huấn luyện trộn lẫn (3 Kinect) . Mô hình thứ 2 được coi là mô hình đánh giá chính theo từng thí nghiệm cụ thể. Trong mô hình này, các dữ liệu huấn luyện được thực hiện với độ phân giải gốc.
* Dữ liệu ảnh để kiểm thử hoàn toàn độc lập với tập huận luyện và được đọc vào một cách tuần tự để việc theo vết có thể hoạt động
* Thuật toán mean shift hoạt động được cần khởi tạo cửa số đối tượng. Nếu ảnh ban đầu chưa có thì có thể khởi tạo bằng cách lấy vào đối tượng đầu tiên có điểm số dự đoán được cao nhất và coi là đối tượng chính xác mà ta cần theo vết trong suốt video (hoặc chuỗi ảnh tuần tự). Việc khởi tạo dựa trên điểm số mạng dự đoán ra đôi khi sai vì vậy em áp dụng một ngưỡng để đảm bảo độ tin vậy cho kết quả (trong trường hợp này em sử dụng ngưỡng là 0.96). Trong quá trình chạy cửa sổ sẽ bị xóa đi khi mà đối tượng không còn được tìm thấy trên ảnh và ở frame tiếp theo em sẽ tiến hành khởi tạo lại.
* Do trong quá trình đánh giá kết quả phát hiện của riêng Mask R-CNN, một số trường hợp Mask R-CNN không phát hiện được đối tượng tuy nhiên nếu crop vùng đối tượng với kích cỡ 291x296 và cho vào mạng huấn luyện với ảnh này thì cho kết quả khá chính xác vì vậy em có sử dụng model crop cho việc phát hiện những frame thiếu này.
* Để đánh giá mức độ sai khác của cửa sổ theo vết và các ứng viên dự đoán bởi MaskR-CNN em so sánh 3 tiêu chí là khoảng cách giữa 2 tâm, và độ sai khác màu sắc giữa 2 vùng thông quá độ sai khác giữa 2 giá trị màu H và V trong không gian màu HSV và điểm số dự đoán của Mask R-CNN. Do việc sai khác về mặt đơn vị nên em chuẩn hóa chúng về dạng tỉ lệ [0, 1]:

*D d H S Score* = + Δ + Δ +

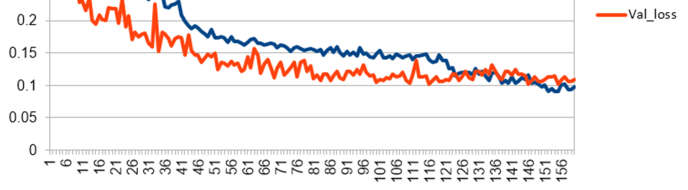
**3.1.1. Quá trình huấn luyện Mask R-CNN**

Để huấn luyện mạng Mask R-CNN em có tham gia vào quá trình chuẩn bị dữ liệu trên tập dữ liệu MICA với khoảng 1000 ảnh.

Em sử dụng mô hình pre-train của tác giả trên tập dữ liệu COCO để khởi tạo các tham số mô hình cho mạng. Các bước huấn luyện như sau:  
*Bước 1: Chuẩn bị dữ liệu ảnh màu và ảnh nhị phân chứa vùng đối tượng trên ảnh màu tương ứng  
Bước 2: Tải bộ trọng số Pre-train và cấu hình các tham số mạng  
Bước 3: Tải dữ liệu lên và tiến hành huấn luyện  
Bước 4: Lưu lại các checkpoint để huấn luyện lại nếu xảy ra lỗi*Từ việc chuẩn bị dữ liệu em có khoảng 1000 ảnh  
Quá trình huấn luyện mạng theo các bước như sau:

* Huấn luyện các lớp đầu với khoảng 40 000 bước lặp với giá trị learning rate 0.001
* Tiếp theo là 80 000 bước lặp trên toàn bộ mạng với giá trị learning rate 0.001
* 40 000 bước lặp tiếp theo huấn luyện trên toàn bộ mạng với learning rate 0.0001

Trong quá trình huấn luyện em có thử trên 2 kích thước với ảnh đầu vào là kích thước gốc 720x960 và kích cỡ 292x261 sau khi resize. Em thực hiện các thí nghiệm trên GPU GTX 1080 Ti với dung lượng Ram 12gb. Với số lượng ảnh lớn sẽ không thể huấn luyện được mạng do giới hạn bởi dung lượng Ram vì vậy em chọn cách thực trao đổi giữa kích cỡ ảnh huấn luyện và số lượng ảnh cho vào huấn luyện. Trong quá trình thực hiện đồ án em có thực hiện các thí nghiệm sau:  
Thí nghiệm 1: Huấn luyện Mask R-CNN với 1000 ảnh (5 loại trái cây khác nhau) tại 3 góc nhìn khác nhau được trộn lẫn. Đầu vào ảnh được resize về kích cỡ 292x261. Dữ liệu test: 1000 ảnh chia đều 3 góc = nhìn.  
Thí nghiệm 2: Huấn luyện với số lượng ảnh như trên nhưng crop lại kích với kích thước 292x261 lan từ tâm vùng trái cây.  
Thí nghiệm 3: Huấn luyện Mask R-CNN trên với khoảng 1000 ảnh (5 loại trái cây) với độ phân giải gốc (292x261). Dữ liệu từ 1 người 5 hành động độc lập tập huấn luyện sẽ được sử dụng cho việc test.  
Thời gian huấn luyện cho mỗi thí nghiệm khoảng 30 tiếng trên GPU với các bước huấn luyện như nhau đã đề cập ở trên.  
**Giá trị hàm mất mất trong quá trình huấn luyện:**Như đã nói trong phần cơ bản mạng neuron quá trình huấn luyện thực chất là quá trình tối ưu hàm mất mát sao cho giá trị nó đi tới cực tiểu. Trong Mask R-CNN giá trị hàm loss được định nghĩa là tổng của 3 giá trị loss khác nhau:  
Lcls: hàm mất mát trong dự đoán lớp  
Lbbox: giá trị mất mát trong dự đoán bao đóng  
Lmask: giá trị mất mát trong dự đoán phân vùng  
Loss = Lcls + Lbbox + Lmask  
Trong quá trình huấn luyện em thấy giá trinh hàm Loss hội tụ sau 160.000 bước lặp. Minh họa dưới đây cho biểu thị hàm loss trong quá trình huấn luyện mạng Mask R-CNN với thí nghiệm huấn luyện ảnh resize về kích cỡ 292x261 sẽ được em trình bày sau đây. Các thí nghiệm khác đồ thị tương tự vì vậy em không report thêm.

**3.1.2. Cài đặt module và huấn luyện Mask R-CNN**

Hình 19: Giá trị hàm mất mát trong quá trình huấn luyện

Ngôn ngữ lập trình dùng cho việc thao tác dữ liệu: Python là một ngôn ngữ lập trình scripting phổ biến và rất thú vị. Hơn nữa nó rất hiệu quả trong việc xử lý dữ liệu, dễ học, dễ làm và dễ cài đặt. Python đang là ngôn ngữ phổ biến nhất hiện tại.  
Công cụ xử lí ảnh và Video em sử dụng là Opencv do việc xử dụng khá đơn giản và Opencv được hỗ trợ trên python trên nền numpy  
- Cấu hình máy tính sử dụng để chạy thử nghiệm: 12 core CPU 2.5GHz, GPU GeForce GTX 1080 Ti, 12GB VRam  
- Môi trường lập trình là python vì vậy cần cài đặt python và các package cần thiết là: tensorflow, keras, opencv2, numpy  
- Để sử dụng Mask R-CNN ta có 2 lựa chọn framework : Tensorflow và Caffe tại: http://caffe.berkeleyvision.org và https://www.tensorflow.org/ .  
Source code của Mask R-CNN được public tại: https://github.com/matterport/Mask\_RCNN  
Hoặc với Caffe tại : https://github.com/facebookresearch/Detectron  
Trong đồ án em sử dụng framework Tensorflow.  
Sau khi cài đặt các gói python cần thiết và framework ta tiến hành clone source từ github bằng lệnh: git clone https://github.com/matterport/Mask\_RCNN  
Tải bộ trọng số mô hình pre-train tại: https://github.com/matterport/Mask\_RCNN/releases  
Chuẩn bị dữ liệu ảnh màu và nhị phân load toàn bộ lên mạng và tiến hành train.  
Quá trình test làm tương tự với train thay đổi mode mô hình từ training -> inference  
Thuật toán Tracking MeanShift ta sử dụng thư viện OpenCV3 để chạy.

**3.2. Kết quả thí nghiệm**

Các thí nghiệm của em có 4 kết quả chính gồm  
- Huấn luyện trộn lẫn các góc nhìn ảnh resize kích cỡ 256x256, kiểm thử trộn lẫn trên  
ảnh gốc và ảnh resize kích cỡ 292x261.  
- Huấn luyện trộn lẫn các góc nhìn với ảnh crop kích cỡ 292x261 và test trên ảnh crop kích cỡ 292x261 và ảnh gốc.  
- Huấn luyện trên tập dữ liệu nhỏ hơn với kích cỡ ảnh gốc với từng góc nhìn và kiểm thử trên tất cả các góc nhìn.  
- Kết hợp thuật toán theo vết đối tượng để tăng độ chính xác đối với mô hìn

Trong các thí nghiệm này, AP là tỷ lệ giữa vùng giao và vùng hợp tạo bởi vùng phát hiện được với groundtruth.  
***Thời gian tính toán:****Đối với các thì nghiệm dưới đây thời gian thử nghiệm trên mỗi ảnh đối với GPU GTX 1080Ti  
Vram 12Gb:***-** Thử nghiệm bộ trọng số được huấn luyện từ Mask R-CNN tốc độ đạt khoảng 5fps.  
- Thử nghiệm kết hợp thuật toán theo vết theo mô hình đề xuất tốc độ khoảng 3fp.

**3.2.1. Kết quả huấn luyện với ảnh resize kích cỡ 256x256**

Như đề cập ở trên phần này em huấn luyện với số lượng ảnh là 100 ảnh hỗn hợp có số lượng gần tương đương nhau.  
Có 2 kết quả là test trên ảnh đã resize về kích cỡ 256x256 và trên ảnh gốc :  
**Kết quả thí nghiệm :**

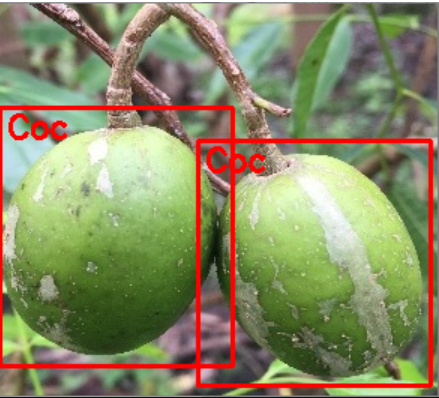
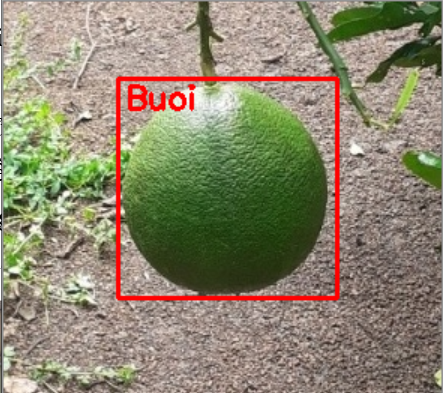
**Table 1: Kết quả thử nghiệm trên ảnh gốc**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Bưởi | Cam | Cóc | Khế | Mít |
| Bưởi | 0 |  |  |  |  |
| Cam |  | 0 |  |  |  |
| Cóc |  |  | 0 |  |  |
| Khế |  |  |  | 0 |  |
| Mít |  |  |  |  | 0 |

**Table 2: Kết quả thử nghiệm với ảnh resize (292x261)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Bưởi | Cam | Cóc | Khế | Mít |
| Bưởi | 18 | 0 | 0 | 0 | 0 |
| Cam | 0 | 16 | 0 | 0 | 0 |
| Cóc | 0 | 0 | 19 | 0 | 0 |
| Khế | 0 | 0 | 0 | 10 | 0 |
| Mít | 0 | 0 | 0 | 0 | 20 |

**Nhận xét:** Từ kết quả trên ta thấy được độ chính xác trên trên góc nhìn thứ 1 là rất thấp , góc nhìn thứ 3 và thứ 5 có độ chính xác tại giá trị AP=0.5 là chấp nhận được. Ta thu được kết quả như vậy là do tại góc nhìn thứ nhất trái cây bị lẫn nhiều bởi màu nền, thêm nữa cử chỉ khá nhanh làm cho hình ảnh bị hiệu ứng mờ (motion blur) làm hệ thống không nhận dạng được.  
Một nguyên nhân nữa là do ta đã resize ảnh huấn luyện đầu vào về kích thước 292x261 làm cho thông tin bị mất mát. Thí nghiệm này là huấn luyện trên tất cả các góc nhìn, sau 160 bước giá trị hàm loss vẫn khá cao cũng là nguyên nhân gây nên kết quả kém.  
Với Mask R-CNN kết quả thử nghiệm trên cho thấy việc resize ảnh trong lúc thử nghiệm không gây ra nhiều sai khác do kiến trúc có sử dụng mạng FPN. Mặc dù vậy kết quả khi thử nghiệm với kích thước ảnh đã resize về kích cỡ 292x261 bằng kích cỡ ảnh huấn luyện vẫn cho kết quả cao hơn khoảng 4%



Hình 20: Kết quả nhận dạng ảnh đã được resize 292x261

**3.2.2. Kết quả huấn luyện với ảnh crop kích cỡ 292x261**

Em tiến hành với tập dữ liệu như trên và kết quả em đánh giá trên ảnh crop kích cớ 292x261và ảnh gốc. Trong thí nghiệm này em huấn luyện ảnh đã crop kích cỡ 292x261với tâm là trọng tâm của vùng trái cây đã được phân vùng thủ công. Trong trường hợp này em chỉ báo cáo kết quả thử nghiệm với tập ảnh thuộc cả 3 góc nhìn với 2 trường hợp là ảnh thử nghiệm cũng được crop kích thước 292x261và ảnh thử nghiệm giữ nguyên kích thước gốc.

Dễ nhận thấy ảnh kích cỡ ban đầu 480x640 so với ảnh đã crop thông tin bị mất đi nhiều so với ảnh gốc vì vậy mà mô hình cho kết quả rất tệ với ảnh kích cỡ gốc. Tuy nhiên với ảnh được crop lại kích cỡ 292x261chứa vùng trái cây thì hầu như là không bị nhận nhầm. Độ chính xác có thể đạt tới 99%

**Nhận xét**: Từ bảng kết quả trên ta thấy việc sử dụng Mask R-CNN cho kết quả tốt nhất khi thực hiện huận luyện trên góc nhìn nào và thử nghiệm trên góc nhìn đó. Trên góc nhìn thứ nhất khi sử dụng huấn luyện một tập dữ liệu chung cho cả 3 góc nhìn cho kết quả khá kém tuy nhiên khi ta kiểm nghiệm với tập dữ liệu chỉ thuộc góc nhìn này thì cho kết quả cao hơn khá nhiều tuy nhiên vẫn còn khá hạn chế so với 2 góc nhìn còn lại.

.

**CHƯƠNG 4: KẾT LUẬN**

**4.1 Kết quả đạt được**

Trong quá trình thực hiện đề tài, em đã tìm hiểu và làm chủ được một số kỹ thuật trong lĩnh vực thị giác máy tính và học máy, cụ thể là mạng nơ ron tích chập. Em đã hiểu về quy trình triển khai của một mô hình mạng nơ ron trên máy chủ. Bên cạnh đó, em đã tìm hiểu thêm về một lĩnh vực mới là Thị Giác Máy Tính cụ thể là lý thuyết cơ bản và một thuật toán để tính toán luồng quang học. Ngoài ra, em đã học được thêm một ngôn ngữ lập trình mới là python và sử dụng để thao tác với dữ liệu và sử dụng các thuật toán trong thư viện opencv, numpy. Kết quả chạy tương đối tốt so với yêu cầu phát hiện vùng đối tượng trái cây trên ảnh

**4.2 Nhược điểm và hướng phát triển**

**4.2.1 Nhược điểm**

* Thời gian chạy chương trình chậm chưa thể đạt đến thời gian thực. Với mô hình đề xuất em chưa nghiên cứu tối ưu thời gian chạy.
* Kết quả đạt được trên góc nhìn thứ nhất còn chưa tốt.
* Một số trường hợp khi nhận dạng bằng hình ảnh gốc không cho ra kết quả

**4.2.2 Hướng phát triển**

Như đã nói trong chương 1 bài toán phát hiện và phân vùng đối tượng trái cây đóng vai trò khá quan trọng vấn đề thị giác máy tính do vậy mà hướng phát triển cho bài toán khác cũng có rất nhiều:

* Cải thiện độ chĩnh xác cho phát hiện và phân vùng đối tượng trên các góc nhìn khó như góc nhìn thứ 1, đánh giá thủ nghiệm trên nhiều các tập dữ liệu khác nhau
* Cải thiện tốc độ tính toán cho thuật toán nhằm ứng dụng cho các bài toán đòi hỏi thờigian thực
* Từ kết quả phân vùng đối tượng áp dụng cho các pha tiếp theo như nhận diện chuyển động, cử chỉ
* Phát triển ứng dụng nhận dạng khác trên nền tảng Mobile

**TÀI LIỆU THAM KHẢO**

[1] He K., Gkioxari G., Dollár P., et al. (2017). Mask R-CNN. *ArXiv170306870Cs*.  
[2] Real-Time Hand Gesture Detection and Recognition Using Bag-of-Features and Support Vector Machine Techniques - IEEE Journals & Magazine.  
<https://ieeexplore.ieee.org/document/5983442>, accessed: 05/20/2018.  
[3] Oualla M., Sadiq A., and Mbarki S. (2014). A survey of Haar-Like feature representation. *2014 International Conference on Multimedia Computing and Systems (ICMCS)*, 1101– 1106, 1101–1106.  
[4] Chouvatut V., Yotsombat C., Sriwichai R., et al. (2015). Multi-view hand detection applying viola-jones framework using SAMME AdaBoost. *2015 7th International Conference on Knowledge and Smart Technology (KST)*, 30–35, 30–35.  
[5] <https://medium.com/analytics-vidhya/training-your-own-data-set-using-mask-r-cnn-for-detecting-multiple-classes-3960ada85079>

[6] https://www.robots.ox.ac.uk/~vgg/software/via/via-1.0.6.html