Code

Deep learning: Nhận diện vật thể qua ảnh chụp

Lê Ngọc Khả Nhi

02 Tháng 7 2018

1 Mở đầu

Là một người ngoại đạo với Machine learning, Nhi chỉ mới bắt đầu (tự) học về nó khoảng 2 năm nay và hằng ngày Nhi vẫn đang cố gắng theo đuổi mục tiêu này với vốn kiến thức ít ỏi của mình (Nhi xuất phát từ phái Thống kê cổ điển). Như một hệ quả tất yếu, sau khi thực hành trên những bài toán đơn giản, người học sẽ dần dần chạm đến những vấn đề thú vị hơn (và phức tạp hơn), thí dụ nhận diện vật thể / trích xuất giá trị định lượng từ ảnh chụp. Bài toán này sử dụng dữ liệu đầu vào là ảnh chụp và cần công cụ là Convolutional neural network (mạng neuron tích chập, CNN).

Gần đây, với những R package như magick (xử lý ảnh chụp), keras (cho phép kết nối với TensorFlow backend từ Python, thậm chí không cần GPU) và lime (giải thích cơ chế mô hình, với ứng dụng cho dữ liệu văn bản/matrix và ảnh chụp), việc thực hiện một mô hình CNN cho các nghiên cứu đơn giản, quy mô vừa và nhỏ là hoàn toàn khả thi.

Tiến sĩ Shirin Glander, một đàn chị trong giới data science tại Đức vừa công bố một thí dụ cho thấy tính khả thi và đơn giản đáng ngạc nhiên khi thực hành Computervision model với keras và lime. Tuy bài viết của chị ấy rất đáng quý,vì có hiệu quả tích cực, khích lệ tinh thần của những người nghiệp dư - tự học trong đó có Nhi; nhưng nội dung của bài mang tích chất biểu diễn, nhưng có thể gây khó hiểu cho nhiều bạn , do đó Nhi quyết định biên tập lại thí dụ minh họa và code của Shirin kèm theo diễn giải bằng tiếng Việt cho các bạn bác sĩ tại VN.

2 Bước 1: Mô tả bài toán và dữ liệu

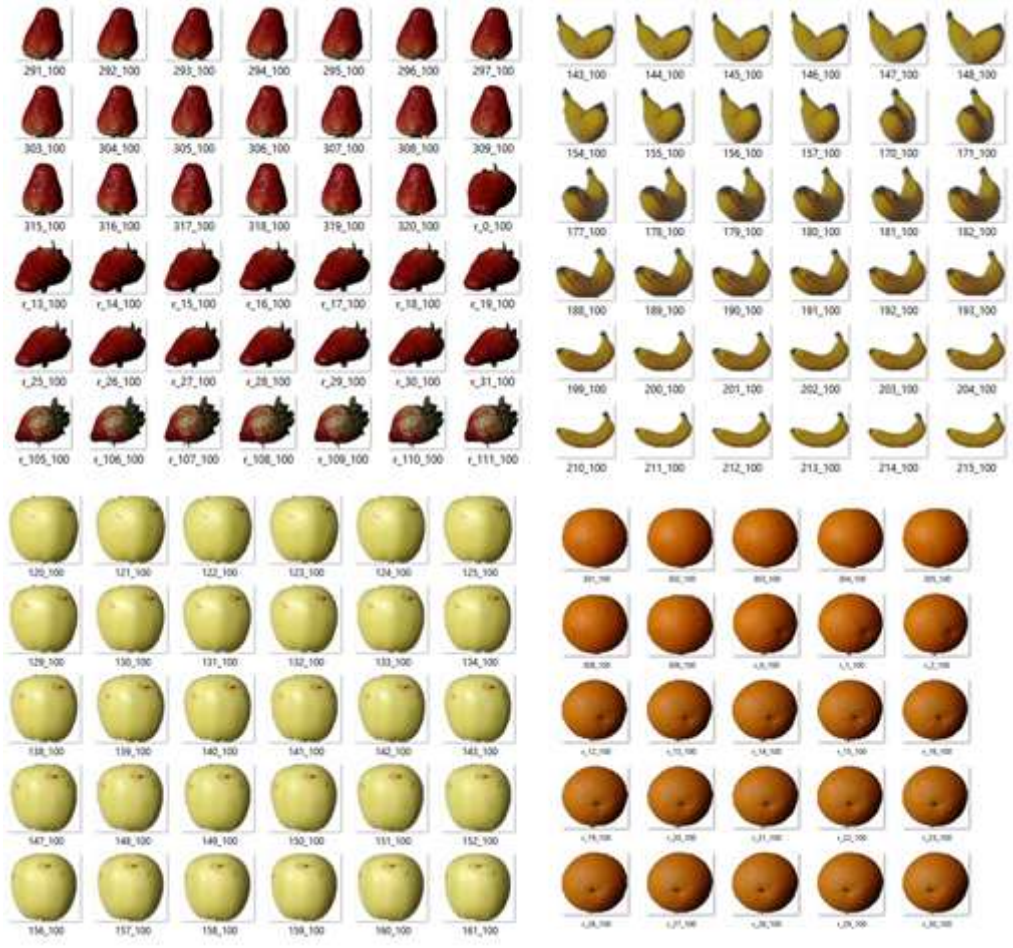
Bài toán minh họa mà chúng ta sẽ giải quyết có Mục tiêu là xây dựng một mô hình mạng neuron cho phép nhận dạng 15 loại quả từ một tấm ảnh chụp. Danh sách 15 loại quả này bao gồm: Kiwi, chuối (Banana),Mơ (Apricot),bơ (Avocado),dừa (Cocos), quít (Mandarine), cam (Orange), Chanh xanh (Limes), chanh vàng (Lemon), đào (Peach), mận (Plum), Raspberry, dâu (Strawberry), Dứa (Pineapple) và lựu (Pomegranate).

Bộ dữ liệu dùng để huấn luyện mô hình được lấy về từ kaggle:

https://www.kaggle.com/moltean/fruits/data (https://www.kaggle.com/moltean/fruits/data)

Đầu tiên, bạn hãy tải file ZIP của dữ liệu về máy tính của mình, nó có kích thước khoảng 390 Mb. Sau khi giải nén, bạn sẽ có một thư mục tên là fruits-360, với cấu trúc gồm 4 thư mục con bên trong, trong đó ta quan tâm đến 2 thư mục là Test (chứa 12642 ảnh JPG, trong 74 thư mục riêng biệt cho từng loại quả) và Training (chứa 37101 file ảnh JPG, trong 74 thư mục riêng).

Khi thăm dò một vài thư mục bất kì, thí dụ: Straberry, Banana,Golden Apple và Orange, bạn sẽ thấy hàng trăm ảnh chụp bên trong. Mỗi ảnh có kích thước đồng nhất là 100x100 pixels, trình bày hình ảnh của đối tượng (thí dụ quả dâu) nằm tại trung tâm tấm ảnh, trên 1 nền trắng và dưới nhiều góc nhìn khác nhau với đầy đủ những chi tiết đặc thù bao gồm thân, cuống, lá và màu sắc.



Một vài nhận xét của Nhi về bài toán và dữ liệu:

1. Đây là một bài toán rất đơn giản, vì để giải quyết nó chúng ta không cần kiến thức chuyên ngành nào cả, khác với những vấn đề phức tạp như ảnh CTscan, ảnh giải phẫu bệnh, hay nhân trắc học… Và nhất là, ảnh chụp đã được xử lý đồng nhất về màu sắc, kích thước, dán nhãn rõ ràng, đối tượng cần nhận diện nằm đúng trung tâm và trên 1 nền trắng hoàn hảo. Số lượng ảnh lớn nhưng kích thước nhỏ (100x100) là một lợi thế rất lớn cho phép giải quyết trên một máy tính không có GPU (như của Nhi) với một cấu trúc mạng neuron đơn giản.

2. Tuy đơn giản, bài toán hết sức thú vị: nó giúp ta hình dung khái quát về cơ chế nhận diện một đối tượng: các loại quả có thể được phân biệt nhờ: Hình dạng của chúng,như cam và Chuối), màu sắc (Dâu và Chanh), chi tiết phụ (lông của quả kiwi, hạt trên quả dâu, lá của quả thơm).Nguy cơ nhầm lẫn hoàn toàn có thể xảy ra (Cam vs quít vs chanh). Tính thú vị còn ở chỗ Hoa quả gợi cho con người cảm giác dễ chịu, khi liên tưởng đến vị ngon và màu sắc của chúng (không phải ngẫu nhiên mà những bài thực hành Deep learning cho ảnh chụp đều liên quan đến những đối tượng vui nhộn, dễ chịu như: nhận diện Mèo, Chó, Món ăn, Hoa)

3. Trong bài toán này, dữ liệu không phải là một bảng tính với con số và các biến như ta thường thấy, nhưng chúng ta làm việc trực tiếp trên ảnh chụp. Như vậy một trong các công đoạn thực hiện mô hình là chuyển từ ảnh chụp thành dữ liệu phù hợp cho công cụ ta đang sử dụng (keras và TensorFlow).

3 Bước 2: Chuẩn bị nguồn cung cấp dữ liệu

Trước hết, bạn cần install Python và package keras cho máy tính, bạn có thể làm theo hướng dẫn tại: https://keras.rstudio.com/index.html (https://keras.rstudio.com/index.html)

Code

Bây giờ, ta bắt đầu sơ chế dữ liệu: Việc đầu tiên cần làm đó là rút gọn danh sách các loại quả cần nhận dạng từ 74 xuống còn 15 mà thôi:

Code

```
##  [1] "Kiwi"        "Banana"      "Apricot"     "Avocado"     "Cocos"
##  [6] "Mandarine"   "Orange"      "Limes"       "Lemon"       "Peach"
## [11] "Plum"        "Raspberry"   "Strawberry"  "Pineapple"   "Pomegranate"
```

(Trong danh sách này, vấn đề khó nhất là: Phân biệt Cam và Quýt, Kiwi/Dừa/quả bơ)

Một object tên là output_n được chuẩn bị, để chỉ số nhãn kết quả cần phân loại:

Code

```
## [1] 15
```

Tiếp theo, ta dự tính sẽ thu nhỏ kích thước ảnh trong quá trình huấn luyện từ 100x100 còn 20x20,ta chuẩn bị 2 tham số: img_width và img_height của tùy chỉnh target_size.

Một tùy chỉnh khác cũng được thiết lập, đó là số kênh màu sắc (khi sử dụng ảnh màu, có 3 kênh tương ứng R,G,B, còn ảnh đen trắng chỉ có 1 kênh, mỗi pixel trong từng kênh có giá trị từ 0:255)

(tác giả Shirin đã rất thông minh khi làm việc một cách thứ bậc - khai báo các tùy chỉnh bên ngoài, từ đó không cần lặp lại chúng khi áp dụng các hàm keras và không sợ nhầm lẫn)

Code

```
##            [,1]
## img_width    20
## img_height   20
```

Code

```
##             [,1] [,2]
## target_size   20   20
```

Code

```
##          [,1]
## channels    3
```

Tiếp theo, Nhi khai báo đường dẫn đến 2 thư mục dùng để huấn luyện và kiểm định mô hình:

Code

```
## [1] "/Deeplearning/fruits-360/Training/"
```

Code

```
## [1] "/Deeplearning/fruits-360/Test/"
```

Hàm image_data_generator được dùng để chuyển từ file ảnh JPEG thành dữ liệu tensor cho keras, hàm này còn cho phép sinh mẫu ngẫu nhiên bằng cách biến dạng/gây nhiễu ngẫu nhiên ảnh gốc, tính năng này chưa được dùng đến trong thí dụ này. Tùy chỉnh rescale cho phép chuyển giá trị pixel từ 0:255 thành thang đo 0:1, để thích hợp cho mạng neuron.

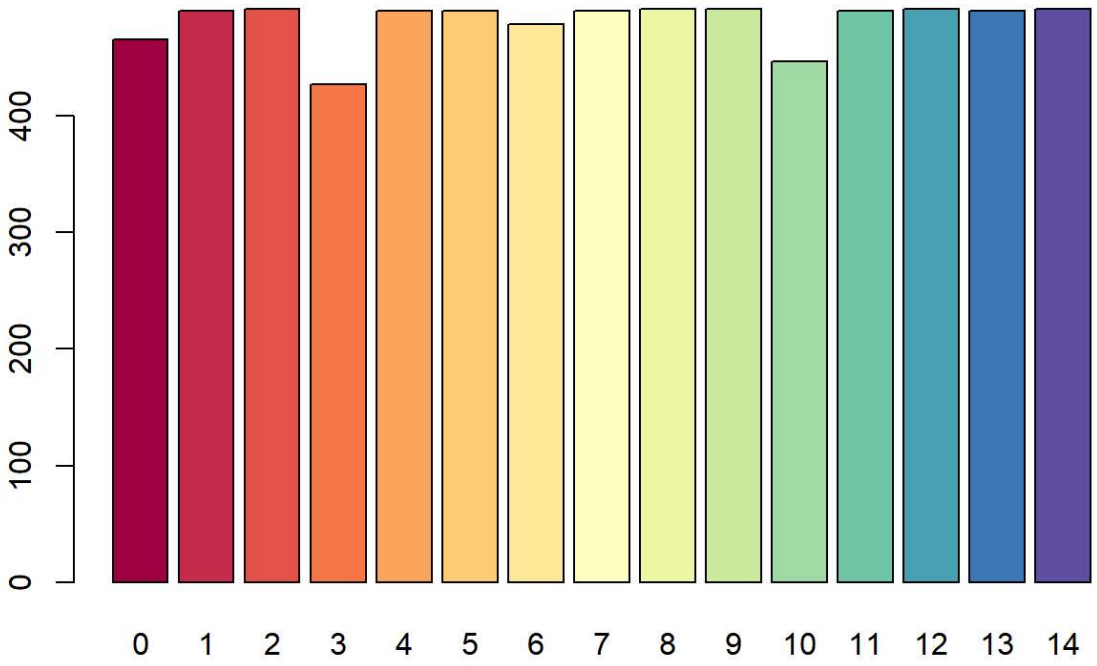
Code

Bây giờ ta có thể lấy mẫu một số ảnh chụp ngẫu nhiên từ 1 trong 2 thư mục Training /Test (qua đường dẫn đã khai báo), sau đó chuyển thành dữ liệu tensor (arrays) , với kích thước ảnh = target size đã được khai báo, class_mode là tính chất của nhãn kết quả = categorical (bài toán phân loại), và danh sách nhãn = fruit_list

Code

Cấu trúc array dữ liệu huấn luyện:

Code

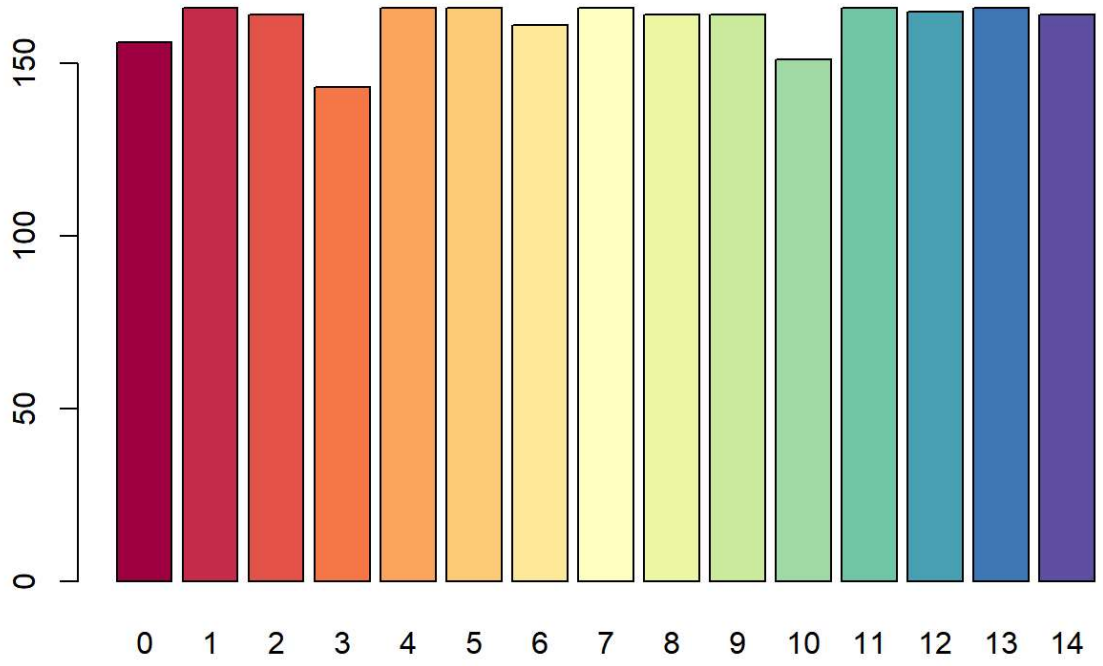


Code

```
##
##   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14
## 466 490 492 427 490 490 479 490 492 492 447 490 492 490 492
```

Cấu trúc dữ liệu kiểm định:

Code



Code

```
##
##   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14
## 156 166 164 143 166 166 161 166 164 164 151 166 165 166 164
```

Nhi sao lưu lại dữ liệu huấn luyện trong 1 file Rdata

Code

Bước tiếp theo, hai tùy chỉnh khác là train_samples và valid_samples hay kích thước của tập huấn luyện/kiểm định được khai báo trước:

Code

```
##               [,1]
## train_samples 7219
## valid_samples 2428
```

Tương tự, ta ấn định trước giá trị 2 tùy chỉnh batch_size và epochs (số lượt huấn luyện)

Code

```
##            [,1]
## batch_size   32
## epochs       10
```

4 Bước 3: Xây dựng cấu trúc mạng neuron

Đây là công đoạn chính của quy trình, ta sẽ xây dựng mạng neuron với cấu trúc gồm các lớp như sau:

1. Đầu tiên là 1 lớp Convolutional tiếp nhận dữ liệu tensor với các thông số: img_width, img_height, channels đã được khai báo ở trên, hàm kích hoạt là RELU;
2. Lớp ẩn tiếp theo là 1 lớp conv khác
3. Tiếp theo là 1 lớp pooling
4. Tiếp theo, 1 lớp flatten nối với 1 lớp dense (100 neuron) và 1 lớp dropout, công dụng của bộ ba này là chuyển giá trị đầu ra của lớp pooling thành vector feature, sau đó tinh chỉnh
5. Lớp cuối cùng: xuất kết quả , với số neuron đúng bằng số nhãn cần phân loại, với hàm kích hoạt là softmax

Mô hình được compile với hàm loss = categorical_crossentropy và tùy chỉnh optimizer = rmsprop, tiêu chí tối ưu hóa là accuracy.

Code

Code

```
## Model
## _________________________________________________________________
## Layer (type)                 Output Shape              Param #
## =================================================================
## conv2d_1 (Conv2D)            (None, 20, 20, 32)        896
## _________________________________________________________________
## activation_1 (Activation)    (None, 20, 20, 32)        0
## _________________________________________________________________
## conv2d_2 (Conv2D)            (None, 20, 20, 16)        4624
## _________________________________________________________________
## leaky_re_lu_1 (LeakyReLU)    (None, 20, 20, 16)        0
## _________________________________________________________________
## batch_normalization_1 (BatchNorm (None, 20, 20, 16)    64
## _________________________________________________________________
## max_pooling2d_1 (MaxPooling2D)   (None, 10, 10, 16)    0
## _________________________________________________________________
## dropout_1 (Dropout)          (None, 10, 10, 16)        0
## _________________________________________________________________
## flatten_1 (Flatten)          (None, 1600)              0
## _________________________________________________________________
## dense_1 (Dense)              (None, 100)               160100
## _________________________________________________________________
## activation_2 (Activation)    (None, 100)               0
## _________________________________________________________________
## dropout_2 (Dropout)          (None, 100)               0
## _________________________________________________________________
## dense_2 (Dense)              (None, 15)                1515
## _________________________________________________________________
## activation_3 (Activation)    (None, 15)                0
## =================================================================
## Total params: 167,199
## Trainable params: 167,167
## Non-trainable params: 32
## _________________________________________________________________
```

5 Bước 4 : Huấn luyện mô hình

Sau khi có cấu trúc mạng CNN, ta bắt đầu thực hiện quy trình huấn luyện cho nó. Quy trình này bắt đầu bằng việc đưa tensor data (train_image_array_gen, được chuẩn bị ở bước 2) vào lớp tiếp nhận dữ liệu, mô hình được huấn luyện 10 lượt để tối ưu hóa accuracy.Diễn tiến được ghi lại trong file fruits_checkpoints.h5 với đường dẫn tùy chọn, và 1 file logs trong thư mục keras (nếu bạn muốn khai thác bằng tensorboard)

Code

Do tính chất đơn giản, mô hình được huấn luyện khá nhanh ngay cả khi dùng CPU. Biểu đồ cho thấy sau 10 lượt huấn luyện/kiểm định, giá trị loss giảm dần và accuracy tăng dần,tức là hiệu năng của mô hình đạt đến trạng thái tối ưu

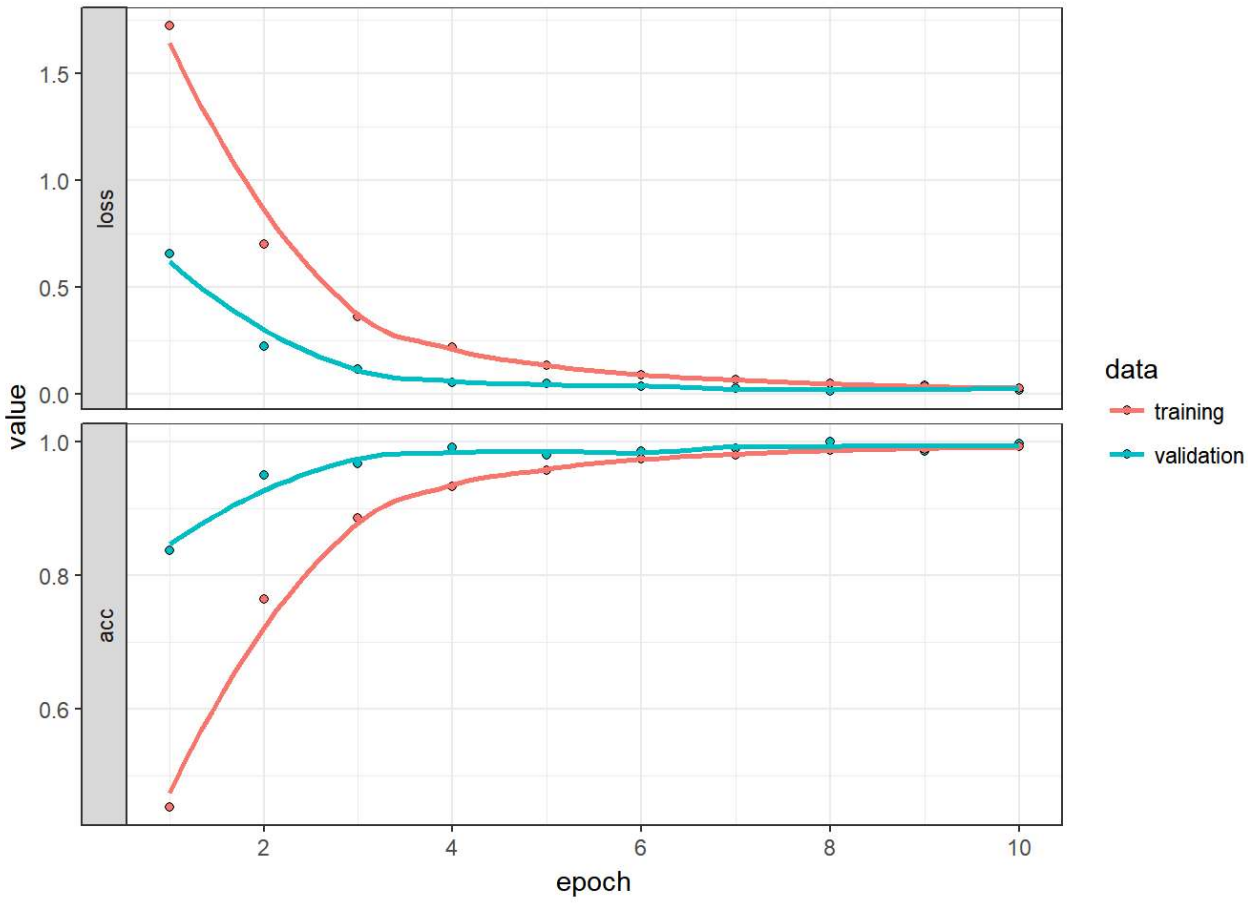
Code

```
## -- Attaching packages ---------------------------------------------------
- tidyverse 1.2.1 --
```

```
## v ggplot2 2.2.1     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.5
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

```
## -- Conflicts ------------------------------------------------------------ tidy
verse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Code



Code

```
## Trained on NULL samples (batch_size=NULL, epochs=10)
## Final epoch (plot to see history):
## val_loss: 0.02168
##  val_acc: 0.9962
##     loss: 0.03006
##      acc: 0.9923
```

6 Bước 5: Sử dụng mô hình và giải thích cơ chế

Đến lúc này, chúng ta đã đi được 80% hành trình, và có trong tay một mô hình CNN có khả năng nhận diện/phân biệt được 15 loại quả khác nhau dựa vào bất kì ảnh chụp mới nào, với điều kiện ảnh phải có nền trắng và chỉ chứa 1 loại quả duy nhất.

Trước hết, Nhi chuẩn bị đường dẫn đến 1 thư mục chứa dữ liệu (ảnh chụp) cần nhận dạng

Code

Quy trình cần 2 package là lime và magick

Code

```
##
## Attaching package: 'lime'
```

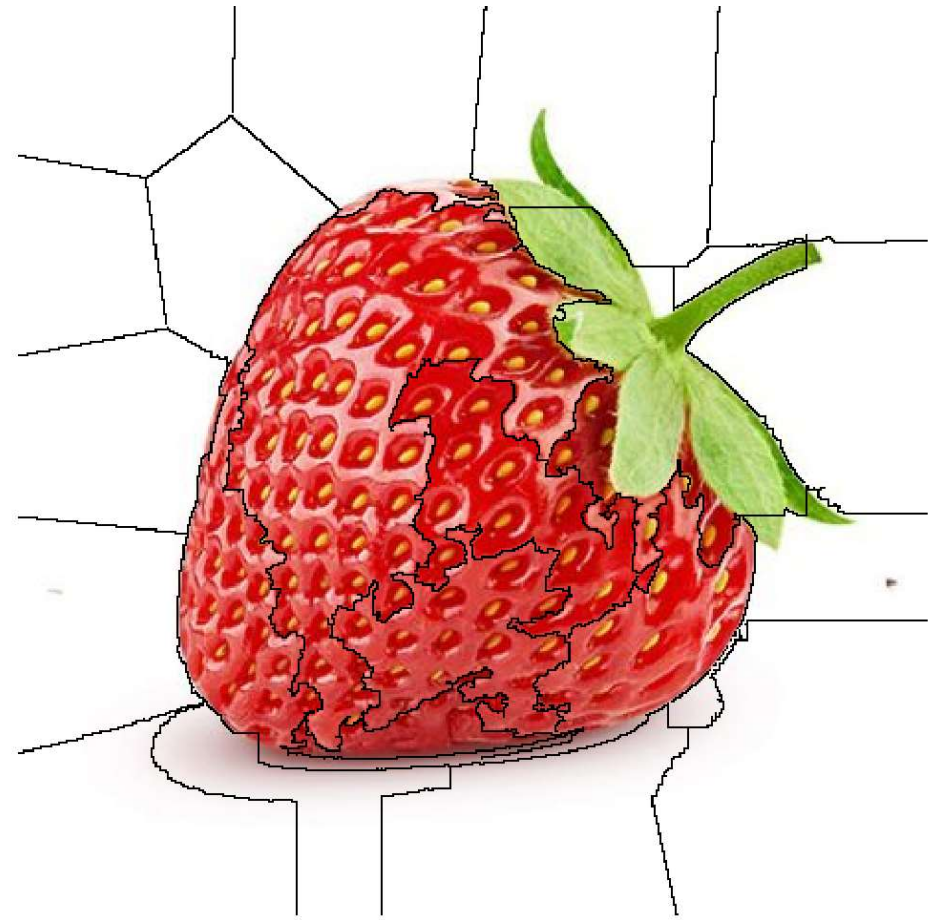
```
## The following object is masked from 'package:dplyr':
##
##     explain
```

Code

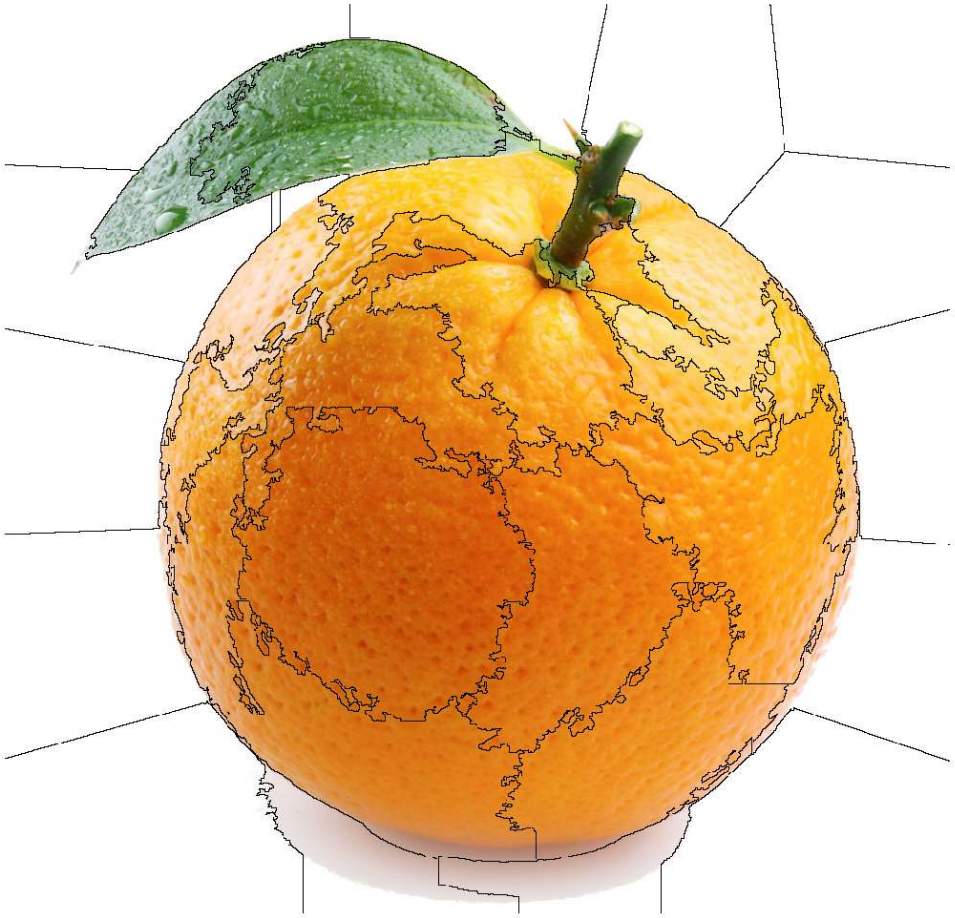
```
## Linking to ImageMagick 6.9.9.14
## Enabled features: cairo, freetype, fftw, ghostscript, lcms, pango, rsvg, webp
## Disabled features: fontconfig, x11
```

Mục tiêu giả định là nhận dạng 1 quả dâu và và 1 quả cam, hình ảnh được lấy ngẫu nhiên từ internet

Code



Code



Ta sử dụng thủ thuật tương tự như bước 2, để thu nhỏ kích thước ảnh còn 20x20, và chuyển giá trị pixel về thang đo 0:1, cho toàn bộ tập kiểm định (bao gồm 2 ảnh mới thêm vào)

Sau đó dùng hàm predict_generator cho mô hình ở trên và dữ liệu cần dán nhãn là test_generator, ta có kết quả và lưu nó lại dưới dạng dataframe

Code

Sau khi kết hợp với danh sách nhãn kết quả fruits_classes_indices, ta có một matrix xác suất cho mỗi nhãn, và tổng kết lại kết quả

Code

```
## 1 :Raspberry
##        12
## 2 :Banana
##      2
## 3 :Pomegranate
##          15
## 4 :Limes
##     8
## 5 :Peach
##    10
## 6 :Pomegranate
##          15
## 7 :Plum
##   11
## 8 :Peach
##    10
## 9 :Lemon
##     9
## 10 :Orange
##      7
## 11 :Lemon
##     9
## 12 :Banana
##      2
## 13 :Limes
##     8
## 14 :Avocado
##       4
## 15 :Avocado
##       4
## 16 :Raspberry
##        12
## 17 :Lemon
##     9
## 18 :Limes
##     8
## 19 :Raspberry
##        12
## 20 :Lemon
##     9
## 21 :Pineapple
##        14
## 22 :Avocado
##       4
## 23 :Peach
##    10
## 24 :Lemon
##     9
## 25 :Avocado
##       4
## 26 :Banana
##      2
```

```
## 27 :Kiwi
##    1
## 28 :Avocado
##       4
## 29 :Avocado
##       4
## 30 :Raspberry
##        12
## 31 :Lemon
##     9
## 32 :Lemon
##     9
```

Để sử dụng được package lime nhằm giải thích cơ chế hoạt động của mô hình, Shirin Glander viết 1 hàm image_prep, nội dung của nó nhằm chuẩn bị dữ liệu - như thu nhỏ kích thước ảnh còn 20x20 pixels, chuyển thang đo của mỗi pixel về 0:1 và chuyển ảnh chụp thành arrays.

Tiếp theo, danh sách nhãn cần phân loại với định dạng long được tạo ra : fruits_classes_indices_l

sau đó hàm lime được dùng với tùy chỉnh gồm đường dẫn đến 2 tấm ảnh mới (Dâu và Cam) cần giải thích , tên mô hình và danh sách nhãn cần phân loại được đặt trong hàm as_classifier nhằm xác định đây là 1 mô hình phân loại, cuối cùng là hàm image_prep

Nội dung kết quả sẽ được lưu lại trong 1 object explainer. Đây là bộ máy dùng để giải thích cơ chế mô hình trên 2 ảnh bằng hàm expain của lime, với tùy chỉnh dữ liệu x=c(đường dẫn ảnh 1 (dâu), đường dẫn ảnh 2 (cam)), n_label = 1 để tập trung vào 1 nhãn duy nhất,số features=20, superpixels (mảnh ghép) = 35, nền màu trắng

Code

```
##            0            1            2            3            4
##       "Kiwi"     "Banana"    "Apricot"    "Avocado"      "Cocos"
##            5            6            7            8            9
##  "Mandarine"     "Orange"      "Limes"      "Lemon"      "Peach"
##           10           11           12           13           14
##       "Plum"  "Raspberry" "Strawberry"  "Pineapple" "Pomegranate"
```

Code

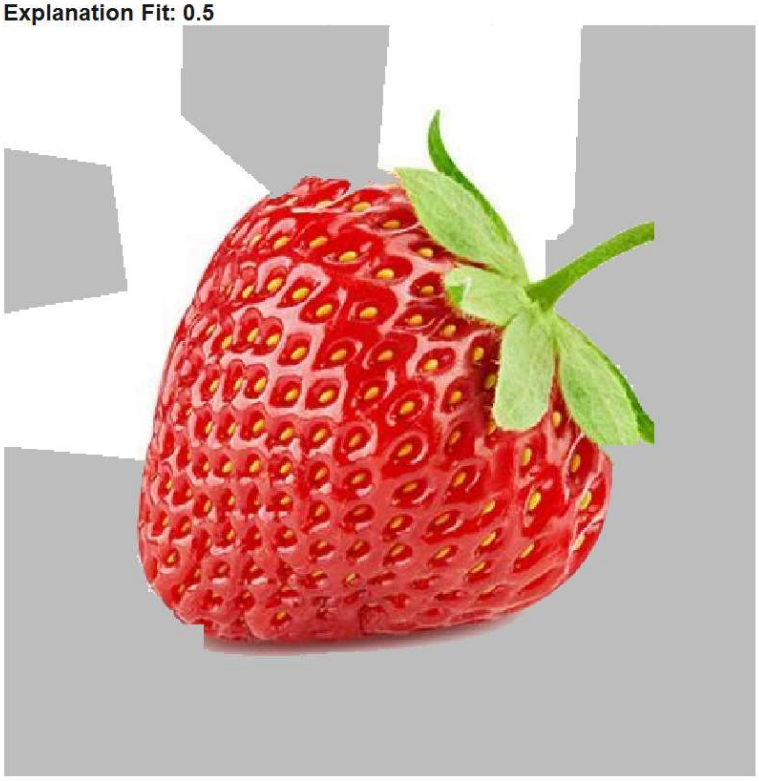
Ta tách riêng 2 đối tượng để giải thích trực quan:

Code

Kết quả giải thích trực quan có thể trình bày dưới 2 hình thức: Phân lập những vùng ủng hộ cho nhãn kết quả, trên 1 nền xám:

Code

Label: Strawberry
Probability: 0.97
Explanation Fit: 0.5



Code

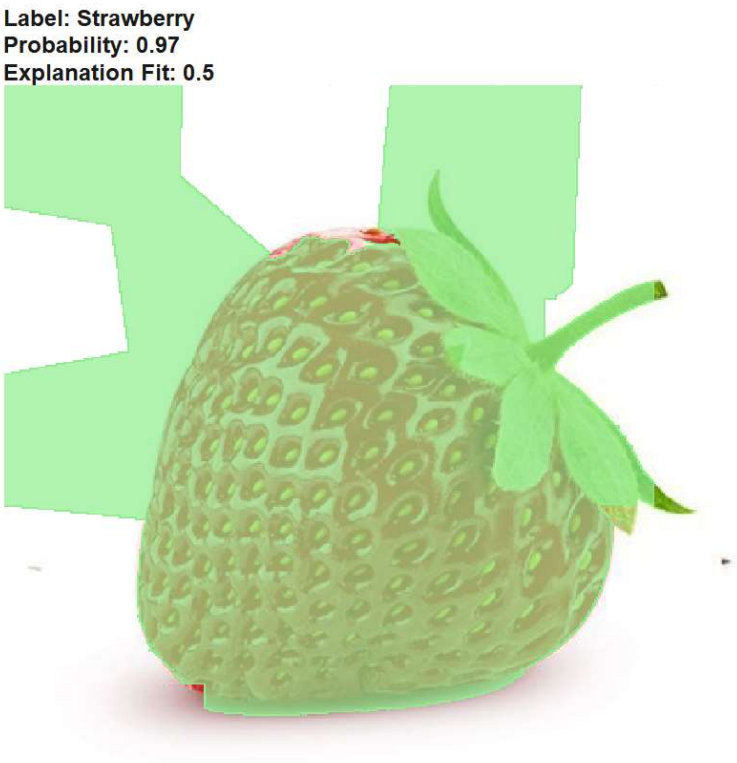
Label: Banana
Probability: 0.89
Explanation Fit: 0.092

Theo kết quả này, ta thấy quả dâu được nhận diện chính xác với prob =1, và toàn bộ quả dâu trong ảnh cũng được phân lập như chứng cứ ủng hộ cho kết quả đó.

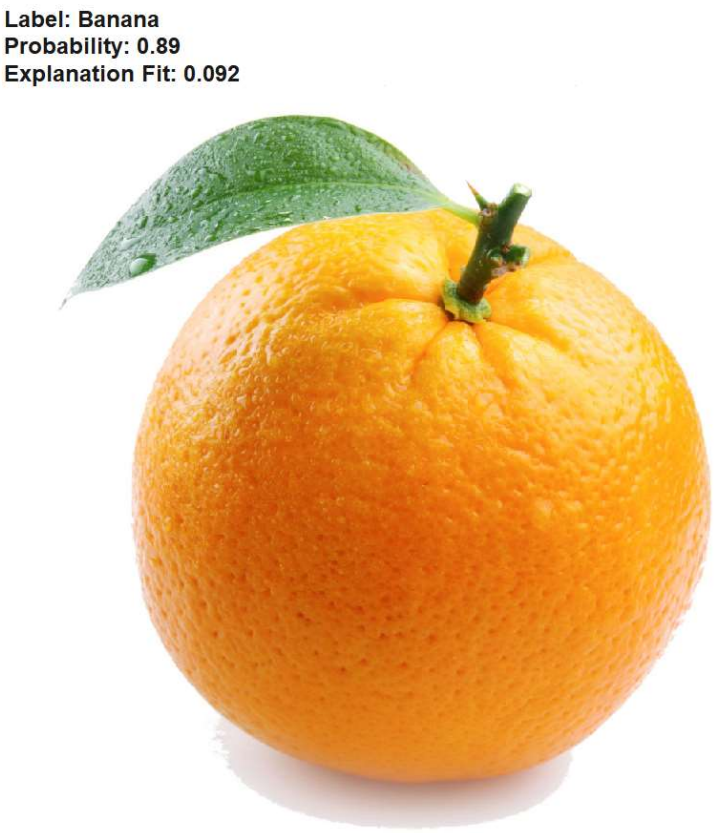
Ngược lại, quả cam bị nhận diện nhầm lẫn thành quả chanh (Lemon) với prob=0.93, và những vùng dữ liệu dẫn đến kết quả (sai lầm) này bao gồm cuống, lá và hình tròn với da màu cam

Hình thức thứ hai là chồng lắp những mảng/vùng dữ liệu quan trọng trên nền một tấm ảnh gốc

Code



Code



7 Kết luận:

Dù đây chỉ là một thí dụ rất đơn giản, chúng ta có thể hy vọng về khả năng thực hiện được những điều phức tạp hơn, vì hành trình dù dài đến đâu cũng bắt đầu bằng 1 bước chân phải không các bạn ?