

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
ĐẠI HỌC CÔNG NGHỆ THÔNG TIN



## **BÁO CÁO ĐỒ ÁN**

Môn học: Máy học

Mã lớp: CS114.L11.KHCL

Học kỳ I (2020-2021)

### **CHỦ ĐỀ:**

## **BANANA DETECTION**

**GVHD:** Phạm Nguyễn Trường An – Lê

Đình Duy

#### **Các thành viên**

Trần Nguyễn Thanh Nguyên - 18521168

Trần Hữu Anh Khoa - 18520937

Lê Huy Đạt - 18520569

# MỤC LỤC

|   |           |
|---|-----------|
| <b>MỤC LỤC</b>                                | <b>2</b>  |
| <b>PHẦN I: ĐẶT VẤN ĐỀ</b>                     | <b>3</b>  |
| <b>PHẦN II:NỘI DUNG</b>                       | <b>3</b>  |
| A. TỔNG QUAN VỀ DỮ LIỆU                       | 3         |
| 1. Giới thiệu bài toán                        | 3         |
| B. ĐỌC-HIỂU DỮ LIỆU                           | 4         |
| 1. Thông tin dữ liệu.                         | 4         |
| 2. Cách thức xây dựng                         | 5         |
| 3. Ảnh minh họa                               | 5         |
| C. TIỀN XỬ LÝ DỮ LIỆU & CHIA DỮ LIỆU          | 8         |
| 1. Tiền xử lý                                 | 8         |
| 2. Tách dữ liệu train-test (Train-test-split) | 8         |
| D. HUẤN LUYỆN DỮ LIỆU                         | 9         |
| 1. Mục tiêu huấn luyện                        | 9         |
| 2. Phương pháp đề xuất                        | 9         |
| 3. Độ đo đánh giá                             | 12        |
| 4. Huấn Luyện Mô Hình                         | 14        |
| <b>PHẦN III: KẾT LUẬN</b>                     | <b>18</b> |
| <b>PHẦN IV: Mở rộng</b>                       | <b>19</b> |
| 1. App  | 19        |

# PHẦN I: ĐẶT VẤN ĐỀ

- Ta có ảnh hoặc video có chuối trong đó, làm sao để xác định vị trí của chúng trong những bức ảnh hay video đó
- Bằng cách ứng dụng công nghệ Deep Learning, nhóm sẽ đi phát hiện và tìm ra vị trí của những quả chuối có trong ảnh hoặc video
- Đề tài có thể phát triển lên để đi phát hiện thêm những quả khác

## PHẦN II:NỘI DUNG

### A. TỔNG QUAN VỀ DỮ LIỆU

#### 1. Giới thiệu bài toán

- Tên đề tài: Banana Detection
- Nguồn dữ liệu: Tự thu thập và tự dán nhãn
- Mục tiêu: Xây dựng model dự đoán và xác định vị trí của chuối trong ảnh/video
- Input: Một bức ảnh có chuối
- Output: Các cặp tọa độ chỉ vị trí của bounding box bao quanh chuối

Input



Output

```
array([[326, 339, 441, 654],  
       [391, 384, 506, 745],  
       [480, 325, 607, 723]], dtype=int32)
```

## **B. ĐỌC-HIỂU DỮ LIỆU**

### **1. Thông tin dữ liệu.**

Tập ảnh:

- Số lượng: 305 tấm

- Độ đa dạng

+ Một quả chuối với nhiều góc độ và hoàn cảnh khác nhau: 30,5%

+ Nhiều quả chuối, nải chuối, chuối bị cắn/cắt, chuối lột vỏ (1 phần và toàn phần) : 69.5%

+ Độ sáng khác nhau:

+ Độ sáng cao : 97.38% (297 tấm)

+ Độ sáng thấp : 2.62% (8 tấm)

+ Trên các nền khác nhau:

+ Trên tay người : 27.74% (85 tấm)

+ Trên đĩa : 57.7% (176 tấm)

+ Trên nền màu đà: 10.81% (33 tấm)

+ Nền trắng, gạch trắng: 2.62% (8 tấm)

+ Cảnh mờ, chuối mờ : 1.13% (4 tấm)

+ Bị khuất bởi các vật thể khác (bình giữ nhiệt, giấy cuộn, laptop, quạt, bánh tráng nướng, quả nhãn, ...) : 12.13% (37 tấm)

+ Màu chuối:

+ Vàng: 68.86% (210 tấm)

+ Xanh: 11.14% (34 tấm)

+ Có thâm đen là phần lớn: 20% (61 tấm)

Tập mô tả:

- Có cùng số lượng với tập ảnh

- File mô tả của ảnh nào sẽ có cùng tên với ảnh đó

## **2. Cách thức xây dựng**

### **- Tập ảnh:**

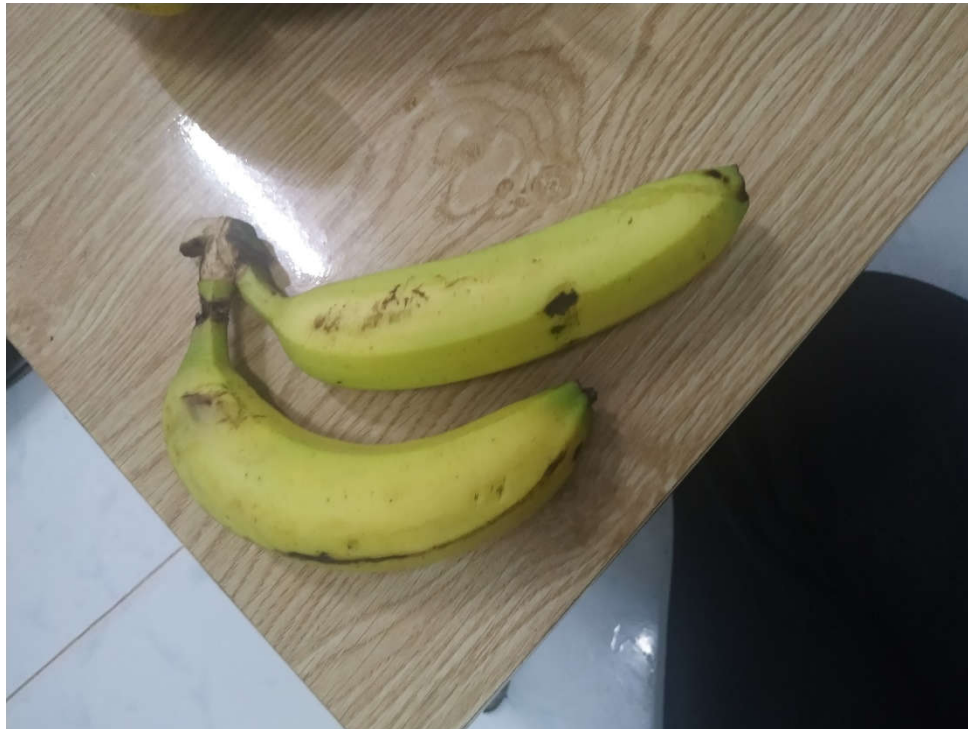
- Mỗi thành viên trong nhóm mỗi ngày mua chuối về và chụp ảnh
- Với một hay nhiều quả chuối thì sẽ có nhiều cách thức tạo ảnh:
  - + Một quả chuối với nhiều góc độ và hoàn cảnh khác nhau (độ sáng khác nhau, góc chụp khác nhau)
  - + Nhiều quả chuối (hoặc nải chuối) và kèm một số vật thể khác (chi tiết ở phần độ đa dạng)
  - + Ảnh từ google

### **- Tập mô tả:**

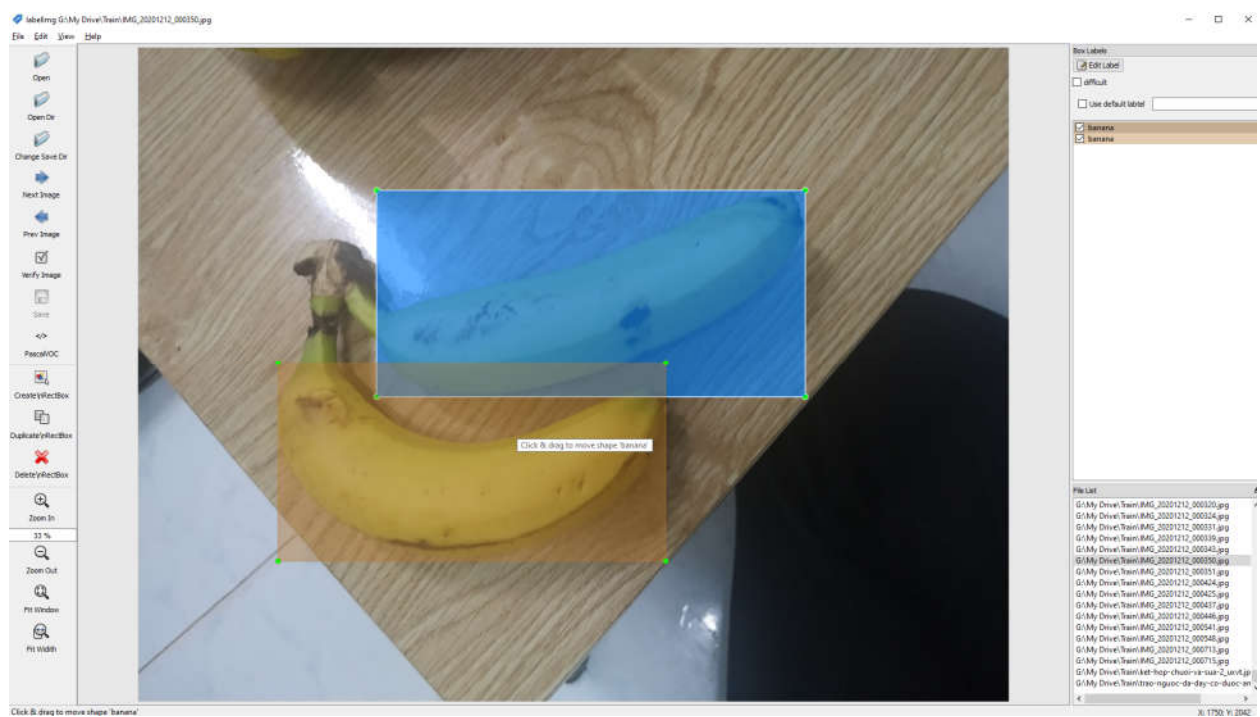
- + Với mỗi ảnh sẽ có một file chú thích riêng về vị trí của từng quả chuối trong ảnh
- + Sử dụng công cụ labelImg để vẽ các ô hình chữ nhật bao quanh từng quả chuối bằng tay
- + Công cụ labelImg sẽ trả về file xml chứa thông tin tọa độ điểm đầu và cuối của từng ô hình chữ nhật đã vẽ
- + Lưu tên của file xml trùng với tên của ảnh gốc

## **3. Ảnh minh họa**

### **- Ảnh gốc:**



- Ảnh khi được gán nhãn bằng công cụ labelImg:



- File mô tả (.xml):

```
IMG_20201212_000350.xml - Notepad
File Edit Format View Help
<annotation>
  <folder>My Drive</folder>
  <filename>IMG_20201212_000350.jpg</filename>
  <path>G:\My Drive\IMG_20201212_000350.jpg</path>
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>4032</width>
    <height>3024</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>banana</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
      <xmin>1105</xmin>
      <ymin>659</ymin>
      <xmax>3089</xmax>
      <ymax>1615</ymax>
    </bndbox>
  </object>
  <object>
    <name>banana</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
      <xmin>649</xmin>
      <ymin>1456</ymin>
      <xmax>2443</xmax>
      <ymax>2374</ymax>
    </bndbox>
  </object>
</annotation>
```

## **C. TIỀN XỬ LÝ DỮ LIỆU & CHIA DỮ LIỆU**

- Vì tập data đều là ảnh nên nhóm em không có tiền xử lý dữ liệu giãn dữ liệu (scaling data), đồng thời tiến hành chia tập dữ liệu train-test (train-test-split).

### **1. Tiền xử lý**

- Không có (bước resize các ảnh trong data về kích cỡ cố định được thực hiện trong mô hình máy học)

### **2. Tách dữ liệu train-test (Train-test-split)**

- Phân chia thành 2 tập train và test theo tỉ lệ 7:3



## D. HUẤN LUYỆN DỮ LIỆU

### 1. Mục tiêu huấn luyện

- Huấn luyện mô hình để đạt được kết quả dự đoán tốt.
- Dùng độ đo mAP để đánh giá mô hình (chi tiết ở dưới)
- Sử dụng mô hình Mask RCNN
- Tinh chỉnh mô hình bằng các tham số: image\_size, backbone.
- Đánh giá mô hình trên tập train, tập test
- Demo phát hiện chuỗi trên một ảnh bất kỳ

### 2. Phương pháp đề xuất

#### - Tên thuật toán

#### **Mask RCNN**

- Mask RCNN là deep neural network nhằm đến việc giải quyết vấn đề về segmentation trong máy học hoặc thị giác máy tính.

- Nghĩa là, nó có thể tách các vật thể khác nhau trong một bức ảnh hoặc video, nó cho bạn bounding boxes, class, mask của vật thể.

- Quá trình phát triển: R-CNN (Region-based Convolutional Neural Networks) -> Fast R-CNN -> Faster R-CNN -> Mask R-CNN.

- Các modules của thuật toán:

- + Backbone
- + Region Proposal Network (RPN)
- + ROI Classifier & Bounding Box Regressor
- + Segmentation Masks

#### **a) Backbone**

- Là convolutional neural network cơ bản (thông thường là ResNet50 hoặc Resnet101) phục vụ cho việc rút trích đặc trưng

- Các layer đầu sẽ phát hiện các đặc trưng low level (góc và cạnh), và các layer về sau sẽ phát hiện các đặc trưng high level (xe, người, vv...)

- Cho ảnh đi qua backbone network, ảnh sẽ được chuyển từ kích thước 1024x1024x3 thành feature map có kích thước 32x32x2048 để phục vụ cho các bước sau.

### **b) Region Proposal Network (RPN)**

- Là lightweight neural network cho phép scan bức ảnh để tìm ra vùng chứa vật thể

- Những vùng mà RPN scan qua được gọi là các archors, mà chính là các boxes được phân bố phủ khắp bức ảnh

- RPN làm việc khá nhanh do scan trên backbone feature map

- Việc này cho phép RPN sử dụng lại các feature đã trích xuất 1 cách hiệu quả và tránh các tính toán trùng lặp.

- RPN tạo ra 2 output với mỗi anchor: Anchor class(foreground or background), bounding box refinement(% change in x,y,width,height)

### **c) ROI Classifier & Bounding Box Regressor**

- Là run regions of interest (ROIs) đề xuất bởi RPN

- Cũng như RPN, trả về 2 output:

- + Anchor class: khác với RPN, network này có thể phân loại các vùng thành các class cụ thể (người, ghế, xe,...vv). Nó cũng thể tạo ra background class, khiến cho ROI bị bỏ đi

- + Bounding box refinement: tương tự như RPN, mục đích cũng là điều chỉnh lại vị trí và kích cỡ của bounding box bao quanh vật thể.

- Các classifiers không xử lý tốt input có kích cỡ thay đổi (yêu cầu fixed input), nhưng ở bước bounding box refinement ở RPN, các ROI boxes có thể có kích cỡ khác nhau

=> ROI Pooling: crop 1 phần của feature map và resize về 1 fixed file

### **d) Segmentation Masks**

- Thực hiện công đoạn mask branch

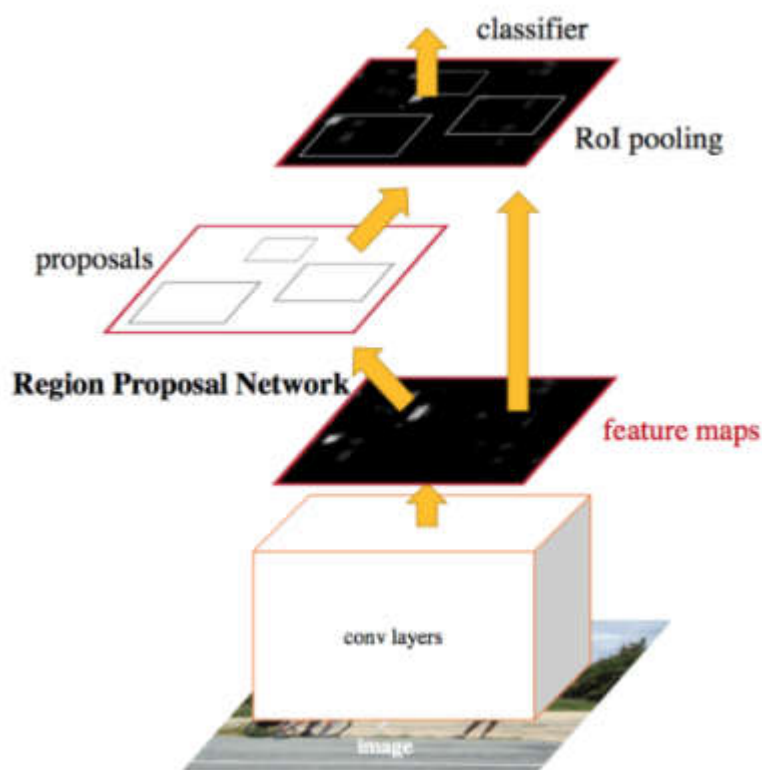
- Là convolutional network mà nó chọn ra các positive mask từ ROI classifier và tạo masks cho chúng.

- Các mask được tạo ra có độ phân giải thấp: 28x28 pixels. Nhưng chúng là các soft masks, biểu diễn bởi số thực, nên chúng sẽ chứa đựng nhiều thông tin hơn binary masks

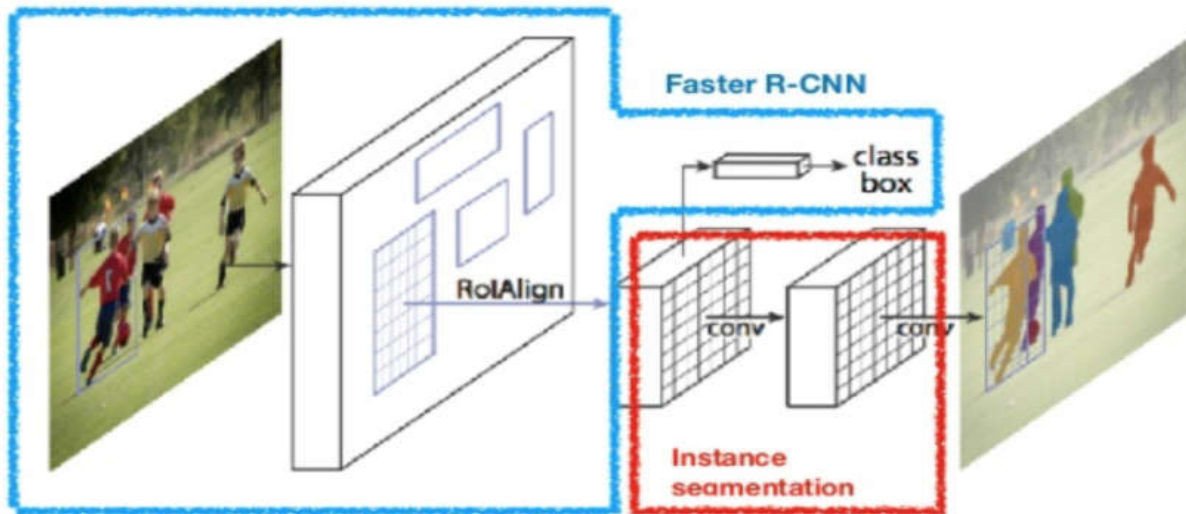
- Kích cỡ các mask nhỏ giúp mask branch nhẹ hơn.

- Sơ đồ:

- + Faster RCNN:



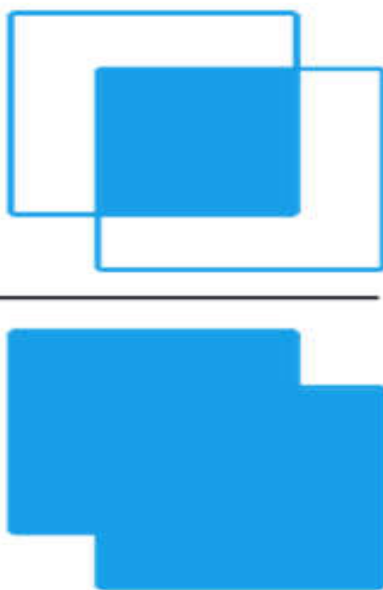
- + Mask RCNN:



### 3. Độ đo đánh giá

#### a. IoU

- IoU (Intersection over Union)
- Dùng để xác định 1 dự đoán có chính xác hay không
- Được tính bằng lấy phần giao giữa ô dự đoán và ô thực tế chia cho phần hợp giữa chúng.
- Một dự đoán được coi là True Positive nếu  $\text{IoU} > \text{threshold}$ , False Positive nếu  $\text{IoU} < \text{threshold}$ .

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$


*Công thức 1*

- Dùng để tính toán các độ đo precision và recall

### **b. Precision và Recall**

- *Precision*, gọi là độ chính xác, trong tập tìm được thì bao nhiêu cái (phân loại) là đúng. Precision còn được gọi là Specificity (độ đặc hiệu), Positive predictive value (Tỉ lệ thực sự Positive trên tổng số các trường hợp được mô hình dán nhãn “Positive” - PPV).

- *Recall*, gọi là độ bao phủ, trong số các tồn tại, tìm ra được bao nhiêu cái (phân loại). Recall còn được gọi là Sensitivity (độ nhạy), hit rate (tỉ lệ trúng đích), True Positive rate (Tỉ lệ phân loại Positive đúng trên tổng số các trường hợp Positive - TPR).

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{\# \text{ ground truths}}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{TP}}{\# \text{ predictions}}$$

*Công thức 2*

### c. mAP

- AP (Average Precision): là mean của giá trị precision tại tập tất cả các độ đo recall

$$AP = \frac{1}{n} \sum_{r > \bar{r}} P(r)$$

*Công thức 3*

-> Ý nghĩa: Giá trị precision tại mỗi độ đo recall  $r$  với  $r >$  một ngưỡng  $\bar{r}$  cho trước,  $n$  là số lượng các giá trị recall  $r$  thỏa

- mAP: trung bình cộng AP của tất cả các class

## 4. Huấn Luyện Mô Hình

### 4.1. Cài đặt

#### a) Huấn luyện

- Cài đặt các thư viện cần thiết: tensorflow, keras,..vv
- Clone thư viện Mask RCNN từ github về
- Cài đặt thông số cho Detector Config:
- Chuyển xml file sang dict
- Load các ảnh tương ứng với mask của nó
- Chuẩn bị tập train và tập test
- Download file COCO weight
- Tạo training model và load weight
- Bắt đầu train
- Lưu file weight cuối cùng sau khi train
- Đánh giá bằng mAP

## b) Dự đoán

- Cài đặt thư viện: như trên
- Cài đặt thông số cho

### 4.2. Huấn luyện mô hình gốc

- Kết quả:
  - + Độ đo mAP của tập train là 0.831
  - + Độ đo mAP của tập test là 0.716
- Một số ảnh minh họa:





### 4.3 Tinh chỉnh tham số

- Thay đổi các giá trị image size, backbone như sau:

- + Case 0 (base case): image size = (256, 256), backbone = 'resnet50'
- + Case 1: image size = (512, 512), backbone = 'resnet50'
- + Case 2: image size = (1024, 1024), backbone = 'resnet50'
- + Case 3: image size = (256, 256), backbone = 'resnet101'
- + Case 4: image size = (512, 512), backbone = 'resnet101'
- + Case 5: image size = (1024, 1024), backbone = 'resnet101'

- Kết quả:

|                      | mAP (train) | mAP (test) |
|----------------------|-------------|------------|
| 256x256 – resnet50   | 0.831       | 0.716      |
| 512x512 – resnet50   | 0.642       | 0.660      |
| 1024x1024 – resnet50 | 0.700       | 0.682      |
| 256x256 – resnet101  | 0.822       | 0.736      |
| 512x512 – resnet101  | 0.846       | 0.787      |



|                       |       |       |
|-----------------------|-------|-------|
| 1024x1024 – resnet101 | 0.821 | 0.767 |
|-----------------------|-------|-------|

- Nhận xét: Model có kích cỡ ảnh 512x512, sử dụng ResNet 101 cho kết quả mAP trên cả 2 tập train và test cao nhất

- Một số ảnh minh họa:

Mask gốc



(256,256)-resnet50



(512,512)-resnet50



(1024,1024)-resnet50



(512,512)-resnet101

(256,256)-resnet101



(1024,1024)-resnet50



## PHẦN III: KẾT LUẬN

### - Về data

- Dữ liệu không có ảnh không có chuối
- Chỉ có một loại quả trong data, nên bổ sung các quả khác để huấn luyện
- Kết quả dự đoán khá tốt đối với mô hình sau khi tinh chỉnh (80%)

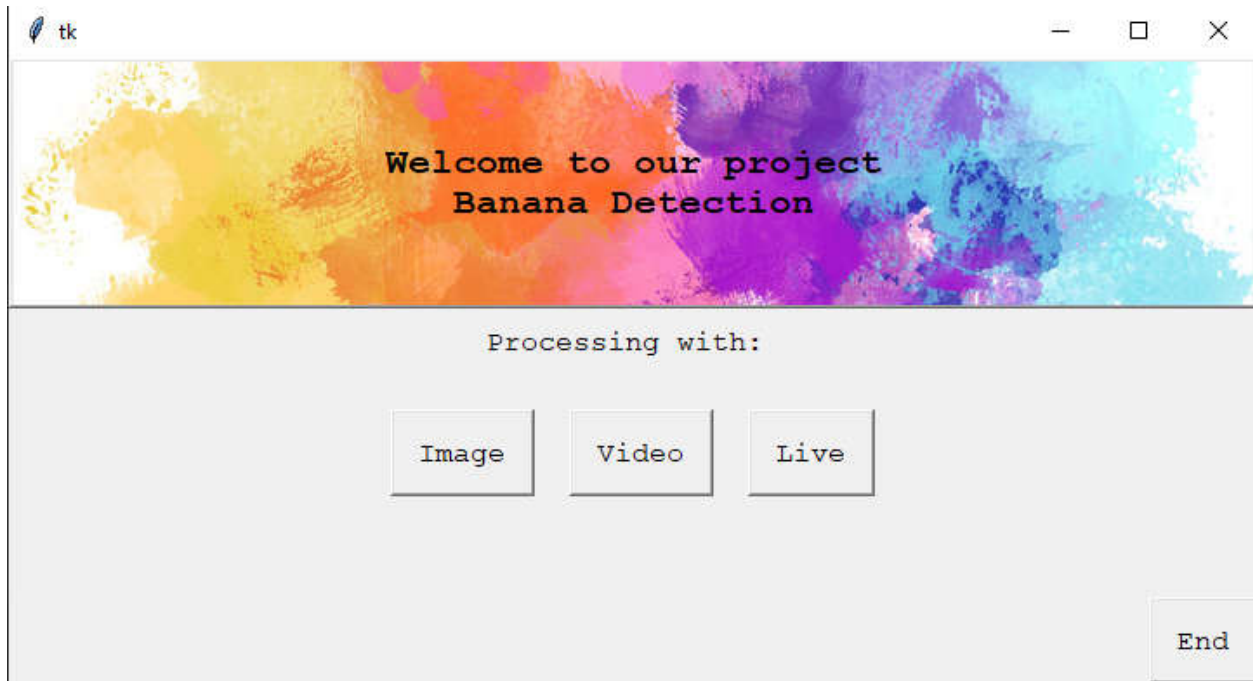
### - Về mô hình

- Có thể train với nhiều epoch hơn để cải thiện kết quả dự đoán

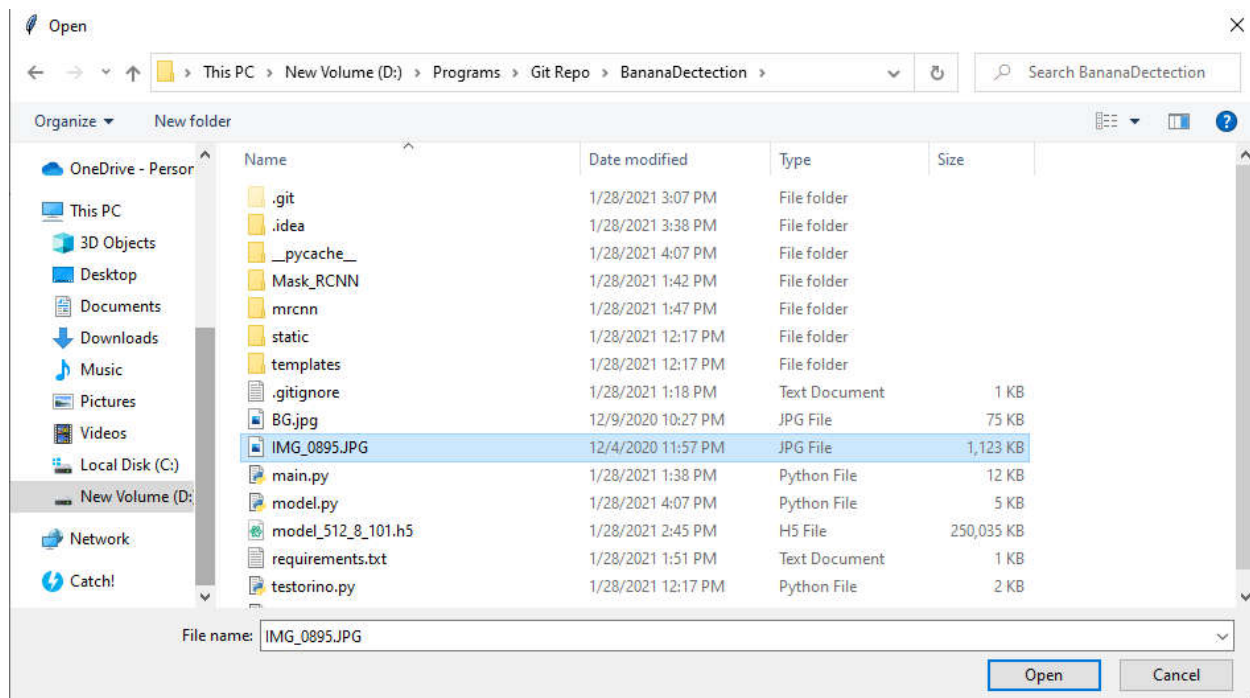
# PHẦN IV: Mở rộng

## 1. App

- Nhóm em đã làm một app cơ bản để thực hiện dự đoán trên một ảnh và video bất kỳ
- Giao diện chương trình:



- Chọn chức năng:
  - + Image:
    - + Chọn ảnh cần dự đoán

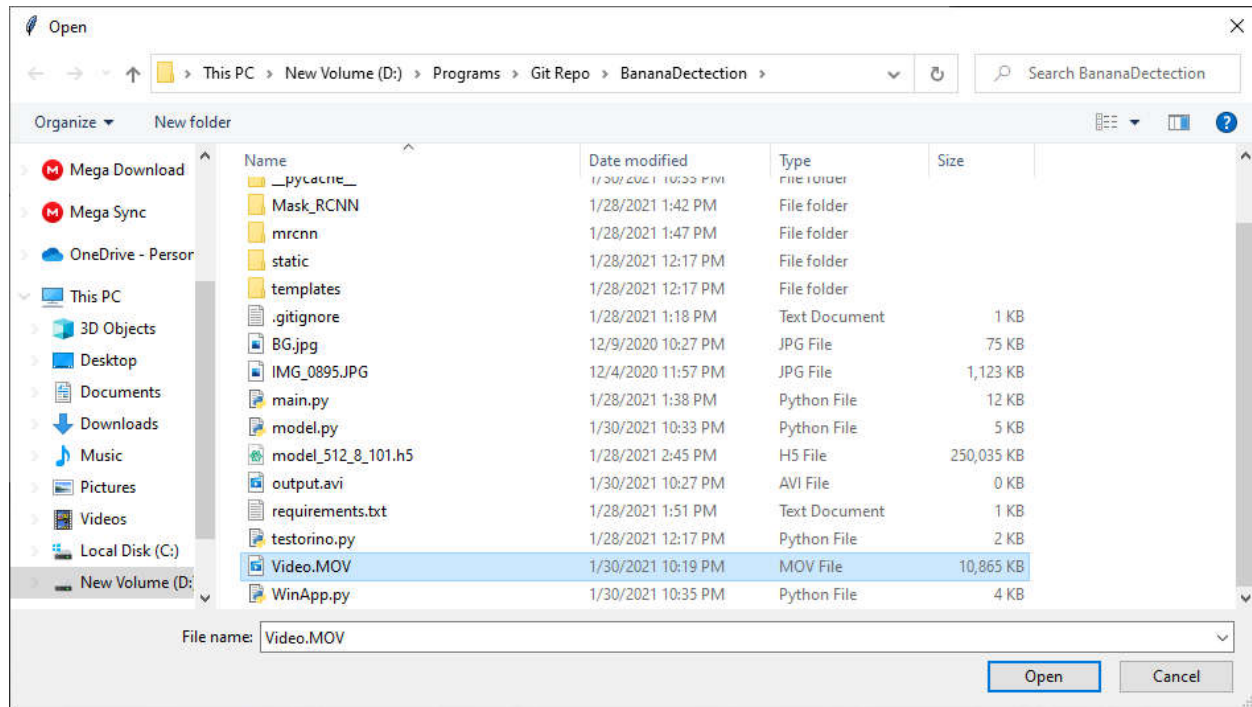


+ Nhận kết quả:



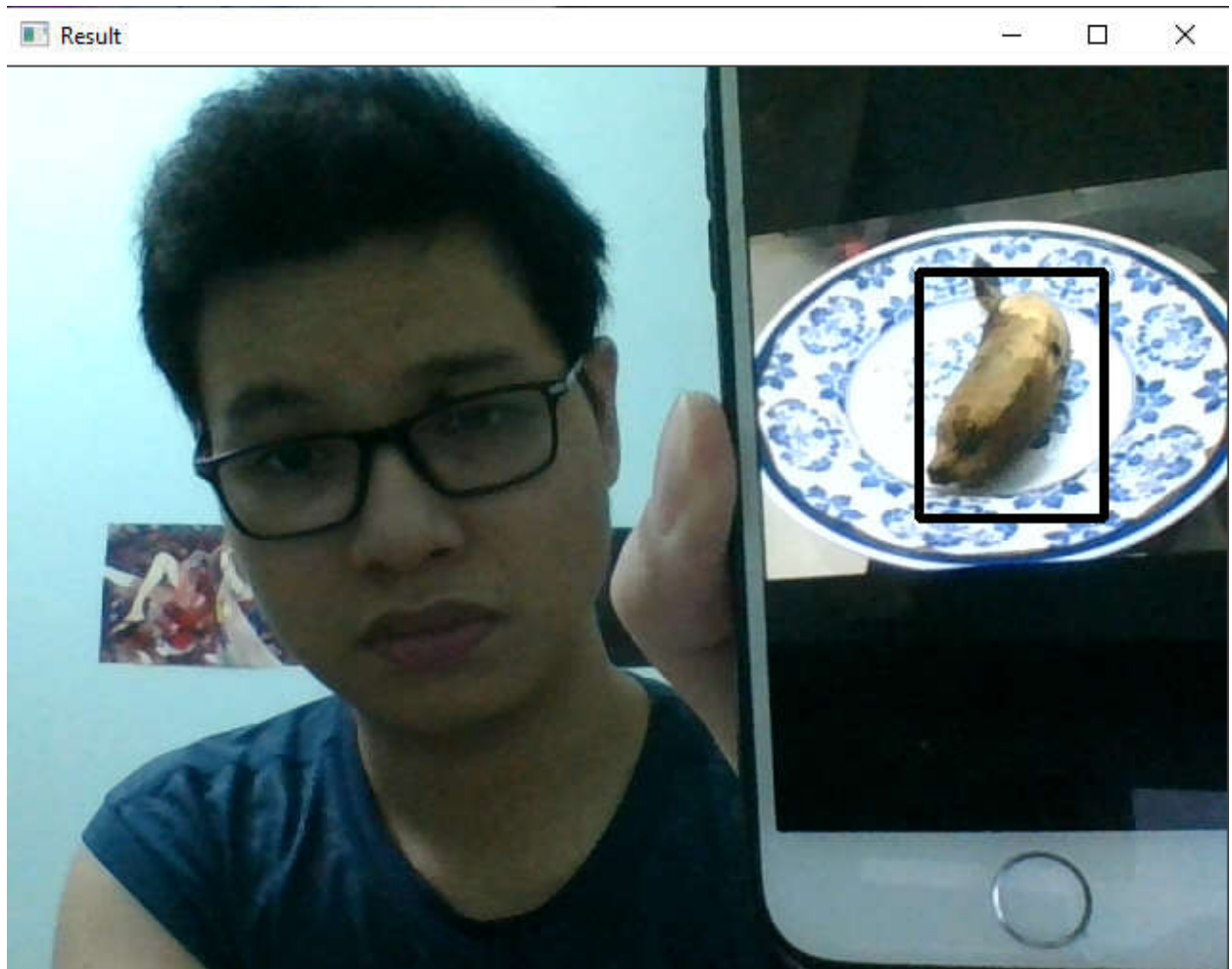
+ Video:

+ Chọn video cần dự đoán



-> Xử lý còn chậm và lag

+ Live:



>- Xử lý còn chậm và lag

# KẾT THÚC PHẦN BÁO CÁO

~ ~ Cảm ơn thầy/cô và các bạn đã theo dõi ~ ~