TRƯỜNG ĐẠI HỌC ĐIỆN LỰC

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**ĐỒ ÁN TỐT NGHIỆP**

**ĐỀ TÀI:**

**XÂY DỰNG HỆ THỐNG NHẬN DẠNG CHỮ IN**

|  |  |
| --- | --- |
|  |  |
| |  |  | | --- | --- | | **Sinh viên thực hiện** | **: HOÀNG ĐỨC LONG** | | **Mã sinh viên** | **: 18810301452** | | **Giảng viên hướng dẫn** | **: Th.S HOÀNG VĂN QUÝ** | | **Ngành** | **: CÔNG NGHỆ THÔNG TIN** | | **Chuyên ngành** | **: CÔNG NGHỆ PHẦN MỀM** | | **Lớp** | **: D13-CNPM5** | | **Khóa** | **: 2018 - 2023** | |  |
|  |  |

***Hà Nội, tháng 01 năm 2023***

**PHIẾU CHẤM ĐIỂM**

Sinh viên thực hiện:

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và tên** | **Chữ ký** | **Ghi chú** |
| 1 | Hoàng Đức Long  (MSV: 18810310452) |  |  |

Giảng viên chấm:

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **Chữ ký** | **Ghi chú** |
| Giảng viên chấm 1: |  |  |
| Giảng viên chấm 2: |  |  |
| Giảng viên chấm 3: |  |  |

**MỤC LỤC**

[**DANH MỤC HÌNH ẢNH** 4](#_Toc124195342)

[**DANH MỤC CÁC BẢNG** 6](#_Toc124195343)

[**CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI** 1](#_Toc124195344)

[1.1. Khảo sát hiện trạng 1](#_Toc124195345)

1.1.1. [Đặc điểm của các dạng văn bản, giấy tờ tại Việt Nam 1](#_Toc124195346)

[1.1.2. Xác định các yêu cầu của module nhận dạng chữ in. 2](#_Toc124195347)

[1.1.3. Quy trình bóc tách và nhận dạng chữ in 2](#_Toc124195348)

[1.2. Mô tả hệ thống nhận dạng chữ in 2](#_Toc124195349)

[**CHƯƠNG 2: XÂY DỰNG MODULE PHÁT HIỆN VÀ NHẬN DẠNG CHỮ IN** 5](#_Toc124195350)

[2.1. Xây dựng module bóc tách biểu mẫu với YOLOV5 5](#_Toc124195351)

[2.1.1. Giới thiệu về YOLO 5](#_Toc124195352)

[2.1.2. Kiến trúc mạng YOLO 5](#_Toc124195353)

[2.1.3. Quy trình huấn luyện model phát hiện chữ in và biểu mẫu với YOLOv5 11](#_Toc124195354)

[2.1.4. Cài đặt môi trường của YOLOv5 23](#_Toc124195355)

[2.2. Xây dựng module nhận dạng chữ in. 24](#_Toc124195356)

[2.2.1. Mô hình nhận dạng Attention-OCR 24](#_Toc124195357)

[2.2.2. Mô hình nhận dạng VietOCR 27](#_Toc124195358)

[2.2.3. Đánh giá kết quả nhận dạng của mô hình VietOCR và Attention-OCR 31](#_Toc124195359)

[2.2.4. Ứng dụng của công nghệ nhận dạng chữ in 31](#_Toc124195360)

[2.3. Sơ đồ quy trình huấn luyện phát hiện, nhận dạng chữ in và biểu mẫu 33](#_Toc124195361)

[**CHƯƠNG 3: KẾT LUẬN** 34](#_Toc124195362)

[3.1. Kết quả đạt được 34](#_Toc124195363)

[3.2. Hướng phát triển 34](#_Toc124195364)

[**TÀI LIỆU THAM KHẢO** 35](#_Toc124195365)

# **DANH MỤC HÌNH ẢNH**

[Hình 1. 1: Đặc điểm một số loại kiểu chữ in trong các văn bản ở Việt Nam 1](#_Toc124195366)

[Hình 1. 2: Quy trình bóc tách và nhận dạng chữ in và các biểu mẫu 2](#_Toc124195367)

[Hình 1. 3: Sơ đồ tổng quát hệ thống 3](#_Toc124195368)

[Hình 2. 1: Sơ đồ kiến trúc mạng YOLO 6](#_Toc124195369)

[Hình 2. 2: Các feature maps của mạng YOLO 7](#_Toc124195370)

[Hình 2. 3: Xác định anchor box cho một vật thể 9](#_Toc124195371)

[Hình 2. 4: Trường hợp non-max suppression trong YOLO 10](#_Toc124195372)

[Hình 2. 5: Công thức tính tỉ lệ IOU (Intersection of Union) 11](#_Toc124195373)

[Hình 2. 6: Sơ đồ đánh giá độ chính xác model của YOLOv5 12](#_Toc124195374)

[Hình 2. 7: Tải xuống src của tool LabelImg 13](#_Toc124195375)

[Hình 2. 8: Cài đặt các thư viện python cần thiết cho tool LabelImg 13](#_Toc124195376)

[Hình 2. 9: Khởi động tool LabelImg 14](#_Toc124195377)

[Hình 2. 10: File nhãn của tool LabelImg 14](#_Toc124195378)

[Hình 2. 11: Danh sách đối tượng segment chữ in 14](#_Toc124195379)

[Hình 2. 12: Danh sách đối tượng khoanh vùng trong CV 15](#_Toc124195380)

[Hình 2. 13: Cách khoanh vùng đối tượng và gán nhãn 16](#_Toc124195381)

[Hình 2. 14: Chế độ lưu nhãn YOLO của tool LabelImg 16](#_Toc124195382)

[Hình 2. 15: Kết quả sau khi gán nhãn 17](#_Toc124195383)

[Hình 2. 16: Cấu trúc thư mục chứa data huấn luyện YOLOv5 17](#_Toc124195384)

[Hình 2. 17: Cách tạo ứng dụng với Colab 18](#_Toc124195385)

[Hình 2. 18: Cấu trúc thư mục train của YOLOv5 với Colab 19](#_Toc124195386)

[Hình 2. 19: Cài đặt gói môi trường YOLOv5 và kiểm tra GPU 19](#_Toc124195387)

[Hình 2. 20: Kiểm tra các thư mục trên Driver 20](#_Toc124195388)

[Hình 2. 21: Chạy huấn luyện YOLOv5 20](#_Toc124195389)

[Hình 2. 22: File coco.yaml 21](#_Toc124195390)

[Hình 2. 23: Model thu được sau khi huấn luyện YOLOv5 21](#_Toc124195391)

[Hình 2. 24: Một số kết quả huấn luyện phát hiện các vùng trong CV YOLOv5 22](#_Toc124195392)

[Hình 2. 25: Một số kết quả phát hiện các vùng trong CV với YOLOv5 23](#_Toc124195393)

[Hình 2. 26: Mô hình mạng của kỹ thuật Attention-OCR 24](#_Toc124195394)

[Hình 2. 27: Cấu trúc mạng CNN (Convolutional Neural Network) 25](#_Toc124195395)

[Hình 2. 28: Mô hình hoạt động của mang Recurrent Neural Network 26](#_Toc124195396)

[Hình 2. 29: Dữ liệu chữ in 27](#_Toc124195397)

[Hình 2. 30: Mô hình huấn luyện của Attention OCR 27](#_Toc124195398)

[Hình 2. 31: Một số kết quả thử nghiệm nhận dạng của mô hình Attention OCR 27](#_Toc124195399)

[Hình 2. 32: Các feature maps trong mô hình VietOCR 28](#_Toc124195400)

[Hình 2. 33: Mô hình nhận dạng VietOCR 29](#_Toc124195401)

[Hình 2. 34: Cấu trúc tập dữ liệu huấn luyện của mô hình VietOCR 29](#_Toc124195402)

[Hình 2. 35: Mô hình huấn luyện của VietOCR 30](#_Toc124195403)

[Hình 2. 36: Một số kết quả nhận dạng của mô hình VietOCR 30](#_Toc124195404)

[Hình 2. 37: Sơ đồ quy trình huấn luyện phát hiện, nhận dạng chữ in và biểu mẫu 33](#_Toc124195405)

# **DANH MỤC CÁC BẢNG**

[Bảng 2. 1: Cài đặt môi trường của YOLOv5 23](#_Toc124195485)

[Bảng 2. 2: Đánh giá kết quả thử nghiệm nhận dạng 31](#_Toc124195486)

# **CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI**

## 1.1. Khảo sát hiện trạng

Hiện nay, trên thế giới các hệ thống nhận dạng chữ in tự động đã được áp dụng một cách rộng rãi, điển hình là hệ thống tự động nhận chữ in ABBYY Recognition Server. Nhưng các hệ thống này yêu cầu những thiết bị chuyên dụng có giá thành cao, đồng thời khó áp dụng ở Việt Nam. Cùng với sự phát triển về kinh tế, mức tăng trưởng về nhu cầu xác thực, nhận dạng văn bản. Văn bản biểu mẫu tăng lên, đòi hỏi tốc độ xử lí các yêu cầu và tìm kiếm phải tăng theo. Đã gây áp lực rất nhiều cho những người phụ trách quản lý tài liệu.

Vì vậy việc áp dụng trí tuệ nhân tạo AI vào nhận dạng đã không còn xa lạ với mọi người, nó giúp giảm bớt thời gian soát vé thủ công, tiếp kiệm chi phí thuê nhân viên, …

Chính vì vậy, việc xây dựng hệ thống nhận dạng chữ in này là hữu ích và cần thiết, ứng dựng được vào thực tiễn.

* + 1. *Đặc điểm của các dạng văn bản, giấy tờ tại Việt Nam*

Với đặc thù đa dạng của văn bản ở Việt Nam. Một số đặc điểm của của cả các giấy tờ ở Việt Nam, bao gồm:

* Giấy tờ về vẳn bản hành chính (có format rõ ràng)
* Chứng minh thư nhân dân, căn cước công dân,… (chữ có nền làm nhiều).
* Sơ yếu lí lịch (nhiều font chữ cỡ cữ khác nhau).
* Các loại giấy tờ có khác



Hình 1. 1: Đặc điểm một số loại kiểu chữ in trong các văn bản ở Việt Nam

Nhận dạng biển số từ các file ảnh định dạng JPG, BMP hay PNG. Có thể hoạt động như một chương trình trên Windows, Linux...

### *1.1.2. Xác định các yêu cầu của module nhận dạng chữ in.*

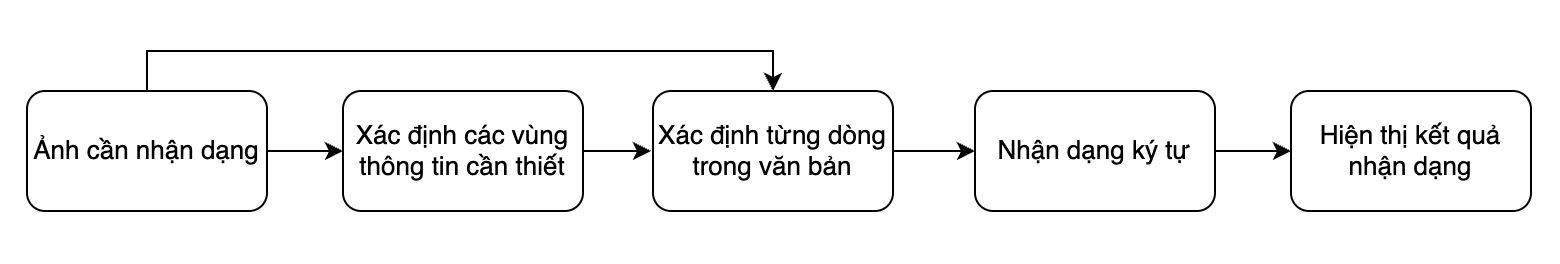
* Bóc tách được các chữ in trong các văn bản, xác định được các vùng văn bản trọng tâm.
* Nhận dạng được ảnh chữ in với độ chính xác cao.
* Tốc độ nhận dạng và bóc tách nhanh.

### *1.1.3. Quy trình bóc tách và nhận dạng chữ in*

Công nghệ nhận dạng chữ in ứng dụng vào hệ thống nhận dạng và bóc tác văn bản với mục đích dùng chuỗi ký tự là thông tin trong văn bản để tích hợp vào cơ sở dữ liệu quản lí. Điều này giúp ích rất nhiều cho việc quản lý giất tờ và tra cứu văn bản:

Khi giấy tờ đã được đưa vào hệ thống thì việc truy lại thông tin giấy tờ trên hệ thống phần mềm chỉ cần nhập chuỗi ký tự trong văn bản thì truy ra thông tin văn bản một cách nhanh chóng, …

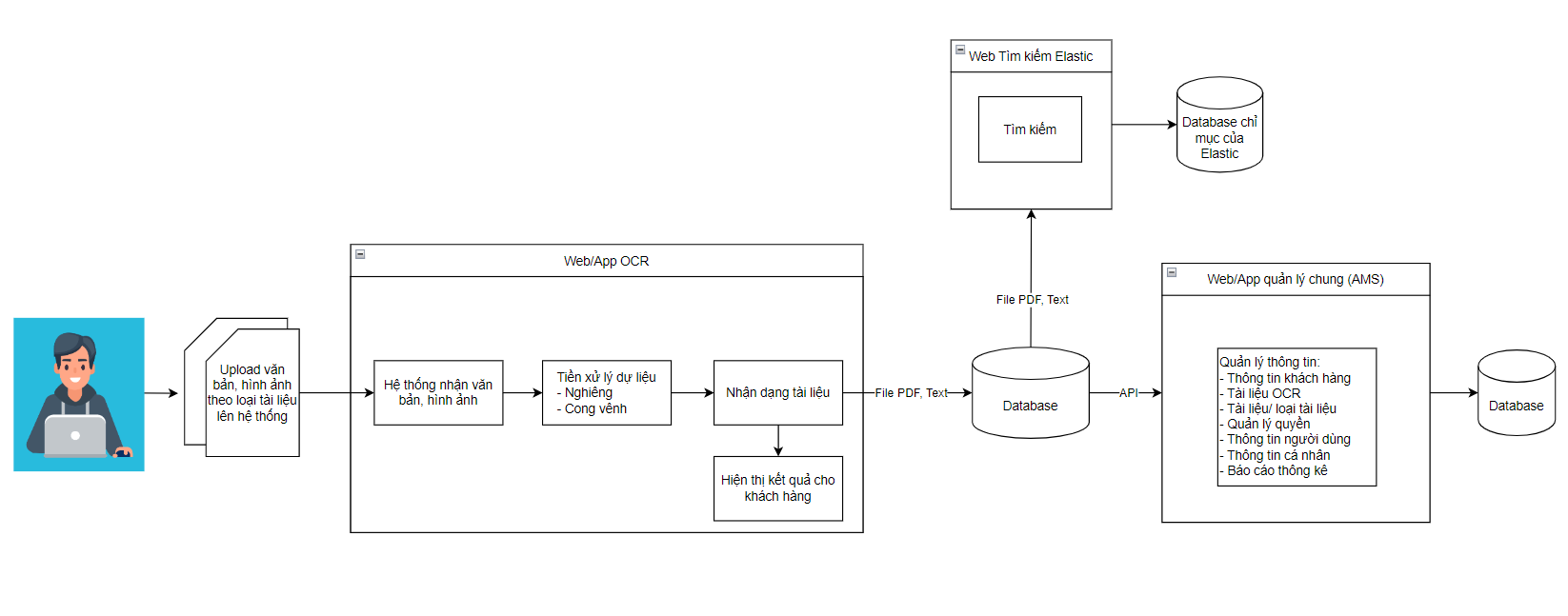
* Module nhận dạng có khả năng nhận dạng và bóc tách được các loại văn bản như: Bóc tách toàn văn, bóc tách biểu mẫu thông tin các nhân, bóc tách chứng minh thư nhân dân.



Hình 1. 2: Quy trình bóc tách và nhận dạng chữ in và các biểu mẫu

## 1.2. Mô tả hệ thống nhận dạng chữ in

Thiết kế hệ thống nhận dạng chữ in và bóc tách biểu mẫu.



Hình 1. 3: Sơ đồ tổng quát hệ thống

Nhận dạng chữ in/chữ viết tay bao gồm công việc nhận diện giấy tờ tùy thân, văn bản theo phương thức hình ảnh hoặc file PDF. Khi khách hàng truy cập vào hệ thống nhận dạng sẽ đăng nhập tài khoản (Đăng ký), chọn hình thức nhận dạng văn bản-upload tài liệu cần nhận dạng và kiểm tra dữ liệu trả về, quản lý thông tin cá nhân và đăng xuất tài khoản khỏi hệ thống.

Bên cạnh những chức năng nghiệp vụ, hệ thống phải đảm bảo những đặc tính sau:

* Tính tích hợp cao: Mọi hoạt động từ của khách hàng đều sẽ được lưu trữ vào hệ thống quản lý chung (AMS) và mã hóa trên cơ sở dữ liệu.
* Hệ thống phải đảm bảo tính bảo mật cao
* Thời gian đáp ứng của hệ thống phải đủ nhanh.
* Hệ thống có giao diện thân thiện với người dung.
* Hệ thống có khả năng mở rộng, dễ dàng tích hợp với các hệ thống khác.
* Chức năng của hệ thống
  + Nhận dạng bất kì các loại tài liệu có chữ in.
  + Bóc tác giấy tờ tuy thần (Chứng minh nhân dân và sơ yếu lí lịch)
* Đối tượng sử dụng
* Các công ty cần số hóa tài liệu.
* Các công ty cần định danh cá nhận tự động thông qua giấy tờ tùy thân
* Các siêu thị, khu vui chơi giải trí
* Các khu chung cư, khách sạn, nhà hàng, công trường, nhà máy, xí nghiệp.
* Phạm vi của hệ thống nhận dạng chữ in:
* Hệ thống nhận dạng chữ in là một giải pháp nhận diện chữ in, văn bản tài liệu thông minh áp dụng công nghệ nhận diện sử dụng trí tuệ nhân tạo (Yolo, Attention-OCR) giúp quản lý giấy tờ, khoa học, thuận tiện, tiết kiệm thời gian và công sức.
* Lợi ích của hệ thống nhận dạng chữ in
* Trợ giúp người mắc bệnh mù và khiếm thị: Vì sao lại nói OCR có khả năng hỗ trợ người mắc bệnh mù và khiếm thị, những người có ảnh hưởng về thị giác. Bởi, OCR có khả năng quét và đọc các từ trên màn hình. Từ đây, những người gặp vấn đề thị giác có thể dễ dàng hiểu được chúng.
* Tìm kiếm và thực hành dữ liệu: Thực tế, OCR có khả năng tạo ra những nội dung văn bản riêng của quét tài liệu giúp chúng có thể dễ dàng tìm kiếm và xác định vị trí tài liệu dựa trên từ khóa. Đồng thời, OCR cũng cho phép nhanh nhẹn hơn trong việc chỉnh sửa và xử lý văn bản.
* Cập nhập dữ liệu nhanh chóng: OCR đảm bảo chức năng cải thiện hiệu quả và nhanh chóng cho công việc văn phòng cũng như năng suất cao. Bởi, hầu hết trong quá trình làm việc ở văn phòng nhu cầu scan (quét) tài liệu ngày một lớn. Điều này sẽ giúp tiết kiệm thời gian, đồng thời cập nhập dữ liệu nhanh, chính xác nhất cho người dùng.
* Cấu hình yêu cầu của hệ thống là cấu hình phù hợp với việc phát triển công nghệ thông tin của doanh nghiệp, không yêu cầu sử dụng các thiết bị chuyên dụng đắt tiền.
* Ngoài ra hệ thống nhận dạng chữ in cho phép nâng cấp một cách dễ dàng, đáp ứng yêu cầu khác như lư trữ ảnh, video kèm theo văn bản, mở rộng số lượng camera được quản lý, thêm các loại tài liệu biểu mẫu yêu cầu của khách hàng, liên kết với các chương trình quản lý khác.

# **CHƯƠNG 2: XÂY DỰNG MODULE PHÁT HIỆN VÀ NHẬN DẠNG CHỮ IN**

## 2.1. Xây dựng module bóc tách biểu mẫu với YOLOV5

### *2.1.1. Giới thiệu về YOLO*

YOLO trong object detection có nghĩa là “You only look once”. Nghĩa là chỉ cần nhìn 1 lần là có thể phát hiện ra vật thể. Về độ chính xác YOLO có thể không phải là thuật toán tốt nhất nhưng nó là thuật toán nhanh nhất trong các lớp mô hình object detection. Nhưng YOLO đạt được tốc độ gần như real time mà độ chính xác không kém so với các model thuộc top đầu.

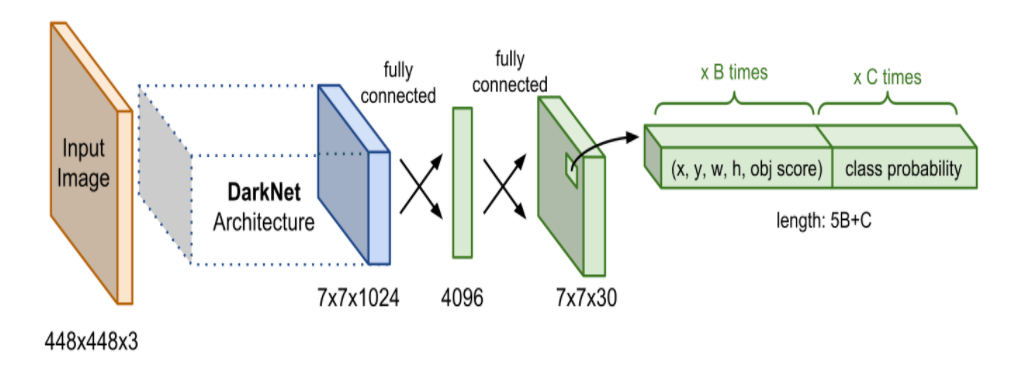
YOLO là thuật toán object detection nên mục tiêu của mô hình không chỉ là dự báo nhãn cho vật thể như các bài toán classification mà nó còn xác định location của vật thể. Do đó YOLO có thể phát hiện được nhiều vật thể có nhãn khác nhau trong một bức ảnh thay vì chỉ phân loại duy nhất một nhãn cho một bức ảnh.

### *2.1.2. Kiến trúc mạng YOLO*

Kiến trúc YOLO bao gồm: base network là các mạng convolution làm nhiệm vụ trích xuất đặc trưng. Phần phía sau là những Extra Layers được áp dụng để phát hiện vật thể trên feature map của base network.

Base network của YOLO sử dụng chủ yếu là các convolutional layer và các fully conntected layer. Các kiến trúc YOLO cũng khá đa dạng và có thể tùy biến thành các version cho nhiều input shape khác nhau. Một số khái niệm cần biết:

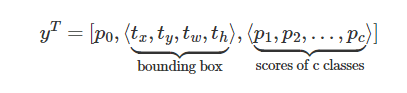
* Bounding box, anchor box: Bounding box là khung hình bao quanh vật thể. Anchor box là những khung hình có kích thước xác định trước, có tác dụng dự đoán bounding box.
* Mạng nơ ron tích chập (Convolutional Neural Network): Đây là mạng nơ ron áp dụng các layer Convolutional kết hợp với Maxpooling để giúp trích xuất đặc trưng của ảnh tốt hơn
* Feature map: Là một khối output mà ta sẽ chia nó thành một lưới ô vuông và áp dụng tìm kiếm và phát hiện vật thể trên từng cell.
* Non-max suppression: Phương pháp giúp giảm thiểu nhiều bounding box overlap nhau về 1 bounding box có xác suất lớn nhất.
* **Ground truth box**: Khung hình được xác định trước từ bộ dữ liệu thông qua tọa độ (cx, cy, w, h) giúp xác định vật thể
* **Offsets**: Các tọa độ (cx, cy, w, h) để xác định vật thể.
* **Positive matching prediction**: Khung được dự báo (predicted box) là vùng có vật thể là đúng, được xác định dựa trên tỷ lệ IoU > 0.5 giữa predicted box với ground truth box.
* **Negative matching prediction**: Khung được dự báo (predicted box) là vùng không chứa vật thể là đúng, cũng được xác định dựa trên IoU < 0.5 giữa predicted box với ground truth box.



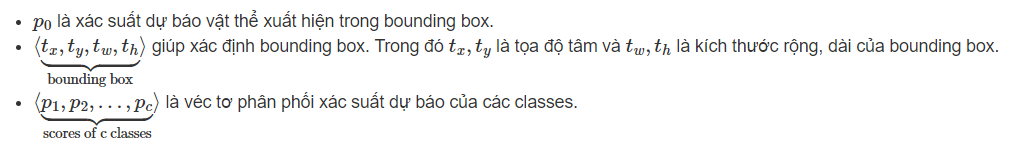
Hình 2. 1: Sơ đồ kiến trúc mạng YOLO

Thành phần Darknet Architechture được gọi là base network có tác dụng trích suất đặc trưng. Output của base network là một feature map có kích thước 7x7x1024 sẽ được sử dụng làm input cho các Extra layers có tác dụng dự đoán nhãn và tọa độ bounding box của vật thể.

Đầu ra của YOLO là một véc tơ sẽ bao gồm các thành phần:

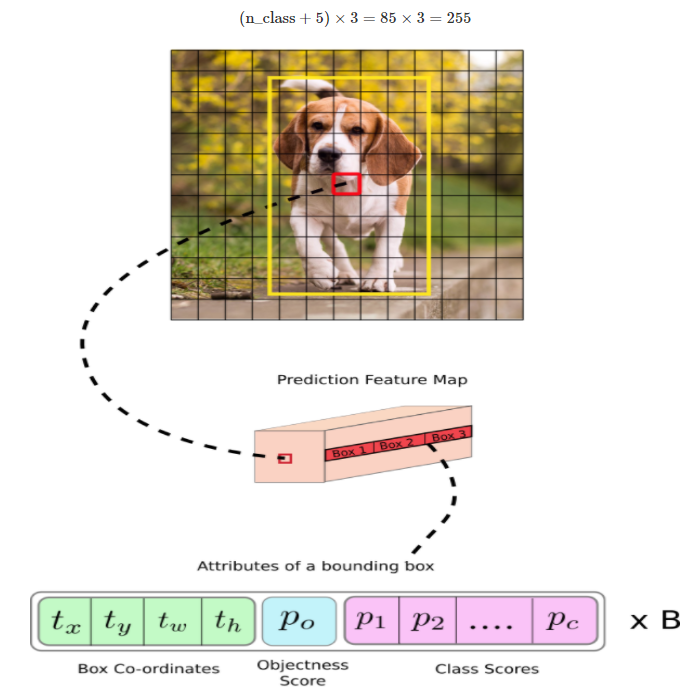


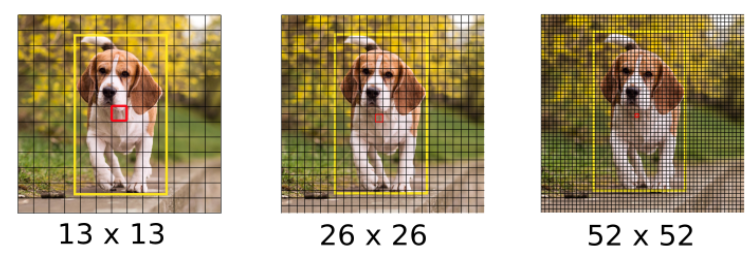
Trong đó



Việc hiểu output khá là quan trọng để chúng ta cấu hình tham số chuẩn xác khi huấn luyện model qua các open source như darknet. Như vậy output sẽ được xác định theo số lượng classes theo công thức (n\_class+5).

Nếu huấn luyện 80 classes thì bạn sẽ có output là 85. Trường hợp bạn áp dụng 3 anchors/cell thì số lượng tham số output sẽ là:





Hình 2. 2: Các feature maps của mạng YOLO

Với input shape là 416x416, output là 3 feature maps có kích thước lần lượt là 13x13, 26x26 và 52x52. Những feature map ban đầu có kích thước nhỏ giúp dự báo được các object kích thước lớn. Những feature map sau có kích thước lớn hơn trong khi anchor box được giữ cố định kích thước nên sẽ giúp dự báo các vật thể kích thước nhỏ.

Trên mỗi một cell của các feature map chúng ta sẽ áp dụng 3 anchor box để dự đoán vật thể. Như vậy số lượng các anchor box khác nhau trong một mô hình YOLO sẽ là 9 (3 featue map x 3 anchor box).

Đồng thời trên một feature map hình vuông S x S, mô hình YOLOv3 sinh ra một số lượng anchor box là: S x S x 3. Như vậy số lượng anchor boxes trên một bức ảnh sẽ là:

(13×13+26×26+52×52) ×3=10647(anchor boxes)

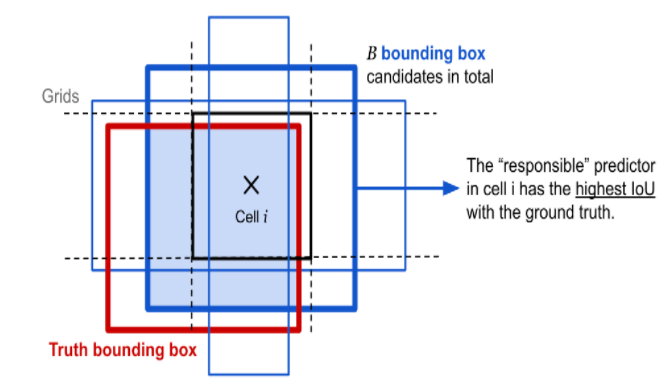
Đây là một số lượng rất lớn và là nguyên nhân khiến quá trình huấn luyện mô hình YOLO vô cùng chậm bởi chúng ta cần dự báo đồng thời nhãn và bounding box trên đồng thời 10647 bounding boxes.

Một số lưu ý khi huấn luyện YOLO:

* Khi huấn luyện YOLO sẽ cần phải có RAM dung lượng lớn hơn để save được 10647 bounding boxes như trong kiến trúc này.
* Không thể thiết lập các batch\_size quá lớn như trong các mô hình classification vì rất dễ Out of memory. Package darknet của YOLO đã chia nhỏ một batch thành các subdivisions cho vừa với RAM.
* Thời gian xử lý của một step trên YOLO lâu hơn rất rất nhiều lần so với các mô hình classification. Do đó nên thiết lập steps giới hạn huấn luyện cho YOLO nhỏ. Đối với các tác vụ nhận diện dưới 5 classes, dưới 5000 steps là có thể thu được nghiệm tạm chấp nhận được. Các mô hình có nhiều classes hơn có thể tăng số lượng steps theo cấp số nhân tùy bạn.

Để tìm được bounding box cho vật thể, YOLO sẽ cần các anchor box làm cơ sở ước lượng. Những anchor box này sẽ được xác định trước và sẽ bao quanh vật thể một cách tương đối chính xác. Sau này thuật toán regression bounding box sẽ tinh chỉnh lại anchor box để tạo ra bounding box dự đoán cho vật thể. Trong một mô hình YOLO:

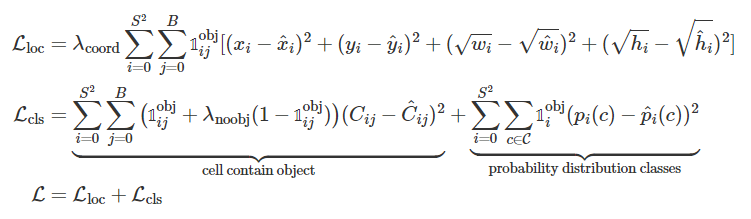
* Mỗi một vật thể trong hình ảnh huấn luyện được phân bố về một anchor box. Trong trường hợp có từ 2 anchor boxes trở lên cùng bao quanh vật thể thì ta sẽ xác định anchor box mà có IoU với ground truth bounding box là cao nhất.



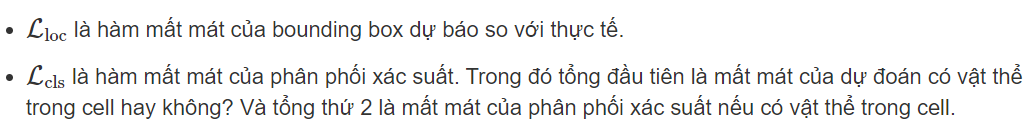
Hình 2. 3: Xác định anchor box cho một vật thể

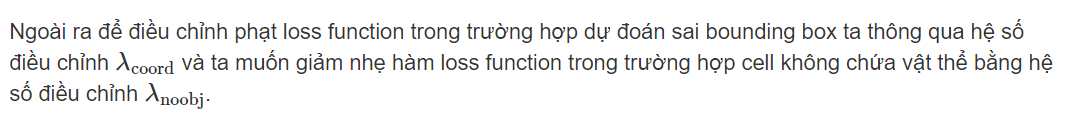
Từ Cell i ta xác định được 3 anchor boxes viền xanh như trong hình. Cả 3 anchor boxes này đều giao nhau với bounding box của vật thể. Tuy nhiên chỉ anchor box có đường viền dày nhất màu xanh được lựa chọn làm anchor box cho vật thể bởi nó có IoU so với ground truth bounding box là cao nhất.

Hàm loss function của YOLO chia làm 2 phần:









Do thuật toán YOLO dự báo ra rất nhiều bounding box trên một bức ảnh nên đối với những cell có vị trí gần nhau, khả năng các khung hình bị overlap là rất cao. Trong trường hợp đó YOLO sẽ cần đến non-max suppression để giảm bớt số lượng các khung hình được sinh ra một cách đáng kể.



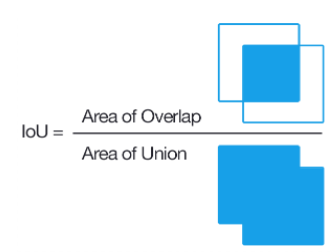
Hình 2. 4: Trường hợp non-max suppression trong YOLO

Từ 3 bounding box ban đầu cùng bao quanh chiếc xe đã giảm xuống còn một bounding box cuối cùng. Các bước của non-max suppression:

* Step 1: Đầu tiên chúng ta sẽ tìm cách giảm bớt số lượng các bounding box bằng cách lọc bỏ toàn bộ những bounding box có xác suất chứa vật thể nhỏ hơn một ngưỡng threshold nào đó, thường là 0.5.
* Step 2: Đối với các bouding box giao nhau, non-max suppression sẽ lựa chọn ra một bounding box có xác xuất chứa vật thể là lớn nhất. Sau đó tính toán chỉ số giao thoa IoU với các bounding box còn lại.

**IoU**: Tỷ lệ Intersection of Union là tỷ lệ đo lường mức độ giao nhau giữa 2 khung hình (thường là khung hình dự báo và khung hình ground truth) để nhằm xác định 2 khung hình overlap không. Tỷ lệ này được tính dựa trên phần diện tích giao nhau gữa 2 khung hình với phần tổng diện tích giao nhau và không giao nhau giữa chúng.

Để tính IOU giữ 2 box chúng ta cần tính diện tích giao nhau giữa 2 box chia cho tổng diện tích của 2 box đó.



Hình 2. 5: Công thức tính tỉ lệ IOU (Intersection of Union)

Mỗi object lại có thế có nhiều boundary box khác nhau do mô hình dự đoán. Để tìm boundary box tốt nhất các object, ta có thể dùng thuật toán non-maximal suppression để loại những boundary box giao nhau nhiều, tức là có IOU giữ 2 boundary box lớn.

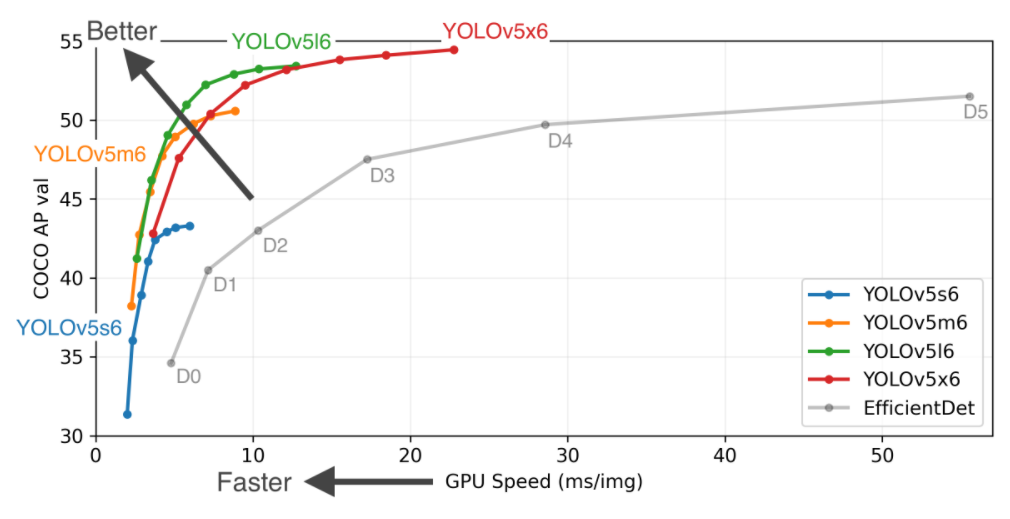
Hàm Loss Function của YOLO bao gồm:

* Độ lỗi của việc dự đoán loại nhãn của đối tượng là hàm lỗi tính trên những ô vuông có xuất hiện object, còn những ô vuông khác ta không quan tâm.
* Độ lỗi của dự đoán tọa độ tâm, chiều dài, rộng của boundary box (x, y, w, h) là hàm lỗi dùng để tính giá trị lỗi cho boundary box được dự đoán bao gồm tọa độ tâm, chiều rộng, chiều cao của so với vị trí thực tế từ dữ liệu huấn luyện của mô hình.
* Độ lỗi của việc dự đoán bounding box đó chứa object so với nhãn thực tế tại ô vuông đó là độ lỗi giữa dự đoán boundary box đó chứa object so với nhãn thực tế tại ô vuông đó. Độ lỗi này tính trên cả những ô vuông chứa object và không chứa object.

### *2.1.3. Quy trình huấn luyện model phát hiện chữ in và biểu mẫu với YOLOv5*

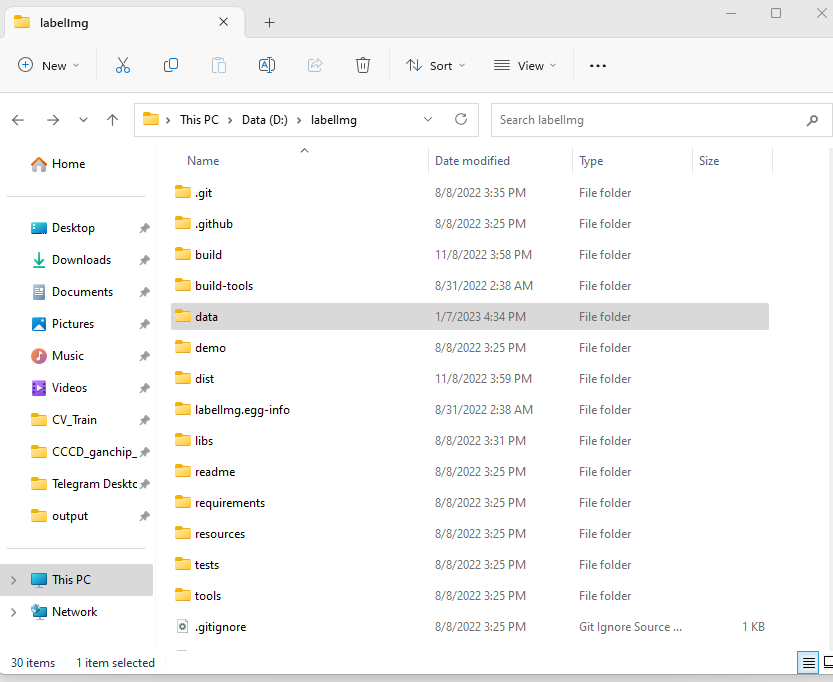
Ở đây tôi sử dụng YOLOv5

YOLOv5 là một họ các mô hình phát hiện đối tượng theo tỷ lệ phức hợp được đào tạo trên tập dữ liệu COCO và bao gồm chức năng đơn giản giúp tăng thời gian thử nghiệm (TTA), tổ hợp mô hình, phát triển siêu tham số và xuất model.



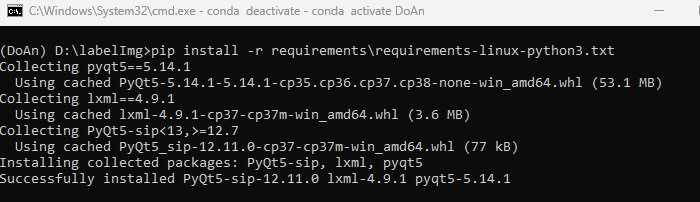
Hình 2. 6: Sơ đồ đánh giá độ chính xác model của YOLOv5

Bước 1: Chuẩn bị và chuẩn hóa dữ liệu

Tải tool LabelImg gán nhãn dữ liệu, qua đường link tải: https://github.com/heartexlabs/labelImg/

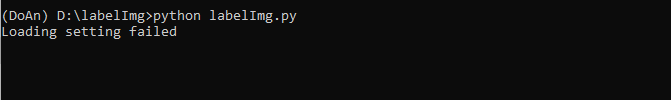
Hình 2. 7: Tải xuống src của tool LabelImg

Mở cmd ở thư mục vừa tải và cài đặt thư viện cần thiết bằng dòng lệnh pip install -r requirements\requirements-linux-python3.txt:



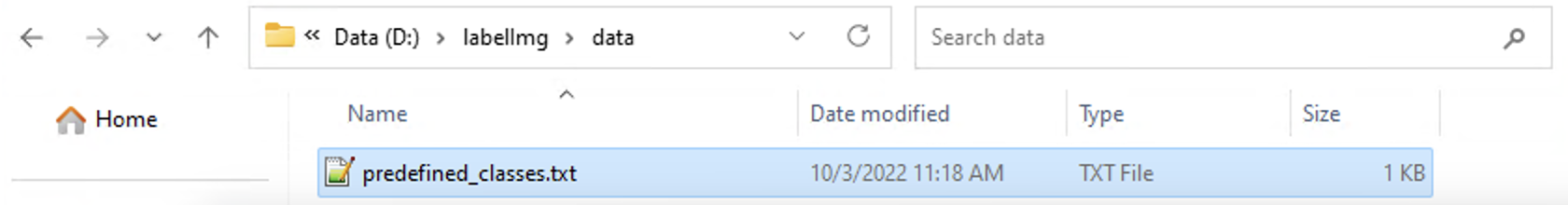
Hình 2. 8: Cài đặt các thư viện python cần thiết cho tool LabelImg

Tiếp tục chạy lệnh python labelImg.py để khởi động tool LabelImg:

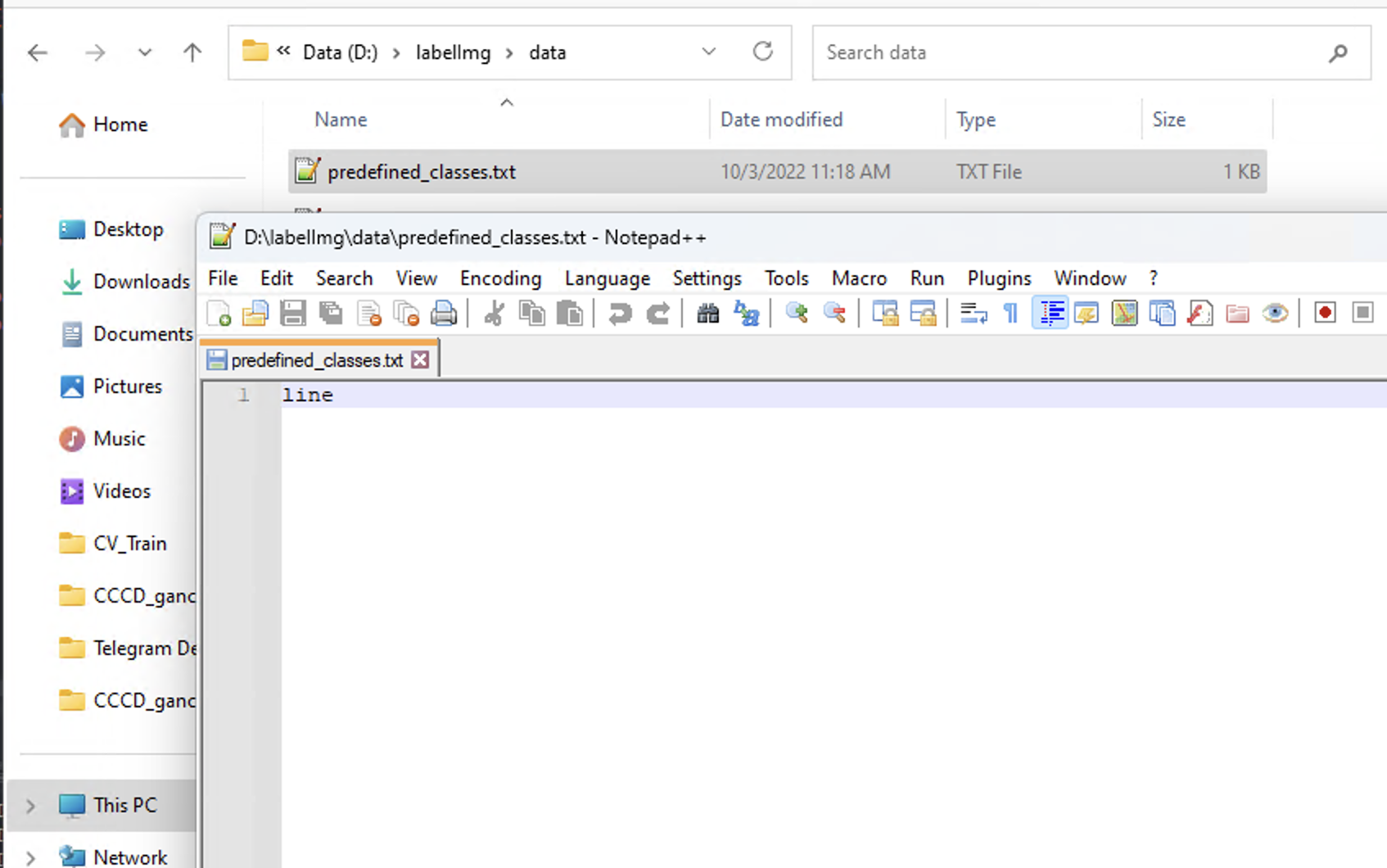


Hình 2. 9: Khởi động tool LabelImg

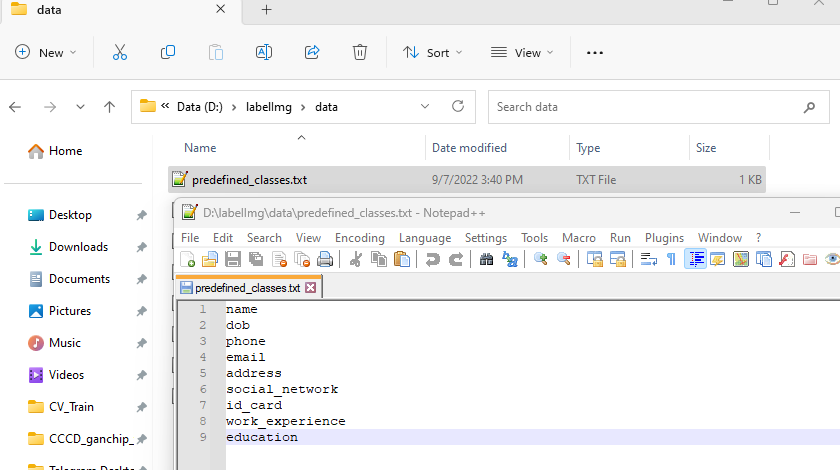
Vào thư mục data, sửa tên nhãn trong file predefined\_classes.txt



Hình 2. 10: File nhãn của tool LabelImg

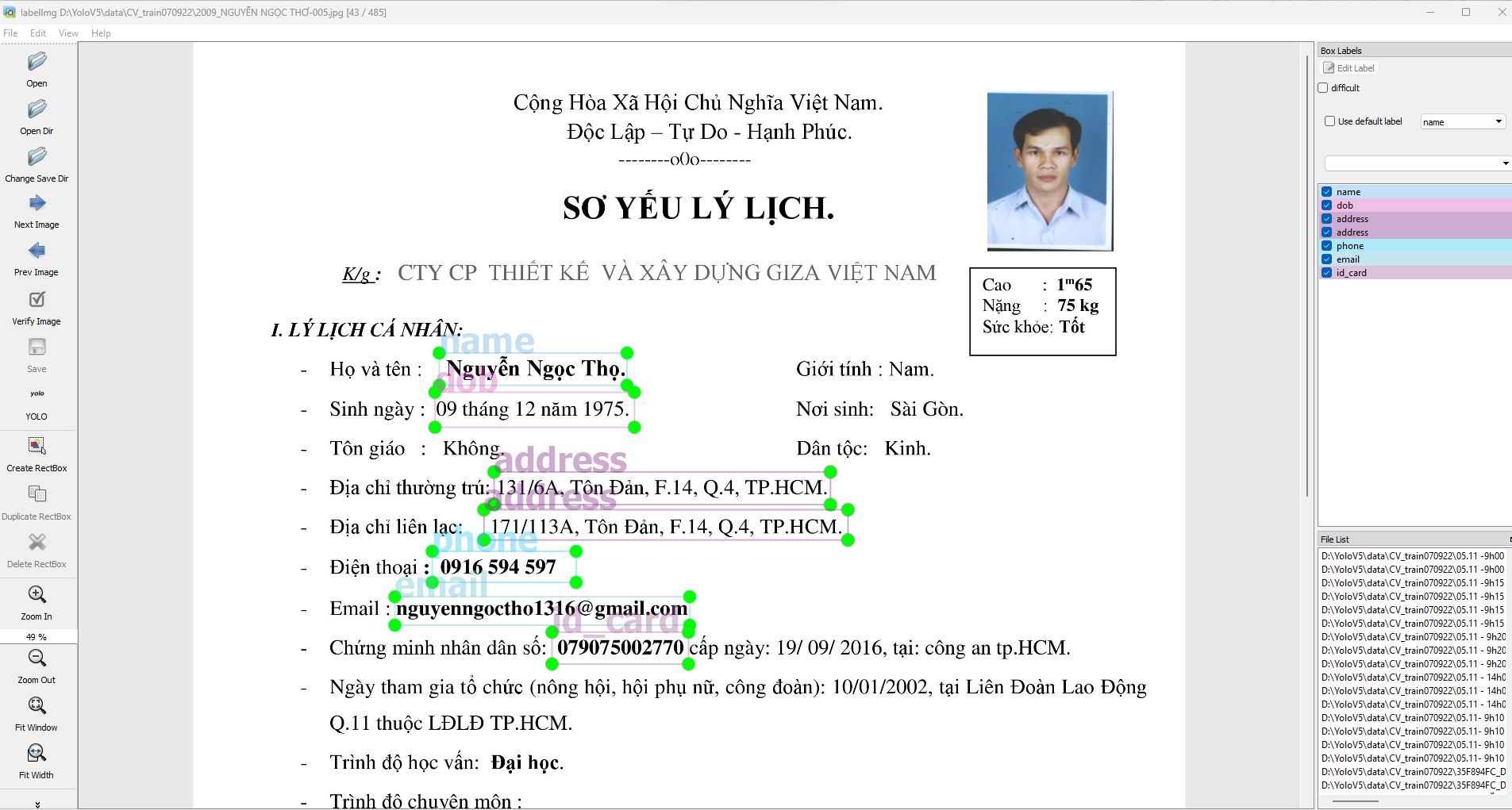


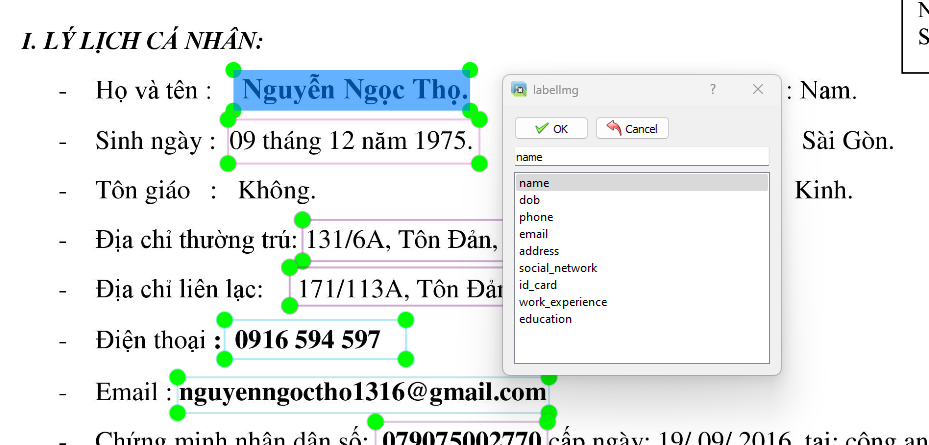
Hình 2. 11: Danh sách đối tượng segment chữ in



Hình 2. 12: Danh sách đối tượng khoanh vùng trong CV

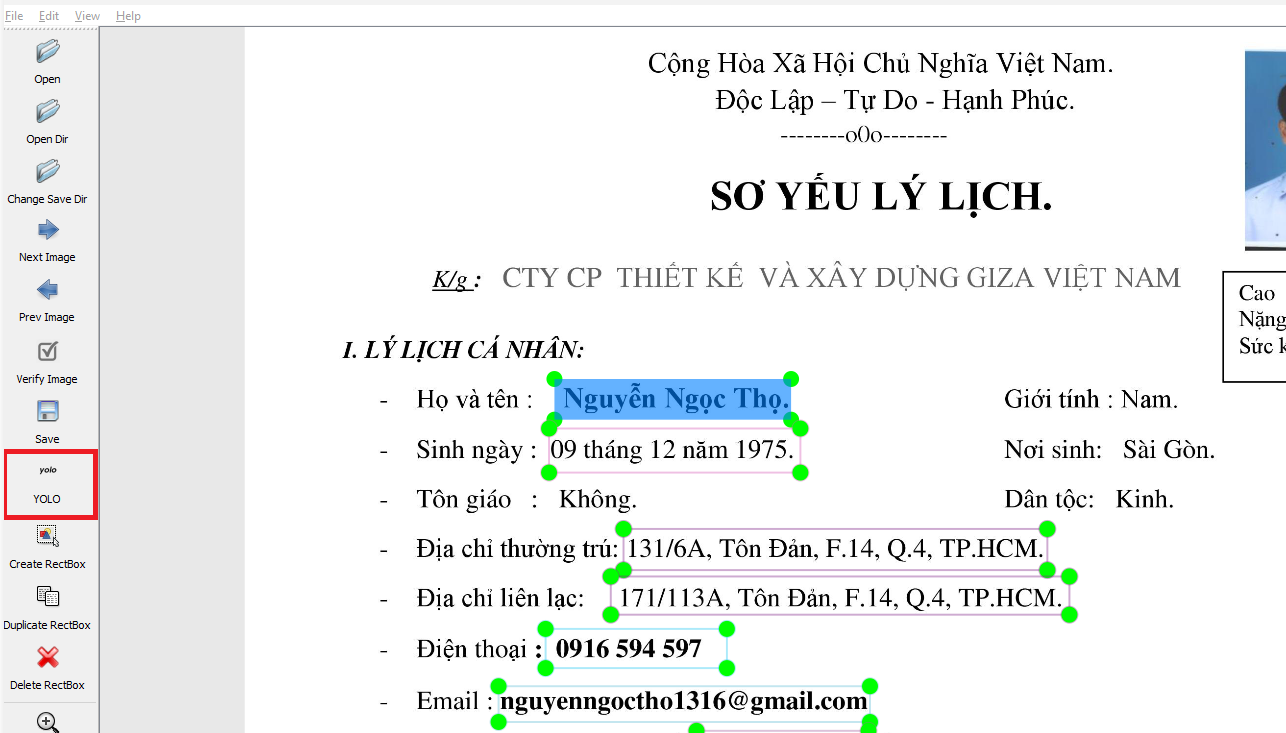
Khởi động giao diện làm việc của labelimg hiện lên ta tiến hành khoanh vùng và gán nhãn cho những vùng đã khoanh. Có thể khéo thả các vùng khoanh cho phù hợp.





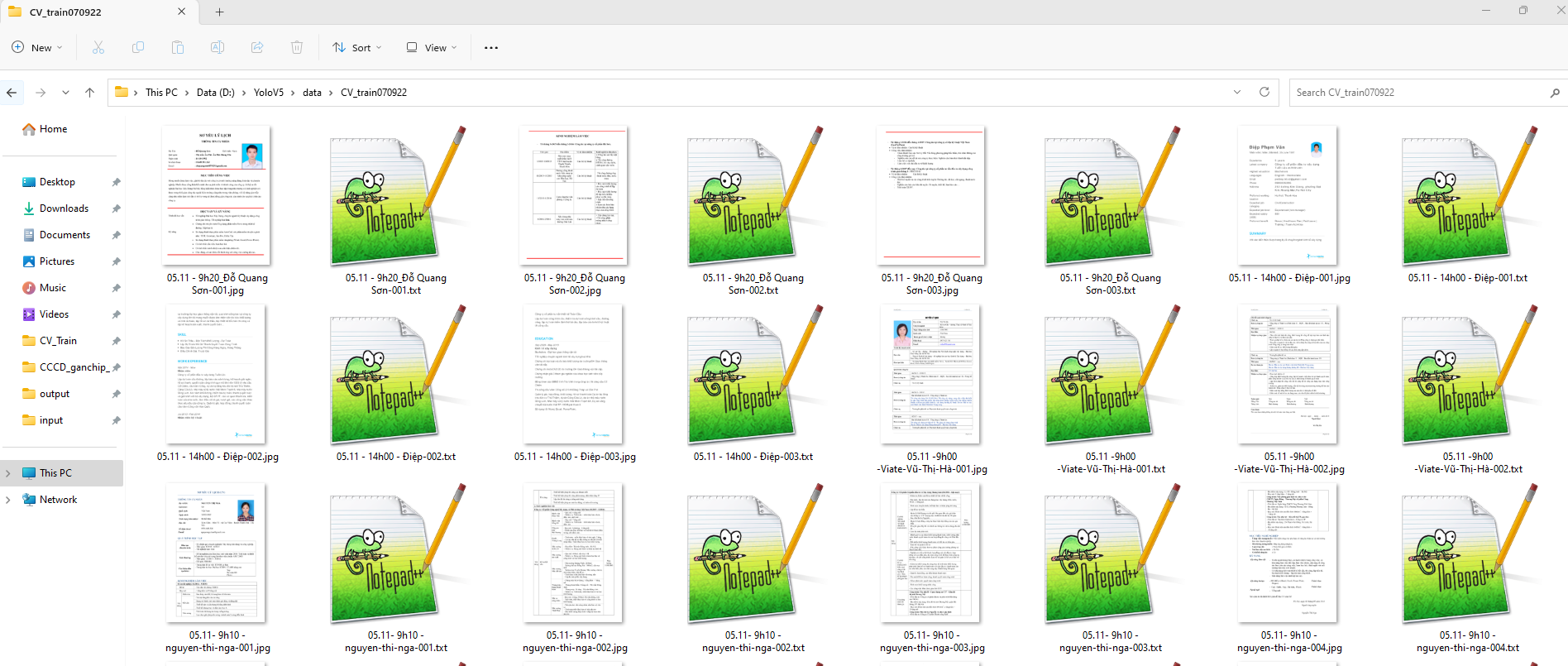
Hình 2. 13: Cách khoanh vùng đối tượng và gán nhãn

Sau khi khoanh vùng và gán nhãn xong ta chọn tính năng YOLO bên góc trái và ấn Save



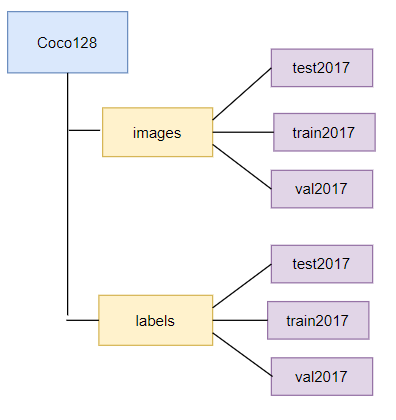
Hình 2. 14: Chế độ lưu nhãn YOLO của tool LabelImg

Ta được một file nhãn đuôi .txt tương ứng



Hình 2. 15: Kết quả sau khi gán nhãn

Bước 2: Tạo các thư mục chứa data

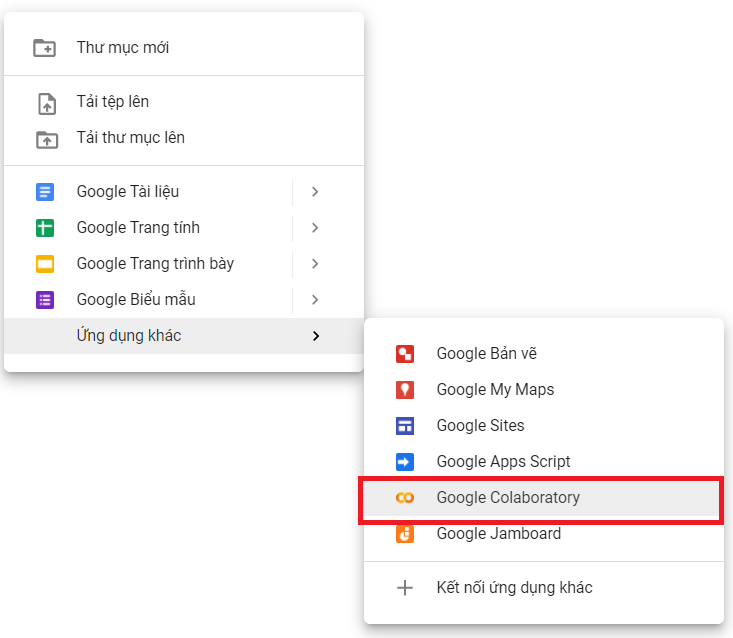


Hình 2. 16: Cấu trúc thư mục chứa data huấn luyện YOLOv5

* Các thư mục con test2017/val2017/train2017 của thư mục images chứa các hình ảnh huấn luyện
* Các thư mục con test2017/val2017/train2017 của thư mục labels chứa các nhãn đuôi .txt tương ứng với từng thư mục hình ảnh huấn luyện

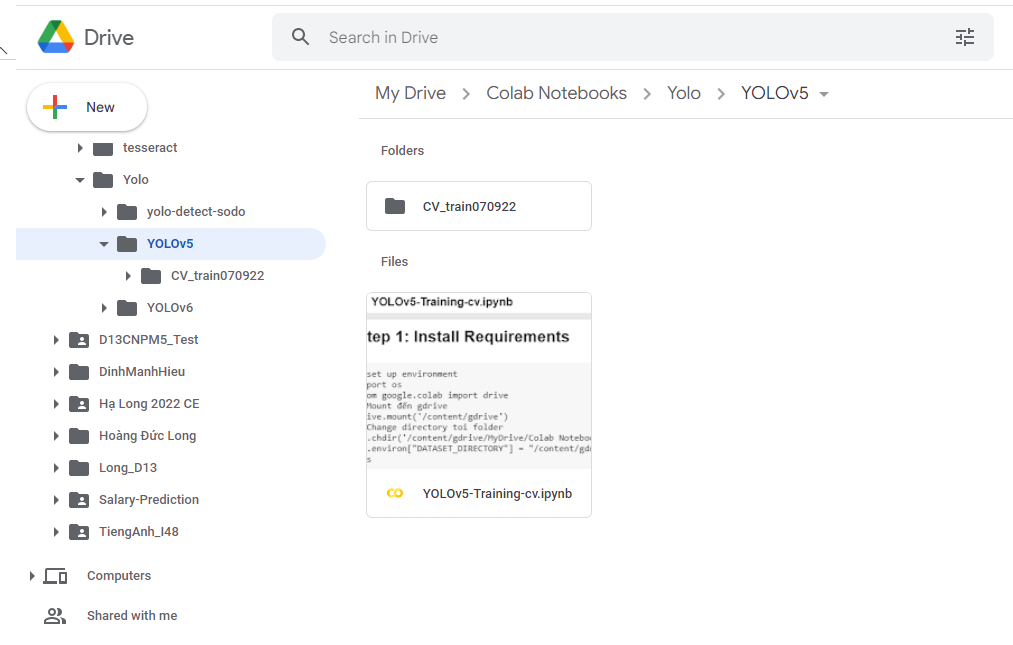
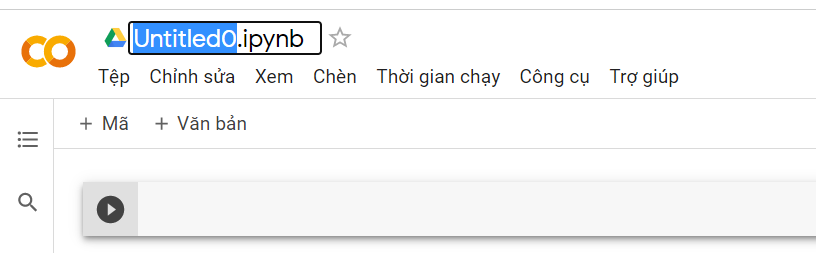
Tải tất cả dữ liệu của thư mục coco128 lên Driver

Bước 3: Huấn luyện model YOLOv5 phát hiện chữ in và vùng thông tin trên biểu mẫu thông tin cá nhân trên Colab



Hình 2. 17: Cách tạo ứng dụng với Colab

Vào Driver, chọn Ứng dụng khác rồi chọn Google Colaboratory. Ta được một file Untitled0.ipynb. Đổi tên Untitled0.ipynb thành YOLOv5-Training-cv.Ipynb ta được:



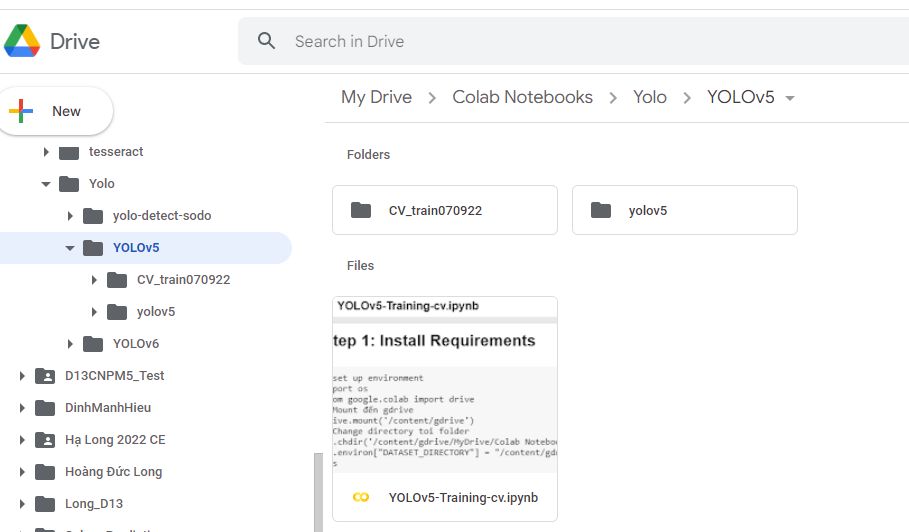
Hình 2. 18: Cấu trúc thư mục train của YOLOv5 với Colab

Thêm các dòng lệnh này và nhấn vào nút mũi tên, dòng lệnh sẽ được thực thi



Hình 2. 19: Cài đặt gói môi trường YOLOv5 và kiểm tra GPU

Sau khi Colab thực thi xong dòng lệnh Driver sẽ có thêm thư mục yolov5.



Hình 2. 20: Kiểm tra các thư mục trên Driver

Ta thêm dòng lệnh này, nhấn nút mũi tên để chạy huấn luyện model phát hiện/định vị biển số các vùng thông tin trong biểu mẫu thông tin cá nhân với YOLOv5



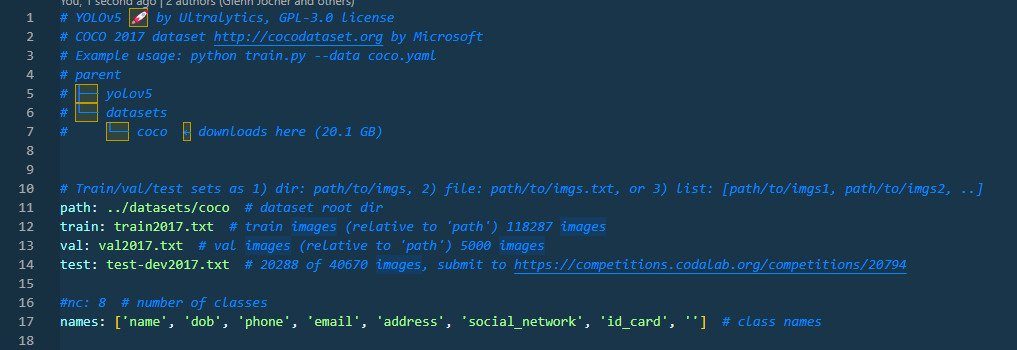
Hình 2. 21: Chạy huấn luyện YOLOv5

Trong đó:

--img là kích thước của dữ liệu huấn luyện

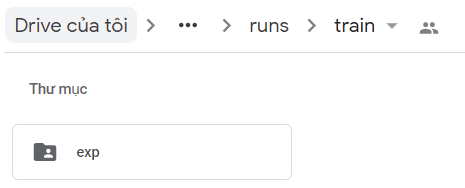
--epochs là số bước huấn luyện

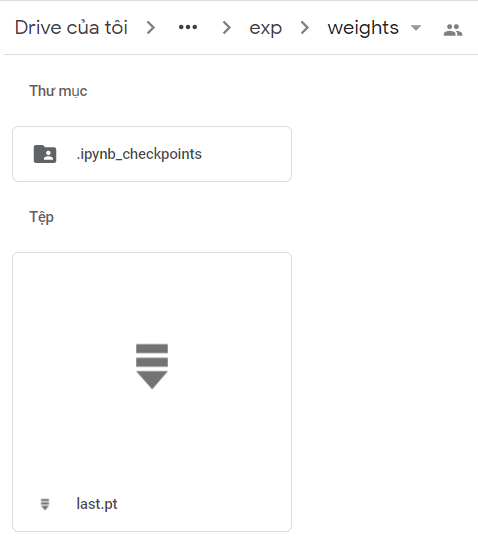
File coco.yaml sẽ có danh sách tên các nhãn đã gán trên tool LabelImg như trong file predefined\_classes.txt



Hình 2. 22: File coco.yaml

Sau khi huấn luyện model được lưu trong thư mục runs/train/exp/weights. Ta được model last.pt và đây là một số hình ảnh đầu ra của model.





Hình 2. 23: Model thu được sau khi huấn luyện YOLOv5



Hình 2. 24: Một số kết quả huấn luyện phát hiện các vùng trong CV YOLOv5

Một số kết quả thử nghiệm mô hình phát hiện/định vị các vùng với YOLOv5 đã huấn luyện





Hình 2. 25: Một số kết quả phát hiện các vùng trong CV với YOLOv5

### *2.1.4. Cài đặt môi trường của YOLOv5*

Cài đặt python >=3.6.5

|  |  |
| --- | --- |
| Các thư viện của python cần cài đặt | |
| matplotlib>=3.2.2 | torch>=1.7.0 |
| numpy>=1.18.5 | torchvision>=0.8.1 |
| opencv-python>=4.1.2 | seaborn>=0.11.0 |
| Pillow | pandas |
| PyYAML>=5.3.1 | thop |
| scipy>=1.4.1 | pycocotools>=2.0 |
| tqdm>=4.41.0 |  |

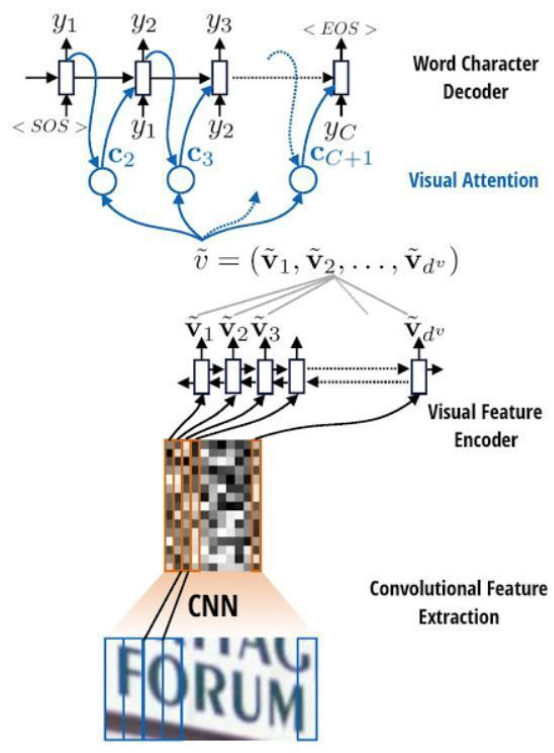
Bảng 2. 1: Cài đặt môi trường của YOLOv5

## 2.2. Xây dựng module nhận dạng chữ in.

Nhận dạng ký tự quang học (OCR hay Optical Character Recognition) đề cập đến một tập hợp các vấn đề về thị giác máy tính. Dùng để chuyển đổi hình ảnh của hình ảnh văn bản viết tay hoặc kỹ thuật số thành văn bản có thể đọc được bằng máy ở dạng máy tính có thể xử lý được, lưu trữ và chỉnh sửa dưới dạng tệp văn bản hoặc một phần của một phần mềm thao tác và nhập dữ liệu.2.2.1. Mô hình nhận dạng Attention-OCR.

### *2.2.1. Mô hình nhận dạng Attention-OCR*

Attention-OCR là một kỹ thuật OCR (Optical Character Recognition) có sẵn trên tensorflow.



Hình 2. 26: Mô hình mạng của kỹ thuật Attention-OCR

Đầu tiên nó sử dụng các lớp mạng phức hợp để trích xuất các tính năng hình ảnh được mã hóa. Các tính năng được trích xuất này sau đó được mã hóa thành chuỗi và chuyển qua mạng lặp lại để cơ chế chú ý xử lý.

Cơ chế này được sử dụng trong quá trình thực hiện được mượn từ mô hình dịch máy Seq2Seq. Cuối cùng là sử dụng bộ giải mã của nó để dự đoán văn bản trong hình ảnh.

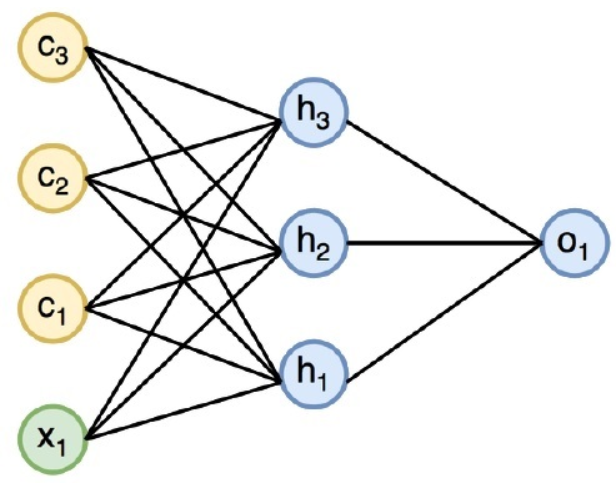
Attention-OCR sẽ sử dụng những tầng CNN (Convolutional Neural Network) để trích xuất các đặc trưng của bức ảnh. Sau đó sẽ dùng một Attention-mask để biến đổi feature maps thành một vector đầu vào của mạng RNN (Recurrent Neural Network) hoặc LSTM (Long Short-Term Memory)



Hình 2. 27: Cấu trúc mạng CNN (Convolutional Neural Network)

Mạng RNN (Recurrent Neural Network) sẽ chuyển đổi feature maps từ tầng CNN (Convolutional Neural Network) thành một vector đầu vào giống như trong Language Model.

Recurrent Neural Network (RNN) là một loại thuật toán định hướng học sâu theo cách tiếp cận tuần tự. Trong mạng nơ-ron, ta luôn giả định rằng mỗi đầu vào và đầu ra là độc lập với tất cả các lớp khác. Loại mạng nơ-ron này được gọi là mạng lặp lại vì chúng thực hiện các phép tính toán học một cách tuần tự hoàn thành nhiệm vụ này đến tác vụ khác.



Hình 2. 28: Mô hình hoạt động của mang Recurrent Neural Network

Trong hình trên, c1, c2, c3 và x1 được coi là đầu vào bao gồm một số giá trị đầu vào ẩn cụ thể là h1, h2 và h3 cung cấp đầu ra tương ứng là o1. Bây giờ ta sẽ tập trung vào việc triển khai PyTorch để tạo ra một sóng sin với sự trợ giúp của các mạng RNN.

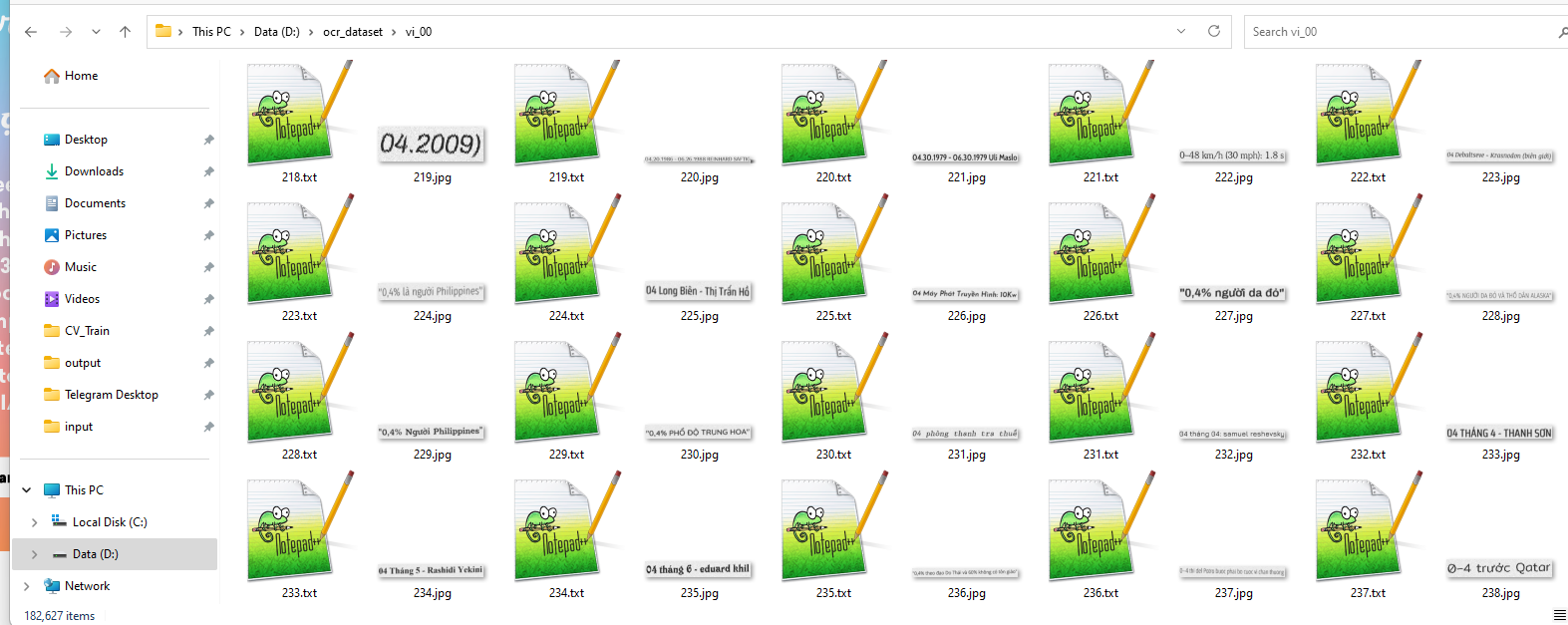
Để đào tạo một mô hình cần có tập hợp các hình ảnh của các dòng chữ in hoặc các vùng thông tin cùng với nhãn của chúng - văn bản có trong các vùng chữ in và tọa độ hộp giới hạn của các vùng đó.

Mạng RNN (Recurrent Neural Network) được ứng dụng nhiều với các bài toán như:

* Chuyển đổi giọng nói sang dạng mà máy tính có thể hiểu được
* Dự đoán đột quỵ
* Nhận diện hành động trong video

Các bước tiến hành:

* Thu thập dữ liệu từ trên mạng. Lấy phần cắt cho từng khung hình của mỗi văn bản.



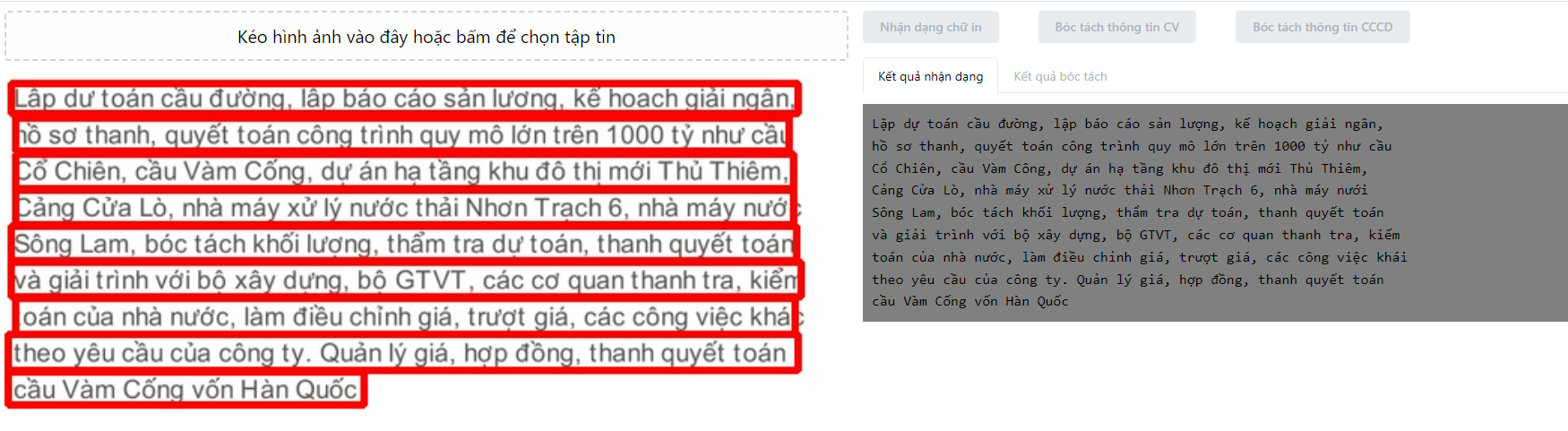
Hình 2. 29: Dữ liệu chữ in

* Huấn luyện mô hình bằng Attention OCR.



Hình 2. 30: Mô hình huấn luyện của Attention OCR

Một số kết quả thử nghiệm nhận dạng:



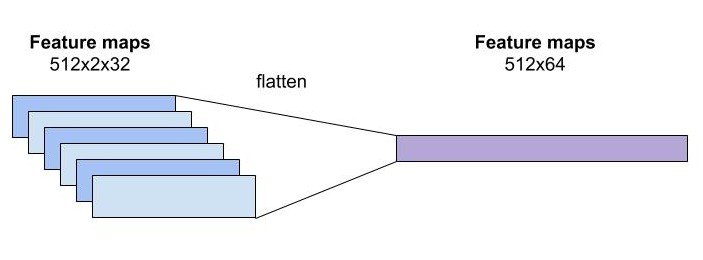
Hình 2. 31: Một số kết quả thử nghiệm nhận dạng của mô hình Attention OCR

### *2.2.2. Mô hình nhận dạng VietOCR*

Mô hình VietOCR là sự kết hợp giữa mô hình CNN (Convolutional Neural Network) và Transformer. Mạng nơ-ron liên kết (ConvNet/CNN) là một thuật toán Deep Learning (học sâu) có thể lấy hình ảnh đầu vào, gán tầm quan trọng (trọng số và thành kiến ​​có thể học được) cho các khía cạnh/đối tượng khác nhau trong hình ảnh và có thể phân biệt hình ảnh này với hình ảnh kia.

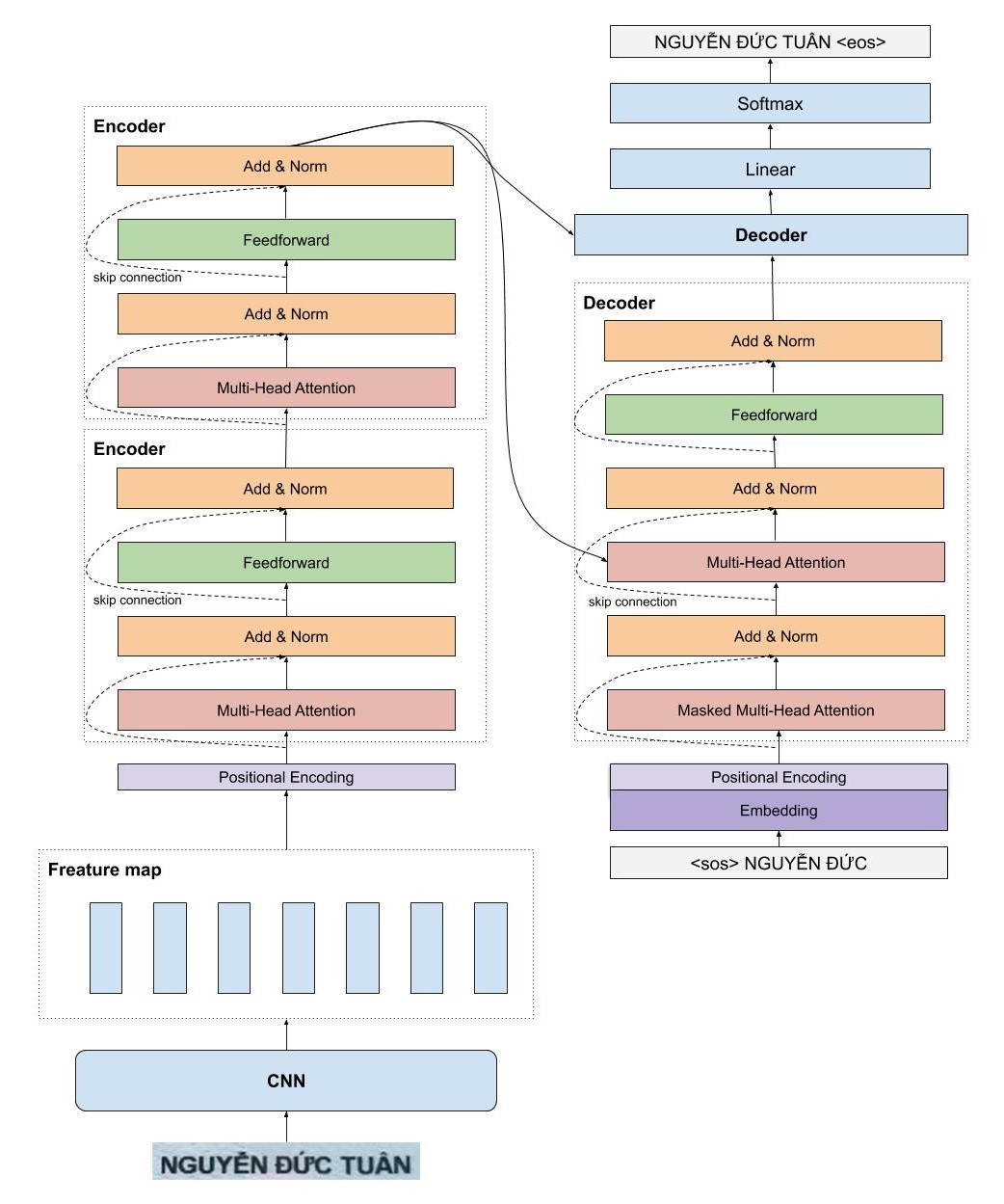
Một ảnh qua mô hình CNN (Convolutional Neural Network), sẽ cho một feature maps có kích thước channel (thang độ xám) x height (chiều cao) x width (chiều rộng), feature maps này sẽ trở thành đầu vào cho mô hình LSTM (Long Short-Term Memory), tuy nhiên, mô hình LSTM (Long Short-Term Memory) chỉ nhận chỉ nhận đầu vào có kích thước là hiddenxtime\_step. Một cách đơn giản và hợp lý là 2 chiều cuối cùng height (chiều cao) x width (chiều rộng) của feature maps sẽ được duổi thẳng.

Feature maps lúc này sẽ có kích thước phù hợp với yêu cầu của mô hình LSTM (Long Short-Term Memory).



Hình 2. 32: Các feature maps trong mô hình VietOCR

VietOCR là phần mềm nhận dạng văn bản tiếng Việt (Phần mềm OCR tiếng Việt, Phần mềm nhận dạng chữ viết tay tiếng Việt, Phần mềm quét chữ trên ảnh cho PC) từ các hình ảnh (bằng cách chụp màn hình hoặc các hình ảnh có chứa văn bản tiếng Việt). Được phát triển dựa trên mô hình Transfomer dành riêng cho tiếng Việt.

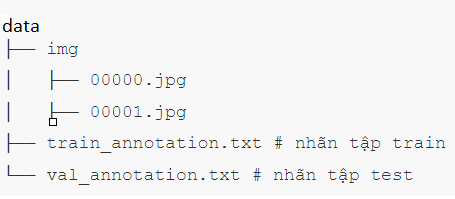


Hình 2. 33: Mô hình nhận dạng VietOCR

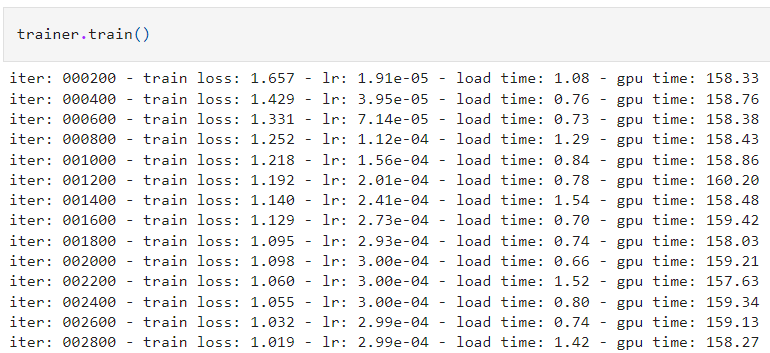
Để cài đặt môi trường, gõ lệnh sau:

pip install vietocr==0.3.5

Cấu trúc thư mục dữ liệu huấn luyện của VietOCR



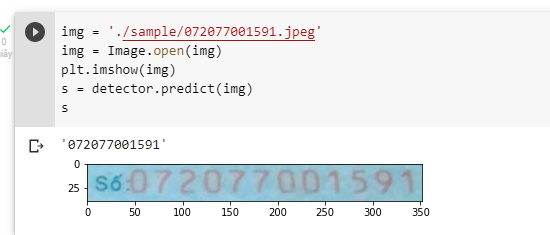
Hình 2. 34: Cấu trúc tập dữ liệu huấn luyện của mô hình VietOCR



Hình 2. 35: Mô hình huấn luyện của VietOCR

Một số kết quả thử nghiệm nhận dạng chữ in của mô hình VietOCR





Hình 2. 36: Một số kết quả nhận dạng của mô hình VietOCR

* Mô hình khá nhạy cảm với sự thay đổi nhỏ của ảnh đầu vào khi sử dụng pretrained model trên tập dữ liệu mới chưa được huấn luyện.
* Mô hình VietOCR không mang lại cải tiến vượt bậc trong bài toán OCR (Optical Character Recognition). Do đó sử dụng language model quá tốt cũng không mang lại hiệu quả.

### *2.2.3. Đánh giá kết quả nhận dạng của mô hình VietOCR và Attention-OCR*

Huấn luyện mô hình Attention-OCR hay VietOCR hoàn toàn giống nhau đều sử dụng cross-entropy loss để tối ưu thay vì sử dụng CTCLoss (Connectionist Temporal Classification Loss) như mô hình CRNN (Convolutional Recurrent Neural Network), tức là tại mỗi thời điểm mô hình dự đoán một từ sau đó so sánh với nhãn để tính loss và cập nhật lại trọng số của mô hình.

|  |  |  |
| --- | --- | --- |
|  | Mô hình nhận dạng VietOCR | Mô hình nhận dạng Attention-OCR |
| Tốc độ | 4-5 giây | 1 giây |
| Khả năng nhận dạng | Phù hợp với nhận dạng các dòng không rõ nét, nền màu thường nhận dạng kém. | Nhận dạng tốt trên cả ảnh màu. |
| Cài đặt môi trường | Môi trường ít hỗ trợ cho các bản python dưới 3.5 và tensorflow <1.15 | Chưa hỗ trợ bản tensorflow trên 1.15 |

Bảng 2. 2: Đánh giá kết quả thử nghiệm nhận dạng

### *2.2.4. Ứng dụng của công nghệ nhận dạng chữ in*

Số hóa nhanh chóng và các tài liệu nâng cao đã hỗ trợ đa chiều cho các ứng dụng của Công nghệ nhân dạng chữ in. Ngoài việc tìm kiếm sử dụng trong thực thi pháp luật, nó cũng đã tìm thấy các ứng dụng thành công trong các lĩnh vực sau:

* Hỗ trợ cuộc sống của người già, người khiếm thị: Phần mềm OCR có khả năng nhận dạng phông chữ này. Ngay lập tức, công nghệ OCR đã được tích hợp với công nghệ tổng hợp giọng nói (giọng máy), giúp máy có khả năng đọc hiểu văn bản. Nói cách khác, văn bản không chỉ được phần mềm OCR giải mã, mà còn được công cụ tổng hợp giọng nói đọc ra thành tiếng. Giọng nói vi tính hóa đã được ứng dụng vào việc đọc văn bản trong sách, báo, tạp chí cho người cao tuổi, người khiếm thị, giúp cuộc sống của họ trở nên nhẹ nhàng hơn.
* Bảo tồn các văn bản có giá trị: Các thư viện cổ, trung tâm văn hóa lịch sử hay viện bảo tàng… là những nơi lưu trữ rất nhiều các bản thảo, tài liệu, hồi kí… Quá trình cất giữ và bảo quản những tài liệu này không hề đơn giản. Chúng rất dễ bị mối mọt và hư hại theo thời gian. Việc nhập liệu thủ công chính xác và đầy đủ lượng văn bản khổng lồ đấy là điều vô cùng khó khăn và vất vả, có thể mất tới hàng chục năm. Tuy nhiên, công nghệ OCR ra đời đã giúp nhiều tổ chức giải quyết bài toán đó một cách đơn giản hơn. Các văn bản, tài liệu quan trọng được chuyển đổi từ dạng giấy sang file mềm, giúp việc lưu trữ và bảo tồn nhiều di sản văn học trở nên dễ dàng hơn.
* Nhận dạng cá nhân: Khi đăng kí mở tài khoản ngân hàng, mở thẻ thành viên hay bất kì một hành động nào cần phải xác minh danh tính, bạn đều phải cung cấp thông tin cá nhân đúng và đầy đủ với các bên đối tác. Quá trình kê tờ khai mất kha khá thời gian của bạn và bên làm dịch vụ. Đôi khi, nhập liệu sai dẫn đến những rắc rối không đáng có về sau.

## 2.3. Sơ đồ quy trình huấn luyện phát hiện, nhận dạng chữ in và biểu mẫu



Hình 2. 37: Sơ đồ quy trình huấn luyện phát hiện, nhận dạng chữ in và biểu mẫu

# **CHƯƠNG 3: KẾT LUẬN**

## 3.1. Kết quả đạt được

Trong quá trình nghiên cứu và xây dựng hệ thống nhận dạng chữ in đã đạt được một số kết quả sau:

* Nắm bắt được nghiệp vụ cũng như cách triển khai một ứng dụng thực tế.
* Bước đầu xây dựng được module phát hiện và nhận dạng chư xin.
* Cách kết nối giữa web với module nhận dạng chữ in bằng API..

## 3.2. Hướng phát triển

Với công nghệ tiến bộ với tốc độ chóng mặt, các thành phần của hệ thống tự động nhận dạng chữ in đã được cải thiện rất nhiều. Từ cảm biến hình ảnh có độ phân giải cao và khả năng xử lý dữ liệu nhanh hơn, độ chính xác phát hiện của hệ thống nhận dạng chữ in đã được cải thiện đáng kể.

* Module phát hiện và nhận dạng chữ in cần được tiếp tục cải thiện độ chính xác
* Có thể phát triển hệ thống nhận dạng chữ in lên hệ thống số hóa tài liệu và biểu mẫu có sẵn hoặc hệ thống định danh cá nhân.
* Tích hợp nhận dạng và gọi API tự động, đóng gói được chương trình.
* Hệ thống quản lý nhận dạng thân thiện với người dùng.
* Ứng dụng hệ thống nhận dạng chữ in vào thực tiễn.

# **TÀI LIỆU THAM KHẢO**

[1]. TS. Đào Nam Anh, Giáo trình Phân tích thiết kế hướng đối tượng, trường Đại học Điện Lực.

[2]. Joseph Redmon, Optimal Speed and Accuracy of Object Detection, năm 2020

[3]. T. He, W. Huang, Y. Qiao, and J. Yao, “Text-attentional convolutional neural network for scene text detection,” IEEE Transactions on Image Processing, vol. 25, no. 6, pp. 2529–2541, 2016.

[4]. K. Wang, B. Babenko, and S. Belongie, “End-to-end scene text recognition,” in 2011 International Conference on Computer Vision. IEEE, 2011, pp. 1457–1464.

[5]. M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, “Reading text in the wild with convolutional neural networks,” International Journal of Computer Vision, vol. 116, no. 1, pp. 1–20, 2016.