

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**  
**KHOA KHOA HỌC MÁY TÍNH**



**BÁO CÁO ĐỒ ÁN CUỐI KÌ**

**SỐ HÓA TỬ SÁCH**

**Giảng viên hướng dẫn:**

**TS. Nguyễn Đình Duy**

**ThS. Phạm Nguyễn Trường An**

**Sinh viên thực hiện:**

**18520767 – Võ Kiều Hoa**

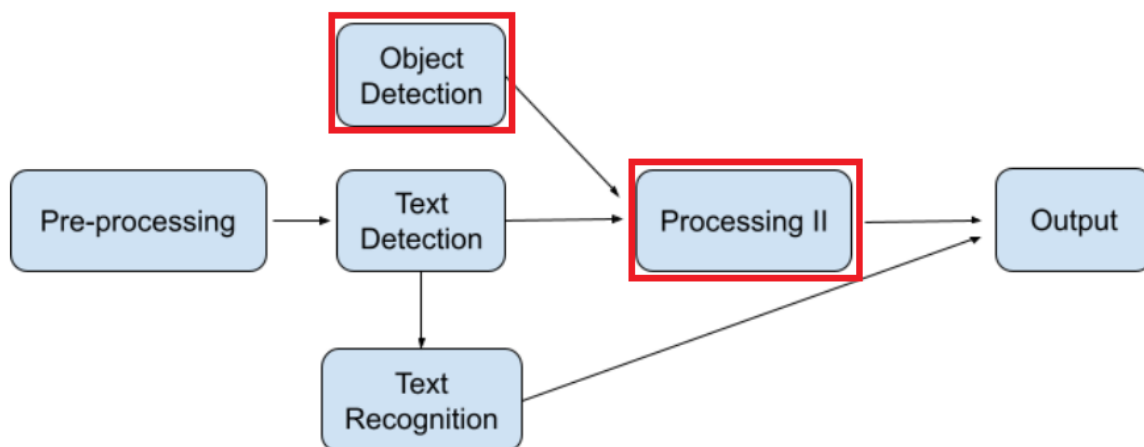
**18521655 – Nguyễn Xuân Vinh**

***TP. Hồ Chí Minh, tháng 01 năm 2022***

## MỤC LỤC

|  |    |
|--|----|
| <b>Giải trình chỉnh sửa sau vấn đáp</b>  | 3  |
| <b>Giới thiệu đề án</b>  | 4  |
| <b>Xây dựng bộ dữ liệu</b>   | 4  |
| Thu thập dữ liệu   | 4  |
| Gán nhãn dữ liệu   | 4  |
| Chi tiết bộ dữ liệu  | 6  |
| <b>Thực nghiệm</b>   | 8  |
| Tổng quan  | 8  |
| Các bước thực hiện   | 8  |
| Tiền xử lý dữ liệu (Pre-processing)  | 8  |
| Phát hiện đối tượng trên bìa sách (Object detection)   | 10 |
| Phát hiện văn bản (Text Detection)   | 13 |
| Nhận dạng văn bản(Text Recognition)  | 17 |
| Xử lý (Processing II): Phân loại văn bản thuộc các lớp đối tượng và chuẩn hóa văn bản đầu ra | 19 |
| <b>Kết quả đánh giá và phân tích lỗi</b>   | 23 |
| Độ đo đánh giá   | 23 |
| Kết quả đánh giá và nhận xét   | 27 |
| <b>Kết luận và hướng phát triển</b>  | 32 |
| <b>TÀI LIỆU THAM KHẢO</b>  | 33 |

## I. Giải trình chỉnh sửa sau vấn đáp



Hình 5. Những thay đổi sau vấn đáp

Ở buổi vấn đáp, chúng em chỉ dừng lại ở việc xây dựng bộ dữ liệu và thực nghiệm các mô hình trên các task: Preprocessing(Detectron2), Text Detection (EasyOCR, PaddleOCR), Text Recognition(EasyOCR, TransformerOCR - VietOCR). Chúng em chỉ detect được những vùng chứa văn bản có trong ảnh và recognize được nội dung text chứa trong các vùng đó.

Sau khi vấn đáp, chúng em đã thực hiện tên task Object Detection được mô tả ở mục [Phát hiện đối tượng trên bìa sách \(Object detection\)](#) và task ProcessingII được mô tả ở mục [Xử lý \(Processing II\): Phân loại văn bản thuộc các lớp đối tượng và chuẩn hóa văn bản đầu ra](#). (Hình 5, vùng được khoanh đỏ)

Ở task Object Detection, chúng em sử dụng mô hình Yolov4 để huấn luyện trên dữ liệu của chúng em. Mục đích của việc đưa thêm task này vào quy trình là để phân loại được các vùng chứa các trường dữ liệu như: tác giả, tên sách và nhà xuất bản.

Ở task ProcessingII, sau khi detect được các vùng chứa văn bản thuộc 3 trường dữ liệu trên, chúng em dùng các kỹ thuật thực hiện phân loại văn bản thuộc các lớp đối tượng đó và chuẩn hóa đầu ra để đưa ra output hoàn thiện nhất.

Chúng em đã bổ sung giải trình về độ đo WER ở mục [WER](#) và các độ đo khác đã sử dụng ở mục [Độ đo đánh giá](#).

Ngoài ra chúng em đã trình bày chi tiết tất cả các nội dung đã làm mà hôm báo cáo không thể nói hết được, mong thầy dành chút thời gian để xem kỹ ạ.

## II. Giới thiệu đề án

Các bìa sách thường mang lại các thông tin liên quan đến tên sách, tác giả hay nhà xuất bản. Việc trích xuất các thông tin trên bìa sách là một điều cần thiết cho một hệ thống tử sách được số hóa tự động để tiết kiệm được thời gian, chi phí và công sức. Optical Character Recognition (OCR) là một công nghệ phục vụ cho việc nhận diện chữ viết, chữ đánh máy trong ảnh một cách tự động và có thể giải quyết bài toán này một cách hiệu quả. Ngày nay OCR được áp dụng đa số vào các bài toán số hóa văn bản, nhận dạng biển số xe, trích xuất thông tin hóa đơn, ngân hàng,... Trong đề án này chúng em thực nghiệm các phương pháp nhận dạng văn bản Tiếng Việt từ ảnh các bìa sách tự chụp.

Bài toán có đầu vào là ảnh chụp một bìa sách và đầu ra là các thông tin bao gồm: tên sách, tên tác giả, nhà xuất bản. Chúng em đã xây dựng được bộ dữ liệu ảnh chụp bìa sách gồm 979 ảnh và thực nghiệm trên nhiều phương pháp, nhiều mô hình để tìm ra phương pháp tối ưu nhất cho bài toán đã đề ra.

## III. Xây dựng bộ dữ liệu

### 1. Thu thập dữ liệu

Để dữ liệu phù hợp với tính ứng dụng đã đặt ra ban đầu, yêu cầu của dữ liệu bìa sách phải là ảnh chụp, không phải ảnh scan, bìa thiết kế,... Chính vì thế chúng em chỉ thu thập ảnh chụp. Bộ dữ liệu được thu thập bằng hai cách: thu thập dữ liệu ảnh trên google (10 ảnh) và chụp thủ công.

Các ảnh được chụp bằng nhiều thiết bị điện thoại khác nhau, đa số đều được chụp ở góc chính diện, bìa sách được tập trung trọn khung hình, tuy nhiên vẫn có những ảnh chụp ở các góc độ khác nhau. Ngoài ra chúng em cũng thu thập với nhiều thể loại khác nhau như sách giáo khoa, sách văn học, sách tham khảo, truyện,... để tăng tính đa dạng và phong phú cho bộ dữ liệu.

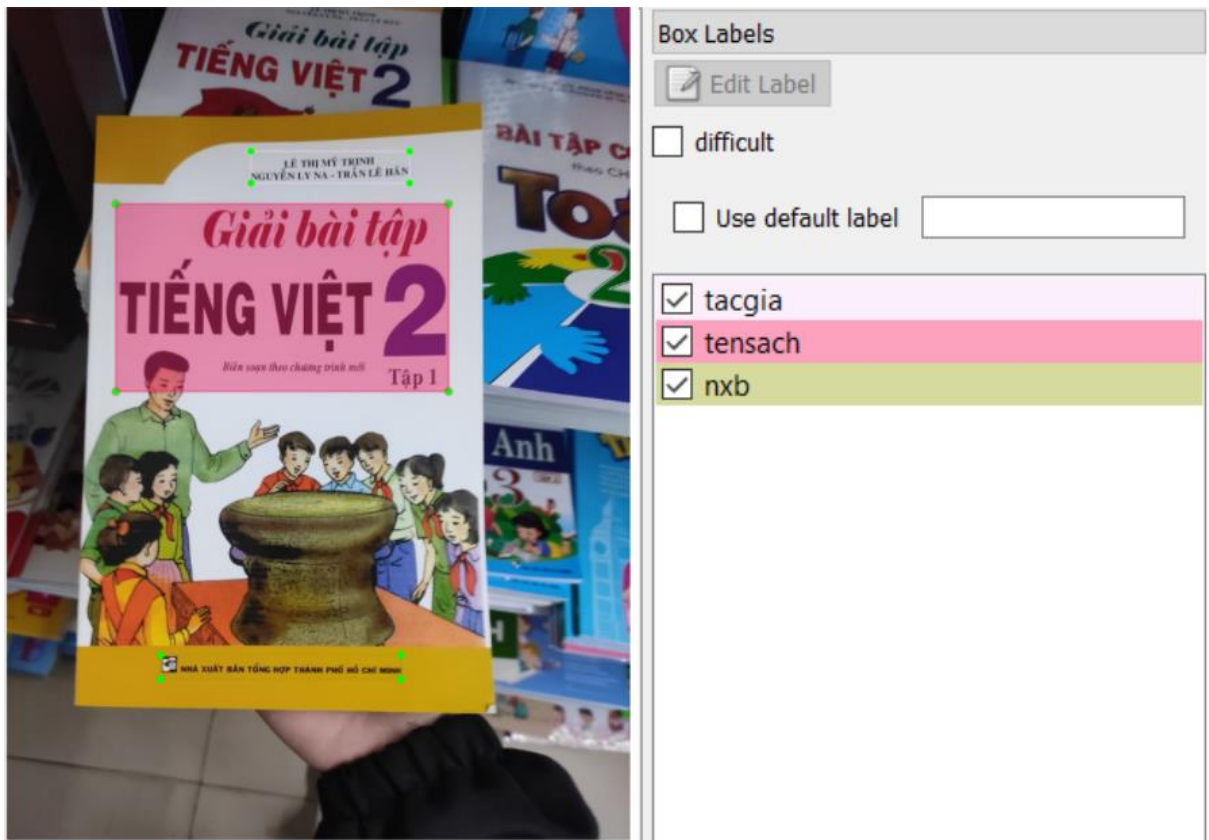
Chúng em đã thu thập hơn 1000 ảnh tuy nhiên sau quá trình kiểm tra lại và loại bỏ nhiều ảnh bị nhiễu, mờ có thể ảnh hưởng xấu đến chất lượng mô hình. Sau cùng chúng em có được bộ dữ liệu với 979 ảnh.

### 2. Gán nhãn dữ liệu

Ở đây, chúng em chuẩn bị dữ liệu huấn luyện cho 2 task là Object Detection và Text Detection.

- **Object Detection**

Để chuẩn bị dữ liệu cho việc huấn luyện ở task này, chúng em sử dụng [LabelImg](#) - một công cụ gán nhãn cho task object detection của Yolo. Các file nhãn cho mô hình YOLOv4 sẽ có định dạng .txt, trong đó có một file classes.txt gồm 3 nhãn sẽ huấn luyện là tác giả (0), tên sách (1) và nhà xuất bản (2). Các file còn lại sẽ là tên các ảnh và đuôi file nhãn sẽ là .txt. Ví dụ ảnh photo1.jpg sẽ có file nhãn là photo1.txt. Trong file nhãn này sẽ chứa thứ tự của nhãn được gán trong file classes.txt, 4 con số thập phân liên qua đến thông tin của bounding box. Hình 1 mô phỏng gán nhãn bằng [LabelImg](#).



Hình 1. Minh họa gán nhãn YOLOv4 bằng labelImg tool

- **Text Detection**

Để chuẩn bị dữ liệu huấn luyện cho task Text Detection, chúng em sử dụng công cụ gán nhãn [PPOCRLabel](#) để tiến hành gán nhãn. Công cụ [PPOCRLabel](#) được phát triển bởi công ty Baidu (Trung Quốc). [PPOCRLabel](#) là điển hình cho việc gán nhãn dữ liệu ảnh cho OCR task. Công cụ này có hỗ trợ polygon bounding box (một trong những khác biệt với các công cụ gán nhãn dữ liệu ảnh khác hiện có). Polygon bounding box phù hợp cho bộ dữ liệu của chúng em vì văn bản trong ảnh với nhiều định dạng khác nhau, polygon bounding box sẽ linh hoạt hơn với định dạng chữ

ngiên, vòng cung... (so với rectangle bounding box thông thường). Hình 2 minh họa gán nhãn dữ liệu bằng [PPOCRLabel](#).



Hình 2. Minh họa công cụ gán nhãn [PPOCRLabel](#).

### 3. Chi tiết bộ dữ liệu

Bộ dữ liệu ảnh bìa sách của chúng em bao gồm 979 ảnh có định dạng “.jpg”. Chúng em cố gắng tạo bộ dữ liệu đa dạng nhất có thể bằng cách chụp nhiều thể loại sách khác nhau và chụp với nhiều góc khác nhau([Hình 3](#)).



Hình 3. Ảnh minh họa chụp nghiêng trái, chính diện và nghiêng phải

Bộ dữ liệu được chia thành 3 tập train set, validation set và test teset. Chúng em chia dữ liệu ngẫu nhiên với tỉ lệ train:val:test = 8:1:1. Tập train chứa 800 ảnh, tập val chứa 89 ảnh và tập test chứa 90 ảnh. Sơ đồ cây phân bố của bộ dữ liệu được mô tả như [Hình 4](#) dưới đây:

```
+---Train
|   classes.txt
|   photo_283@13-01-2022_22-53-59.jpg
|   photo_283@13-01-2022_22-53-59.txt
|   .....
|   photo_312@13-01-2022_22-54-28.jpg
|   photo_312@13-01-2022_22-54-28.txt
+---Val
|   classes.txt
|   photo_326@13-01-2022_22-54-29.jpg
|   photo_326@13-01-2022_22-54-29.txt
|   .....
|   photo_81@13-01-2022_22-51-36.jpg
|   photo_81@13-01-2022_22-51-36.txt
+---Test
|   classes.txt
|   photo_268@05-01-2022_08-43-53.jpg
|   photo_268@05-01-2022_08-43-53.txt
|   .....
|   photo_175@05-01-2022_08-43-03.jpg
|   photo_175@05-01-2022_08-43-03.txt
+---Label
|   Val_Lavel.txt
|   Test_Lavel.txt
|   Train_Lavel.txt
```

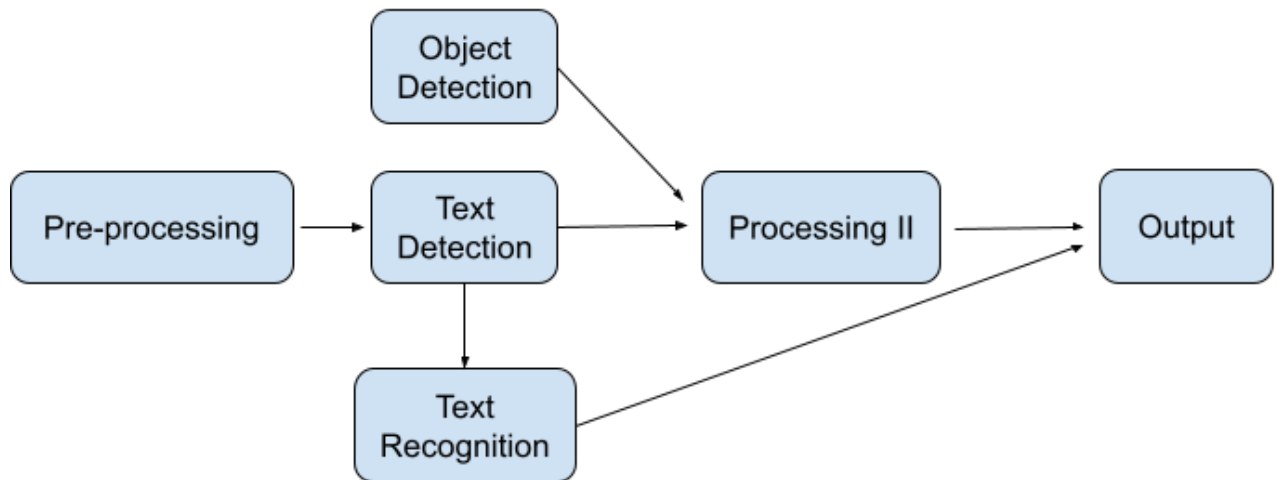
*Hình 4. Sơ đồ cây phân bố của bộ dữ liệu*



## IV. Thực nghiệm

### 1. Tổng quan

Để có thể nhận dạng chữ trên bìa sách và phân loại chúng thành các lớp: Tên tác giả, tên sách, nhà xuất bản. Chúng em thực hiện theo quy trình ở **Hình 6** Chúng em thực nghiệm trên các mô hình được mô tả ở **Bảng 1**.



Hình 6. Tổng quan quy trình thực hiện

|       | Pre-processing | Object Detection | Text Detection | Text Recognition |
|-------|----------------|------------------|----------------|------------------|
| Model | Detectron2     |                  | EasyOCR        | EasyOCR          |
|       |                | Yolov4           | PaddleOCR      | VietOCR          |

Bảng 1. Các mô hình thực nghiệm

### 2. Các bước thực hiện

#### a. Tiền xử lý dữ liệu (Pre-processing)

Pre-processing là giai đoạn tiền xử lý dữ liệu, ở đây chúng em thực hiện xóa background ảnh với mục đích để mô hình không bị nhầm lẫn những vùng chứa văn bản không thuộc nội dung bìa sách dẫn đến kết quả không mong muốn (Hình 7).



Chúng em sử dụng [Detectron2](#) để xóa background. Detectron 2 là một framework cho bài toán Image Segmentation được phát triển bởi nhóm nghiên cứu của Facebook vào năm 2019. Phiên bản Detectron2 được cải tiến từ phiên bản Detectron đã có trước đó. Đặc biệt nó có pre-trained model có sẵn tại [Model Zoo](#). Hơn nữa Model Zoo của Detectron2 lại rất phong phú, được huấn luyện với nhiều mô hình SOTA như Mask R-CN, RetinaNet, Faster R-CNN,...

Cụ thể trong đồ án này, chúng em sử dụng [Detectron2](#) với mô hình **R50-FPN** cho task Image Segmentation. Image Segmentation cũng có chung mục tiêu như Object Detection là phát hiện vùng ảnh chứa vật thể. Tuy nhiên độ chính xác của Image Segmentation ở mức cao hơn (đúng đến từng pixel) so với bounding box của Object Detection. Mặt khác, dữ liệu thu thập được của chúng em có nhiều bìa sách chụp nghiêng, nên việc sử dụng Object Detection là không tối ưu. Mô hình chúng em sử dụng cho kết quả khả quan. Hình 10 minh họa ảnh predict bằng EasyOCR trước và sau khi xử lý xóa background. Ảnh sau khi xóa background có thể loại bỏ được những nội dung không thuộc bìa sách mà chúng ta không quan tâm đến.



Hình 7. Minh họa xóa background cho bìa sách

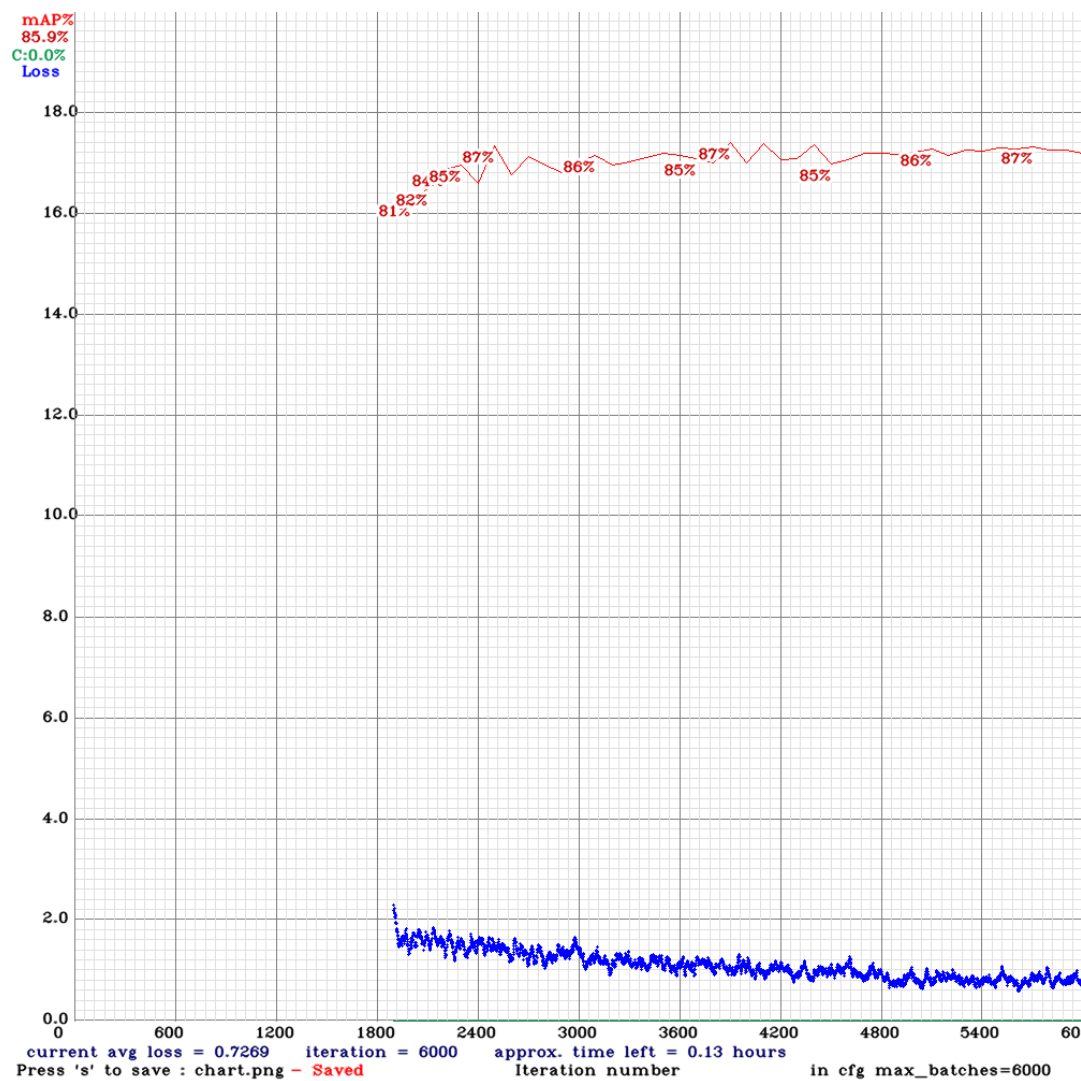
### **b. Phát hiện đối tượng trên bì sách (Object detection)**

Ở bước này chúng em sử dụng mô hình [Yolov4](#) để thực nghiệm tác vụ phát hiện đối tượng trên bì sách, cụ thể ở đây là tên sách, tên tác giả, nhà xuất bản. Chúng em huấn luyện mô hình trên tập train và hiệu chỉnh trên tập validation.

Chúng em đã tiến hành huấn luyện mô hình [Yolov4](#) khi gán nhãn được 400 ảnh trên tập train và 89 ảnh trên tập val. Sau đó khi hoàn thành gán nhãn 800 ảnh trên tập train, chúng em đã huấn luyện lại một lần nữa, các thông số khác hoàn toàn giống nhau. Và kết quả thu được kết quả ở [Bảng 8](#) và [Bảng 9](#)

Clone source [darknet](#) có chứa mô hình [Yolov4](#) và sửa lại các file theo đúng mục đích của mình lần lượt là các file makefile, yolo.names, yolo.data, cfg/yolov4-custom.cfg và upload tất cả các ảnh train và val vào thư mục data. Tạo file backup để lưu lại mô hình huấn luyện.

Chi tiết hơn, chúng em sử dụng pre-train model với weights “yolov4.conv.137” để bắt đầu train bộ dữ liệu của chúng em. Các thông số được tinh chỉnh như sau: width = 416, height = 416 (Giảm kích thước ảnh để tránh trường hợp out memory); batch\_size = 64; Subdivisions=16; Max\_batches = 6000; classes = 3 (tên tác giả, nhà xuất bản, tên sách); filters=(số class+5)\*3 = 24. Bắt đầu huấn luyện mô hình và tính mAP(Hình 8) của tập val sau 1000 iterations. Kết thúc 6000 iterations và thu được mô hình tốt nhất được lưu lại ở file backup được ban đầu.



Hình 8. Visualize mAP và Loss trong quá trình huấn luyện YOLOv4

Dưới đây là một số ảnh minh họa cho quá trình đánh giá trên tập test (Hình 9).



Hình 9. Ảnh được dự đoán trên tập test

| image                                 | Width | Height | tacgia                    | tensach                 | nxb                       |
|---------------------------------------|-------|--------|---------------------------|-------------------------|---------------------------|
| photo_43<br>@05-01-2022_08-41-24.jpg  | 961   | 1280   |                           | ['244,374,7<br>72,650'] | ['262,998,6<br>62,1132']  |
| photo_313<br>@05-01-2022_08-45-39.jpg | 961   | 1280   | ['519,1108,723<br>,1180'] | ['296,222,6<br>44,544'] | ['365,1177,<br>543,1227'] |
| .....                                 | ....  | .....  | .....                     | .....                   | .....                     |
| photo_353<br>@05-01-2022_08-45-59.jpg | 961   | 1280   | ['286,188,7<br>98,232']   | ['252,220,8<br>22,522'] | ['352,1058,<br>668,1116'] |

Bảng 2. Mô tả đầu ra của Yolov4 sau khi dự đoán trên tập test

Sau khi dự đoán toàn bộ ảnh trên tập test, chúng em xuất ra kết quả và lưu về một file .csv, đầu ra của csv được mô tả như trong [bảng 2](#), sẽ gồm có các thuộc tính sau: image(tên ảnh), width(chiều rộng), height(chiều cao), tacgia(tên tác giả), tensach(tên

sách), nxb(nhà xuất bản). Trong đó các cột taggia, tensach, nxb chứa tọa độ của các bounding box ứng với các class đó. Mỗi class có thể có nhiều bounding box.

### c. Phát hiện văn bản (Text Detection)

Ở bước phát hiện văn bản(Text Detection) chúng em thực nghiệm trên 2 mô hình là [EasyOCR](#) và [PaddleOCR](#).

#### ● EasyOCR model

Dữ liệu ảnh sau khi tiền xử lý xóa background sẽ được đưa vào [EasyOCR](#) model để phát hiện văn bản. [EasyOCR](#) detection là mô hình pre-train trên hơn 80 ngôn ngữ, trong đó có tiếng Việt, nên chúng em sử dụng để dự đoán mà không huấn luyện lại. Hình 10 cho thấy [EasyOCR](#) detection model detect khá tốt với ảnh đã xóa background.



Hình 10. Minh họa ảnh dự đoán của mô hình EasyOCR khi xóa background

#### ● PaddleOCR Text Detection model



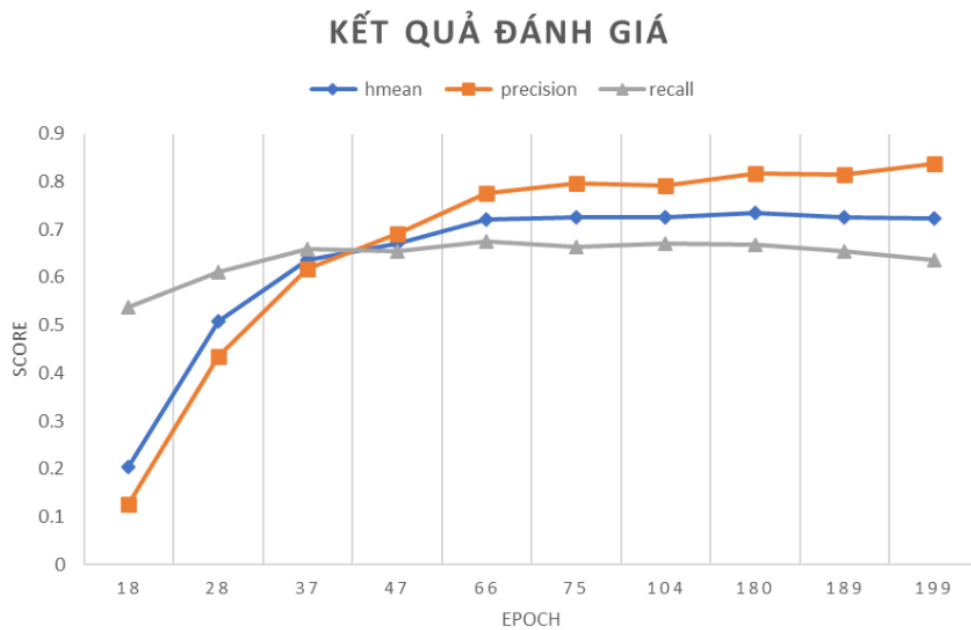
Dữ liệu ảnh sau khi được gán nhãn với [PPOCRLabel](#) (được mô tả ở mục [Gán nhãn dữ liệu](#)) sẽ được huấn luyện với mô hình [PaddleOCR Text Detection](#) nhằm mục đích phát hiện văn bản trong ảnh.

PaddleOCR là mô hình SOTA trong lĩnh vực OCR, được phát triển bởi công ty Baidu, Trung Quốc và được xuất bản vào tháng 9 năm 2020. Đây là pre-trained model được huấn luyện và dự đoán rất tốt trên tiếng Trung, tiếng Anh. [PaddleOCR](#) cũng có mô hình đa ngôn ngữ hỗ trợ trên các ngôn ngữ: tiếng Hàn, Nhật, Đức Pháp,...

Sở dĩ chúng em chọn mô hình [PaddleOCR detection](#) cho task detection vì qua quá trình gán nhãn bằng [PPOCRLabel](#), chúng em thấy chức năng automatic detection của [PPOCRLabel](#) detect rất tốt trên tiếng anh, nhưng với tiếng việt có dấu thì không detect được dấu vì thế bounding box không bao trùm hết được các từ. Ngoài ra mô hình chưa được huấn luyện với dữ liệu bìa sách của chúng em nên khi detect sẽ detect cả những vùng chứa văn bản không thuộc nội dung bìa sách. Hình 12 minh họa chức năng detection trên.

Từ các vấn đề trên, chúng em đã chọn mô hình pre-trained [PaddleOCR](#) để huấn luyện lại trên dữ liệu của chúng em. Cụ thể là mô hình

**ResNet50\_vd\_ssld\_pretrained** - mô hình detection có kích thước lớn của PaddleOCR. Chúng em sử dụng gpu của Google Colab để huấn luyện mô hình trên tập train và hiệu chỉnh trên tập validation với các thông số được tinh chỉnh như sau: *epoch\_num = 200, log\_smooth\_window = 20, learning\_rate = 0.001, eval\_batch\_step = 100, score\_thresh = 0.5, image\_shape = [512, 512]* và một số thông số khác như, *train\_dir, val\_dir, test\_dir, save\_model\_dir* (trong file *config/det\_r50\_vd\_sast\_totaltext.yml*). Hình 11 mô tả kết quả đánh giá trên tập validation lúc huấn luyện mô hình. Mô hình cho kết quả tốt nhất trên tập validation ở epoch 180.

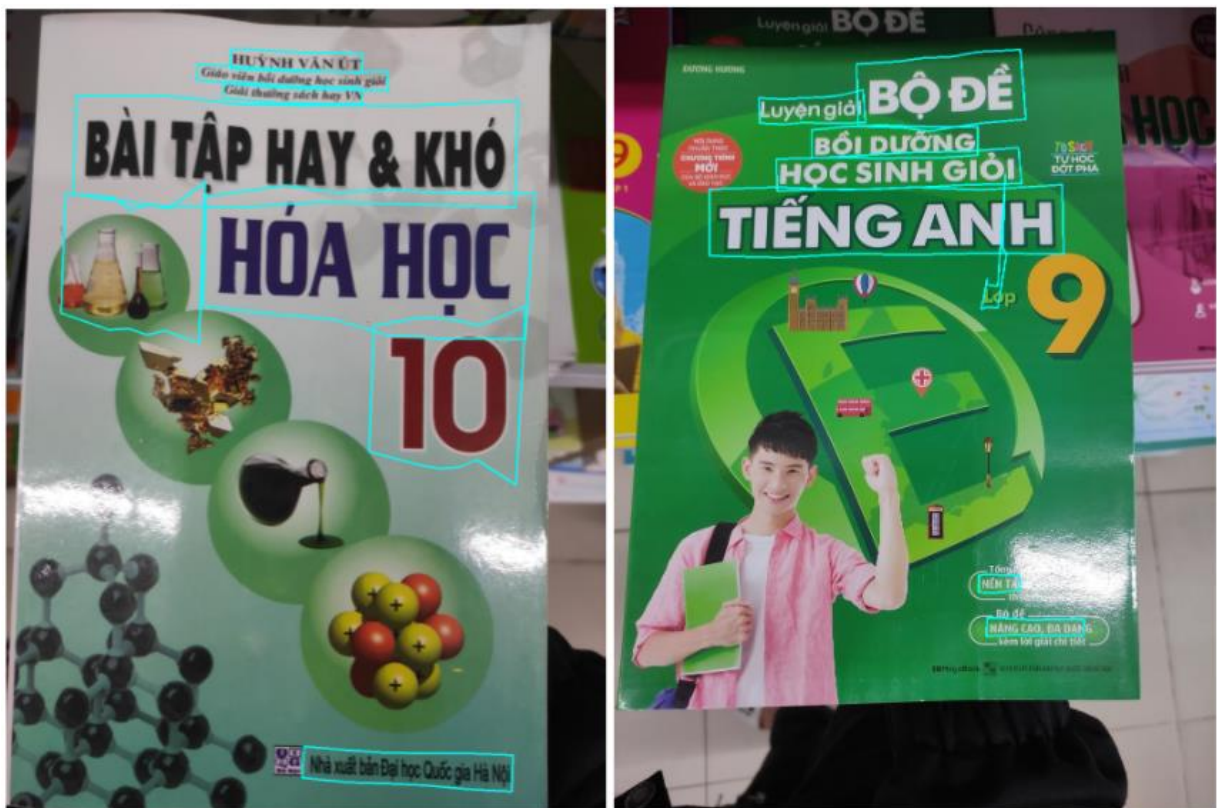


Hình 11. Kết quả đánh giá trên tập validation của [PaddleOCR](#)



Hình 12. Ảnh [PaddleOCR](#) detection model predict được trước khi huấn luyện





Hình 13. Ảnh predict bằng [PaddleOCR](#) sau khi huấn luyện với custom data.

Chúng em thấy sau khi huấn luyện, mô hình có thể detect tốt vùng chứa văn bản trên bìa sách, có thể cải thiện được những vấn đề như bỏ sót dấu và detect nhầm những vùng không mong muốn đã được đề cập trước đó (Hình 13).

Đầu vào của bước Text Detection bằng [PaddleOCR](#) là ảnh chụp bìa sách, đầu ra là tọa độ của polygon bounding box mà mô hình detect được của mỗi bìa sách, có định dạng như hình sau:

```
det_results_all_test/photo_100@05-01-2022_08-41-58.jpg
[{"transcription": "", "points": [[154.7133026123047,
196.5032958984375], [281.5513087272644, 190.26013692220053],
[364.2052720308304, 178.74051411946616], [443.80327520370486,
188.62945556640625], [521.2636551380158, 185.50989786783856],
[639.6499633789062, 187.21104431152344], [645.145751953125,
379.7010192871094], [523.5079338550568, 389.9393717447917],
[445.0019865512848, 386.78708394368493], [367.55556004047395,
388.83427937825525], [282.85019145011904, 383.9176686604818],
[154.0120849609375, 381.4969787597656]]]}
```

Bảng 3. Minh họa định dạng đầu ra Text Detection bằng [PaddleOCR](#)

Trong đó “**det\_results\_all\_test/photo\_100@05-01-2022\_08-41-58.jpg**” là đường dẫn đến file ảnh, **points** là tọa độ các đỉnh của polygon bounding box. Trong ảnh minh họa chỉ chứa 1 bounding box.

#### d. Nhận dạng văn bản(Text Recognition)

Sau khi qua bước Text Detection chúng em có được các vùng chứa văn bản trên bìa sách. Ở bước Text Recognition chúng em sử dụng hai thư viện với hai hướng tiếp cận là EasyOCR và VietOCR được trình bày chi tiết như bên dưới đây.

- **EasyOCR Text Recognition model.**

Dữ liệu ảnh sau khi detect vùng chứa văn bản bằng EasyOCR Text Detection, chúng em sử dụng EasyOCR để thực hiện task Text Recognition và so sánh với những phương pháp khác sau đó tìm ra phương pháp tối ưu. Đầu vào của task này là ảnh bìa sách và tọa độ của bounding box đã detect được ở bước trước. Đầu ra là text ứng với từng bounding box. [Bảng 4](#) mô phỏng đầu ra với tọa độ đã được chuẩn hóa của EasyOCR Text Recognition model.

| name                             | points  | text   |
|----------------------------------|---|--|
| photo_93@05-01-2022_08-41-58.jpg | [[[396, 126], [578, 126], [578, 154], [396, 154]], [[707, 135], [745, 135], [745, 147], [707, 147]], [[696, 143], [749, 143], [749, 158], [696, 158]], [[702, 156], [750, 156], [750, 164], [702, 164]],... [[440.6546544120074, 1052.5201198532081], [474.7253959401953, 1049.9885610005751], [475.3453455879926, 1066.4798801467919], [442.2746040598047, 1069.0114389994249]]] | ['Nguyễn Thị Thu Huệ', 'Bilra Soan', 'rto (ucat', 'nfsra', 'THE LANGMASTER', '(aro Gioour', 'oloao', 'JSack KoC IoT Tieng anh', 'BỘ ĐỀ KIỂM TRA', 'TIẾNG', 'ANH', 'Lớp', 'tập', '9', 'Có đáp án', '2', 'ENGLISH', 'XUẤT BẢN', '0G', 'ĐẠI HỌC QUỐC GIA HÀ NỘI', '7oabulary', 'reldin', 'Speakine', 'NHA'] |

*Bảng 4. Minh họa đầu ra với tọa độ đã được chuẩn hóa EasyOCR Text Recognition model*

- **VietOCR**

Sau khi qua bước Text Detection bằng [PaddleOCR Text Detection](#) với custom dataset, chúng em có được các vùng chứa văn bản trên bìa sách. Chúng em sử dụng thư viện [VietOCR](#) để thực hiện task Text Recognition. [VietOCR](#) là thư viện được ra mắt phiên bản đầu tiên vào tháng 3 năm 2020 và đã cập nhật phiên bản mới nhất vào tháng 11 năm 2021. Thư viện cung cấp cả 2 kiến trúc **AttentionOCR** và **TransformerOCR**. Mô hình pre-trained của VietOCR được huấn luyện trên tập dữ

liệu tiếng Việt gồm 10 triệu ảnh và được tác giả đánh giá có tính tổng quát cực tốt, thậm chí có độ chính xác khá cao trên một bộ dataset mới.

Sau khi có được tọa độ từ task Text Detection bằng **PaddleOCR** model, được mô tả ở *Hình 13* Chúng em xử lý cắt ảnh theo tọa độ bounding box và đưa vào mô hình Transformer của VietOCR. *Hình 14* ảnh bên trái mô tả ảnh sau khi qua bước Text Detection. *Hình 14* ảnh bên phải mô tả ảnh đã được cắt ra. Bounding box của PaddleOCR là polygon chứ không phải rectangle như thông thường nên chúng em đã tạo rectangle bounding box chứa polygon đó để cắt ảnh.

Ngoài ra, để tận dụng được polygon bounding box( để mô hình không bị nhận nhầm vùng bên ngoài polygon bounding box có chứa văn bản) chúng em tạo mask và chuyển phần ngoài polygon về black background và được kết quả như *Hình 14*.



*Hình 14. Mô phỏng các ảnh được cắt ra từ tọa độ bounding box của PaddleOCR Detection*

Đầu vào của mô hình **TransformerOCR** là ảnh vừa cắt được. Đầu ra của của mô hình là nội dung text ứng với ảnh đó. [Bảng 5](#) minh họa đầu ra của task Text Recognition. Ứng với mỗi bounding box ở cột “**points**” là một phần tử của list ở cột “**text**”.

| name | points | text |
|------|--------|------|
|------|--------|------|

|                                  |   |   |
|----------------------------------|---|---|
| photo_93@05-01-2022_08-41-58.jpg | [[[165, 827], [308, 809], [429, 813], [556, 806], [675, 799], [833, 792], [839, 973], [684, 985], [561, 979], [434, 976], [315, 984], [176, 985]], [[590, 589], [612, 586], [623, 587], [635, 587], [647, 585], [670, 585], [670, 621], [650, 636], [638, 637], [625, 637], [613, 635], [592, 632]], ..., [[334, 595], [348, 589], [359, 592], [372, 593], [384, 592], [405, 591], [405, 640], [384, 637], [372, 635], [361, 634], [348, 635], [334, 632]]] | ['ENGLISH', 'TIẾNG', 'BỘ ĐỀ KIỂM TRA', 'ANH', '9', 'NGUYỄN THỊ THU HUẾ', 'CÓ ĐÁP ÁN', 'MM', 'TẬP I', 'LỚP'] |
|----------------------------------|---|---|

Bảng 5. Minh họa đầu ra của mô hình TransformerOCR

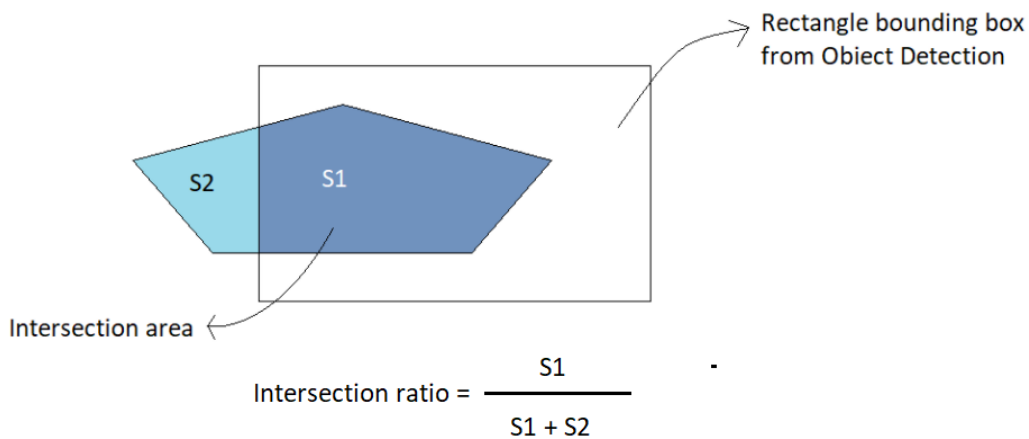
#### e. Xử lý (Processing II): Phân loại văn bản thuộc các lớp đối tượng và chuẩn hóa văn bản đầu ra

Sau khi thực nghiệm các mô hình, các phương pháp, chúng em được kết quả ở [Bảng 8, 9, 10 và 11](#). Chúng em chọn mô hình tốt nhất trên task Text Detection là [PaddleOCR](#) (đã được huấn luyện với dữ liệu của chúng em); mô hình tốt nhất trên task Text Recognition là [TranformerOCR](#) của **VietOCR** và mô hình cho task Object Detection là [Yolov4](#) đã huấn luyện phía trên.

##### • Phân loại văn bản thuộc các lớp đối tượng

Đầu ra của task Text Detection là tọa độ các bounding box. Chúng ta cần biết các bounding box đó chứa nội dung thuộc trường nào trong các trường: tên sách, tên tác giả, nhà xuất bản. Để làm được điều đó, chúng em kết hợp với task **Object Detection**.

Đầu ra của mô hình [Yolov4](#) là tọa độ các bounding box thuộc 3 classes: tên sách, tên tác giả, nhà xuất bản (mục [Object detection](#)). Đầu tiên chúng em chuyển tọa độ đầu ra về format tọa độ 4 đỉnh để dễ xử lý. Tiếp đến, chúng em tiến hành ánh xạ các polygon bounding box đã detect được từ task detection. Nếu tỉ lệ phần giao nhau (intersection ratio)  $> 0.8$  thì bounding box của text detection thuộc class đang xét. Hình 15 minh họa cách tính intersection ratio. Đa số intersection ratio tập trung vào 0.9 đến 1.0.



Hình 15. Minh họa cách tính intersection ratio

Trong quá trình làm chúng em thấy nếu cắt ảnh từ task Object Detection làm đầu vào cho Text Detection thì sẽ mất đi một số thông tin khi Object Detection không detect hết được các vùng thông tin. Trong khi đó [PaddleOCR](#)(Text Detection) detect được khá tốt vùng chứa văn bản. Vì thế chúng em kết hợp kết quả của Object Detection và Text Detection. Nếu các polygon bounding box ở task Text Detection không thuộc class nào trong 3 class: Tên sách, tên tác giả, nhà xuất bản sẽ được giữ lại ở cột note. Sau khi phân loại các bounding box về đúng class, chúng em được dữ liệu như [Bảng 6](#).

| image                            | author_points  | author_text            | title_points   | title_text  |
|----------------------------------|--|------------------------|--|---|
| photo_93@05-01-2022_08-41-58.jpg | [[[396, 125], [432, 127], [467, 126], [503, 124], [534, 124], [576, 123], [576, 151], [534, 151], [504, 152], [468, 154], [432, 154], [396, 155]]] | ['NGUYỄN THỊ THU HUỆ'] | [[[263, 300], [371, 290], [443, 289], [526, 289], [604, 288], [706, 282], [711, 436], [609, 436], [530, 441], [446, 447], [373, 438], [268, 444]],..., [[334, 595], [348, 589], [359, 592], [372, 593], [384, 592], [405, 591], [405, 640], [384, 637], [372, 635], [361, 634], [348, 635], [334, 632]]] | ['TIẾNG', 'BỘ ĐỀ KIỂM TRA', 'ANH', '9', 'CÓ ĐÁP ÁN', 'TAPI', 'LỚP'] |



| publisher_points | publisher_text | note_points  | note_text         |
|------------------|----------------|--|-------------------|
| []               | []             | [[[165, 827], [308, 809], [429, 813], [556, 806], [675, 799], [833, 792], [839, 973], [684, 985], [561, 979], [434, 976], [315, 984], [176, 985]], [[719, 600], [744, 593], [762, 592], [774, 592], [792, 593], [826, 596], [826, 638], [792, 642], [775, 643], [762, 643], [744, 642], [719, 634]]] | ['ENGLISH', 'MM'] |

Bảng 6. Đầu ra dữ liệu sau khi phân loại về các class

Chúng em phân loại các bounding box trong task Text Detection vào 3 classes: author(tác giả), title(tên sách), publisher(nhà xuất bản). Ở cột author\_points, title\_points, publisher\_points, note points chứa tọa độ các bounding box tương ứng với text trong các cột author\_text, title\_text, publisher\_text, note\_text. Trong đó, note\_points và note\_text là các cột chứa bounding box và text trong bounding box đó mà [PaddleOCR](#)(task Text Detection) detect được nhưng lại không thuộc class nào của Yolo Detection.

### • Chuẩn hóa văn bản đầu ra

Sau khi chia các bounding box về các class của nó, chúng em thấy thứ tự của text recognize được bị đảo trật tự (nếu ghép các chuỗi đó lại với nhau thì mất đi ý nghĩa, không tạo nên được nội dung đúng cho bìa sách). Ví dụ ở [Bảng 6](#), ở cột title\_text: “TIẾNG BỘ ĐỀ KIỂM TRA ANH 9 CÓ ĐÁP ÁN TAPI LỚP”. Rõ ràng chuỗi trên không đúng với nội dung bìa sách, lẽ ra phải là “BỘ ĐỀ KIỂM TRA TIẾNG ANH LỚP 9 TAPI CÓ ĐÁP ÁN”.

Vậy nên ở bước này chúng em sẽ áp dụng phương pháp chuẩn hóa như sau để giải quyết vấn đề trên. Chúng em sử dụng tọa độ của bounding box ở [Bảng 6](#) để xác định vị trí của các bounding box đó. Đầu tiên chúng em sẽ xác định center point của các bounding box (thư viện hỗ trợ [GeoPandas](#)).

Sau khi có được tọa độ của center point chúng em viết python script sắp xếp thứ tự các bounding box sao cho tọa độ y của center point giảm dần, x tăng dần. Nghĩa là

sẽ đọc văn bản trên xuống dưới và từ trái qua phải theo đúng quy tắc đọc tiếng Việt. Tiếp đến chúng em ánh xạ lại text của các box và ghép các chuỗi lại với nhau. Phương pháp trên cho kết quả rất khả quan. Sau khi thực hiện quá trình trên chúng em được kết quả như [Bảng 7](#) sau:

| image                             | author                          | title   | publisher           | note  |
|-----------------------------------|---------------------------------|---|---------------------|---|
| photo_93@05-01-2022_08-41-58.jpg  | NGUYỄN THỊ THU HUẾ              | BỘ ĐỀ KIỂM TRA TIẾNG ANH TAPI LỚP 9 CÓ ĐÁP ÁN                         |                     | MM ENGLISH  |
| photo_489@05-01-2022_08-47-23.jpg | URYASHI PITRE                   | NẤU ĂN DỄ DÀNG VỚI NỒI CHIẾN KHÔNG DẦU 100 CÔNG THỨC BÙNG NỔ HƯƠNG VỊ |                     |   |
| photo_456@05-01-2022_08-47-04.jpg | SIMONE GRIFFIN - DIANNE SANDLER | THÚC ĐẦY GIAO TIẾP  | NHÀ XUẤT BẢN PHỤ NỮ | 300 TRÒ CHƠI VÀ CÁC HOẠT ĐỘNG CHO TRẺ TỰ KỶ DỊCH VÀ HIỆU ĐỈNH: TRẦN BÍCH PHƯƠNG & NGUYỄN KIM ĐIỀU |

*Bảng 7. Đầu ra dữ liệu sau khi chuẩn hóa*



## V. Kết quả đánh giá và phân tích lỗi


### 1. Độ đo đánh giá

- **IoU:**

Intersection over Union là chỉ số đánh giá được sử dụng để đo độ chính xác của detection task trên tập dữ liệu cụ thể. IOU thường được đánh giá hiệu năng của các mô hình detection.

Để tính IoU ta cần có hai bounding box: Thứ nhất là bounding box thực sự của đối tượng (chính là bounding box đã gán nhãn). Thứ hai là bounding box của đối tượng được dự đoán bởi mô hình.

Cách tính IoU sẽ là tỉ lệ diện tích giao nhau giữa bounding box thực sự và bounding box dự đoán với diện tích hợp của hai bounding box đó. Được minh họa ở Hình 16 dưới đây.

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


Hình 16. Minh họa cách tính độ đo IoU

- **True/false positive/negative:**

Kết quả trả về của IoU sẽ nằm trong khoảng từ 0 đến 1, và mỗi detection sẽ có một IoU riêng. Để xác định được wrong detection hay correct detection thì chúng ta phải dựa vào một ngưỡng cho trước tùy vào bài toán (0.25, 0.5, 0.75,...). Ở đề án này chúng em chọn 2 ngưỡng là **0.5 và 0.75 cho Object Detection model - YOLOv4** và **0.5 cho Text Detection - PaddleOCR & EasyOCR model**. Nếu IoU của detection lớn hơn hoặc bằng so với ngưỡng cho trước thì đó là correct detection, ngược lại thì wrong detection.

Từ đó ta có định nghĩa về True/false positive/negative:

- True Positive (TP): IoU lớn hơn hoặc bằng ngưỡng, là một correct detection. Trong trường hợp có nhiều box có IoU lớn hơn ngưỡng thì box có IoU lớn nhất sẽ được tính là TP, còn lại là FP.
- False Positive (FP): IoU bé hơn ngưỡng (là một wrong detection).
- False Negative (FN): trường hợp mà có bounding box nhưng không dự đoán.
- **Precision:** là thang đo độ chính xác của dự đoán.

$$Precision = \frac{TP}{TP + FP}$$

- **Recall:** là thang đo độ tốt của khả năng tìm thấy được các correct detection.

$$Recall = \frac{TP}{TP + FN}$$

- **F1- score, H-mean** là trung bình điều hòa của Precision và Recall

$$F1\ score = H\ mean = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

- **AP (Average Precision):**

Từ precision và recall được định nghĩa ở trên chúng ta có thể đánh giá mô hình bằng cách thay đổi ngưỡng và giá trị quan sát của precision và recall. Khái niệm Area Under the Curve (AUC) cũng được định nghĩa tương tự. Với Precision-Recall Curve, AUC còn có một tên khác là Average precision (AP).

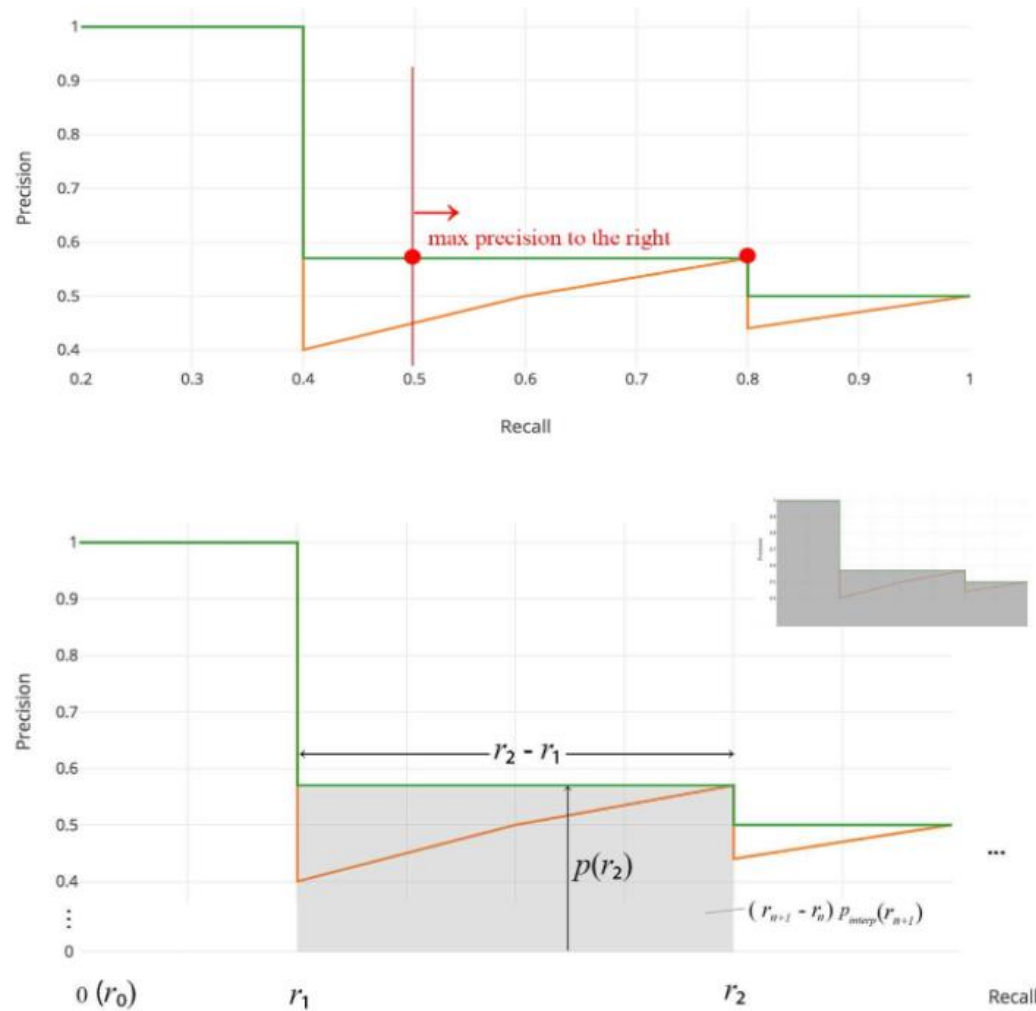
Giá trị AP là giá trị phía dưới đường biểu diễn mối quan hệ precision – recall. Tại mỗi recall level, ta thay giá trị precision bằng giá trị precision lớn nhất tại recall level đó.

AP sẽ bằng phần diện tích phía dưới đường biểu diễn precision-recall bằng cách tính tổng các hình chữ nhật xấp xỉ.

Công thức:

$$P_{interp}(r_{n+1}) = \max_{r' \geq r_{n+1}}(r')$$

$$AP = \sum (r_{n+1} - r_n) P_{interp}(r_{n+1})$$



Hình 17. Sơ đồ trực quan cách tính AP dựa vào chỉ số precision và recall

- **mAP:**

Bài toán Object Detection của chúng ta có nhiều class, mỗi class ta sẽ tiến hành đo AP, sau đó lấy trung bình của tất cả các giá trị AP của các class ta được chỉ số mAP của mô hình.

- **Accuracy (Text Recognition):**

Sau khi tính được các box thỏa mãn (True Positive), sẽ chuyển qua tính kết quả của bước Text Recognition. Các chuỗi GroundTruth sẽ được chuyển sang dạng chữ viết hoa (upper case) và so sánh. Kết quả được đánh giá đúng khi hai chuỗi ký tự đó giống hệt nhau. Sau đó sẽ tính trung bình trên mỗi ảnh và trên các ảnh.

$$Accuracy = \frac{\text{Số chuỗi dự đoán đúng}}{\text{Tổng số chuỗi}}$$

- **WER:**

Ở đây, sau khi classify ra 3 class: author (tên tác giả), title(tên sách), publisher(nhà xuất bản). Chúng em dùng độ đo WER(word error rate - tỉ lệ lỗi trên cấp độ từ) để tính độ lỗi. WER được tính như công thức dưới đây.

$$WER = \frac{S + D + I}{N}$$

Trong đó: *S* là số token bị sai chính tả

*D* là số token bị xóa mất, thiếu

*I* là số token được chèn thêm vào.

*C* là số token dự đoán đúng

$N = S + D + C$  là số token của chuỗi ground truth.

Ý niệm đằng sau xóa và chèn là những biến đổi nhằm chuyển từ chuỗi thực tế sang chuỗi dự đoán, nghĩa là những lỗi khiến cho chuỗi ground truth bị sai thành chuỗi dự đoán.

Ví dụ:

| Chuỗi thực tế                                     | Chuỗi dự đoán                                  |
|---|--|
| GIẢI ĐÓ GIẢI NGỒ CÙNG TRUYỆN NGỰ<br>NGÔN TOẢN HỌC | GIÁ ĐI GIAING CÙNG TRUYỆN NGỰ<br>NGÔN TOẢN HỌC |

Bảng 13. Ví dụ tính WER

Trong ví dụ trên:

Số token bị sai chính tả:  $S = 3$  (GIẢI -> GIÁ, ĐÓ -> ĐI, GIẢI -> GIAING)

Số token bị mất, thiếu, xóa:  $D = 1$  (NGỐ)

Số token được thêm vào:  $I = 0$

Số token được dự đoán đúng:  $C = 6$

$N = S + D + C = 10$

$\Rightarrow WER = 4/10 = 0.4$

```
from jiwer import wer

ground_truth = "GIẢI ĐỐ GIẢI NGỐ CÙNG TRUYỆN NGỰ NGÔN TOÁN HỌC"
hypothesis = "GIÁ ĐI GIAING CÙNG TRUYỆN NGỰ NGÔN TOÁN HỌC"

error = wer(ground_truth, hypothesis)
print(error)
```

0.4

Hình 20. Minh họa tính WER bằng thư viện jiwer

Chúng em sử dụng thư viện jiwer để tính WER và cũng được kết quả 0.4 như hình 20.

## 2. Kết quả đánh giá và nhận xét

### a) Object Detection

| Model                           | IoU | Precision | Recall | F1-score | mAP(%)       |
|---------------------------------|-----|-----------|--------|----------|--------------|
| Yolov4<br>(400<br>ảnh<br>train) | 50% | 0.86      | 0.85   | 0.85     | 86.96        |
|                                 | 75% | 0.71      | 0.70   | 0.71     | 63.53        |
| Yolov4<br>(800<br>ảnh<br>train) | 50% | 0.91      | 0.91   | 0.91     | <b>92.57</b> |
|                                 | 75% | 0.73      | 0.73   | 0.73     | <b>63.59</b> |

Bảng 8. Kết quả tập test trên mô hình Yolov4 với hai lần huấn luyện

| Model                  | IoU | AP (%)<br>class: Tác giả | AP (%)<br>class: Tên sách | AP (%)<br>class: Nhà xuất bản |
|------------------------|-----|--------------------------|---------------------------|-------------------------------|
| Yolov4 (400 ảnh train) | 50% | 74.25                    | 92.27                     | 94.37                         |
|                        | 75% | 52.42                    | 69.75                     | 68.43                         |
| Yolov4 (800 ảnh train) | 50% | 87.52                    | <b>95.46</b>              | 94.72                         |
|                        | 75% | 56.52                    | <b>77.56</b>              | 56.68                         |

Bảng 9. Kết quả đánh giá độ đo AP (%) trên từng class

Dựa vào kết quả ở [Bảng 8](#) ta có thể thấy với việc có nhiều dữ liệu để huấn luyện hơn thì mô hình sẽ có kết quả tốt hơn.

Ngoài ra, ta có thể nhận thấy class có độ chính xác cao nhất là tên sách và thấp nhất là tác giả như ở [Bảng 9](#).

Đối với class tên sách: Đa số các tên sách đều là các chữ in, to và dễ nhận thấy nên việc nhận dạng chúng là dễ dàng hơn so với các class còn lại. Vì vậy nên độ chính xác khi nhận dạng class tên sách là tốt nhất. Với class tên tác giả thường bị nhận nhầm vì tên tác giả xuất hiện ở nhiều vị trí khác nhau và cỡ chữ nhỏ; còn nhà xuất bản thường được in khá nhỏ nên dẫn đến việc không nhận dạng được ([Hình 19](#)) hoặc nhận dạng nhầm khi có các chữ nhỏ ở góc bên dưới bìa sách (như [Hình 18](#) ảnh bên phải).



Hình 18. Class tác giả không nhận diện được ở hình bên phải và nhầm lẫn ở hình bên trái



Hình 19. Không nhận dạng được class nhà xuất bản vì quá nhỏ



### b) Text Detection

| Model                | Precision(%) | Recall(%)    | Hmean(%)     |
|----------------------|--------------|--------------|--------------|
| EasyOCR (Detection)  | 37.86        | <b>71.40</b> | 49.48        |
| Detectron2 + EasyOCR | 50.18        | 61.08        | 55.10        |
| PaddleOCR            | <b>88.61</b> | 69.23        | <b>77.74</b> |

*Bảng 10. Kết quả của các mô hình trên task Text Detection*

Nhìn vào bảng 10, có thể thấy qua task Text Detection, mô hình cho kết quả H-mean cao nhất là PaddleOCR(77.74 %). Mô hình này sau khi được huấn luyện có thể khắc phục được 2 nhược điểm: ở PaddleOCR(chưa được huấn luyện) thì bounding box detect được bị sót dấu (Hình 12) và ở EasyOCR thì detect nhầm những vùng không mong muốn (Hình 10).

Sở dĩ ở mô hình EasyOCR cho kết quả độ đo Recall(độ phủ) cao nhất vì mô hình này detect hết tất cả các vùng chứa văn bản có trong ảnh hay nói cách khác mô hình này có tính bao phủ tốt(khi xóa background thì khả năng bỏ sót tăng lên), tuy nhiên vì thế nên độ chính xác lại không cao.

### c) Text Recognition

| Model                   | Accuracy (%) |
|-------------------------|--------------|
| EasyOCR (Recognition)   | 37.86        |
| TransformerOCR(VietOCR) | <b>71.25</b> |

*Bảng 11. Kết quả các mô hình trên task Text Recognition*

Từ bảng trên có thể thấy mô hình Recognition TransformerOCR(VietOCR) cho kết quả tốt hơn EasyOCR rất nhiều, với kết quả 71.25%. Mô tả cách tính độ đo Accuracy ở mục [Accuracy \(Text Recognition\)](#)

### d) Output

| Model  | WER - author(%) | WER- title (%) | WER - publisher (%) | WER - all (%) |
|--|-----------------|----------------|---------------------|---------------|
| PaddleOCR + TransformerOCR(VietOCR) +Yolov4 +Processing II | <b>20.80</b>    | 34.72          | 29.13               | <b>28.22</b>  |

*Bảng 12. Kết quả đánh giá trên tập test của tổng hợp các mô hình*

Sự kết hợp của các mô hình **PaddleOCR** (Text Detection), **TransformerOCR**(VietOCR, Text Recognition), **Yolov4** (Object Detection), **ProcessingII** cho kết quả WER là 28.22 (trên cả 3 classes. WER thấp nhất đối với tên tác giả(author) và cao nhất đối với title(tên sách). Dễ dàng thấy, title (tên sách) rất đa dạng về font chữ, cỡ chữ cũng như họa tiết, hoa văn, màu sắc, chính điều đó gây nên khó khăn cho mô hình.

## **VI. Kết luận và hướng phát triển**

Trong đồ án này, chúng em đã xây dựng được bộ dữ liệu ảnh chụp bìa sách gồm 979 ảnh với đa dạng các thể loại, kiểu chụp khác nhau để phục vụ cho bài toán nhận dạng, trích xuất thông tin từ bìa sách.

Chúng em đã thực nghiệm các mô hình ở 4 tác vụ: Object Detection(Yolov4), Text Detection (EasyOCR, PaddleOCR), Text Recognition (EasyOCR, TranformerOCR - VietOCR), ProcessingII (quá trình biến đổi và chuẩn hóa để cho ra output cuối cùng).

Cuối cùng chúng em nhận thấy sự kết hợp cho kết quả tốt nhất là Yolov4, PaddleOCR, TranformerOCR(VietOCR), ProcessingII cho đầu ra có tỉ lệ lỗi 28.22% (độ đo WER). Nhìn chung, kết quả đầu ra có chứa đựng được thông tin khá chính xác về nội dung trên bìa sách, tuy nhiên vẫn còn sai sót.

Trong tương lai, chúng em sẽ cố gắng thử nghiệm nhiều mô hình mới, nhiều phương pháp xử lý mới và phát triển lên thành ứng dụng để có thể hỗ trợ trong việc quản lý sách, số hóa sách.

## TÀI LIỆU THAM KHẢO

- [1] AI-Challenge: <https://aichallenge.hochiminhcity.gov.vn/>
- [2] VietOCR: <https://github.com/pbcquoc/vietocr>
- [3] EasyOCR: <https://github.com/JaidedAI/EasyOCR>
- [4] Detectron2: <https://github.com/facebookresearch/detectron2>
- [5] jiwer library: <https://pypi.org/project/jiwer/>
- [6] Opencv: <https://docs.opencv.org/4.x/>
- [7] PPOCRLabel:  
<https://github.com/PaddlePaddle/PaddleOCR/tree/release/2.3/PPOCRLabel>
- [8] PaddleOCR detection:  
[https://github.com/PaddlePaddle/PaddleOCR/blob/release/2.1/doc/doc\\_en/detection\\_en](https://github.com/PaddlePaddle/PaddleOCR/blob/release/2.1/doc/doc_en/detection_en)
- [9] WER vs CER:  
[https://openaccess.thecvf.com/content\\_CVPRW\\_2020/papers/w34/Baek\\_CLEval\\_Character-Level\\_Evaluation\\_for\\_Text\\_Detection\\_and\\_Recognition\\_Tasks\\_CVPRW\\_2020\\_paper.pdf](https://openaccess.thecvf.com/content_CVPRW_2020/papers/w34/Baek_CLEval_Character-Level_Evaluation_for_Text_Detection_and_Recognition_Tasks_CVPRW_2020_paper.pdf)
- [10] GeoPandas:  
<https://geopandas.org/en/stable/docs/reference/api/geopandas.GeoSeries.centroid.html>
- [11] Yolov4: <https://arxiv.org/pdf/2004.10934v1.pdf>