ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

TRÍCH XUẤT THÔNG TIN TỪ THẺ SINH VIÊN UIT

Sinh viên thực hiện:		
STT	Họ tên	MSSV
1	Huỳnh Nhật Nam	19520750
2	Huỳnh Hữu Hào	19520521

TP. HỒ CHÍ MINH – 5/2022

1. GIỚI THIỆU

Đề tài: Trích xuất thông tin từ thẻ sinh viên Công nghệ Thông tin (UIT).

Phương pháp thực hiện:

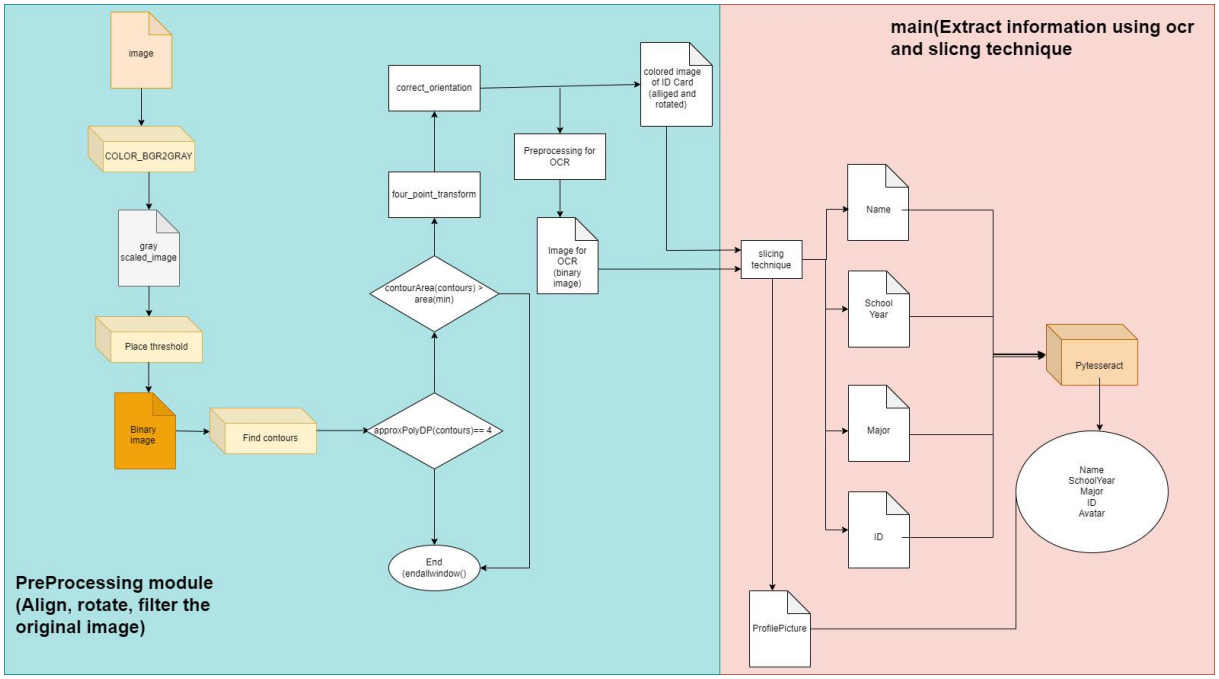
- Hình ảnh được đọc sẽ được chuyển sang thang đo xám để thiết lập ngưỡng threshold từ đó có kết quả contours (các đường viền bao quanh object) chính xác hơn.
- Tìm các contours để xác định thẻ sinh viên bằng hàm cv2.findcontours. sau đó dùng hàm cv2.approxPolyDP để đơn giản hoá contours (từ dạng vector thành các điểm).
- Thông qua hàm four_point_transform để detect 4 đỉnh, và dùng phép xoay matrix tạo ra hình ảnh của thẻ.
- Dùng hàm correct_orientation để xoay ảnh về đúng vị trí mong muốn (xoay 90 độ nếu hình không ngang, sau đó tính toán pixel để xác định hình có bị ngược hay không để xoay 180 độ), đồng thời đưa hình về chuẩn 480x280.
- Hình ảnh sau đó được lưu thành 2 dạng: một hình là hình ảnh gốc chưa được xử lý tách background và một hình đã được xử lý để dễ đọc cho ocr.
- Dùng kĩ thuật slicing cho hình ảnh thu được ở trên để tạo ra các vùng ảnh sẵn sàng cho máy học ocr.
- Sử dụng pytetesseract ocr với config cho từng vùng ảnh để nhận ra text.

Kết quả thu được: có thể dùng để đọc được thẻ sinh viên UIT trong trường hợp có điều kiện tốt (Background đơn giản, không có dị vật).

2. NỘI DUNG

2.1. Quá trình lên ý tưởng

Ban đầu nhóm chúng tôi thực hiện theo hướng chỉ thuần xử lý hình ảnh, sử dụng các hàm argument để làm thay đổi hình ảnh về dạng ảnh từ trên xuống như chúng tôi mong muốn. Cụ thể là sử dụng phương pháp Edge detection bằng cách tìm contours. Sơ đồ pipeline ban đầu của nhóm chúng tôi như sau:

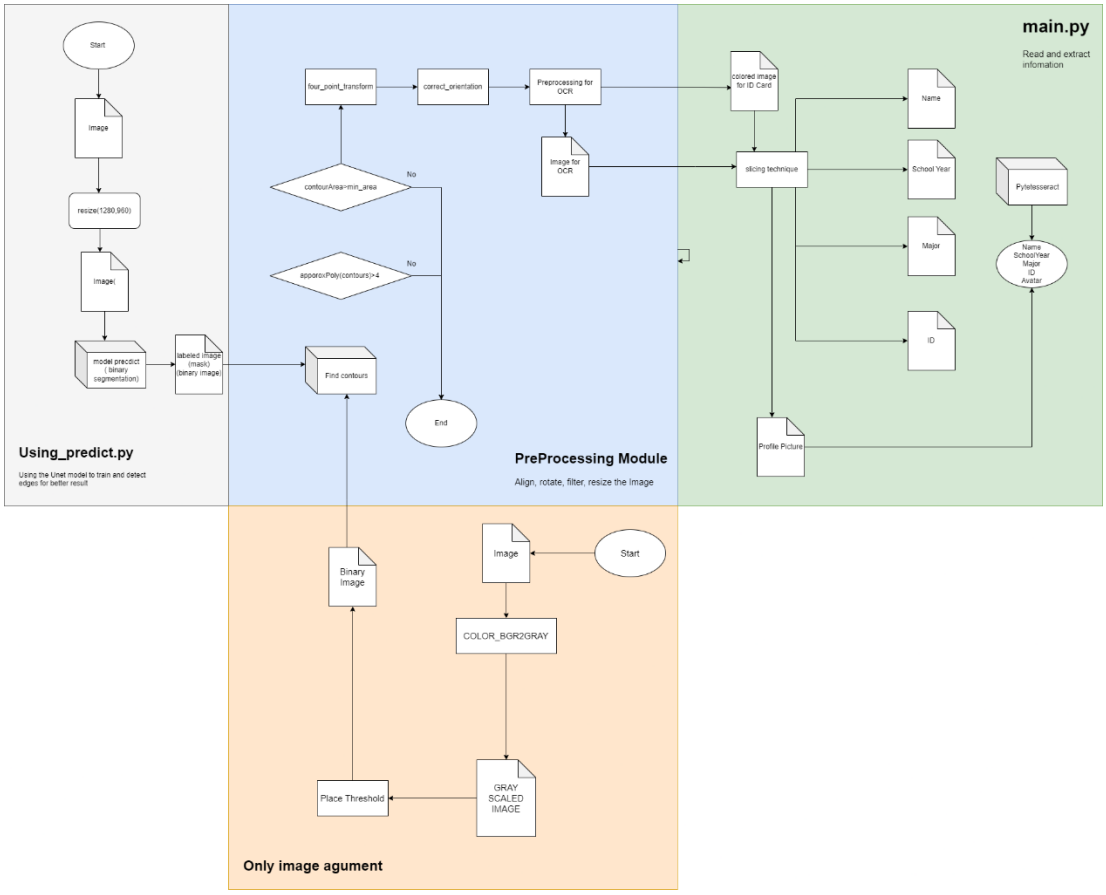


Hình 1: Sơ đồ pipeline ban đầu

Tuy nhiên vì tính dễ bị ảnh hưởng của các điều kiện bên ngoài dẫn đến kết quả đạt được không tốt, các hình ảnh bị biến dạng và không thể nhận diện được do sự nhiễu của hình ảnh, điều kiện ánh sáng, hay đơn giản là vật thể lạ nằm trong khung hình…

Để khắc phục điều này. Chúng tôi tìm được một hướng giải quyết đó là thực hiện một dự án máy học nho nhỏ, cụ thể là binary segmentation, giúp tách biệt hình ảnh thẻ ra khỏi môi trường một cách dễ dàng hơn.

Chúng tôi đưa đến quyết định kết hợp cả 2 phương pháp trên để đạt được kết quả tốt nhất.



Hình 2: Sơ đồ pipeline hiện tại

2.2. Nhị phân hóa hình ảnh

Đầu tiên, chúng tôi sẽ chuyển hình ảnh sang thang đo xám để đặt threshold (ngưỡng). Ngưỡng được đặt sẽ là 127, với các giá trị pixel < 127 sẽ được gán bằng 0 (màu đen); ngược lại, các giá trị pixel >=127 sẽ được gán bằng rgb(255,255,255) (màu trắng). Bước này sẽ thực hiện khi mô hình đưa ra dự đoán không tốt.

2.3. Tìm và đơn giản hóa Contours để xác định ảnh thẻ

Chúng tôi dùng hàm cv2.findContours để tìm các contours (là một vector điểm nối các điểm liên tục lại tạo thành một đường cong – curve). Việc tìm contours sẽ giúp ta tìm ra được các đối tượng (object) có trong ảnh.

Tiếp theo, chúng tôi đơn giản hóa contours bằng cách sử dụng hàm cv2.approxPolyDP, từ các đường cong (curve) chỉ lấy điểm đầu và điểm cuối. Từ đây, các đối tượng (object) có trong ảnh sẽ được xác định bằng các tập hợp điểm. Việc này cũng góp phần giúp cho việc xây dựng tập toạ độ góc đơn giản hơn nhiều.

2.4. Xác định ảnh thẻ

Từ các object đã tìm được, ta xác định ảnh thẻ bằng cách lựa chọn object có contours_approx = 4, tức là có thể nối 4 điểm để tạo ra hình ảnh contour của nó (là một hình bình hành) và có diện tích lớn nhất (trong các object có diện tích > 1/10 diện tích ảnh gốc).

2.5. Xoay hình về góc nhìn thẳng đứng

Thông qua 4 toạ độ điểm, đã được xét trên. Chúng tôi sẽ tìm ra 2 điểm có toạ độ nằm cách xa nhau nhất để xác định đỉnh của đường chéo hình bình hành. Khi đã có toạ độ 2 đầu mút đường chéo. Thông qua hàm np.diff với 1 điểm để tìm đc điểm thuộc chiều dài, chiều rộng. Việc này được thực hiện thông qua hàm order_points.

Chúng tôi gọi lần lượt 4 điểm là TopLeft, TopRight, BottomRight, BottomLeft. Từ 4 điểm đó, từ đó tính toán được chiều dài, chiều rộng của khung ảnh và thực hiện phép xoay ma trận thông qua hàm four_point_transform. Lúc này ảnh thẻ đã được xoay về góc nhìn thẳng đứng.

2.6. Xoay hình về đúng chiều và đưa về kích thước chuẩn

Sau khi hình đã được xoay thẳng đứng nhưng có thể chiều của ảnh chưa chính xác, chúng tôi sử dụng hàm correct_orientation để xoay hình về đúng chiều và đưa về kích thước chuẩn. Đầu tiên, chúng tôi so sánh chiều rộng (width - w) và chiều dài (height - h) của ảnh, nếu w < h (nghĩa là lúc này 2 giá trị bị ngược nhau) thì sẽ xoay hình 90 độ. Tiếp đó, chúng tôi dùng hàm resize để đưa hình thẻ về kích thước chuẩn (480x280). Cuối cùng, chúng tôi chia hình làm 2 phần và thực hiện so sánh số pixel, nếu số pixel bên trái > số pixel bên phải thì thực hiện xoay ảnh 180 độ (vì theo chiều ảnh đúng thì nội dung bên phải nhiều hơn bên trái). Đặc thù của ảnh sinh viên là số lượng nội dung nằm bên tay phải sẽ nhiều hơn tay trái nên chúng tôi xác định được điều trên.

2.7. Dùng slicing để tách vùng văn bản và đọc ocr

Để đọc được nội dung: tên, ngành, mssv,… Chúng tôi sẽ dựng nên một khung có kích thước chuẩn như trên(480x280) và kẻ khung hình ứng với từng nội dung trên.

Chúng tôi thực hiện việc trên thông qua kĩ thuật slicing.

Sau nhiều lần thử nghiệm, chúng tôi đã nhận ra Ocr engine hoạt động tốt nhất trong môi trường ảnh nhị phân chính vì vậy chúng tôi đã đưa hình ảnh về dạng nhị phân để đạt được kết quả tốt hơn.

Dùng kĩ thuật slicing để cắt vùng văn bản, hình ảnh và lưu vào các biến: name (họ tên), date (khóa), major (ngành học), mssv (mã số sinh viên), avatar (ảnh sinh viên).

Cuối cùng, chúng tôi sử dụng pytetesseract ocr với từng giá trị config khác nhau và trích xuất kết quả.

2.8. Xây dựng mô hình máy học Unet và áp dụng vào dự án

Việc chỉ sử dụng các phương pháp xử lý ảnh thông thường đem lại kết quả không tốt chính vì vậy chúng tôi hướng đến một giải pháp là sử dụng máy học.

Nhận thấy khả năng thu thập dữ liệu sẽ rất khó vì dự án ít người và thời gian gấp. Chúng tôi bỏ qua ý tưởng áp dụng máy học cho bài toán đa nhãn. Chúng tôi đã đi đến kết luận sử dụng bài toán segmentation (phân loại ảnh) cho 2 nhãn duy nhất đó là thẻ sinh viên và môi trường (tức là background).

2.9. Quá trình thu thập dữ liệu và dán nhãn

Nhóm chúng tôi sẽ dụng điện thoại cá nhân và thẻ sinh viên để thu thập dữ liệu, cố gắng thử càng nhiều càng tốt và được khoảng 100 hình ảnh.

Việc dán nhãn được thử hiện thủ công bằng phần mềm labelme(hay trước đay là labelimg)

2.10. Quá trình huấn luyện

Việc thực hiện huấn luyện diễn ra khá tốt, tuy nhiên do đặc thù của bài toán nhóm chúng tôi có tuỳ biến lại hàm loss, hàm accuracy và mô hình để phù hợp với bài toán

2.11. Hướng phát triển

Chúng tôi đưa ra một số hướng đi trong tương lai:

- Tiếp tục thu thập dữ liệu để tăng độ chính xác, tin cậy cho mô hình và không cần phải phụ thuộc vào các xử lý hình ảnh đơn thuần.
- Có thể phát triển, đóng gói dạng phần mềm.

–

3. KẾT LUẬN

Nhìn chung chúng tôi đã thực hiện được hầu hết các tiêu chí của đồ án như: xây dựng được bộ tiền xử lý dữ liệu, xây dựng được bộ khung để đọc thẻ cũng như đã sử dụng nhiều các hàm thư viện có sẵn.

Tuy kết quả đạt được chưa được như mong đợi, chúng tôi đã có hướng đi đúng cho đồ án và thực hiện một cách ổn thoả.

Tuy đã đạt được mục tiêu ban đầu nhưng một số điểm chưa hài lòng như: chưa sắp xếp tốt thư mục file đồ án, chưa áp dụng tốt OOP vào dự án.

Tổng kết lại thì kết quả khá là khả quan và vượt qua mong đợi của chúng tôi.

Thông qua đề tài, chúng tôi đã rút ra nhiều kiến thức về deep learning, các thuật toán sử dụng và khả năng áp dụng kĩ thuật lập trình python. Trong tương lai các kiến thức nhận được có thể áp dụng vào các dự án khác lớn hơn.

TÀI LIỆU THAM KHẢO

[1] Edge Detection Using OpenCV - Edge Detection Using OpenCV | LearnOpenCV #.

[2] 4 Point OpenCV getPerspective Transform Example - 4 Point OpenCV getPerspective Transform Example - PyImageSearch.

[3] Phạm Duy Tùng - Contour - Phạm Duy Tùng Machine Learning Blog (phamduytung.com) – 24/5/2019.

[4] Finding extreme points in contours with OpenCV - Finding extreme points in contours with OpenCV - PyImageSearch.

[5] Nguyễn Chiến Thắng - Nhận dạng Tiếng Việt bằng thư viện Tesseract OCR- Mì AI (miai.vn) – 22/8/2019.

[6] Bui Tien Tung - U-net : Kiến trúc mạnh mẽ cho Segmentation (viblo.asia) – 15/1/2020.

[7] Pham Dinh Khanh - Khoa học dữ liệu (phamdinhkhanh.github.io) – 23/8/2020.

[8] Python Code Examples for mean pixel accuracy - Python mean pixel accuracy (programcreek.com).

PHỤ LỤC

PHỤ LỤC PHÂN CÔNG NHIỆM VỤ

STT	Thành viên	Nhiệm vụ
1	Huỳnh Nhật Nam	Tìm hiểu mô hình Unet, áp dụng mô hình, dán nhãn. Xây dựng hàm loss, accuracy, train và giám sát mô hình. Thay đổi trọng số, sử dụng mô hình sau khi huấn luyện để segmentation hình ảnh của nhóm. Xây dựng module Prediction và file using predict.py.
2	Huỳnh Hữu Hào	Thu thập dữ liệu cho mô hình Unet. Xây dựng module PreProcessing và file main.py. Tìm hiểu và sử dụng Tesseract OCR. Thay đổi các thông số để phù hợp với bài toán.