Web Trích Xuất Thông Tin ảnh thẻ CCCD

Đinh Nhật Thành

Ngày 29 Tháng 2 Năm 2024

Mục lục

1 Hệ Thống Trích Xuất Thông Tin Từ Ảnh Thẻ CCCD

1.1 Mô Tả Công Việc

Xây dựng hệ thống trích xuất thông tin từ ảnh thẻ CCCD Việt Nam 10 số ID.

1.2 Phương Pháp Thực Hiện

- **Object Detection và YOLO:** Tìm hiểu về IoU, NMS, mAP và cách thức hoạt động của YOLO.
- **Thực hành Object Detection và Chuẩn bị Dữ liệu:** Thực hành với IoU, NMS, mAP.
- **Tham khảo các tài liệu:**
  - ”Trích xuất thông tin từ chứng minh thư”[1]
  - ”Alignment ảnh chứng minh thư với PyTorch”[2]
  - ”Invoice Information ExtracExtraction”[3]
  - ”Chinh phục bài toán Object Detection với Tensorflow V2 API trong 5 phút”[4]
- Thiết kế kiến trúc hệ thống cơ bản cho cả huấn luyện và triển khai
- Tìm hiểu và chọn dataset và mô hình huấn luyện. Làm sạch và gán nhãn dữ liệu, Data Augumentation, huấn luyện mô hình nhận diện 4 góc ảnh CCCD.
- **Tối ưu và Triển khai Backend:** Cải thiện dataset, tối ưu mô hình với xử lý song song và CUDA, triển khai backend với FastAPI.
- **Nhận diện ROI và Tiền xử lý ảnh:** Chọn lọc, chỉnh sửa dataset. Huấn luyện mô hình nhận diện ROI, cải thiện tiền xử lý ảnh để làm nổi bật chữ.
- Cải thiện kiến trúc hệ thống cho tích hợp Front-End

- **Tích hợp OCR và Hoàn thiện Hệ thống:** Tích hợp VietOCR, kết nối CSDL, tích hợp API lên frontend, kiểm thử và hoàn thiện hệ thống.

Cấu trúc dự án

Dự án được tổ chức thành các thư mục và module rõ ràng, với sự phân tách trách nhiệm rõ ràng:

- **`app` (thư mục gốc):** Chứa file `app.py`, file chính điều phối toàn bộ hoạt động của ứng dụng.
- **`database`:** Chứa các file liên quan đến tương tác CSDL:
  - `database.py`: Thiết lập kết nối CSDL và cung cấp cơ chế dependency injection.
  - `models.py`: Định nghĩa các data model cho các bảng (`ocr_texts` và `detection_logs`).
- **`templates`:** Chứa các file liên quan đến giao diện người dùng (frontend): `index.html`, `scripts.js`, và `styles.css`.
- **`utils`:** Chứa các module tiện ích:
  - `detect`: Thư mục hiện tại không có file.
  - `calculate_missed_corners.py`: Tính toán các góc bị thiếu của thẻ CCCD.
  - `crop_image.py`: Thực hiện quá trình cắt và chỉnh phối cảnh ảnh thẻ CCCD.
  - `database_operations.py`: Chứa các hàm liên quan đến database (lưu trữ, log).
  - `image_processing.py`: Chứa các hàm tiền xử lý ảnh, phát hiện vùng chứa thẻ và vùng văn bản.

- `IOU.py`: Tính toán Intersection Over Union để đánh giá các bounding box.
- `NMS.py`: Thực hiện thuật toán Non-Maximum Suppression.
- `ocr_processing.py`: Các hàm liên quan đến xử lý OCR (lưu ảnh, xử lý và lưu kết quả trích xuất văn bản).

- **`validation`:** Nơi lưu trữ các ảnh đã xử lý trong quá trình kiểm thử.
- **`weights`:** Chứa các file trọng số đã train của các mô hình học sâu.
- **Các file khác:** `app.py`, `.gitignore`, `corners-config.yaml`, `roi-config.yaml`, `run.sh`, `testrun.py`.

1.3 Pipeline xử lý và logic lưu CSDL

1. **Đầu vào:** Ảnh thẻ CCCD gốc (có thể đã được gán nhãn).
2. **Triển Khai Hệ Thống:** Hệ thống kiểm tra tính thích ứng của mô hình YOLO11n, VietOCR với các weights từng tương ứng. Kết nối database và các thư viện liên quan.
3. **Module 1: Text Area Detection (Phát hiện vùng chứa ID Card):**
   - **`image_processing.py` → `warp_image(image, file_name)`:**
     - **Tiền xử lý ảnh:** `resize_image`, chuyển ảnh xám, CLAHE, chuyển về BGR.
     - **Phát hiện 4 góc thẻ:** Sử dụng weight `yolov11s-4corners.pt`.
     - **Cắt và chỉnh phối cảnh (`crop_image.py` → `crop_image()`):**
       - Lọc bounding box bằng NMS, xác định các điểm góc, tái tạo góc (nếu cần), annotation ảnh, chỉnh phối cảnh.
   - **Đầu ra:** Ảnh vùng thẻ CCCD đã được cắt và chỉnh phối cảnh.

4. **Module 2: Text Region Detection (Phát hiện các vùng văn bản):**
    - `image_processing.py` → `get_roi(image)`:
        - **Tiền xử lý ảnh cho vùng văn bản:** Chuyển ảnh xám, xóa bóng, làm mờ, cân bằng độ sáng và độ tương phản, chuyển về BGR.
        - **Phát hiện vùng chứa văn bản:** Sử dụng `yolo11n-roi-30.pt`.
        - **Lọc bounding box bằng NMS.**
    - **Đầu ra:** Danh sách các vùng chứa văn bản (bounding box).
5. **Module 3: Text Extraction (Trích xuất thông tin từ các vùng văn bản):**
    - `app.py`:
        - **OCR (VietOCR):** Sử dụng mô hình VietOCR để nhận dạng và trích xuất nội dung văn bản từ các vùng đã detect.
        - **Xử lý kết quả OCR (`ocr_processing.py` → `regex_ocr()`):**
            - **Lấy thông tin lớp:** Xác định loại thông tin từ class id của bounding box (`roi_config`).
            - **Định dạng ngày tháng:** Chuyển đổi các ngày tháng năm về dạng `dd/mm/yyyy` nếu có thể nhận dạng, hoặc đặt thành `Không Thời Hạn` nếu không nhận dạng được hoặc ít hơn 8 số.
            - **Xóa từ trùng lặp:** Sử dụng `remove_duplicate_words_before_comma.py` để loại bỏ các từ trùng lặp trước dấu phẩy (,) hoặc chấm (.).
            - **Chỉnh sửa lỗi OCR:** Sử dụng `resub` để sửa các lỗi chính tả phổ biến (ví dụ, ”Hà Nôi”thành ”Hà Nội”).
            - **Kết hợp các kết quả:** Kết hợp các kết quả OCR của cùng

loại thông tin lại thành 1 chuỗi.

* **Tạo output:** Kết quả sau cùng sẽ được đưa vào từ điển chứa các key `id`, `name`, `dob`, `gender`, `nationality`, `origin_place`, `current_place`, và `expire_date`.

– **Lưu trữ thông tin vào CSDL (`database_operations.py` → `save_to_db()`):**

* Dữ liệu văn bản đã xử lý từ output của `regex_ocr` được tạo thành đối tượng `OCRText` và lưu vào bảng `ocr_texts`.

– **Lưu log vào CSDL (`database_operations.py` → `save_db_log`):**

* Ghi lại log về quá trình trích xuất và lưu dữ liệu vào bảng `detection_logs`, bao gồm cả thành công và thất bại.

- **Hiển thị:** Dữ liệu từ bảng `ocr_texts` có thể được truy vấn và hiển thị trên giao diện người dùng.

Logic lưu trữ dữ liệu vào CSDL chi tiết

- **Bảng `ocr_texts`:**
  – **Mục đích:** Lưu trữ thông tin chi tiết được trích xuất từ ảnh thẻ CCCD sau khi xử lý OCR, có các cột: `id_text`, `id_number`, `name`, `dob`, `gender`, `nationality`, `origin_place`, `current_place`, `expire_date`.
  – **Vị trí lưu:** Dữ liệu được lưu vào bảng này sau khi quá trình OCR và trích xuất thông tin hoàn tất, tại hàm `save_to_db` trong `database_operations.py`, được gọi trong `app.py`, sử dụng output từ `regex_ocr`.
- **Bảng `detection_logs`:**
  – **Mục đích:** Lưu trữ log về quá trình xử lý ảnh thẻ CCCD, bao gồm

trạng thái thành công, thất bại, và các thông báo liên quan, có các cột: `log_id`, `timestamp`, `status`, `message`, `file_name`.

- **Vị trí lưu:** Dữ liệu được lưu vào bảng này trong quá trình xử lý ảnh, tại các hàm `save_db_log` được gọi từ các khối `try-except` trong `app.py` và sau khi lưu data thành công.

1.4 Khó khăn và cách khắc phục

1.4.1 Khó khăn gặp phải

- **Tiền xử lý ảnh (Preprocessing):**
  - *Biến đổi ánh sáng và độ tương phản*: Ảnh CCCD có thể được chụp trong các điều kiện ánh sáng khác nhau, gây khó khăn cho việc phát hiện các góc và vùng văn bản.
  - *Nhiễu ảnh*: Ảnh có thể bị nhiễu do chất lượng camera hoặc điều kiện chụp, ảnh hưởng đến độ chính xác của các bước xử lý tiếp theo.
- **Cắt ảnh (Crop Image):**
  - *Góc khuất*: Một số góc của thẻ CCCD có thể bị che khuất hoặc không rõ ràng do cách chụp, gây khó khăn cho việc xác định chính xác các điểm góc.
  - *Biến dạng phối cảnh*: Ảnh CCCD có thể bị biến dạng do góc chụp không thẳng, đòi hỏi phải có phép biến đổi phối cảnh chính xác.
- **Tính góc thiếu (Calculate Missing Corners):**
  - *Sai số tích lũy*: Sai số trong việc xác định các góc có thể dẫn đến sai số lớn trong việc tính toán góc thiếu.
  - *Trường hợp quá nhiễu*: Trong trường hợp ảnh có quá nhiều nhiễu, hoặc các bounding box được detect không chính xác, việc tái tạo có

thể cho ra kết quả sai.

- **Xử lý OCR (regex_ocr):**
  - *Lỗi chính tả và định dạng*: Kết quả OCR có thể chứa các lỗi chính tả, lỗi định dạng ngày tháng hoặc các ký tự đặc biệt không mong muốn.
  - *Từ trùng lặp*: OCR có thể trả về các từ trùng lặp, đặc biệt là trong trường hợp các vùng văn bản nhỏ.
- **Thu thập dữ liệu:**
  - *Số lượng dữ liệu hạn chế*: Khó khăn trong việc thu thập đủ dữ liệu ảnh thẻ CCCD để huấn luyện các mô hình học sâu.
  - *Đa dạng dữ liệu*: Ảnh thẻ CCCD có thể có nhiều biến thể về chất lượng, ánh sáng và góc chụp.

1.4.2 Cách khắc phục

- **Tiền xử lý ảnh (Preprocessing):**
  - *Cân bằng histogram và tăng cường độ tương phản*: Sử dụng CLAHE và các hàm cân bằng độ sáng và độ tương phản tự động để cải thiện chất lượng ảnh và làm rõ các chi tiết.
  - *Lọc nhiễu*: Sử dụng các bộ lọc trung bình, Gaussian hoặc morphological operations để giảm nhiễu ảnh.
- **Cắt ảnh (Crop Image):**
  - *Reconstruction góc*: Sử dụng hàm `calculate_missing_with_homography` để tái tạo góc bị thiếu dựa trên các góc đã detect được, giúp xác định chính xác vùng thẻ.
  - *Perspective Transform*: Sử dụng hàm `perspective_transformation`

để chỉnh phối cảnh ảnh, đưa ảnh về dạng chính diện, giảm thiểu sai số do biến dạng.

- **Tính góc thiếu (Calculate Missing Corners):**
  - *Tính toán trung điểm và clamp kết quả*: Sử dụng các tính toán hình học dựa trên trung điểm để ước lượng vị trí góc và đảm bảo kết quả luôn nằm trong phạm vi ảnh.
  - *Input Validation:* Thêm input validation để kiểm tra các tọa độ của bounding box, và không tiếp tục nếu các tọa độ không hợp lệ.
- **Xử lý OCR (regex_ocr):**
  - *Sử dụng regular expressions (regex)*: Dùng regex để chuẩn hóa định dạng ngày tháng, sửa lỗi chính tả thông thường.
  - *Loại bỏ từ trùng lặp*: Sử dụng hàm `remove_duplicate_words_before_comma` để loại bỏ các từ trùng lặp xuất hiện trước dấu phẩy.
- **Thu thập dữ liệu:**
  - *Data Augmentation (Roboflow):* Sử dụng Roboflow để tự động gán nhãn và tăng cường dữ liệu, tạo thêm các biến thể của ảnh (xoay, lật, thay đổi độ sáng).
  - *Auto-labeling (Roboflow):* Dùng Roboflow auto-labeling để gán nhãn nhanh chóng trên một phần dữ liệu, giảm thời gian và công sức cho việc thu thập dữ liệu.

1.5 Kết Quả Đạt Được

- Tự động hóa quá trình trích xuất thông tin từ ảnh thẻ CCCD một cách chính xác và hiệu quả.

- Đảm bảo dữ liệu được trích xuất và lưu trữ có cấu trúc, dễ dàng sử dụng và truy vấn.
- Cung cấp một hệ thống linh hoạt, dễ dàng mở rộng và bảo trì.

2 KẾT LUẬN VÀ KIẾN NGHỊ

2.1 Kết luận

Đây là 1 kì thực tập đáng nhớ và đáng học hỏi. Dù em không được chia đúng ban để được tư vấn, có 1 mentor giúp và hỗ trợ thực sự. Dù ban đầu có loạn định hướng lẫn kiến thức thực sự cần học, em đã cố gắng tìm hiểu phương pháp và các khóa học để cải thiện kiến thức nền tảng chuẩn bị cho kì chuyên ngành tới và hoàn thành dự án kì thực tập đề ra.

2.2 Kiến nghị

Ban thực tập nên được chia theo thái độ và mong muốn của sinh viên để đạt được hiệu quả học tập tốt nhất. Em cũng nghĩ tiền học của cũng nên giảm trong kì thực tập vì phần lớn kiến thức và công sức hoàn thành kì này cũng là của sinh viên hoặc chất lượng của kì thực tập được cải thiện để khớp với sô tiền học của kì thực tập.

3 Tài Liệu Tham Khảo

Tài liệu

[1] B. Q. Manh, *Trích xuất thông tin từ chứng minh thư*, Tutorial on Viblo, Accessed: 2025-01-01, 2023. **url**: https://viblo.asia/p/trich-xuat-thong-tin-tu-chung-minh-thu-bJzKmaRwK9N.

[2] P. V. Toàn, *[PyTorch Tutorial] #3 - Alignment ảnh chứng minh thư với PyTorch. Hướng dẫn dễ như ăn kẹo*, Tutorial on Viblo, Accessed: 2025-01-01, 2023. **url**: https://viblo.asia/p/pytorch-tutorial-3-alignment-anh-chung-minh-thu-voi-pytorch-huong-dan-de-nhu-an-keo-4dbZNJ8mZYM.

[3] S. Ghosh, *Invoice Information Extraction*, Article on Medium (Analytics Vidhya), Accessed: 2025-01-01, 2023. **url**: https://medium.com/analytics-vidhya/invoice-information-extraction-using-ocr-and-deep-learning-b79464f54d69.

[4] B. Q. Manh, *Chinh phục bài toán Object Detection với Tensorflow V2 API trong 5 phút*, Tutorial on Viblo, Accessed: 2025-01-01, 2023. **url**: https://viblo.asia/p/chinh-phuc-bai-toan-object-detection-voi-tensorflow-v2-api-trong-5-phut-1VgZvMRrKAw.