BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC HOA SEN
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO
ĐỒ ÁN LẬP TRÌNH MÁY HỌC

Tên đề tài:

Giảng viên hướng dẫn : Bùi Ngọc Lê
Thời gian thực hiện : Từ 16/09/2024 đến 21/12/2024
Nhóm sinh viên thực hiện : Phạm Thị Yến Ngọc MSSV: 22205600
: Lương Thảo Nhi MSSV: 22203891
Số nhóm : 1

THÁNG 12 / NĂM 2024

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC HOA SEN
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO
ĐỒ ÁN LẬP TRÌNH MÁY HỌC

Tên đề tài:

Giảng viên hướng dẫn : Bùi Ngọc Lê
Thời gian thực hiện : Từ 16/09/2024 đến 21/12/2024
Nhóm sinh viên thực hiện : Phạm Thị Yến Ngọc MSSV: 22205600
: Lương Thảo Nhi MSSV: 22203891
Số nhóm : 1

THÁNG 12 / NĂM 2024

TRÍCH YẾU

Trong thời đại công nghệ phát triển như hiện nay, việc tự động hóa trong xử lý tài liệu và số hóa thông tin trở thành nhu cầu thiết yếu, đặc biệt đối với các ngành yêu cầu quản lý dữ liệu lớn như hành chính, tài chính, và kế toán. Cùng với sự gia tăng của các tài liệu số hóa, việc trích xuất thông tin tự động, đặc biệt là ngày, tháng, năm, trở thành một thách thức quan trọng trong việc tối ưu hóa quy trình làm việc. Hệ thống nhận diện ngày, tháng, năm từ ảnh ra đời như một giải pháp hiệu quả để giải quyết vấn đề này.

Hệ thống nhận diện ngày, tháng, năm từ ảnh không chỉ giúp giảm thiểu thời gian và công sức trong việc nhập liệu mà còn tăng cường độ chính xác trong xử lý thông tin. Với sự tích hợp của các công nghệ hiện đại như YOLO, Tesseract OCR, và mô hình học sâu MNIST, hệ thống có khả năng nhận diện chính xác các vùng thông tin thời gian trên tài liệu và chuyển đổi chúng thành dữ liệu số một cách tự động. Sự kết hợp này mang lại nhiều lợi ích từ việc giảm sai sót trong nhập liệu thủ công đến tăng hiệu suất làm việc.

Đồ án này tập trung nghiên cứu và phát triển hệ thống nhận diện ngày, tháng, năm từ ảnh, bao gồm các bước xử lý ảnh, phát hiện vùng chứa thông tin, và nhận diện chữ số. Hệ thống được thiết kế với khả năng mở rộng, có thể ứng dụng không chỉ trong nhận diện ngày tháng mà còn trong các lĩnh vực khác như nhận diện văn bản, nhận diện chữ viết tay, hoặc phân tích tài liệu tổng thể. Mặc dù nội dung chính của đồ án tập trung vào trích xuất thông tin thời gian, các giải pháp được đề xuất hoàn toàn có thể mở rộng để đáp ứng nhu cầu xử lý tài liệu đa dạng.

Với những lợi ích rõ rệt và khả năng ứng dụng rộng rãi, hệ thống nhận diện ngày, tháng, năm từ ảnh hứa hẹn sẽ trở thành một công cụ hữu ích trong việc hiện đại hóa quản lý thông tin, đáp ứng nhu cầu ngày càng cao về tự động hóa và số hóa trong xã hội phát triển hiện nay. Hệ thống này không chỉ cải thiện hiệu suất làm việc mà còn góp phần thúc đẩy sự phát triển bền vững trong quản lý tài liệu và công nghệ xử lý thông tin.

MỤC LỤC

LỜI CẢM ƠN

Chúng tôi xin chân thành cảm ơn thầy Bùi Ngọc Lê, người đã tận tình hướng dẫn và hỗ trợ chúng tôi trong suốt quá trình thực hiện đồ án này. Nhờ có sự chỉ dẫn chi tiết và những kiến thức quý báu mà thầy truyền đạt, chúng tôi đã có thể hoàn thành tốt đề tài nghiên cứu của mình.

Thầy không chỉ giúp chúng tôi hiểu rõ về chuyên môn mà còn định hướng cách giải quyết các vấn đề khó khăn, giúp nhóm từng bước tiến bộ và hoàn thiện sản phẩm. Sự tận tâm và nhiệt huyết của thầy chính là động lực lớn để chúng tôi cố gắng đạt được kết quả tốt nhất.

DANH MỤC HÌNH ẢNH

NHẬP ĐỀ

Trong thời đại kỹ thuật số, việc tự động hóa các tác vụ xử lý dữ liệu văn bản là nhu cầu thiết yếu. Trong các văn bản hành chính, hóa đơn hoặc tài liệu viết tay, ngày tháng đóng vai trò quan trọng, phục vụ việc quản lý thời gian và phân loại dữ liệu. Tuy nhiên, nhận dạng ngày tháng viết tay thường gặp khó khăn do sự đa dạng trong cách viết của con người, từ đó đặt ra bài toán tối ưu hóa nhận diện bằng công nghệ trí tuệ nhân tạo.

Đề tài tập trung vào việc áp dụng các kỹ thuật học máy và học sâu để phát triển một hệ thống nhận dạng ngày tháng viết tay chính xác, phục vụ cho nhiều ứng dụng như tự động hóa hành chính, lưu trữ hóa đơn điện tử và quản lý tài liệu.

Mục tiêu nghiên cứu: Phát triển hệ thống nhận diện ngày tháng trên văn bản viết tay, cải thiện độ chính xác thông qua các kỹ thuật học sâu (Deep Learning), ứng dụng hệ thống vào thực tế qua giao diện người dùng trực quan.

Phạm vi ứng dụng: Nhận diện ngày tháng từ văn bản viết tay như: hoá đơn thanh toán, giấy tờ hành chính, biên lai, giấy hẹn; Đưa kết quả vào cơ sở dữ liệu hoặc xuất ra định dạng sử dụng được.

Phương pháp thực hiện: Sử dụng các công nghệ và công cụ như OpenCV, OCR, mô hình CNN tư thư viện TensorFlow/Keras để xử lý ảnh và nhận dạng ký tự quang học, nhận dạng số.

1. Giới thiệu đề tài

Trong thời đại công nghệ phát triển, việc tự động hóa trong xử lý dữ liệu và số hóa thông tin trở thành nhu cầu cần thiết. Đồ án này tập trung nghiên cứu và phát triển một hệ thống nhận diện ngày, tháng, năm từ ảnh, áp dụng các kỹ thuật học sâu như YOLO, Tesseract OCR, và MNIST để trích xuất thông tin thời gian từ tài liệu viết tay một cách chính xác và nhanh chóng.

1.1. Các thư viện sử dụng

os: Thao tác với hệ thống tệp và thư mục; Cung cấp chức năng quản lý file như tạo, xóa, di chuyển thư mục.

cv2 (OpenCV): Xử lý và thao tác với ảnh; Hỗ trợ các tác vụ như phát hiện cạnh, lọc ảnh, nhận diện đối tượng.

shutil: Quản lý tệp và thư mục cấp cao hơn os; Dùng để sao chép, di chuyển, đổi tên hoặc xóa file/thư mục.

numpy: Xử lý dữ liệu số học hiệu quả với mảng (array); Hỗ trợ các phép toán như đại số tuyến tính, ma trận.

pytesseract: Nhận diện ký tự từ hình ảnh (OCR); Tích hợp Tesseract OCR để chuyển hình ảnh thành văn bản.

keras: Thư viện deep learning đơn giản hóa việc xây dựng và huấn luyện mô hình; Hỗ trợ dự đoán với các bộ dữ liệu như MNIST.

PyQt5: Framework phát triển giao diện đồ họa (GUI)

ultralytics: Sử dụng YOLOv8 để nhận diện và tạo bounding box cho đối tượng; Phục vụ cho các ứng dụng về computer vision và phát hiện đối tượng.

1.2. Hệ thống file

Thư mục lưu ảnh gốc: Thư mục images/val/ lưu trữ ảnh mà người dùng đã chọn để xử lý.

Thư mục lưu kết quả: Thư mục result/ lưu các ảnh sau khi đã được xử lý và nhận diện.

Thư mục debug: Thư mục debug/ lưu các bước trung gian, bao gồm ảnh đã tách chữ số và ảnh resize.

2. Cơ sở lý thuyết

2.1. Xử lý ảnh cơ bản

Xử lý ảnh là bước quan trọng đầu tiên trong hệ thống nhận diện ngày, tháng, năm. Mục tiêu của bước này là chuẩn bị dữ liệu đầu vào sao cho phù hợp nhất để các thuật toán xử lý tiếp theo đạt hiệu quả cao. Các bước chính bao gồm:

Lọc nhiễu: Sử dụng Gaussian Blur để làm mịn hình ảnh, giảm thiểu các yếu tố không mong muốn như hạt nhiễu, đảm bảo các đặc trưng quan trọng không bị mất đi.

Chuyển đổi ảnh: Hình ảnh đầu vào thường được chuyển đổi sang thang độ xám (Grayscale) để giảm độ phức tạp và nhị phân hóa (Binarization) nhằm phân tách nền và chữ số rõ ràng hơn.

Phát hiện đường viền (Edge Detection): Áp dụng các thuật toán như Canny Edge Detection để phát hiện các đường viền, hỗ trợ quá trình cắt vùng quan tâm (ROI).

Cắt và chuẩn hóa: Sử dụng thuật toán Contour để xác định và cắt vùng chứa thông tin thời gian, sau đó chuẩn hóa kích thước để phục vụ nhận diện.

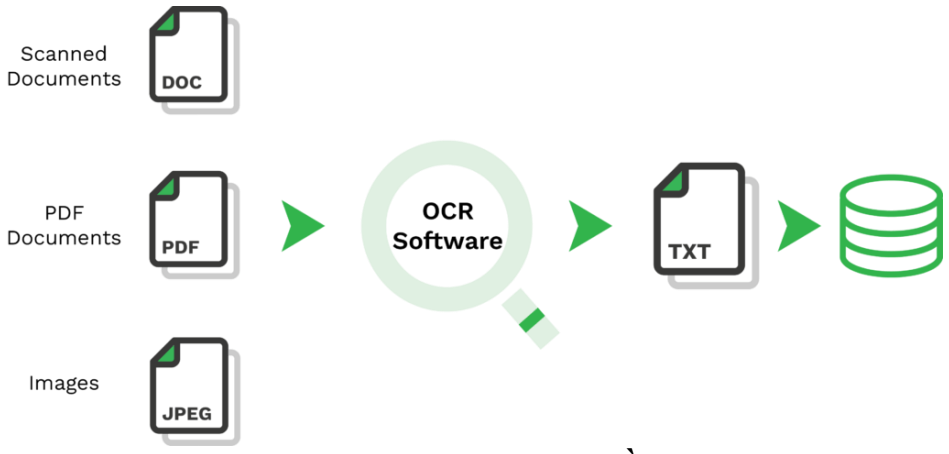
2.2. Nhận diện ký tự quang học (OCR)

OCR (Optical Character Recognition) là công nghệ chuyển đổi hình ảnh chứa văn bản thành dữ liệu dạng văn bản số. Trong hệ thống, Tesseract OCR được sử dụng để nhận diện ký tự và số từ hình ảnh.

Nguyên tắc hoạt động của OCR:

- Phân đoạn (Segmentation): Tách hình ảnh thành từng phần nhỏ chứa ký tự.
- Trích xuất đặc trưng (Feature Extraction): Phân tích và trích xuất các đặc trưng như đường cong, góc cạnh.
- Nhận diện (Recognition): So sánh các đặc trưng với dữ liệu mẫu để xác định ký tự.

Tùy chỉnh Tesseract OCR: Tesseract được tinh chỉnh để tối ưu nhận diện các ký tự trong vùng ngày, tháng, năm, phù hợp với kiểu dữ liệu mà hệ thống yêu cầu.



Hình 1 - Quy trình xử lý bằng OCR

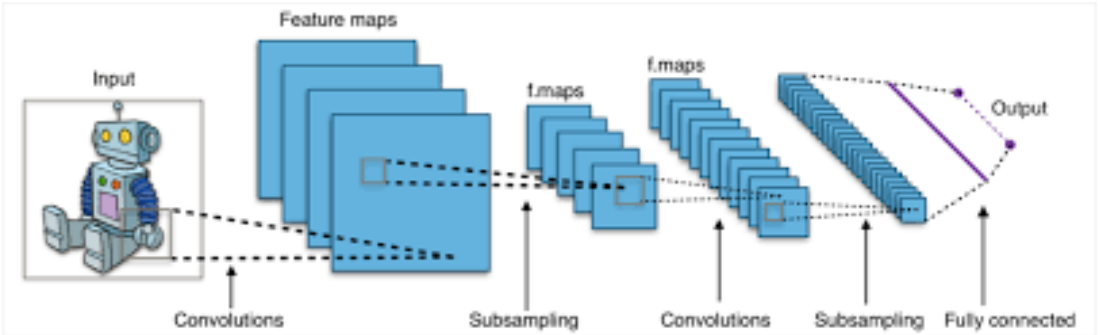
2.3. Mạng nơ-ron tích chập (CNN)

Mạng nơ-ron (Neural Network) là một trong những công nghệ cốt lõi được sử dụng trong hệ thống. Đặc biệt, mạng nơ-ron tích chập (CNN - Convolutional Neural Network) là loại mạng được tối ưu hóa cho xử lý ảnh.

Cấu trúc CNN: CNN bao gồm các lớp chính:

- Lớp tích chập (Convolution): Phát hiện các đặc trưng của ảnh như đường nét, góc cạnh thông qua các bộ lọc (filters).
- Lớp gộp (Pooling): Giảm kích thước của dữ liệu, giữ lại các đặc trưng quan trọng, giúp giảm tải tính toán.
- Lớp liên kết đầy đủ (Fully Connected): Tổng hợp thông tin từ các đặc trưng để đưa ra kết quả dự đoán.

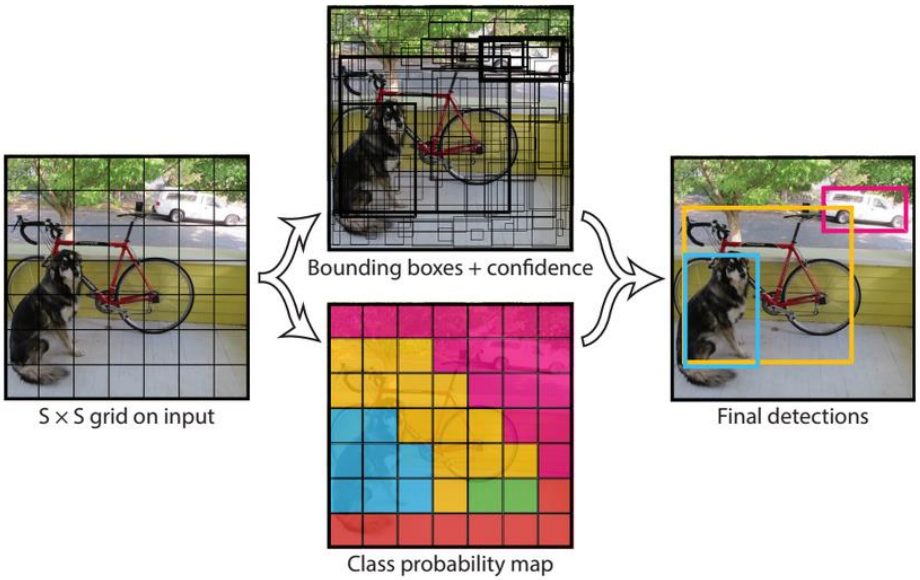
Ứng dụng CNN: Trong hệ thống này, CNN được sử dụng để nhận diện từng chữ số trong vùng ngày, tháng, năm. Các mô hình được đào tạo trên bộ dữ liệu như MNIST giúp tăng độ chính xác khi nhận dạng chữ số viết tay.



Hình 2 - Cấu trúc của mạng nơ-ron tích chập (CNN)

2.4. YOLO và phát hiện đối tượng (Object Detection)

YOLO (You Only Look Once) là một mô hình phát hiện đối tượng nhanh và hiệu quả, được sử dụng để phát hiện vùng chứa thông tin thời gian trong ảnh.



Hình 3 – Cách hoạt động của YOLO

Cách hoạt động:

- Ảnh đầu vào được chia thành các lưới (grids).
- Từng ô lưới được phân tích để dự đoán bounding box và phân loại đối tượng.
- Các bounding box được tối ưu hóa để xác định chính xác vị trí và kích thước của vùng chứa thông tin.

Ưu điểm của YOLO:

- Xử lý nhanh, phù hợp với hệ thống thời gian thực.
- Phát hiện chính xác cả trong trường hợp ảnh có độ phức tạp cao.

Ứng dụng: Trong đề tài này, YOLO được sử dụng để tìm vùng ngày, tháng, năm

2.5. Gaussian Blur

Gaussian Blur là một phương pháp làm mờ ảnh phổ biến trong xử lý ảnh, giúp giảm nhiễu và chi tiết không cần thiết, từ đó cải thiện chất lượng phân tích ảnh.

Hình 4 – Cách hoạt động của Gaussian Blur

Cách hoạt động:

- Gaussian Blur áp dụng hàm Gaussian hai chiều để tính toán mức độ làm mờ tại mỗi điểm ảnh.
- Mỗi điểm ảnh được thay thế bởi một giá trị trung bình trọng số của các điểm ảnh xung quanh, với trọng số được xác định bởi hàm Gaussian.
- Kernel (lõi lọc) Gaussian được sử dụng để thực hiện phép tính này, với kích thước và độ lệch chuẩn (sigma) quyết định mức độ làm mờ.

Ưu điểm của Gaussian Blur:

- Giảm nhiễu: Loại bỏ các chi tiết nhỏ không cần thiết trong ảnh mà không làm mất quá nhiều thông tin quan trọng.
- Mịn màng hơn so với các bộ lọc làm mờ khác: Gaussian Blur không tạo ra các cạnh sắc hay hiện tượng răng cưa như một số bộ lọc khác.
- Tiền xử lý hiệu quả: Tăng độ chính xác khi thực hiện các tác vụ như phát hiện cạnh, nhận diện đối tượng.

Ứng dụng:

Trong đề tài này, Gaussian Blur được sử dụng để:

- Loại bỏ nhiễu nền: Làm mờ các chi tiết nhỏ gây nhiễu, giúp các bước xử lý tiếp theo chính xác hơn.
- Chuẩn bị ảnh đầu vào: Tăng hiệu quả của các thuật toán phát hiện vùng chứa thông tin ngày, tháng, năm.

2.6. Cắt vùng quan tâm (ROI)

Cắt Vùng Quan Tâm (Region of Interest - ROI) là một bước quan trọng trong xử lý ảnh, được sử dụng để tập trung vào các vùng chứa thông tin cần phân tích, từ đó giảm tải xử lý và tăng độ chính xác của hệ thống.

Cách hoạt động:

- Xác định vùng quan tâm: Dựa vào các thuật toán phát hiện, như YOLO hoặc các phương pháp khác, để tìm bounding box bao quanh vùng cần phân tích.
- Trích xuất vùng: Bounding box được sử dụng để cắt một phần ảnh, tạo ra một vùng con chứa thông tin cần xử lý.
- Tiền xử lý vùng ROI: Vùng trích xuất được chuẩn hóa (chẳng hạn thay đổi kích thước hoặc làm mờ) để phù hợp với yêu cầu của các bước xử lý tiếp theo.

Ưu điểm của cắt ROI:

- Giảm tải tính toán: Hạn chế xử lý toàn bộ ảnh, chỉ tập trung vào vùng có thông tin cần thiết.
- Tăng độ chính xác: Loại bỏ các nhiễu bên ngoài vùng quan tâm, cải thiện hiệu suất của các mô hình phân tích.
- Tích hợp linh hoạt: ROI có thể dễ dàng áp dụng cho nhiều bài toán xử lý ảnh khác nhau, từ nhận diện đối tượng đến OCR.

Trong đề tài này, việc cắt ROI được sử dụng để:

- Xác định vùng ngày, tháng, năm: Dựa trên bounding box được phát hiện bởi YOLO, các vùng chứa thông tin thời gian được cắt ra để xử lý riêng.
- Tăng hiệu quả nhận diện chữ số: Cung cấp dữ liệu đầu vào chính xác hơn cho mô hình nhận diện MNIST hoặc OCR.

2.7. MNIST

MNIST là một tập dữ liệu phổ biến được sử dụng để huấn luyện và kiểm tra các mô hình nhận diện chữ số viết tay.

Hình 5 – Các ảnh mẫu từ tập thử nghiệm MNIST

Cách hoạt động:

- Dữ liệu đầu vào: Tập dữ liệu MNIST chứa 70.000 ảnh grayscale của chữ số từ 0 đến 9, kích thước mỗi ảnh là 28x28 pixel.
- Tiền xử lý: Ảnh được chuẩn hóa về khoảng giá trị [0,1] và chuyển đổi thành các ma trận đầu vào cho mô hình học máy.
- Huấn luyện mô hình: Các mô hình như mạng nơ-ron tích chập (CNN) được sử dụng để học đặc trưng của các chữ số từ dữ liệu đầu vào.
- Dự đoán: Mô hình phân loại một ảnh đầu vào thành một trong 10 lớp (từ 0 đến 9).

Ưu điểm của MNIST:

- Dễ tiếp cận: Là tập dữ liệu cơ bản nhưng hiệu quả để nghiên cứu nhận diện chữ số viết tay.
- Độ tin cậy cao: Cung cấp kết quả tốt khi tích hợp với các mô hình mạng nơ-ron hiện đại như CNN.
- Ứng dụng rộng rãi: MNIST thường được sử dụng để kiểm chứng và so sánh hiệu suất của các mô hình học máy.

Ứng dụng:

Trong đề tài này, MNIST được sử dụng để:

- Nhận diện chữ số trong vùng ngày và tháng: Tập dữ liệu huấn luyện mô hình nhận diện các chữ số tách ra từ ảnh chứa thông tin thời gian.
- Kết hợp với các phương pháp khác: Đảm bảo tính chính xác cao trong việc phân loại các chữ số được cắt từ ROI (vùng quan tâm).

2.8. Các công nghệ hỗ trợ khác

Hệ thống còn sử dụng một số công nghệ và thư viện hỗ trợ để đảm bảo hiệu quả và độ chính xác:

OpenCV: Thư viện xử lý ảnh mạnh mẽ với các công cụ như chuyển đổi ảnh, phát hiện cạnh, và thao tác với contour.

PyQt5: Xây dựng giao diện người dùng (GUI), giúp hệ thống trở nên trực quan, dễ sử dụng.

Keras: Một thư viện deep learning đơn giản, hỗ trợ việc thiết kế và huấn luyện các mô hình CNN.

Ultralytics: Thư viện YOLO, giúp thực hiện phát hiện đối tượng nhanh chóng và chính xác.

2.9. Bộ dữ liệu (Dataset)

Bộ dữ liệu là yếu tố quyết định đến hiệu quả của hệ thống. Trong đề tài này:

Dữ liệu huấn luyện: Sử dụng MNIST cho nhận dạng chữ số và bộ dữ liệu tùy chỉnh chứa hình ảnh ngày, tháng, năm từ các tài liệu thực tế.

Dữ liệu kiểm thử: Các hình ảnh chứa ngày tháng từ văn bản hành chính.

Quá trình gán nhãn: Hình ảnh được gán nhãn thủ công để đảm bảo dữ liệu huấn luyện chính xác.

3. Các vấn đề trong đề tài

3.1. Quy trình xử lý tổng thể

Quy trình xử lý được xây dựng gồm các bước chính:

- Tiền xử lý ảnh đầu vào để nâng cao chất lượng.
- Phát hiện vùng chứa thông tin thời gian trên ảnh bằng YOLO.
- Xử lý vùng ảnh đã phát hiện để nhận diện chữ số.
- Tổng hợp kết quả và xuất dữ liệu ra dạng văn bản số.

3.2. Tiền xử lý

Tiền xử lý là bước quan trọng để chuẩn bị dữ liệu đầu vào cho mô hình nhận diện, giúp giảm thiểu nhiễu và đảm bảo mô hình hoạt động hiệu quả. Dưới đây là các bước tiền xử lý và cách vận hành.

3.2.1. Lọc nhiễu

Mục đích: Giảm thiểu các nhiễu ảnh (noise) như hạt nhỏ, vùng đốm không liên quan, giúp cải thiện chất lượng nhận diện.

Phương pháp:

Sử dụng Gaussian Blur, một bộ lọc làm mờ ảnh, để làm mịn các vùng nhiễu.

Gaussian Blur hoạt động bằng cách tính trung bình trọng số các điểm ảnh lân cận, với điểm trung tâm được ưu tiên cao hơn. Kích thước kernel (ví dụ: 5x5) được tùy chỉnh để đảm bảo cân bằng giữa làm mờ nhiễu và giữ lại chi tiết quan trọng.

Hiệu quả:

Giảm thiểu ảnh hưởng của các nhiễu nhỏ.

Hỗ trợ cải thiện kết quả nhị phân hóa ở bước sau.

```
# Chuyển sang grayscale và binary
gray_img = cv.cvtColor(cropped_img, cv.COLOR_BGR2GRAY)

# Áp dụng Gaussian Blur để giảm nhiễu
blurred_img = cv.GaussianBlur(gray_img, (5, 5), 0)

# Nhị phân hóa ảnh
_, binary_img = cv.threshold(blurred_img, 0, 255, cv.THRESH_BINARY_INV + cv.THRESH_OTSU)
```

Hình 6 – Dùng Gaussian Blur để lọc nhiễu và nhị phân hóa ảnh

3.2.2. Chuyển đổi ảnh

Mục đích: Đơn giản hóa thông tin màu sắc và chuẩn bị ảnh để phân tích các đặc trưng cần thiết.

Các bước:

Chuyển sang thang độ xám (Grayscale): Ảnh đầu vào RGB hoặc BGR được chuyển đổi thành ảnh thang độ xám, trong đó mỗi pixel chỉ mang thông tin độ sáng (intensity) từ 0 (đen) đến 255 (trắng). Giảm kích thước dữ liệu (chỉ còn 1 kênh màu thay vì 3) và tăng tốc độ xử lý.

Nhị phân hóa (Binarization): Áp dụng thuật toán Otsu Thresholding để phân chia ảnh thành hai mức: đen (pixel giá trị 0) và trắng (pixel giá trị 255). Nhị phân hóa dựa trên ngưỡng tự động xác định, giúp tách rõ đối tượng (chữ hoặc số) khỏi nền.

```
# Đọc ảnh gốc và xử lý
img = cv.imread(new_image_path)

# Chuyển sang grayscale và binary
gray_img = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
_, binary_img = cv.threshold(gray_img, 0, 255, cv.THRESH_BINARY_INV + cv.THRESH_OTSU)

# Đảo bit ảnh (chữ đen nền trắng)
inverted_img = cv.bitwise_not(binary_img)
```

Hình 7 – Quá trình chuyển đổi ảnh

Hiệu quả:

- Làm nổi bật thông tin cần nhận diện (các ký tự).
- Chuẩn bị ảnh cho bước tách vùng quan tâm (ROI).

3.2.3. Tách vùng quan tâm (ROI)

Mục đích: Xác định và tách riêng từng vùng chứa thông tin ngày, tháng, năm để đưa vào mô hình nhận diện.

Phương pháp:

Áp dụng thuật toán Contour Detection: Tìm các đường viền (contours) của các đối tượng trong ảnh nhị phân hóa. Mỗi contour đại diện cho một vùng có thông tin cần thiết (ví dụ: chữ số ngày, tháng).

Sử dụng Bounding Box để bao quanh từng contour. Xác định tọa độ (x, y, w, h) của các bounding box. Lọc bỏ các vùng nhỏ không phải chữ số bằng cách thiết lập ngưỡng về chiều cao và chiều rộng (ví dụ: width > 5, height > 10).

Hiệu quả:

- Cắt riêng từng vùng chứa thông tin ngày, tháng, năm.
- Giảm nhiễu từ các vùng không liên quan.

```
for contour in contours:
    x, y, w, h = cv.boundingRect(contour)
    padding = 2
    if w > 5 and h > 10:  # Loại bỏ các vùng nhiễu quá nhỏ
        padding = 2
        x = max(0, x - padding)
        y = max(0, y - padding)
        w += 2 * padding
        h += 2 * padding
        digit_img = binary_img[y:y + h, x:x + w]
        digit_images.append((x, digit_img))
digit_images = sorted(digit_images, key=lambda x: x[0])  # Sắp xếp từ trái sang phải
```

Hình 8 – Tách vùng quan tâm (ROI) bằng Contour Detection

3.3. Nhận diện vùng dữ liệu ngày tháng

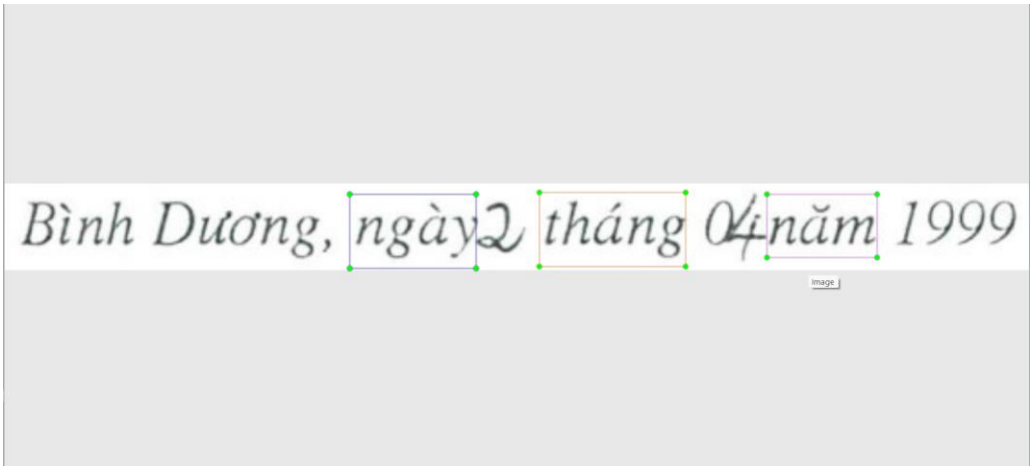
Mô hình YOLO (You Only Look Once) được huấn luyện để phát hiện các vùng chứa thông tin ngày, tháng, năm trên ảnh. Quy trình vận hành và cấu trúc của việc nhận diện vùng dữ liệu ngày tháng được thực hiện như sau:

3.3.1. Dữ liệu huấn luyện và gán nhãn

Trước tiên, ảnh được thu thập từ các nguồn thực tế (ví dụ: hóa đơn, tài liệu, chứng từ).

Bounding box của các vùng chứa chữ “ngày”, “tháng”, “năm” được vẽ thủ công bằng các công cụ như LabelImg. Trong quá trình này, từng bounding box được gán nhãn chính xác (class labels) là ngày, tháng, hoặc năm.

Dữ liệu sau khi gán nhãn được chuẩn hóa và chia thành tập huấn luyện, tập kiểm tra.



Hình 9 – Bounding box được vẽ thủ công để đưa vào YOLO nhận diện

3.3.2. Huấn luyện mô hình YOLO

Mô hình YOLO được huấn luyện trên tập dữ liệu đã được gán nhãn. YOLO học cách phát hiện các bounding box cũng như phân loại các box dựa trên nhãn (ngày, tháng, năm).

Trong quá trình huấn luyện, YOLO tối ưu hai đầu ra chính:

Bounding Box Regression: Dự đoán vị trí (tọa độ) của các bounding box chứa ngày, tháng, năm.

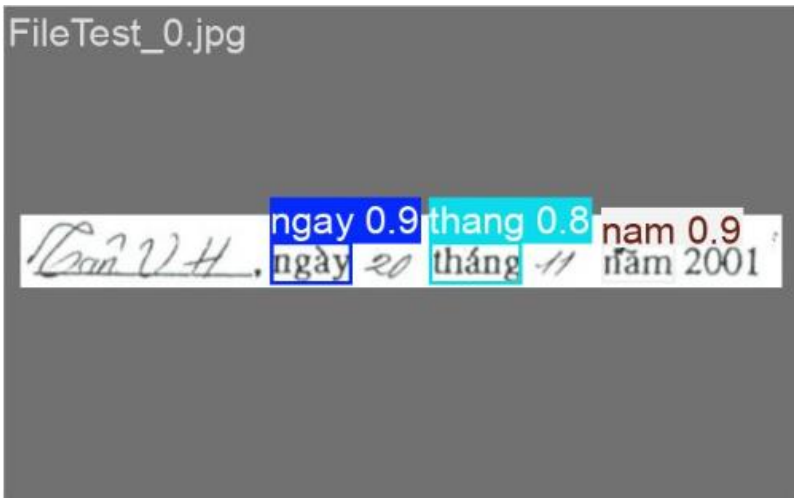
Class Prediction: Phân loại nhãn của bounding box.

3.3.3. Quá trình nhận diện trong thực tế

Khi nhận một ảnh đầu vào, mô hình YOLO chia ảnh thành một lưới và tính toán các thuộc tính như:

- Confidence Score: Xác suất một bounding box chứa thông tin quan trọng.
- Class Score: Xác suất bounding box thuộc các lớp ngày, tháng, hoặc năm.

YOLO sử dụng cơ chế Non-Maximum Suppression (NMS) để loại bỏ các bounding box dư thừa hoặc bị chồng lấn, chỉ giữ lại các box có độ chính xác cao nhất.



Hình 10 – Quá trình nhận diện các chữ ngày, tháng, năm của YOLO

3.3.4. Xử lý sau nhận diện

Sau khi YOLO xác định các bounding box, hệ thống thực hiện bước tiền xử lý:

Cắt các vùng trong ảnh tương ứng với bounding box.

Xác định thứ tự các box theo cấu trúc thông thường: ngày, tháng, năm (nếu cấu trúc bị xáo trộn, cần điều chỉnh thủ công hoặc dùng thuật toán sắp xếp).

Các vùng ảnh cắt được từ bounding box sẽ được chuyển sang các bước tiếp theo, như:

Nhận diện chữ số bằng mô hình MNIST hoặc mô hình tùy chỉnh.

Kiểm tra tính hợp lệ của thông tin ngày, tháng, năm (ví dụ: ngày phải từ 1–31, tháng từ 1–12).

3.4. Nhận dạng chữ số

3.4.1. Cắt vùng dữ liệu và chuẩn bị nhận dạng

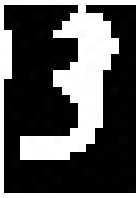
Sau khi phát hiện và cắt vùng chứa số ngày, tháng, năm (bounding box) bằng mô hình YOLO, các vùng này thường chứa một hoặc nhiều chữ số cần được xử lý

Để nhận dạng các chữ số riêng lẻ, thuật toán Contour Detection được áp dụng để tìm các đường biên xung quanh từng chữ số bên trong bounding box. Điều này cho phép hệ thống tách riêng từng chữ số, tạo điều kiện thuận lợi cho quá trình nhận dạng.

3.4.2. Chuẩn hóa kích thước

Mỗi chữ số được tách ra thường có kích thước không đồng nhất (về chiều cao, chiều rộng, hoặc tỷ lệ).

Ví dụ như ảnh số 3 dưới đây là ảnh được cắt ra từ dữ liệu gốc, vẫn chưa được chuẩn hóa.



Hình 11 – Chữ số khi chưa chuẩn hóa

Các chữ số này được chuẩn hóa bằng cách:

Resize về kích thước 28x28: Kích thước chuẩn của ảnh đầu vào phù hợp với cấu trúc của mô hình CNN đã huấn luyện.

Nhị phân hóa và chuẩn hóa cường độ pixel: Ảnh được đưa về dạng nhị phân (đen-trắng) và các giá trị cường độ pixel được chia cho 255 để nằm trong khoảng [0, 1], giúp tối ưu hóa hiệu quả tính toán của mô hình.

```
# Resize về 28x28
resized_digit = cv.resize(digit_img, (28, 28), interpolation=cv.INTER_AREA)

# Lưu từng chữ số đã tách
digit_path = os.path.join(debug_dir, f"{filename_prefix}_{i + 1}.jpg")
cv.imwrite(digit_path, resized_digit)

# Chuẩn hóa ảnh đầu vào
normalized_img = resized_digit.astype("float32") / 255.0
normalized_img = np.expand_dims(normalized_img, axis=-1)  # Thêm kênh màu
normalized_img = np.expand_dims(normalized_img, axis=0)   # Thêm batch size
```

Hình 12 – Chuẩn hóa ảnh đầu vào truóc khi đưa vào nhận dạng

Sau khi thực hiện các bước trên, ảnh số 3 đã được chuẩn hóa.

Hình 13 – Chữ số khi đã thực hiện việc chuẩn hóa

3.4.3. Nhận dạng chữ số

Mạng CNN (được huấn luyện bằng tập dữ liệu MNIST) được sử dụng để nhận dạng từng chữ số riêng lẻ.

Đầu vào: Một ảnh chữ số chuẩn hóa kích thước 28x28.

Xử lý: Mạng CNN xử lý ảnh đầu vào thông qua các lớp tích chập (convolutional layers), lớp gộp (pooling layers), và các lớp kết nối đầy đủ (fully connected layers).

Đầu ra: Giá trị được dự đoán của đầu vào.

Ví dụ: Một chữ số 3 khi đưa vào mạng CNN có thể trả về kết quả dự đoán sau.

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 38ms/step
3
```

Hình 14 – Giá trị dự đoán sau khi thực hiện quá trình nhận dạng chữ số

3.4.4. Ghép thành ngày, tháng, năm hoàn chỉnh

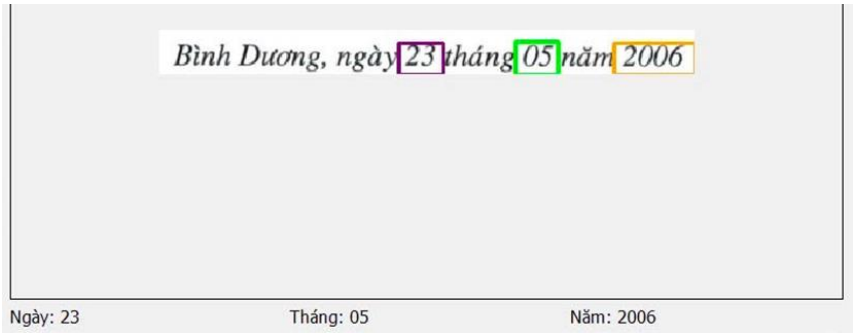
Từng chữ số được ghép lại để tạo thành kết quả hoàn chỉnh của ngày, tháng, năm.

Ví dụ:

Vùng "ngày" chứa các chữ số 2 và 3, hệ thống ghép thành ngày 23.

Vùng "tháng" chứa chữ số 0 và 5, hệ thống trả về tháng 05.

Vùng "năm" chứa các chữ số 2, 0, 0, 6, hệ thống trả về năm 2006.



Hình 15 – Kết quả hệ thống trả về sau quá trình nhận dạng

4. Thực nghiệm trên sản phẩm

4.1. Giới thiệu các chức năng

4.1.1. Nhận diện ngày, tháng, năm trên ảnh:

Mô hình YOLO được sử dụng để phát hiện các vùng chứa ngày, tháng, và năm. Những vùng này được biểu diễn dưới dạng bounding box được vẽ chính xác xung quanh các khu vực cần thiết.

Sau khi phát hiện bounding box, hệ thống cắt các vùng ảnh này để xử lý riêng lẻ, đảm bảo việc xử lý diễn ra chính xác mà không bị ảnh hưởng bởi các khu vực không liên quan.

4.1.2. Nhận diện chữ số từ ngày, tháng, năm:

Sử dụng OpenCV để chuyển đổi các vùng ảnh được cắt sang định dạng grayscale, giúp giảm nhiễu và tập trung vào các chi tiết cần nhận diện.

Sau đó, ảnh được nhị phân hóa (binarization) để làm nổi bật các chữ số, tiếp theo là đảo ngược màu (chữ đen trên nền trắng) nhằm phù hợp với đầu vào của mô hình MNIST.

Hệ thống áp dụng thuật toán tìm contour để tách riêng từng chữ số trong vùng ảnh. Những chữ số này được resize về kích thước chuẩn (28x28) trước khi đưa vào mô hình MNIST để nhận diện.

4.1.3. Dự đoán kết quả chính xác:

Mô hình MNIST CNN được sử dụng để nhận diện từng chữ số riêng lẻ. Mỗi chữ số sẽ được dự đoán và kết quả được tổng hợp lại thành ngày, tháng, và năm hoàn chỉnh.

Việc dự đoán được tối ưu để đảm bảo tính chính xác, ngay cả với dữ liệu viết tay có độ phức tạp khác nhau.

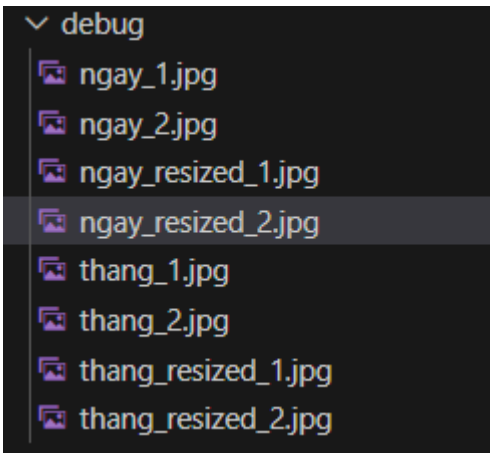
4.1.4. Hiển thị giao diện đơn giản:

Giao diện người dùng được xây dựng bằng PyQt5, cung cấp các nút bấm trực quan như "Chọn Ảnh" để tải ảnh vào hệ thống và hiển thị kết quả nhận diện trực tiếp trên giao diện.

Khung hiển thị ảnh và các trường hiển thị ngày, tháng, năm giúp người dùng dễ dàng theo dõi và kiểm tra kết quả nhận diện.

4.1.5. Lưu các bước debug:

Hệ thống lưu lại tất cả các bước xử lý vào thư mục debug/ để tiện cho việc kiểm tra. Các tệp debug bao gồm ảnh đã nhị phân hóa, ảnh đã đảo màu, ảnh đã tách riêng từng chữ số và ảnh được resize về kích thước chuẩn.

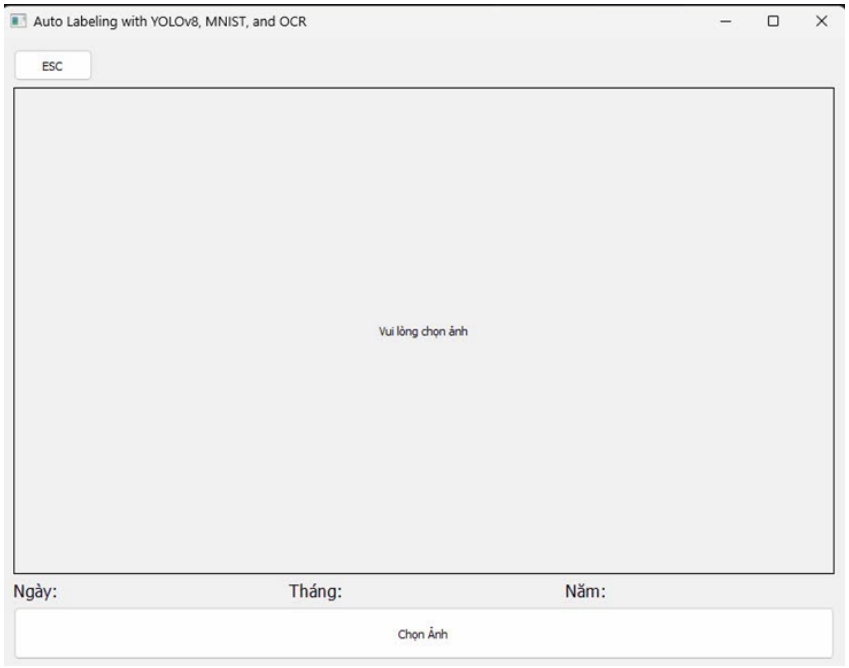


Hình 16 – Thư mục debug dùng để lưu ảnh trong quá trình nhận dạng

4.2. Kết quả chạy thực nghiệm

Khi chạy chương trình, người dùng sẽ thấy giao diện bao gồm:

- Một khung hiển thị ảnh ở giữa để xem kết quả nhận diện.
- Các trường ngày, tháng, năm nằm bên dưới để điền kết quả nhận diện tự động.
- Một nút bấm "Chọn Ảnh" ở dưới cùng để tải lên ảnh cần nhận dạng, đồng thời hỗ trợ phím ESC để thoát ứng dụng.



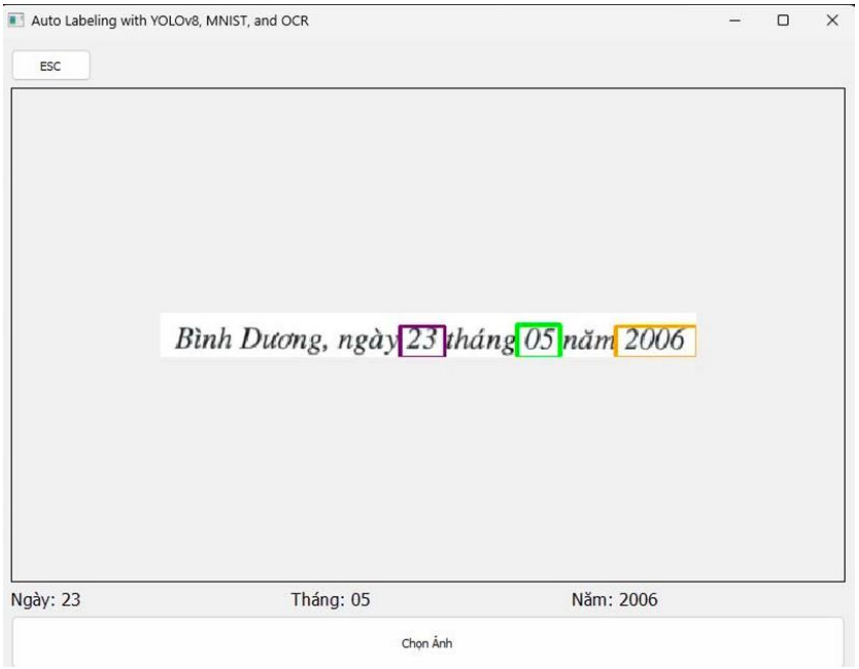
Hình 17 – Giao diện chính của ứng dụng

Quy trình thực nghiệm:

Người dùng nhấn vào nút "Chọn Ảnh" và tải lên một ảnh viết tay chứa ngày, tháng, năm cần nhận diện.

Hệ thống tự động thực hiện các bước như phát hiện vùng chứa ngày, tháng, năm; cắt và xử lý từng khu vực riêng biệt; nhận diện chữ số thông qua mô hình MNIST.

Kết quả được hiển thị trực tiếp trên giao diện người dùng dưới dạng các trường ngày, tháng, năm. Đồng thời, ảnh đã xử lý được hiển thị trong khung ảnh ở giao diện.



Hình 18 – Kết quả sau khi chạy chương trình

Khi đưa vào ảnh chứa ngày viết tay “23/05/2006”, hệ thống sẽ hiển thị kết quả nhận diện chính xác ngày tháng năm lên giao diện và lưu lại các hình ảnh trong quá trình nhận dạng chữ số vào thư mục debug.

5. Tổng kết

5.1. Thành tựu

Hệ thống nhận diện ngày, tháng, năm từ ảnh đã đạt được những thành tựu đáng kể, minh chứng cho sự kết hợp hiệu quả giữa các công nghệ xử lý ảnh và học sâu. Hệ thống sử dụng YOLO để phát hiện chính xác các vùng chứa ngày, tháng, năm trong ảnh, đảm bảo việc xác định vị trí đối tượng nhanh chóng và chính xác. Đồng thời, tích hợp Tesseract OCR hỗ trợ nhận diện văn bản, mở rộng khả năng xử lý dữ liệu phức tạp, và mô hình MNIST CNN giúp nhận diện chính xác các chữ số từ vùng trích xuất. Với các bước tiền xử lý như lọc nhiễu, nhị phân hóa và chuẩn hóa kích thước ảnh, hệ thống hoạt động hiệu quả trong các điều kiện đầu vào khác nhau, đạt độ chính xác cao trong việc nhận diện. Hơn nữa, hệ thống còn được thiết kế linh hoạt để xử lý đa dạng dữ liệu ảnh, dễ dàng mở rộng và tích hợp vào các bài toán thực tế như trích xuất dữ liệu tự động, quản lý tài liệu, hay số hóa thông tin. Đây là một minh chứng rõ ràng về việc ứng dụng thành công các kỹ thuật tiên tiến như phát hiện đối tượng, xử lý ảnh và học sâu, không chỉ giải quyết tốt bài toán hiện tại mà còn mở ra nhiều tiềm năng ứng dụng trong tương lai.

Thông qua việc thiết kế và triển khai hệ thống, các kỹ thuật như xử lý ảnh, phát hiện đối tượng, và nhận diện chữ số đã được ứng dụng một cách tối ưu, tạo nên một công cụ mạnh mẽ trong việc quản lý và số hóa dữ liệu. Hệ thống cũng cho thấy tính khả dụng cao khi có thể được mở rộng để xử lý các loại dữ liệu khác hoặc tích hợp vào các ứng dụng thực tế khác nhau.

5.2. Cải tiến và mở rộng trong tương lai

5.2.1. Tăng độ chính xác của nhận diện:

Huấn luyện lại mô hình YOLO với nhiều dữ liệu hơn, bao gồm các trường hợp thực tế đa dạng.

Tối ưu hóa mô hình MNIST bằng cách sử dụng các kỹ thuật xử lý trước ảnh như cân bằng sáng hoặc giảm nhiễu nhằm nâng cao độ chính xác trong điều kiện ảnh bị mờ hoặc ánh sáng kém.

5.2.2. Xử lý các trường hợp ngoại lệ:

Tích hợp cơ chế thông báo lỗi khi không nhận diện được hoặc khi ảnh đầu vào không hợp lệ.

5.2.3. Mở rộng nhận diện ký tự:

Phát triển khả năng nhận diện các ký tự chữ cái và ký tự đặc biệt (như dấu chấm, gạch chéo) sẽ giúp hệ thống có khả năng xử lý nhiều loại dữ liệu khác nhau, tăng tính ứng dụng.

Sử dụng các mô hình tiên tiến hơn như CRNN (Convolutional Recurrent Neural Network) hoặc các mô hình Transformer-based để cải thiện khả năng nhận diện văn bản.

5.2.4. Cải thiện giao diện người dùng:

Tích hợp thêm tính năng kéo-thả ảnh trực tiếp vào giao diện, giúp người dùng dễ dàng thao tác mà không cần sử dụng hộp thoại chọn file.

5.2.5. Tối ưu hóa hiệu năng:

Sử dụng GPU hoặc đa luồng để tăng tốc độ xử lý, đặc biệt là khi nhận diện ảnh có độ phân giải cao. Điều này đảm bảo hệ thống hoạt động nhanh chóng và hiệu quả trong môi trường thời gian thực.

Áp dụng các kỹ thuật tối ưu hóa ảnh đầu vào như giảm kích thước ảnh hoặc sử dụng bộ lọc tăng cường để giảm thời gian xử lý mà không làm giảm chất lượng nhận diện.

5.2.6. Hỗ trợ đa ngôn ngữ:

Mở rộng hỗ trợ nhận diện văn bản cho nhiều ngôn ngữ khác nhau, đặc biệt là các ngôn ngữ có ký tự đặc biệt như tiếng Nhật, Trung Quốc hoặc tiếng Việt. Điều này làm tăng khả năng ứng dụng của hệ thống trên phạm vi toàn cầu.

5.2.7. Xuất và lưu kết quả:

Thêm tính năng xuất kết quả nhận diện dưới dạng file CSV hoặc PDF, giúp người dùng lưu trữ và chia sẻ thông tin dễ dàng hơn.

Cung cấp báo cáo chi tiết sau mỗi lần xử lý, bao gồm các thông tin về ngày, tháng, năm cùng các bước xử lý mà hệ thống đã thực hiện, tạo sự minh bạch và đáng tin cậy.

5.2.8. Tích hợp ứng dụng thực tế:

Mở rộng ứng dụng trong các lĩnh vực như quản lý hóa đơn, tài liệu chứng từ hoặc nhận diện văn bản từ tài liệu giấy, giúp giảm bớt công sức nhập liệu thủ công và tăng hiệu quả làm việc.

Phát triển thêm API để tích hợp hệ thống vào các ứng dụng lớn hơn hoặc dịch vụ web, tạo điều kiện cho các doanh nghiệp và tổ chức sử dụng trong quy trình số hóa của mình.

5.3. Kết luận

Báo cáo này đã cung cấp chi tiết về cách thức hoạt động, cấu trúc hệ thống, và các ý tưởng cải tiến có thể thực hiện trong tương lai để nâng cao độ chính xác và hiệu quả của hệ thống. Hy vọng rằng kết quả nghiên cứu và phát triển này sẽ đóng góp vào việc phát triển các giải pháp tự động hóa thông minh, đồng thời trở thành một nguồn tham khảo hữu ích cho các dự án tương tự.

Nếu có bất kỳ thắc mắc hoặc đóng góp nào, rất mong nhận được sự phản hồi để hệ thống có thể tiếp tục được hoàn thiện và phát triển.

TÀI LIỆU THAM KHẢO

1. OpenCV. "Image Thresholding" [Online]. Available:https://docs.opencv.org/4.x/d7/d4d/tutorial_py_thresholding.html. [Accessed: 19-Dec-2024].
2. Ultralytics. "YOLOv8 Models Documentation" [Online]. Available:https://docs.ultralytics.com/vi/models/yolov8/. [Accessed: 19-Dec-2024].
3. Keras. "Model Training APIs - Predict Method" [Online]. Available:https://keras.io/api/models/model_training_apis/#predict-method. [Accessed: 19-Dec-2024].