TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN - ĐHQG-HCM

KHOA CÔNG NGHỆ THÔNG TIN

REPORT

Course: Introduction to Machine Learning

Giảng viên hướng dẫn

Bùi Tiến Lên

Võ Nhật Tân

Ngô Đức Bảo

Mục lục

1 Lời cảm ơn

Nhóm chúng em xin gửi lời chân thành cảm ơn nhất đến các thầy đã hướng dẫn chúng em trong quá trình thực hiện đồ án lần này.

Chúng em trân trọng sự hướng dẫn tận tình của thầy Bùi Tiến Lên trong cả một quá trình vừa qua.

Chúng em cám ơn sự hỗ trợ và tiếp sức thầm lặng từ thầy Ngô Đức Bảo và thầy Võ Nhật Tân trong quá trình thực hiện.

Nhờ những kiến thức quý báu thầy Bùi Tiến Lên, thầy Võ Nhật Tân đã truyền đạt trên lớp và thông qua các bài tập về nhà được thầy hướng dẫn rất chi tiết đã giúp chúng em có một nền tảng vững chắc thực hiện đồ án lần này.

Mặc dù quá trình thực hiện là rất gian nan và nhiều chông gai nhưng nhóm đã vượt qua và đi đến được kết quả cuối cùng. Tất cả cũng một phần không nhỏ nhờ vào sự hỗ trợ của các thầy.

2 Thông tin thành viên

STT	Họ và tên	Mã số sinh viên	Ghi chú
1	Phan Lê Đức Anh	22127020	
2	Lý Liên Hoa	22127117	
3	Trần Thị Thiên Kim	22127225	
4	Nguyễn Thị Ngọc Trang	22127421	Trưởng nhóm
5	Nguyễn Gia Phúc	22127482	

Bảng 1: Bảng danh sách các thành viên

3 Tổng quan

3.1 Giới thiệu

Trong bối cảnh trí tuệ nhân tạo (AI) ngày càng phát triển mạnh mẽ, nhiều bài toán phức tạp đã và đang được giải quyết một cách hiệu quả và tối ưu. Tuy nhiên, vẫn còn một bộ phận lớn người dùng gặp khó khăn trong việc truyền đạt suy nghĩ, ý tưởng của mình thông qua ngôn ngữ viết. Xuất phát từ thực tế này, nhóm chúng em đề xuất phát triển nền tảng trang web với hai chức năng chính:

3.1.1 Viết lại đoạn văn theo phong cách yêu cầu của người dùng

Chức năng này được triển khai dựa trên mô hình ngôn ngữ Gemini – một mô hình LLM tiên tiến hỗ trợ đầu vào đa phương tiện, có khả năng hiểu ngữ cảnh sâu sắc từ hình ảnh và văn bản, đồng thời tạo ra nội dung ngôn ngữ tự nhiên, mạch lạc và giàu thông tin. Người dùng có thể nhập đoạn văn bản cần viết lại, chọn phong cách mong muốn (hài hước, học thuật, xúc tích, v.v.) và nhận lại một phiên bản tối ưu theo mục tiêu đã chọn.

3.1.2 Dịch thuật giữa tiếng Việt hiện đại và chữ Hán-Nôm

Nhằm phục vụ những người dùng có hứng thú hoặc nhu cầu nghiên cứu Hán-Nôm, hệ thống hỗ trợ dịch hai chiều giữa quốc ngữ và văn tự Hán-Nôm. Bên cạnh hình thức nhập văn bản thông thường, người dùng cũng có thể tải lên hình ảnh chứa văn bản Hán-Nôm để xử lý. Chức năng này ứng dụng mô hình YOLO11 – một kiến trúc nhận dạng đối tượng (object detection) nổi bật với độ chính xác cao và tốc độ xử lý nhanh, đặc biệt phù hợp cho các ứng dụng thời gian thực trên nền tảng web.

3.2 Công nghệ thực hiện

Ngôn ngữ sử dụng: Python
Front-end: React, TailwindCSS, TypeScript

- **React:**Với thư viện JavaScript phổ biến cho việc xây dựng giao diện người dùng theo mô hình component-based đã cho phép tái sử dụng và tách biệt rõ ràng từng phần của giao diện giúp cho việc thiết kế UI dễ dàng hơn. Ngoài ra, còn hỗ trợ xây dựng UI động, quản lý state và điều hướng mượt mà.

- **TailwindCSS:**Cung cấp sẵn nhiều class CSS tiện ích để nhanh chóng tạo layout và style, giúp phát triển giao diện nhanh chóng, nhất quán và dễ bảo trì.
- **TypeScript:**Giúp phát hiện lỗi sớm ngay khi biên dịch, từ đó giảm thiểu bug runtime và nâng cao độ tin cậy của ứng dụng.

Back-end: Django, PostgreSQL

- **Django:** Được viết bằng ngôn ngữ Python và là một trong nhiều khung web Python. Django quản lý mã cho hệ thống yêu cầu và phản hồi bằng cách sử dụng kiến trúc Mô hình-Khung nhìn-Mẫu (MVT). Nó đóng vai trò là giao diện giữa cơ sở dữ liệu và mã máy chủ. Ngoài ra, khung nhìn của Django sử dụng các mô hình để xử lý yêu cầu.
- **PostgreSQL:** PostgreSQL là hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở, tuân thủ chuẩn SQL và hỗ trợ ACID. Nó cung cấp các tính năng mạnh như khóa ngoại, trigger, stored procedures và mở rộng qua extensions. Hỗ trợ lưu trữ dữ liệu dạng JSON/JSONB giúp kết hợp giữa quan hệ và NoSQL. Phù hợp với cả ứng dụng nhỏ lẫn hệ thống doanh nghiệp có yêu cầu cao về độ tin cậy và hiệu năng..

4 Mô hình Gemini

4.1 Multimodel & Long-Context

Gemini được thiết kế theo kiến trúc multimodal native, cho phép xử lý nhiều loại dữ liệu bao gồm văn bản, hình ảnh, audio và video mà không cần chuyển đổi trung gian Google AI for Developers [1]. Mô hình được huấn luyện từ đầu trên các modalities khác nhau và tinh chỉnh với dữ liệu đa phương thức bổ sung để nâng cao hiệu quả, vượt trội hơn so với các mô hình trước đây. Điều này giúp Gemini vượt trội so với các mô hình đa phương thức khác, đạt hiệu suất hàng đầu trên 30/32 tiêu chuẩn học thuật, bao gồm MMLU (90%) và MMMU (59.4%).

Kiến trúc này dựa trên Transformer và Mixture-of-Experts (MoE), chia mạng nơ-ron thành các "chuyên gia" nhỏ hơn, chỉ kích hoạt các chuyên gia phù hợp với đầu vào, từ đó tăng hiệu quả tính toán, rất hữu ích cho việc tóm tắt hoặc phân tích nội dung dài (Gemini 2.0 Technical Report).

Bên cạnh đó, mô hình hỗ trợ long-context lên đến một triệu tokens và mở rộng thử nghiệm hơn 2 triệu tokens trong nghiên cứu trong khi đó các mô hình như GPT giới hạn tầm ở 128k tokens (với GPT-4-1106-preview). Khả năng này rất hữu ích cho trợ lý chat khi phục vụ các tác vụ như tóm tắt hoặc phân tích tài liệu dài (PDF up to 1000 pages). [2]

4.2 Reasoning & Chain-of-Thought

Các biến thể cao cấp (Ultra, Pro) của Gemini tích hợp cơ chế **chain-of-thought prompting (CoT)**, cho phép mô hình thực thi các bước reasoning nội bộ trước khi xuất kết quả, nâng cao độ chính xác cho các bài toán logic, lập luận và lập kế hoạch. Theo Prompt Engineering Guide, CoT chia nhỏ các bài toán phức tạp thành các bước logic, cải thiện độ chính xác trong các tác vụ như giải toán, lập luận, và lập kế hoạch.

4.3 Nền tảng triển khai

Gemini API hoạt động trên hai nền tảng chính:

- **Google AI Studio:** Nền tảng trực quan cho phép các nhà phát triển thử nghiệm và prototyping nhanh với các mô hình. Với giao diện thân thiện, người dùng có thể gửi các yêu cầu đa phương thức (văn bản, hình ảnh, video,...) và nhận phản hồi ngay lập tức. Nền tảng này

hỗ trợ các ứng dụng như Tldraw và Toonsutra, tận dụng khả năng dịch đa ngôn ngữ và xử lý ngữ cảnh lớn.

- **Vertex AI:** Nền tảng cấp sản xuất của Google Cloud, được thiết kế để triển khai sản phẩm thực tế với hiệu suất cao và khả năng mở rộng. Các tính năng chính bao gồm:
  - Tự động mở rộng (Auto-scaling): Điều chỉnh tài nguyên dựa trên lưu lượng truy cập, đảm bảo trợ lý chat hoạt động mượt mà ngay cả khi có nhiều người dùng.
  - A/B Testing: so sánh hiệu suất của các phiên bản mô hình để chọn phiên bản tốt nhất
  - Fine-tuning: tùy chỉnh mô hình với dữ liệu cụ thể để cải thiện phản hồi cho các trường hợp sử dụng đặc thù.
  - Observability: giám sát hiệu suất và lỗi thông qua Cloud Monitoring.

4.4 Cách gọi

Hệ thống viết lại nội dung sử dụng API Gemini để tạo nội dung viết lại (rewritten content) từ văn bản gốc và các hướng dẫn bổ sung từ người dùng. API này được tích hợp vào backend Django thông qua lớp ‘GeminiClient‘.

4.4.1 Quy trình hoạt động

Quá trình xử lý viết lại nội dung bao gồm các bước sau:

- Người dùng gửi nội dung gốc và (tùy chọn) hướng dẫn viết lại qua frontend.
- Backend nhận yêu cầu thông qua API ‘/rewrite/‘.
- Nội dung được chuyển vào lớp ‘GeminiClient‘, nơi tạo (prompt) phù hợp.
- Prompt này được gửi đến API Gemini thông qua phương thức HTTP POST.
- Kết quả phản hồi từ API là nội dung viết lại, được lưu vào cơ sở dữ liệu cùng với thông tin người dùng.
- Nội dung viết lại được trả về frontend để hiển thị.

4.4.2 Giao tiếp với API Gemini

Lớp ‘GeminiClient‘ thực hiện việc gọi API bằng cách xây dựng một prompt từ nội dung và hướng dẫn, sau đó gửi dưới dạng JSON đến endpoint của Gemini. API phản hồi lại một danh sách các gợi ý (candidates), trong đó lớp này trích xuất phần nội dung viết lại.

Ví dụ: Người dùng có thể gửi một câu như:

"The weather is nice today."

Với hướng dẫn:

"Make it sound more poetic."

Sau khi gọi API, hệ thống có thể trả về nội dung đã được viết lại theo hướng dẫn, ví dụ như:

"The sun smiles gently upon the earth, painting the day with warmth and grace."

4.5 Kết quả

Link mô tả kết quả của chức năng "Viết lại nội dung" : https://docs.google.com/spreadsheets/d/1ShF8MwxtdHaiQXI8qLHUQO94e7rnRUxpQEfrwNW6l18/edit?usp=sharing

4.6 Đánh giá

Tiêu chí	**Đánh giá chi tiết**
Độ chính xác nội dung	Hầu hết các đoạn văn viết lại đều giữ được ý nghĩa gốc. Đây là một điểm mạnh. (Tỷ lệ giữ nguyên ý $> 80\%$)
Khả năng đáp ứng yêu cầu	Khi yêu cầu viết dễ thương, học thuật, truyền cảm hứng,... mô hình biến đổi ngôn ngữ tương đối tốt. Tuy nhiên, một số phong cách khó (ví dụ: báo chí giật tít) đôi khi chưa đạt độ "sắc nét" như kỳ vọng.
Độ tự nhiên của ngôn ngữ	Ngôn ngữ viết lại tự nhiên, giàu cảm xúc, đặc biệt ở yêu cầu như "truyền cảm hứng".
Chính tả - Ngữ pháp	Các đoạn sinh ra rất ít lỗi chính tả/ngữ pháp — cho thấy Gemini API xử lý tốt ngôn ngữ tiếng Việt.
Mạch lạc và logic	Hầu hết đoạn văn đều giữ được mạch ý, sắp xếp câu từ hợp lý
Tính sáng tạo	Đặc biệt cao ở các yêu cầu "truyền cảm hứng", "nhân hóa AI"
Nhất quán phong cách	Một số yêu cầu cần giữ phong cách xuyên suốt (ví dụ: "AI kể chuyện") thì mô hình làm tương đối ổn, dù đôi khi còn hơi chung chung.

Bảng 2: Bảng đánh giá mô hình Gemini

5 Mô hình yolo11

5.1 Tổng quan và thu thập dữ liệu:

- OCR (hay Optical Character Recognition) là công nghệ chuyển đổi hình ảnh văn bản thành văn bản số hóa, và công nghệ OCR được ứng dụng rất nhiều trong đời sống của chúng ta từ những tiện ích như google dịch hay sao chép văn bản nhanh chóng.

- Nhận thấy được tính ứng dụng cao của công nghệ OCR để giúp con người chúng ta tìm hiểu, học tập được sâu sắc và nhanh chóng hơn về văn hóa, lịch sử của những thế hệ cha ông, tổ tiên đi trước; người Việt Nam chúng ta đã ứng dụng và tìm hiểu công nghệ này nhanh chóng. Từ đó, việc tìm hiểu các công cụ có sẵn giúp hiện thực hóa việc OCR chữ Nôm là một việc thiết yếu. Đề tài OCR chữ Nôm với các mô hình hay công cụ khác nhau là một để tài thú vị để chúng ta tìm hiểu kỹ hơn về các công cụ tiềm năng có thể sử dụng để giúp quá trình tìm hiểu về văn hóa, văn học chữ Nôm ngày xưa của chúng ta được thuận lợi hơn, giúp số hóa các tài liệu cổ, tạo điều kiện cho việc nghiên cứu ngôn ngữ học và văn học.

- Vậy, tại sao chúng ta lại chọn YOLO?
  - Tốc độ xử lý của mô hình nhanh.
  - Hiệu quả cao để nhận diện từng chữ một thay vì cả một câu như các mô hình để nhận diện chữ Nôm khác.

- Mục tiêu:
  - Giúp xây dựng hệ thống nhận diện chữ Nôm.
  - Các dữ liệu còn có thể được dùng để huấn luyện các mô hình khác.

- Bộ dữ liệu được sử dụng để train model:
  - Lục Vân Tiên
  - Truyện Kiều bản 1866
  - Truyện Kiều bản 1871
  - Truyện Kiều bản 1872

- Đại Việt Sử Ký Toàn Thư – Quyển Thủ
- Đại Việt Sử Ký Toàn Thư – Ngoại kỷ toàn thư
- Đại Việt Sử Ký Toàn Thư – Bản kỷ toàn thư
- Đại Việt Sử Ký Toàn Thư – Bản kỷ thực lục
- Đại Việt Sử Ký Toàn Thư – Bản kỷ tục biên

Từ các dữ liệu đã được cung cấp, tiến hành xử lý sơ bộ để tăng cường chất lượng hình ảnh như cân bằng sáng, tăng độ tương phản, giảm nhiễu và tiến hành sắp xếp các dữ liệu và gán nhãn dữ liệu lần lượt ở các mức độ câu và ký tự.

5.2 Công cụ hỗ trợ dán nhãn

- Roboflow: giúp trực quan hóa và hiển thị các nhãn dữ liệu được gán, phát hiện và kiểm tra sai sót trong nhãn dữ liệu.
- PPOCRLabel (custom model): OCR các ký tự.
- YOLOv5 (custom model): phát hiện ký tự, lấy bounding box của từng ký tự

5.3 Quy trình gán nhãn dữ liệu

- Đầu tiên, thực hiện trích xuất hình ảnh từ các file pdf, tách các trang chữ Nôm và chữ Quốc Ngữ thành các folder riêng.
- Từ hình ảnh trích xuất được, tiến hành phát hiện từng ký tự chữ Nôm bằng mô hình YOLOv5, thu được bounding box của từng ký tự. Dữ liệu được lưu dưới dạng file txt và có định dạng YOLOv5 PyTorch TXT (nhưng các thông số không được chuẩn hóa để giảm tính sai số khi chuyển đổi định dạng dữ liệu):

```
class_id x_center y_center width height
```

- Tạo project trên Roboflow, tiến hành upload data lên để kiểm tra khả năng phát hiện kí tự của mô hình YOLOv5. Ta nhận thấy với ngữ liệu hiện tại mô hình nhận diện chồng box, thiếu box ở khu vực tối và nhận diện cả những kí tự dư thừa, hoặc bị xoay và mất trang.

- Sau đó, chia ngữ liệu Nôm ra các folder khác nhau để xử lí theo tính chất của từng ảnh, thực hiện các kỹ thuật tiền xử lí ngữ liệu như: xoay trang theo đúng chiều, cắt ảnh để giảm thiểu độ nhiễu do các nội dung khác không liên quan, xử lý độ sáng và độ tương phản theo thông số phù hợp với ảnh để lấy được cả vùng tối và vùng sáng của ảnh, tăng giảm kích thước ảnh.

- Dùng mô hình YOLOv5 đã được train sẵn để lấy bounding box của từng kí tự.

- Tải lại bộ ngữ liệu lên Roboflow để kiểm tra các bounding box và nhận thấy đã xử lí được khá tốt vấn đề chồng box, hạn chế các kí tự không liên quan và nhận diện được tốt hơn các vùng dữ liệu.

- Với các bounding box tốt hơn thu được, tiến hành sắp xếp dựa trên tọa độ theo thứ tự từ trên xuống dưới, từ phải sang trái. Tuy nhiên, do ngữ liệu có số lượng các box nghiêng khá lớn nên chọn một thông số "tolerance" phù hợp với bộ ngữ liệu để lấy được nhiều box trong một câu, giúp tăng độ chính xác khi gióng hàng các chữ. Tiếp tục dùng Roboflow để kiểm tra ngữ liệu đến khi đạt yêu cầu tốt nhất có thể.

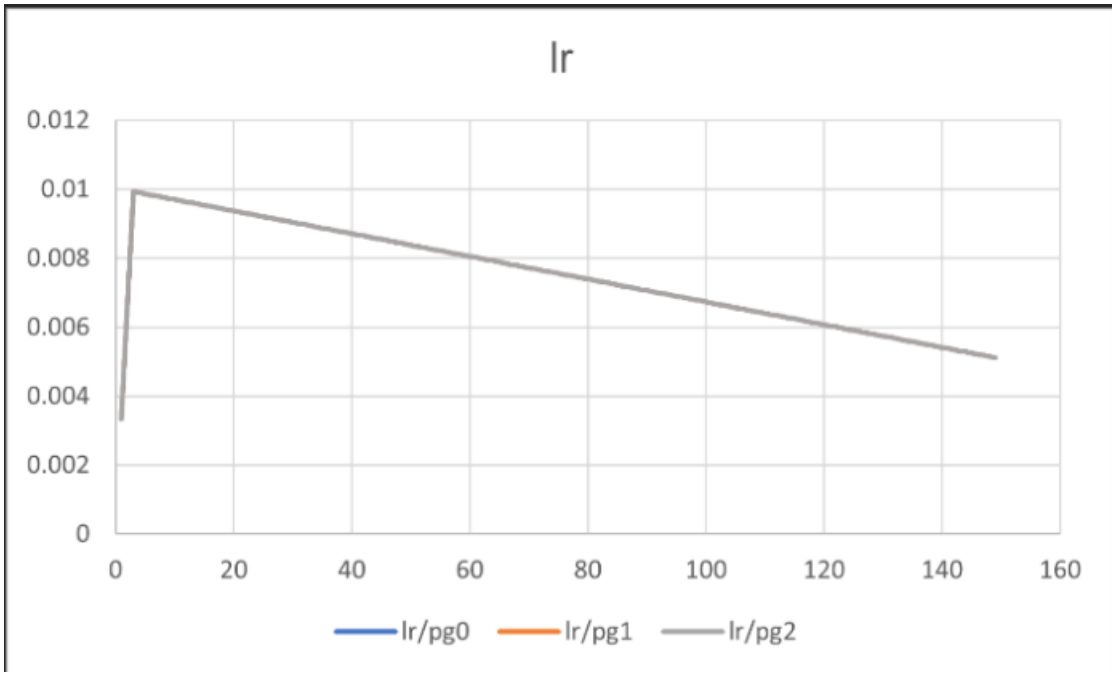
- Tiếp tục gán các class cho các box chữ theo thứ tự đi từ mức trang, mức câu rồi đến mức chữ như cách gióng hàng để tô màu chữ cho ngữ liệu giữa kì vì ngữ liệu giữa kì đã được gán nhãn khá chính xác (được tiền xử lí nhiễu hiệu quả nên chỉ trừ những trường hợp đặt biệt hoặc những trường hợp do công cụ OCR chữ Quốc ngữ không chính xác dẫn đến việc không khớp ở một số dòng). Cụ thể, bộ ngữ liệu giữa kì được xử lí gióng hàng như sau :

  - Gióng hàng mức trang: Chia các box trong một ảnh và đánh số trang Nôm ngược chiều với số trang Quốc ngữ. Đồng thời trong quá trình OCR chữ Quốc ngữ, việc cắt ảnh có để lại số trang (được đưa vào đánh dấu header cho nội dung thuộc một trang), ta gióng hàng với trang được đánh số tương ứng trong các trang chữ Nôm được OCR.
  - Gióng hàng mức dòng: Gióng dựa trên độ dài các box nếu dữ liệu bị mất box câu. Nếu dữ liệu không bị mất box câu thì có thể gióng trực tiếp để đảm bảo câu được khớp hơn.
  - Gióng hàng mức chữ: dùng thuật toán M.E.D Leveinshtein để gióng hàng chữ Nôm trong câu với chữ Quốc ngữ trong câu dựa vào hai từ điển

    `SinoNom_similar_Dic và QuocNgu_SinoNom_Dic`

- Sau khi gióng hàng chữ, ta dùng các box chữ đã được xếp theo câu gióng hàng theo thứ tự với cặp Quốc ngữ và Nôm.
- Cuối cùng ta được nhãn là chữ Quốc ngữ tương ứng với box chữ Nôm
- Tạo file yaml tổng hợp các class chữ Quốc ngữ rồi đánh số thứ tự để gán đúng class id cho dataset theo cấu trúc của YOLOv5 PyTorch TXT.
- Kiểm lại bộ ngữ liệu một lần nữa và có thể điều chỉnh trong Roboflow nếu cần thiết.

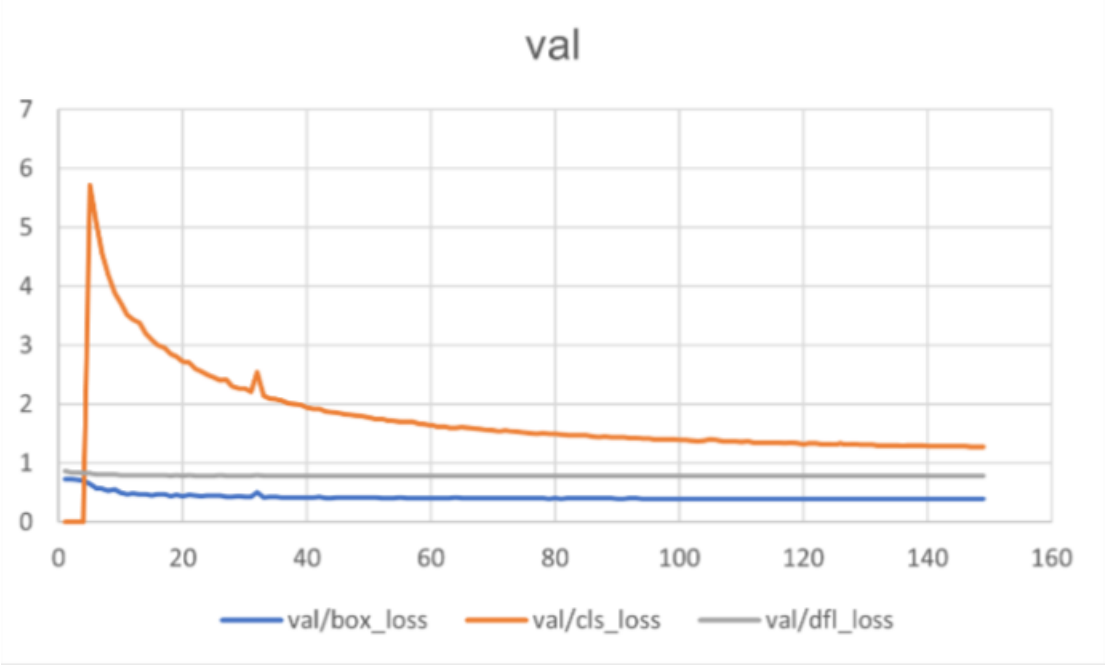
5.4 Train mô hình:



Learning rate(lr)

- Learning rate ban đầu cao (khoảng 0.01) giúp mô hình học nhanh trong giai đoạn đầu.
- Sau khoảng 20 epoch, learning rate giảm dần tuyến tính, xuống mức khoảng 0.004 ở epoch cuối. Đây là một chiến lược scheduler giảm tuyến tính (linear decay), phổ biến trong việc tối ưu hóa mô hình.
- Các nhóm tham số khác nhau (pg0, pg1, pg2) được áp dụng learning rate đồng bộ, cho thấy không có sự ưu tiên đặc biệt nào giữa các nhóm tham số.

Giai đoạn đầu, learning rate cao giúp mô hình nhanh chóng học được các thông số cơ bản.
Khi loss giảm dần, việc giảm learning rate giúp mô hình tránh overshooting (nhảy qua giá trị tối ưu) và tinh chỉnh các thông số tốt hơn.

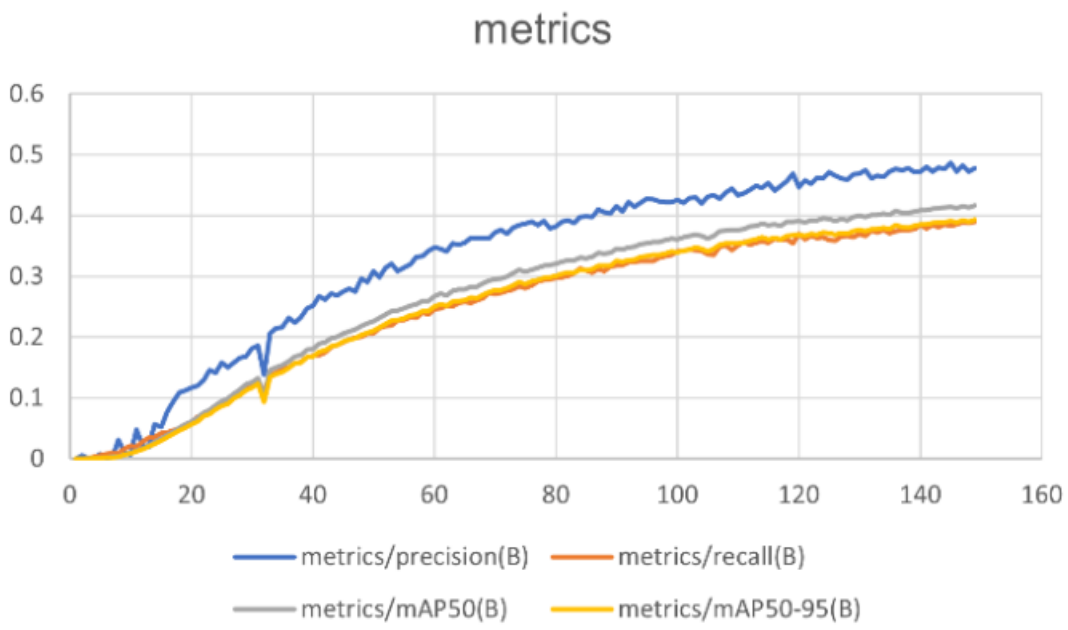


Loss

- Các đồ thị train/val loss (box_loss, cls_loss, dfl_loss) cho thấy quá trình hội tụ ổn định của mô hình.
- Loss (train/val): Loss bắt đầu ở mức khoảng 1.0, giảm nhanh chóng trong 20 epoch đầu, sau đó duy trì ở mức thấp dưới 0.5. Đây là loss dùng để tối ưu hóa vị trí và kích thước bounding box, phản ánh khả năng mô hình học cách định vị đối tượng tốt.
- Classification Loss (train/val): Ban đầu rất cao (khoảng 7.0), sau đó giảm đều và đạt mức 2.0 sau khoảng 80 epoch, ổn định dần về sau. Đây là chỉ số quan trọng vì nó đánh giá khả năng phân loại chính xác của mô hình. Việc loss giảm chứng tỏ mô hình dần hiểu được cách phân loại các đối tượng.
- DFL Loss (train/val): DFL Loss bắt đầu ở mức thấp (khoảng 0.5) và hầu như không thay đổi nhiều trong toàn bộ quá trình huấn luyện. Điều này cho thấy mô hình đã tối ưu tốt khía

cạnh này ngay từ đầu, nhấn mạnh hiệu quả của mô hình với loss này.

- Sự đồng nhất giữa train/val loss: Train và val loss gần như đồng nhất ở mọi giai đoạn, cho thấy mô hình không bị overfitting hay underfitting.



Metrics

- Precision: Precision tăng ổn định, đạt mức 0.55 ở cuối quá trình huấn luyện. Mô hình có khả năng dự đoán chính xác các bounding box thuộc về đối tượng thực tế, tức là mô hình ít đưa ra dự đoán sai (false positives thấp).

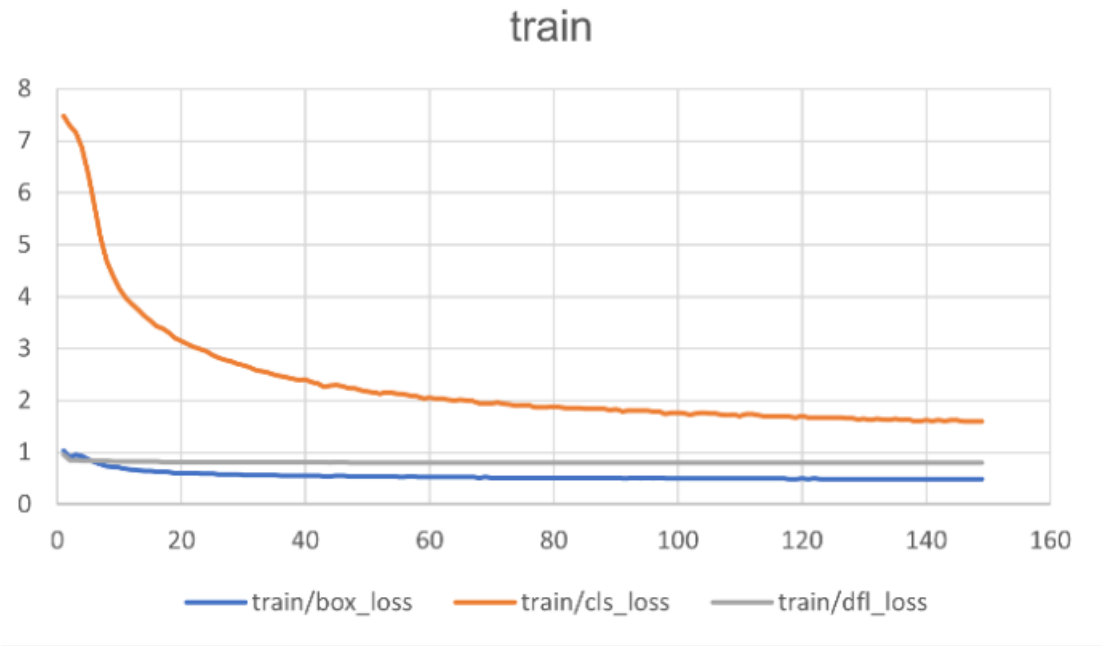
- Recall: Recall tăng dần, đạt mức 0.45 ở epoch cuối. Mô hình phát hiện được 45% số đối tượng thực tế có mặt trên ảnh. Tuy nhiên, Recall thấp hơn Precision, điều này nghĩa là mô hình bỏ sót một số đối tượng (false negatives).

- mAP50: mAP50 (Mean Average Precision tại IoU = 0.50) tăng đều, đạt 0.50 ở cuối. Mô hình dự đoán chính xác bounding boxes với mức độ chồng lấp (IoU) tối thiểu là 50%.

- mAP50-95: mAP50-95 tăng ổn định, đạt 0.40 ở epoch cuối. Đây là chỉ số khó đạt được hơn vì nó đòi hỏi mô hình phải đạt hiệu quả trên nhiều mức IoU khác nhau (từ 0.50 đến 0.95). Chỉ số này đạt 0.40 cho thấy mô hình có khả năng tổng quát hóa tốt.

Thời gian huấn luyện

- Thời gian huấn luyện trung bình cho mỗi epoch ổn định, chứng tỏ tài nguyên phần cứng và quá trình huấn luyện không gặp vấn đề về bottleneck.
- Biểu đồ thời gian phản ánh tính đồng nhất và ổn định của môi trường huấn luyện.



Tổng hợp đánh giá

- Ưu điểm
  - Hội tụ tốt: Loss giảm đều, ổn định và không có dấu hiệu overfitting.
  - Hiệu suất cao: Precision (0.55) và mAP50 (0.50) cao, phù hợp với nhiều ứng dụng thực tế.
  - Chiến lược learning rate hợp lý: Điều chỉnh giảm dần learning rate giúp mô hình hội tụ ổn định.
- Hạn chế
  - Recall thấp hơn Precision: Một số đối tượng bị bỏ sót, gây ảnh hưởng đến khả năng bao quát của mô hình.
  - Thời gian hội tụ lâu: Cần tối ưu thêm để rút ngắn thời gian huấn luyện.

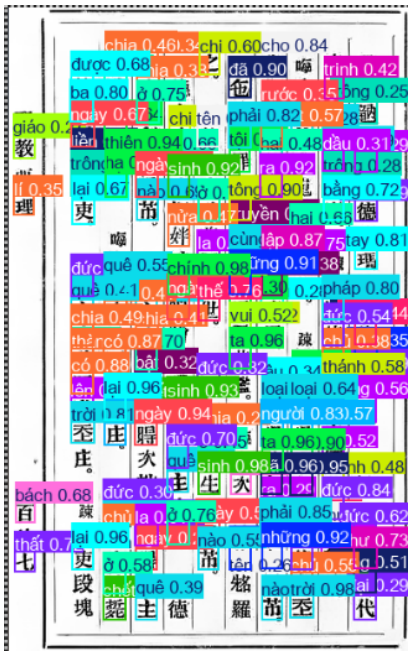
Cải thiện:

- Tăng recall
  - Sử dụng dữ liệu đa dạng hơn, tập trung vào các mẫu khó nhận diện (đối tượng nhỏ, bị che khuất).
  - Điều chỉnh trọng số của classification loss (cls_loss) để tăng ưu tiên phát hiện các đối tượng.
- Tối ưu thời gian huấn luyện
  - Áp dụng mixed-precision training để giảm thời gian tính toán.
  - Sử dụng distributed training nếu có nhiều GPU.
- Tăng dữ liệu: Bổ sung các tập dữ liệu từ nhiều nguồn khác nhau để cải thiện khả năng tổng quát hóa.
- Hyperparameter Tuning: Kiểm tra các giá trị khác nhau cho learning rate, batch size, và các tham số khác để tìm cấu hình tốt nhất.

5.5 Kết quả

Hình 1: Kết quả phát hiện được tổng 81/168 kí tự được gán chính xác cho hình trên.

Hình 2: Kết quả phát hiện được tổng 67/160 kí tự được gán chính xác cho hình trên.



Hình 3: Kết quả phát hiện được tổng 111/224 kí tự được gán chính xác cho hình trên.

5.6 Đánh giá

Dùng model huấn luyện để dự đoán kết quả trên một ảnh bất kỳ có cùng tính chất với các ảnh trong dataset.

Kết quả cho ra class và bounding box của các ký tự. Ta trích xuất các thông số ra để xử lý.

Tiến hành sắp xếp các bounding box dựa trên tọa độ theo thứ tự từ trên xuống dưới, từ phải sang trái.

Sau khi sắp xếp các bounding box, ta lấy ra class là các chữ Quốc ngữ dự đoán được để kiểm tra.

Tiến hành dùng thuật toán M.E.D Leveinshtein để gióng hàng các chữ Quốc ngữ dự đoán với các chữ Quốc ngữ trong văn bản gốc của ảnh đánh giá.

Đếm số cặp mà 2 chữ đều giống nhau và đếm tổng số chữ trong văn bản Quốc ngữ gốc, ta được tỉ lệ chính xác của kết quả dự đoán.

5.7 Khó khăn và giải pháp

Các khó khăn

- Hình ảnh nhiễu, mờ
- Ánh sáng không đều dẫn đến các khu vực tối màu không thể OCR hoặc OCR thiếu sót.

- Nhiều ký tự có cấu trúc tương đồng làm ảnh hưởng đến quá trình xác định.
- Ngữ liệu thô là các tài liệu chữ Nôm viết tay nên dẫn đến các trường hợp như sau:
  - Các ký tự Nôm bị viết dính lại với nhau làm cho model phát hiện chữ hoạt động với độ chính xác thấp.
  - Các ký tự Nôm viết tay xấu dẫn đến việc nhận dạng chữ với độ chính xác thấp.
  - Phông chữ khác nhau làm tăng độ khó của việc nhận diện
  - Chữ Quốc ngữ được trích xuất bằng công cụ OCR nên độ chính xác bị giảm đi đáng kể.
- Kỹ thuật gióng hàng dữ liệu còn chưa hiệu quả dẫn đến độ chính xác của bộ ngữ liệu không cao.
- Chữ Nôm là một ngôn ngữ cổ dẫn đến hạn chế về số lượng tài liệu cũng như các công cụ để giúp kiểm tra độ chính xác

Ta có thể thấy rằng độ chính xác của bộ ngữ liệu ở mỗi bước trong quy trình gán nhãn bị giảm đi đáng kể. Do đó, bộ ngữ liệu được gán nhãn cuối cùng có độ chính xác không cao và bị mất dữ liệu khá nhiều.
Cách xử lý:

- Áp dụng các kỹ thuật giúp tăng cường độ chính xác khi nhận diện như xoay, phóng to, cắt ảnh.
- Sử dụng các kỹ thuật xử lý ảnh như cân bằng sáng, giảm nhiễu và làm mịn hình ảnh.
- Chuẩn bị bộ ngữ liệu chữ Quốc ngữ với độ chính xác cao nhất có thể để bù cho độ chính xác thấp của bộ ngữ liệu chữ Nôm:
  - Thử các công cụ OCR hiệu quả hơn.
  - Sau khi OCR, bộ ngữ liệu được tinh chỉnh bằng công cụ sửa lỗi chính tả để cho ra bộ ngữ liệu đúng chính tả.
  - Kiểm tra thủ công bộ ngữ liệu.

Tài liệu

[1] Tổng quan về model Gemini

[2] The new way to cloud

[3] NomNaOCR

[4] Introduction to Levenshtein

[5] ultralytics