BÁO CÁO ĐÁNH GIÁ VÀ LỰA CHỌN CÔNG CỤ TRÍCH XUẤT NỘI DUNG PDF

1. Mục tiêu thí nghiệm

Mục tiêu của thí nghiệm này là đánh giá và so sánh các công cụ trích xuất nội dung từ tài liệu PDF nhằm lựa chọn công cụ phù hợp nhất cho bài toán nghiên cứu, phục vụ các bước tiền xử lý văn bản, làm sạch ngôn ngữ và phân đoạn ngữ nghĩa trong hệ thống xử lý ngôn ngữ tự nhiên.

Ba công cụ được xem xét bao gồm:

- **PyMuPDF**: công cụ parsing PDF truyền thống, hoạt động dựa trên cấu trúc nội tại của file
- **Marker**: công cụ chuyển đổi PDF sang Markdown dựa trên pipeline OCR và mô hình học sâu
- **Nougat**: mô hình OCR học sâu chuyên biệt cho tài liệu học thuật và bài báo khoa học

2. Thiết lập và phương pháp đánh giá

2.1. Môi trường thực nghiệm

Các thí nghiệm được triển khai trong môi trường **Google Colab**, nhằm đảm bảo khả năng tái lập và kiểm soát tài nguyên:

- CPU
- GPU được cấp phát động (Google Colab Free)
- Hệ điều hành Linux
- Python 3.12

Tài liệu đầu vào là **một PDF chuẩn gồm 3 trang**, có cấu trúc heading và bảng biểu, được tạo bằng thư viện reportlab, nhằm đảm bảo tuân thủ chuẩn PDF và tương thích với các thư viện parsing nghiêm ngặt như PDFium.

2.2. Chỉ số đánh giá

Các công cụ được đánh giá dựa trên các chỉ số định lượng và định tính sau:

- Số ký tự trích xuất (Character Count)
- Số từ (Word Count)
- Số dòng (Line Count)
- Số đoạn văn (Paragraph Count)
- Thời gian xử lý (Processing Time)

- Khả năng giữ cấu trúc nội dung (heading, bảng biểu)
- Định dạng đầu ra (Plain text / Markdown)

3. Phân tích đặc trưng từng công cụ

3.1. PyMuPDF

PyMuPDF là một công cụ trích xuất PDF truyền thống, hoạt động dựa trên việc phân tích cấu trúc file PDF thay vì sử dụng mô hình học sâu.

Ưu điểm:

- Tốc độ xử lý rất cao, nhanh nhất trong các công cụ được đánh giá
- Chạy hoàn toàn trên CPU, không yêu cầu GPU hay bộ nhớ lớn
- Hoạt động ổn định với cả các PDF không hoàn toàn tuân thủ chuẩn
- Phù hợp cho trích xuất văn bản thô với quy mô lớn

Nhược điểm:

- Không bảo toàn cấu trúc logic của tài liệu
- Không phân biệt rõ tiêu đề, đoạn văn hay bảng biểu
- Không sinh trực tiếp đầu ra Markdown có cấu trúc

Phạm vi phù hợp:
PyMuPDF phù hợp cho các pipeline cần tốc độ cao, trích xuất văn bản thô và làm baseline đối chiếu nội dung.

3.2. Nougat

Nougat là một mô hình OCR học sâu được thiết kế chuyên biệt cho các bài báo khoa học.

Ưu điểm:

- Chuyển đổi công thức toán học sang LaTeX hiệu quả
- Hiểu cấu trúc bài báo khoa học tốt

Nhược điểm:

- Yêu cầu phần cứng GPU mạnh và môi trường CUDA ổn định
- Khó tái lập kết quả trong môi trường chuẩn hóa
- Không phù hợp cho các tài liệu không thuần bài báo khoa học

Phạm vi phù hợp:
Nougat phù hợp cho các nghiên cứu chuyên sâu về tài liệu học thuật có nhiều công thức toán học, với hạ tầng phần cứng phù hợp.

3.3. Marker

Marker được phát triển nhằm cân bằng giữa chất lượng trích xuất và khả năng bảo toàn cấu trúc tài liệu.

Ưu điểm:

- Chuyển đổi PDF sang Markdown với chất lượng cao
- Giữ tốt cấu trúc nội dung như heading và phân đoạn đoạn văn
- Phù hợp cho các pipeline chuẩn bị dữ liệu đầu vào cho LLM và hệ thống RAG

Nhược điểm:

- Thời gian xử lý dài do phải tải và khởi tạo các mô hình học sâu
- Phụ thuộc vào việc tài liệu đầu vào phải tuân thủ chuẩn PDF
- Yêu cầu tài nguyên tính toán cao hơn PyMuPDF

Phạm vi phù hợp:
Marker phù hợp cho các bài toán cần bảo toàn cấu trúc nội dung và chuyển đổi sang Markdown phục vụ phân đoạn ngữ nghĩa.

4. Kết quả thực nghiệm định lượng

4.1. Kết quả trích xuất nội dung

Công cụ	Trang	Ký tự	Từ	Dòng	Đoạn	Định dạng	Thời gian (giây)
PyMuPDF	3	736	108	29	3	Plain Text	**0.078**
Marker	N/A	945	132	19	8	Markdown	**330.308**
Nougat	–	–	–	–	–	–	–

Nhận xét:

- PyMuPDF xử lý nhanh hơn Marker **hơn ba bậc độ lớn**.
- Marker trích xuất được nội dung dài hơn và có mức độ phân đoạn cao hơn (8 đoạn so với 3 đoạn).
- Marker sinh Markdown có cấu trúc, trong khi PyMuPDF chỉ sinh văn bản phẳng.

4.2. Phân tích chất lượng đầu ra

- Đầu ra của **PyMuPDF** phù hợp cho các tác vụ xử lý văn bản thô, nhưng cần thêm bước hậu xử lý để phục hồi cấu trúc.
- Đầu ra của **Marker** giữ được cấu trúc logic của tài liệu, thuận lợi cho các bước chunking theo ngữ nghĩa và xây dựng hệ thống RAG.

Sự khác biệt giữa hai công cụ chủ yếu nằm ở **cách biểu diễn và tổ chức nội dung**, không phải ở thông tin cốt lõi.

5. Phân tích khả năng triển khai của Nougat

Trong môi trường Google Colab, Nougat không thể được triển khai ổn định do:

- Phụ thuộc mạnh vào kiến trúc cũ và phiên bản thư viện học sâu
- Yêu cầu tài nguyên GPU lớn và ổn định
- Khó đảm bảo khả năng tái lập kết quả

Do đó, Nougat chỉ được phân tích ở mức tham chiếu kiến trúc và không được đưa vào so sánh định lượng.

6. So sánh tổng hợp

Tiêu chí	PyMuPDF	Marker	Nougat
Chạy ổn định	Rất tốt	Tốt	Không ổn định
Tốc độ xử lý	**Rất nhanh**	Chậm	Rất chậm
Giữ cấu trúc nội dung	Thấp	**Cao**	Cao (lý thuyết)
Định dạng đầu ra	Text	**Markdown**	Markdown
Phụ thuộc môi trường	Thấp	Trung bình	Rất cao
Phù hợp triển khai	Cao	Cao	Thấp

7. Kết luận và lựa chọn công cụ

Kết quả thực nghiệm cho thấy **không tồn tại công cụ trích xuất PDF tối ưu cho mọi kịch bản**. Mỗi công cụ phù hợp với những mục tiêu sử dụng khác nhau.

Trong phạm vi đề tài, với mục tiêu xây dựng pipeline tiền xử lý văn bản phục vụ phân đoạn ngữ nghĩa và hệ thống dựa trên mô hình ngôn ngữ lớn, **Marker được lựa chọn là công cụ chính** nhờ khả năng bảo toàn cấu trúc nội dung và sinh đầu ra Markdown chất lượng cao.

PyMuPDF đóng vai trò baseline hiệu quả, đảm nhiệm trích xuất nhanh và ổn định văn bản thô. **Nougat** được ghi nhận là công cụ tiềm năng nhưng chưa khả thi trong điều kiện triển khai hiện tại.

8. Kết luận ngắn gọn (dùng cho bảng tổng kết)

Marker là công cụ ***phù hợp nhất*** để bảo toàn cấu trúc nội dung khi trích xuất PDF, trong khi **PyMuPDF** là lựa chọn tối ưu về hiệu năng và độ ổn định. Nougat không được đưa vào so sánh định lượng do hạn chế về môi trường triển khai.