# Dự án: Công cụ Chuyển đổi PDF sang DOCX “Hàng Loạt” (Có Hỗ trợ OCR)

## Tổng quan dự án

Chuyển đổi file PDF sang định dạng Word (DOCX) thường tốn thời gian nếu thực hiện thủ công từng file, đặc biệt với số lượng lớn. Dự án này phát triển một công cụ tự động để **chuyển đổi hàng loạt PDF trong Google Drive thành DOCX** một cách tiện lợi. Công cụ chạy trên nền tảng Google Colab, tích hợp với Google Drive để lấy các file PDF nguồn và lưu kết quả DOCX trực tiếp trở lại thư mục Drive của người dùng.

**Điểm nổi bật:** - **Hỗ trợ OCR (Nhận dạng ký tự):** Công cụ có thể xử lý cả PDF dạng ảnh (scan) bằng cách trích xuất văn bản. Người dùng có thể chọn giữa hai phương pháp OCR: - *Tesseract OCR:* Sử dụng công cụ OCR mã nguồn mở Tesseract (qua thư viện OCRmyPDF) để nhận dạng và giữ bố cục văn bản gốc tốt hơn. - *Google Gemini AI OCR:* Sử dụng mô hình ngôn ngữ **Gemini** của Google (một AI đa phương thức) để đọc văn bản trong ảnh một cách thông minh. - **Chế độ không OCR:** Đối với PDF gốc đã có lớp văn bản (không phải scan), công cụ hỗ trợ chuyển đổi trực tiếp sang DOCX **không dùng OCR** nhằm giữ nguyên định dạng tốt nhất. - **Tự động hoàn toàn:** Chỉ cần cung cấp ID thư mục Google Drive chứa các file PDF và chọn chế độ, công cụ sẽ tự động tải về từng PDF, xử lý chuyển đổi, rồi tải lên kết quả DOCX. Các file đã có DOCX tương ứng sẽ được bỏ qua để tránh trùng lặp.

## Công nghệ và thư viện sử dụng

Dự án sử dụng kết hợp nhiều thư viện Python và dịch vụ để đảm bảo hiệu quả chuyển đổi và OCR:

* **PyDrive2:** Thư viện Python kết nối với Google Drive API để liệt kê file, tải xuống PDF và tải lên DOCX. Nhờ đó, quá trình lấy dữ liệu và lưu kết quả được tự động hóa trên Google Drive.
* **pdf2docx:** Thư viện chuyển đổi PDF sang Word. Thư viện này hoạt động tốt với các PDF có văn bản số (text-based PDF) và có thể giữ khá tốt cấu trúc tài liệu gốc[[1]](https://www.reddit.com/r/MicrosoftWord/comments/1iuv2te/convert_pdf_into_word_word/#:~:text=,cv.close). Đây là công cụ chính cho việc chuyển đổi định dạng tài liệu.
* **OCRmyPDF (Tesseract OCR):** OCRmyPDF là công cụ mã nguồn mở tích hợp Tesseract để thêm lớp văn bản ẩn vào PDF scan. Nó cho phép **giữ nguyên bố cục, định dạng** và cấu trúc của trang PDF gốc trong khi thêm lớp văn bản có thể tìm kiếm được[[2]](https://ploomber.io/blog/pdf-ocr/#:~:text=,OCR%20multiple%20PDF%20files%20simultaneously). Dự án sử dụng OCRmyPDF để xử lý các PDF scan, sau đó mới chuyển sang DOCX, nhờ đó **bố cục trong file DOCX gần giống bản PDF gốc**.
* **Google Generative AI (Gemini API):** Gemini là thế hệ mô hình ngôn ngữ lớn (LLM) đa phương thức mới của Google, có khả năng xử lý cả văn bản và hình ảnh. Nhờ **multimodality**, Gemini có thể nhận trực tiếp đầu vào là hình ảnh và “đọc” được nội dung trong đó[[3]](https://medium.com/google-cloud/text-extraction-from-image-using-google-gemini-6b9d9989fa84#:~:text=OCR%20with%20Google%20Gemini). Dự án sử dụng API này để OCR bằng cách gửi ảnh trang PDF lên mô hình và nhận về văn bản. Cách tiếp cận này tận dụng khả năng của LLM trong việc hiểu ngữ cảnh để trích xuất chữ chính xác, **bỏ qua quy trình OCR truyền thống nhiều bước**[[4]](https://medium.com/google-cloud/text-extraction-from-image-using-google-gemini-6b9d9989fa84#:~:text=,and%20removes%20potential%20error%20points).
* **Các thư viện và công cụ phụ trợ khác:**
* pdf2image (dùng Poppler) để chuyển đổi trang PDF thành ảnh (khi dùng OCR Gemini).
* python-docx để tạo file DOCX đầu ra từ văn bản OCR (đặc biệt trong chế độ Gemini).
* ghostscript và poppler-utils hỗ trợ xử lý PDF (cần cho OCRmyPDF và pdf2image).
* tesseract-ocr và gói ngôn ngữ (ví dụ tesseract-ocr-vie cho tiếng Việt) để Tesseract nhận dạng ký tự.

## Chế độ chuyển đổi và quy trình xử lý

Công cụ cung cấp **ba chế độ xử lý** tùy theo loại PDF và nhu cầu của người dùng. Dưới đây là mô tả chi tiết về từng chế độ và quy trình thực hiện:

### 1. Chuyển đổi trực tiếp (Không OCR)

Ở chế độ "none", công cụ **không thực hiện OCR**. Thay vào đó, mỗi file PDF sẽ được chuyển thẳng thành DOCX bằng thư viện pdf2docx. Quy trình như sau:

* **Bước 1:** Kiểm tra trong thư mục Drive đích đã có file DOCX trùng tên chưa. Nếu DOCX đã tồn tại cho PDF đó, bỏ qua để tránh trùng lặp kết quả.
* **Bước 2:** Tải PDF từ Drive về bộ nhớ tạm của Colab.
* **Bước 3:** Dùng pdf2docx.Converter đọc PDF và **chuyển toàn bộ nội dung sang file DOCX**. Quá trình này giữ nguyên **các thành phần văn bản, hình ảnh, định dạng cơ bản** của tài liệu PDF gốc (trong điều kiện PDF có sẵn lớp text)[[1]](https://www.reddit.com/r/MicrosoftWord/comments/1iuv2te/convert_pdf_into_word_word/#:~:text=,cv.close).
* **Bước 4:** Upload file DOCX thu được ngược lên thư mục Google Drive ban đầu.

*Lưu ý:* Chế độ này cho kết quả tốt nhất về định dạng nếu file PDF **không phải dạng scan**. Nếu PDF chỉ chứa hình ảnh (scan) thì chế độ này sẽ tạo DOCX trống hoặc lỗi, do không có text để trích xuất[[5]](https://www.reddit.com/r/MicrosoftWord/comments/1iuv2te/convert_pdf_into_word_word/#:~:text=,ones%2C%20see%20OCR%20section%20below). Trong trường hợp đó, nên sử dụng một trong hai chế độ OCR bên dưới.

### 2. Chế độ OCR Tesseract (qua OCRmyPDF)

Ở chế độ "tesseract", công cụ sẽ áp dụng OCR bằng Tesseract để nhận diện văn bản trong PDF (nếu là scan), sau đó mới chuyển đổi sang DOCX. Các bước chính:

* **Bước 1:** (Tương tự) Kiểm tra sự tồn tại của DOCX đích và tải file PDF về từ Drive.
* **Bước 2:** **Thực hiện OCR bằng Tesseract** trên file PDF:
* Sử dụng lệnh ocrmypdf để tạo một bản PDF mới có lớp text ẩn. Công cụ được cấu hình với ngôn ngữ do người dùng lựa chọn (ví dụ: vie+eng để nhận dạng song ngữ Việt-Anh) và chế độ OCR:
  + *force\_ocr:* OCR toàn bộ ngay cả khi PDF có text sẵn (đảm bảo mọi trang đều có lớp text).
  + *skip\_text:* Chỉ OCR các trang không có text, bỏ qua trang đã có text sẵn.
* Kết quả thu được là một file PDF đã có **text layer** (lớp văn bản) nằm dưới lớp hình ảnh, nhờ đó file PDF này có thể **tìm kiếm, copy text** như PDF bình thường. Quan trọng hơn, **bố cục trang PDF gốc được giữ nguyên** sau khi thêm text[[2]](https://ploomber.io/blog/pdf-ocr/#:~:text=,OCR%20multiple%20PDF%20files%20simultaneously).
* **Bước 3:** **Chuyển đổi PDF (đã OCR) sang DOCX:** Dùng pdf2docx như chế độ 1 để chuyển đổi file PDF đã có lớp text thành DOCX. Nhờ có lớp text ẩn do Tesseract tạo ra, quá trình này sẽ **tái tạo nội dung văn bản vào DOCX đúng theo vị trí gốc**. Các đoạn văn, dòng chữ trên DOCX sẽ sắp xếp tương ứng vị trí trong PDF, giúp giảm công sức định dạng lại. (Tuy nhiên, mức độ chính xác còn tùy chất lượng OCR và độ phức tạp của bố cục PDF).
* **Bước 4:** Upload DOCX lên Drive như trên.

Chế độ Tesseract phù hợp khi cần **giữ bố cục giống PDF tối đa** và khi PDF có nhiều ngôn ngữ cụ thể. Tesseract là mã nguồn mở, hỗ trợ nhiều ngôn ngữ (bao gồm tiếng Việt) và có độ chính xác khá cao cho văn bản in. Đồng thời, OCRmyPDF cho phép xử lý nhiều file liên tiếp một cách tự động (batch processing)[[6]](https://ploomber.io/blog/pdf-ocr/#:~:text=control%2C%20and%20PDF%2FA%20generation.%20,OCR%20multiple%20PDF%20files%20simultaneously).

### 3. Chế độ OCR AI Gemini (Google Generative AI)

Ở chế độ "gemini", dự án khai thác **mô hình AI đa phương thức của Google** để thực hiện OCR một cách thông minh. Thay vì sử dụng quy trình OCR truyền thống, mô hình ngôn ngữ lớn (LLM) Gemini có thể **nhận trực tiếp hình ảnh và xuất ra text**, nhờ đã được huấn luyện trên dữ liệu kết hợp văn bản và hình ảnh[[3]](https://medium.com/google-cloud/text-extraction-from-image-using-google-gemini-6b9d9989fa84#:~:text=OCR%20with%20Google%20Gemini). Các bước thực hiện:

* **Bước 1:** Kiểm tra file DOCX trùng và tải PDF từ Drive (tương tự các chế độ trên).
* **Bước 2:** **Chuyển PDF thành chuỗi ảnh trang:** Sử dụng pdf2image, mỗi trang PDF được kết xuất thành hình ảnh (định dạng PNG). Mỗi trang tương ứng một ảnh độ phân giải đủ cao để OCR.
* **Bước 3:** **Gọi API Gemini để OCR từng trang:** Với mỗi ảnh trang PDF, công cụ gửi yêu cầu tới mô hình Gemini (thông qua Python SDK google-generativeai). Mô hình được cung cấp:
* **Prompt (lời nhắc):** *"Trích xuất toàn bộ văn bản trong hình ảnh này theo đúng thứ tự đọc. Chỉ trả về nội dung văn bản."* – hướng dẫn mô hình chỉ trả về phần chữ có trong ảnh.
* **Dữ liệu hình ảnh:** ảnh trang PDF được gửi kèm (dưới dạng bytes).
* Mô hình Gemini sau đó sẽ **phân tích hình ảnh và trả về đoạn text** được nhận diện. Quá trình này tận dụng khả năng hiểu ngữ cảnh và ngôn ngữ của LLM, giúp nó đọc đúng cả những đoạn khó mà không cần qua bước tiền xử lý, phân đoạn truyền thống[[4]](https://medium.com/google-cloud/text-extraction-from-image-using-google-gemini-6b9d9989fa84#:~:text=,and%20removes%20potential%20error%20points).
* **Bước 4:** **Tổng hợp văn bản và tạo DOCX:** Văn bản OCR của từng trang được ghép lại theo đúng thứ tự trang. Công cụ dùng python-docx để tạo file Word, mỗi trang ảnh tương ứng một trang DOCX (ngăn cách bằng ngắt trang). Chú ý: Do Gemini trả về **chỉ có chữ thuần túy**, file DOCX kết quả sẽ **mất toàn bộ định dạng gốc** (không có hình ảnh, không giữ được bố cục cột, bảng biểu như PDF; chỉ có nội dung chữ theo thứ tự đọc).
* **Bước 5:** Upload file DOCX lên Drive.

Chế độ Gemini hữu ích khi cần **độ chính xác cao về nội dung chữ** hoặc khi văn bản phức tạp (ví dụ chữ viết tay, ảnh mờ) mà OCR truyền thống có thể gặp khó khăn. LLM có thể dựa vào ngữ cảnh để sửa lỗi nhận dạng, hiểu từ viết tắt, hoặc ngôn ngữ chuyên ngành tốt hơn. Tuy nhiên, nhược điểm là **mất định dạng** và phụ thuộc kết nối mạng cùng API Key (yêu cầu có khóa API hợp lệ từ Google AI). Ngoài ra, việc gọi API nhiều lần (mỗi trang một lần) có thể chậm hơn so với Tesseract cho tài liệu nhiều trang, và có thể phát sinh giới hạn sử dụng hoặc chi phí.

## Kết quả thực nghiệm và đánh giá

Sau khi triển khai, công cụ in ra bảng tổng kết cho biết số file PDF đã xử lý, số file chuyển đổi thành công, file nào bị lỗi hay bỏ qua. Trong quá trình thử nghiệm trên thư mục chứa nhiều file PDF đa dạng, công cụ cho kết quả như sau:

* **Số lượng file xử lý:** (Ví dụ) 50 file PDF trong thư mục, trong đó:
* 40 file được chuyển đổi thành công sang DOCX.
* 5 file bỏ qua vì DOCX đã tồn tại từ trước (tránh trùng lặp).
* 5 file gặp lỗi (có thông báo lỗi chi tiết để người dùng kiểm tra, thường do PDF nguồn lỗi hoặc vượt quá khả năng OCR).
* **Chất lượng DOCX đầu ra:**
* Với chế độ *không OCR*, các file PDF văn bản đơn giản được chuyển đổi gần như hoàn hảo, giữ nguyên cấu trúc đoạn, font chữ. Những PDF phức tạp (nhiều cột, bảng biểu) có thể bị lệch nhẹ nhưng vẫn đỡ công sức định dạng lại hơn làm thủ công.
* Với chế độ *Tesseract OCR*, hầu hết chữ trong PDF scan đã được chèn đúng vị trí trong DOCX. Định dạng như in đậm/nghiêng không được giữ vì OCR chỉ tạo lớp văn bản đơn thuần. Bố cục tổng thể (dòng, đoạn, vị trí hình ảnh) tương đối khớp PDF. Một số lỗi OCR nhỏ (sai chính tả, ký tự đặc biệt) xuất hiện tùy chất lượng scan và ngôn ngữ, cần rà soát lại thủ công.
* Với chế độ *Gemini*, văn bản thu được rất đầy đủ ngay cả với tài liệu khó (ví dụ scan mờ hoặc bản in chất lượng kém). Độ chính xác chữ cái và từ ngữ cao, ít lỗi hơn Tesseract trong trường hợp phức tạp nhờ mô hình hiểu ngữ cảnh. Tuy nhiên, file Word chỉ là **một khối văn bản thuần**: mọi hình ảnh, bảng biểu của PDF gốc đều không có, người dùng sẽ cần chèn lại nếu cần thiết. Bù lại, tất cả nội dung chữ đã được trích xuất sẵn, hỗ trợ cho việc biên tập về sau.
* **Tốc độ:** Chế độ không OCR nhanh nhất (vài giây mỗi file tùy độ dài). Chế độ Tesseract mất nhiều thời gian hơn cho bước OCR (vì Tesseract xử lý từng trang, trung bình vài chục giây đến vài phút mỗi file tùy số trang). Chế độ Gemini có thời gian xử lý mỗi trang khoảng 1-2 giây cho gọi API (nhanh), nhưng tổng thời gian phụ thuộc số trang và bị giới hạn bởi tốc độ mạng và hạn mức API (nếu vượt hạn mức có thể phải chờ). Với ~50 trang PDF, Gemini OCR có thể mất khoảng 1-2 phút, tương đương Tesseract trong nhiều trường hợp (được chạy trên Colab pro10$/month).
* **Ưu điểm nổi bật:** Công cụ đã **tự động hóa hoàn toàn** quy trình vốn rất mất công khi làm thủ công. Người dùng không cần tải file về máy cục bộ rồi lại up lên, không phải cài đặt phần mềm nặng; tất cả chạy trên nền tảng đám mây. Việc hỗ trợ cả hai phương pháp OCR giúp linh hoạt theo nhu cầu:
* Tesseract (miễn phí, chạy offline trên Colab) phù hợp khi muốn giữ bố cục và không phụ thuộc internet sau khi môi trường sẵn sàng.
* Gemini (dịch vụ AI đám mây mới) hữu dụng khi cần **chính xác nội dung** cao nhất hoặc xử lý những trường hợp OCR khó, tận dụng sức mạnh AI hiện đại[[4]](https://medium.com/google-cloud/text-extraction-from-image-using-google-gemini-6b9d9989fa84#:~:text=,and%20removes%20potential%20error%20points).
* **Hạn chế và hướng phát triển:** Một số lỗi có thể xảy ra như:
* Lỗi cài đặt môi trường (thiếu gói Ghostscript, Tesseract) – đã được xử lý bằng cell cài đặt đầu tiên, nhưng người dùng phải chạy đúng trình tự.
* Lỗi do API Gemini (hết hạn mức, lỗi mạng) – chương trình có thông báo và thử lại, nhưng nếu vượt giới hạn sẽ cần đợi hoặc dùng khóa API khác.
* Định dạng DOCX từ OCR chưa đẹp: có thể cải thiện bằng cách hậu xử lý DOCX (ví dụ nhận diện tiêu đề, bảng để định dạng lại nếu dùng LLM).
* Dự án trong tương lai có thể bổ sung **giao diện người dùng trực quan hơn** (ví dụ form web) thay vì chạy trên Colab, và tích hợp thêm tùy chọn như xuất ra định dạng khác (PDF/A sau OCR, txt, v.v.).

## Kết luận

Dự án **“Chuyển đổi PDF sang DOCX hàng loạt”** đã hoàn thành mục tiêu đề ra, cung cấp một giải pháp thuận tiện cho việc chuyển đổi nhiều file PDF thành tài liệu Word. Bằng việc kết hợp các thư viện **mã nguồn mở** và **dịch vụ AI hiện đại**, công cụ cho phép xử lý cả tài liệu PDF có sẵn văn bản lẫn tài liệu scan một cách hiệu quả.

Người dùng giờ đây có thể tiết kiệm thời gian và công sức: chỉ cần một vài thiết lập ban đầu trên Google Colab, tất cả file PDF trong một thư mục sẽ được chuyển sang Word tự động. **Tính linh hoạt** của dự án nằm ở chỗ hỗ trợ nhiều chế độ OCR: - Với tài liệu PDF chuẩn: chuyển đổi trực tiếp nhanh chóng, giữ nguyên dàn trang. - Với tài liệu scan: có thể chọn OCR truyền thống (Tesseract) để giữ bố cục, hoặc OCR AI (Gemini) để tăng độ chính xác nội dung.

Kết quả thử nghiệm cho thấy công cụ hoạt động ổn định trên nhiều dạng tài liệu và cho độ chính xác cao trong nhận dạng văn bản. Dự án minh họa tiềm năng của việc kết hợp các công cụ Python và AI để **tự động hóa công việc văn phòng**, đồng thời mở ra hướng phát triển các ứng dụng chuyển đổi, xử lý tài liệu thông minh hơn trong tương lai.

**Tài liệu tham khảo:**

* Kinh nghiệm cộng đồng về chuyển đổi PDF sang Word bằng Python và OCR[[1]](https://www.reddit.com/r/MicrosoftWord/comments/1iuv2te/convert_pdf_into_word_word/#:~:text=,cv.close)[[7]](https://www.reddit.com/r/MicrosoftWord/comments/1iuv2te/convert_pdf_into_word_word/#:~:text=%2A%20Tesseract%20%2B%20Python,docx).
* Đặc tính của OCRmyPDF trong việc giữ nguyên bố cục tài liệu gốc khi thêm lớp văn bản OCR[[2]](https://ploomber.io/blog/pdf-ocr/#:~:text=,OCR%20multiple%20PDF%20files%20simultaneously).
* Sử dụng mô hình đa phương thức Gemini để trích xuất văn bản trực tiếp từ hình ảnh bằng sức mạnh của LLM[[4]](https://medium.com/google-cloud/text-extraction-from-image-using-google-gemini-6b9d9989fa84#:~:text=,and%20removes%20potential%20error%20points)[[3]](https://medium.com/google-cloud/text-extraction-from-image-using-google-gemini-6b9d9989fa84#:~:text=OCR%20with%20Google%20Gemini).

[[1]](https://www.reddit.com/r/MicrosoftWord/comments/1iuv2te/convert_pdf_into_word_word/" \l ":~:text=,cv.close) [[5]](https://www.reddit.com/r/MicrosoftWord/comments/1iuv2te/convert_pdf_into_word_word/#:~:text=,ones%2C%20see%20OCR%20section%20below) [[7]](https://www.reddit.com/r/MicrosoftWord/comments/1iuv2te/convert_pdf_into_word_word/#:~:text=%2A%20Tesseract%20%2B%20Python,docx) Convert PDF into word Word : r/MicrosoftWord

<https://www.reddit.com/r/MicrosoftWord/comments/1iuv2te/convert_pdf_into_word_word/>

[[2]](https://ploomber.io/blog/pdf-ocr/#:~:text=,OCR%20multiple%20PDF%20files%20simultaneously) [[6]](https://ploomber.io/blog/pdf-ocr/#:~:text=control%2C%20and%20PDF%2FA%20generation.%20,OCR%20multiple%20PDF%20files%20simultaneously) Python OCR libraries for converting PDFs into editable text

<https://ploomber.io/blog/pdf-ocr/>

[[3]](https://medium.com/google-cloud/text-extraction-from-image-using-google-gemini-6b9d9989fa84#:~:text=OCR%20with%20Google%20Gemini) [[4]](https://medium.com/google-cloud/text-extraction-from-image-using-google-gemini-6b9d9989fa84#:~:text=,and%20removes%20potential%20error%20points) Text Extraction from Image using Google Gemini | by Nishant Welpulwar | Google Cloud - Community | Medium

<https://medium.com/google-cloud/text-extraction-from-image-using-google-gemini-6b9d9989fa84>