Báo cáo thực tập

Trích xuất thông tin dựa trên mô hình ngôn ngữ lớn

1. Tổng quan
2. Giới thiệu

Hiện nay, xu hướng về khoa học dữ liệu đang phát triển nhanh chóng và có nhiều khía cạnh đáng chú ý như trí tuệ nhân tạo và ML, xử lí ngôn ngữ tự nhiên(NLP), dữ liệu lớn(Big data), khoa học dữ liệu trong ngành công nghiệp, tự động hóa và AI Ethics. Do đó, dữ liệu là vô cùng cần thiết để đáp ứng nhu cầu. Song các dữ liệu thu thập được chỉ dưới dạng dữ liệu thô và không có nhiều giá trị cho khách hàng. Vì vậy ta cần đưa ra được giải pháp để xử lí dữ liệu thô đó. Cùng với sự phát triển của các mô hình ngôn ngữ lớn, một giải pháp khả thi là sử dụng những mô hình này để trích xuất thông tin.

1. Mô hình ngôn ngữ lớn

Mô hình ngôn ngữ lớn là một mô hình học sâu được huấn luyện trên khối lượng dữ liệu và văn bản khổng lồ. Chúng sử dụng cấu trúc mạng nơ-ron, thường là mạng nơ-ron transformer, để học các mẫu và cấu trúc trong ngôn ngữ. Những mô hình này có khả năng xử lý và tạo ra văn bản tự nhiên một các rất chính xác.

Một số mô hình khả nổi tiếng như: GPT-3.5, GPT-4, Llama2, Llama3, vinallama, PhoGPT,....

Kiến trúc transfromer chính là thứ công nghệ đằng sau được áp dụng cho tất cả các mô hình ngôn ngữ lớn.

A diagram of a software algorithm

Description automatically generated

(Cấu trúc transformer)

1. Phương pháp
2. Các phương pháp truyền thống

Thật ngạc nhiên, nhiều công ty vẫn dựa vào việc nhập dữ liệu thủ công, mặc dù có sẵn các công nghệ tiên tiến hơn. Phương pháp này bao gồm việc nhập thông tin bằng tay vào hệ thống mục tiêu. Nò thường dễ chấp nhận hơn do chi phí ban đầu thấp hơn. Tuy nhiên, nhập dữ liệu thủ công không chỉ tốn thời gian mà còn rất dễ mắc lỗi. Ngoài ra, nó còn đạt ra rủi ro bảo mật khi xử lý dữ liệu nhạy cảm, khiến nó trở thành một lựa chọn kém hấp dẫn trong thời đại của tự động hóa và an ninh kỹ thuật số.

1. Phương pháp nhận diện quang học(OCR)

Công nghệ OCR, chuyển đổi hình ảnh và nội dụng viết tay thành dữ liệu có thể đọc được bằng máy, cung cấp giải pháp nhanh hơn và tiết kiệm chi phí hơn cho việc trích xuất dữ liệu. Tuy nheien, chất lượng có thể không đáng tin cậy. Hiệu suất của OCR bị ảnh hưởng đáng kể bởi độ phức tạp và đặc điểm của dữ liệu đầu vào, nó hoạt động không tốt với các văn bản viết tay, đặc biệt khi các hình ảnh phức tạp và khó xử lý. Các công cụ trích xuất dữ liệu với OCR trên thị trường thường được thêm nhiều lớp xử lý hậu kỳ để kiếm tra độ chính xác của dữ liệu trích xuất. Nhưng giải pháp này không thể đảm bảo chính xác 100%.

1. Khớp mẫu văn bản

Khớp mẫu văn bản là phương pháp xác định và trích xuất thông tin cụ thể từ văn bản bằng cách sử dụng các quy tắc hoặc mẫu được định sẵn. Nó nhanh hơn và mang lại ROI cao hơn so với các phương pháp khác. Phương pháp này hiệu quả ở mọi mức độ phức tạp và đạt độ chính xác 100% cho các tập tin có bố cục tương tự. Tuy nhiên, tính cứng nhắc trong việc khớp từng từ một có thể hạn chế khả năng thích ứng, yêu cầu khớp chính xác 100% để trích xuất thông tin thành công. Với các ngữ cảnh khác nhau việc khớp mẫu có thể gây ra hiểu nhầm. Việc tìm được sự cân bằng giữa tính cứng nhắc và khả năng thích ứng vẫn là một thách thức.

1. Nhận diện thực thể(NER)

Nhận diện thực thể (NER), một kỹ thuật trong xử lý ngôn ngữ tự nhiên (NLP), xác định và phân loại thông tin quan trọng trong văn bản.

Các trích xuất của NER bị giới hạn ở các thực thể được định sẵn như tên tổ chức, địa điểm, tên cá nhân và ngày tháng. Nói cách khác, các hệ thống NER hiện tại thiếu khả năng nội tại để trích xuất các thực thể tùy chỉnh ngoài bộ định sẵn này, điều này có thể cụ thể cho một lĩnh vực hoặc trường hợp sử dụng cụ thể. Thứ hai, sự tập trung của NER vào các giá trị chính liên quan đến các thực thể được nhận dạng không mở rộng đến việc trích xuất dữ liệu từ các bảng, làm hạn chế khả năng áp dụng của nó đối với các loại dữ liệu phức tạp hoặc có cấu trúc hơn.

Khi các tổ chức phải xử lý lượng dữ liệu phi cấu trúc ngày càng tăng, những thách thức này làm nổi bật nhu cầu về một phương pháp trích xuất toàn diện và có khả năng mở rộng.

1. Khai phá dữ liệu phi cấu trúc với LLMs

Tận dụng các mô hình ngôn ngữ lớn là một giải pháp hấp dẫn với các ưu điểm riêng biệt nhằm giải quyết các thách thức về nhận biết ngữ cảnh, khả năng tính toán song song, thích ứng với các loại dữ liệu đa dạng.

1. Ứng dụng
2. Trích xuất, tổng hợp thông tin

Đầu tiên và quan trọng nhất đó chính là chuẩn bị dữ liệu thô. Sau đó là lựa chọn mô hình ngôn ngữ lớn phù hợp với việc trích xuất thông tin. Trong khi sử dụng các LLMs, song song với đó ta cần sử dụng các kĩ thuật tinh chỉnh để tối ưu hóa các tác vụ trích xuất thông tin cụ thể như quantization, prompt, n-shot,... Việc này có thể bao gồm huấn luyện mô hình trên một tập dữ liệu đặc thù hoặc sử dụng mô hình đã được huấn luyện sẵn.

Cuối cùng sẽ áp dụng LLMs để tổng hợp thông tin tạo ra các tóm tắt từ văn bản dài, giúp rút ra các điểm chính và thông tin quan trọng

VD dưới đây là việc sử dụng Meta-Llama-3-8B-Intruct để trích xuất thông tin từ dữ liệu nhà cao tầng đã được crawl từ các trang web.

Tải và import các thư viện cần thiết



A black screen with white text

Description automatically generated

Tạo api-key của huggingface và thực hiện login

A screenshot of a computer program

Description automatically generated

Quantization model để nhẹ hợn và có tốc độ nhanh hơn

A screen shot of a computer

Description automatically generated

Cấu hình để thực hiện tác vụ sinh văn bản bằng cách sử dụng một mô hình ngôn ngữ và một tokenizer cụ thể. Khi bạn gọi pipeline này, nó sẽ nhận một đoạn văn bản đầu vào, sử dụng mô hình để sinh ra một hoặc nhiều đoạn văn bản mới (trong trường hợp này là 1), và trả về kết quả đã được chuyển đổi từ các token trở lại văn bản dễ đọc.

A screenshot of a computer program

Description automatically generated

Định nghĩa hàm để lấy phản hồi từ model

A screen shot of a computer program

Description automatically generated

Load dữ liệu và lấy cột mô tả, định dạng cho excel table cần lưu, đưa ra ví dụ và dùng few-shot để hướng dẫn mô hình đưa ra dữ liệu

A screenshot of a computer program

Description automatically generated

Định nghĩa hàm để phân tích dữ liệu json nhận được từ model

A computer screen shot of a program code

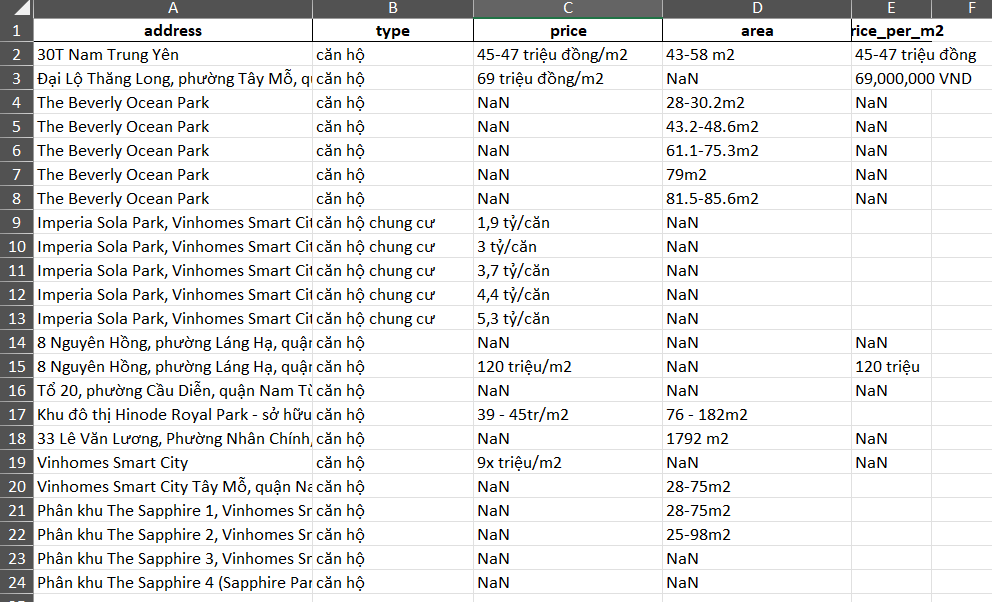
Description automatically generated

Cuối cùng ta thực hiện trích xuất từng mô tả

A screen shot of a computer program

Description automatically generated

Kết quả thu được khi chạy thử với 10 đoạn mô tả đầu tiên



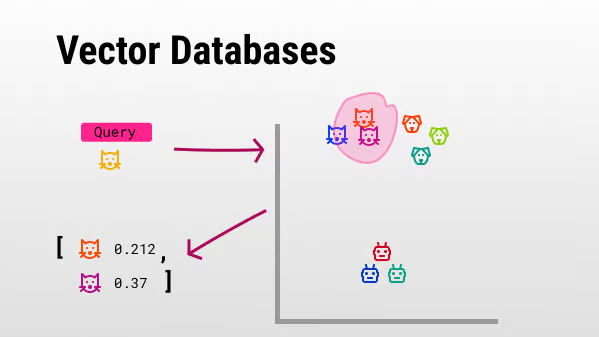
1. Sử dụng RAG

Trong thực tế, các mô hình học máy chỉ học được dữ liệu đến thời điểm nào đó nhất định. Ví dụ chat gpt-3 chỉ đến tháng 12 năm 2021. Do đó những dữ liệu dung cấp sẽ không còn chính xác với hiện tại. Vì vậy ta cần một phương pháp để các mô hình có thể học được những dữ liệu riêng của mình.

Retrieval-Augmented Generation (RAG) là một kỹ thuật giúp nâng cao khả năng của mô hình sinh (language model generation) kết hợp với tri thức bên ngoài (external knowledge)

Phương pháp này thực hiện bằng cách truy xuất thông tin liên quan từ kho tài liệu (tri thức) và sử dụng chúng cho quá trình sinh câu trả lời dựa trên LLMs.

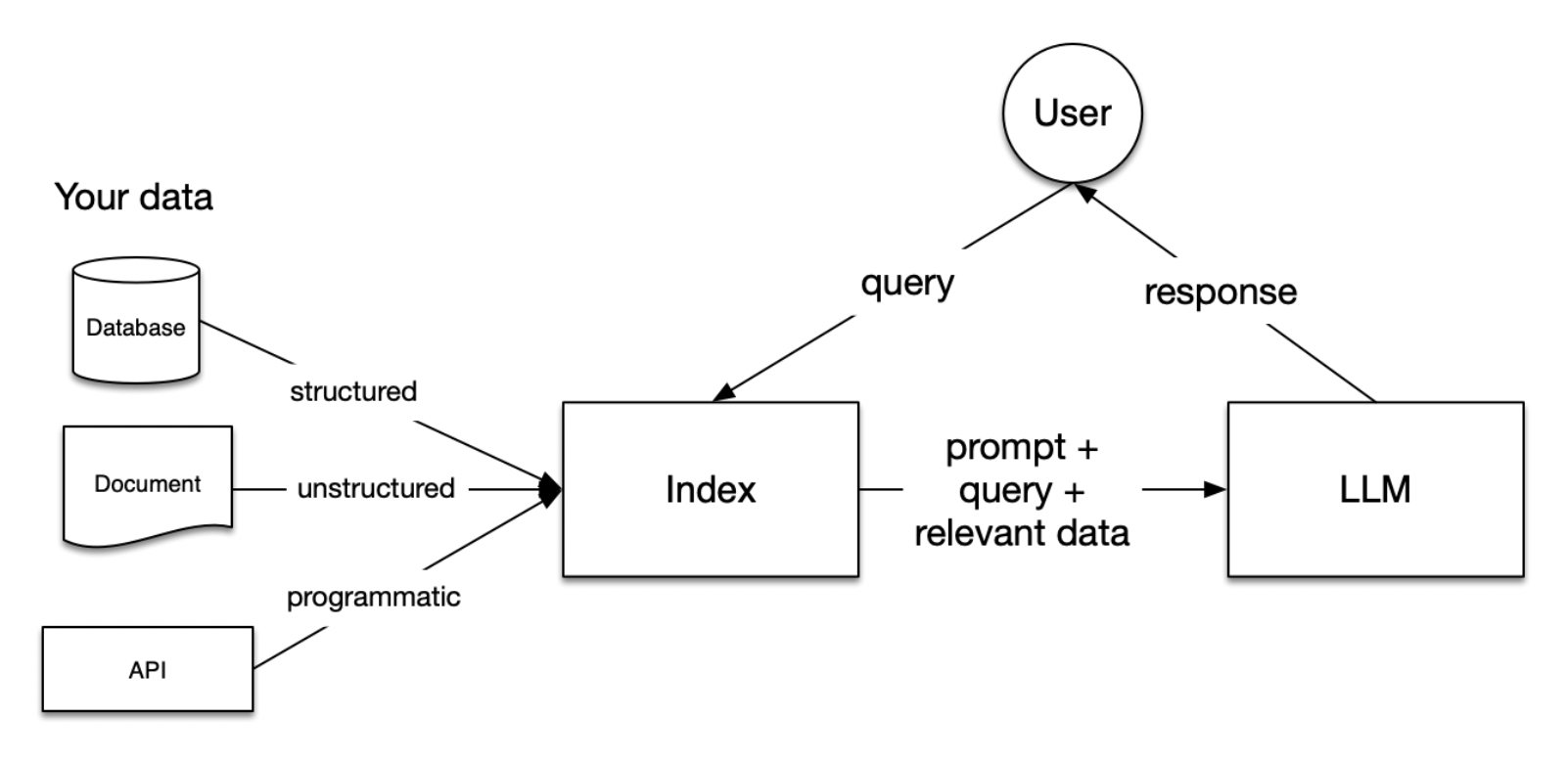
Trong phương pháp này, cần sử dụng một vector database để lưu trữ các thông tin. Vector database là một hệ thống lưu trữ dữ liệu được tối ưu hóa để lưu trữ và truy vấn các vector, tức là các tập hợp các giá trị số, thường là các số thực, được sắp xếp theo một thứ tự cố định.

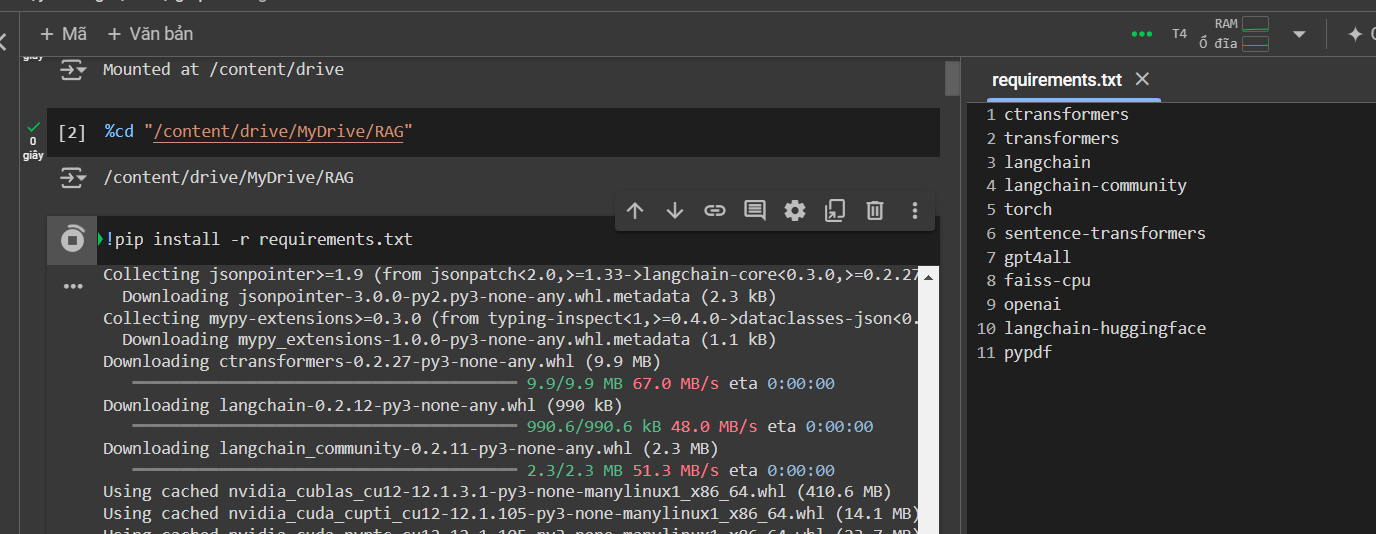


A diagram of embedding model

Description automatically generated

Các bước thực hiện:



Tải và import các thư viện cần thiết  


Load data và chia thành các chunks

A screenshot of a computer program

Description automatically generated

Định nghĩa hàm và model\_embedding cần dùng để tạo và lưu trữ vào vector database  
A screen shot of a computer program

Description automatically generated

Load vector database và in thử ra 1 giá trị vector  
A screenshot of a computer

Description automatically generated

Đăng nhập và sử dụng api-key của hugging face

A screenshot of a computer program

Description automatically generated

Quantization để có tốc độ nhanh và mô hình nhẹ hơn

A screenshot of a computer program

Description automatically generated

Cấu hình để thực hiện tác vụ sinh văn bản bằng cách sử dụng một mô hình ngôn ngữ và một tokenizer cụ thể. Khi bạn gọi pipeline này, nó sẽ nhận một đoạn văn bản đầu vào, sử dụng mô hình để sinh ra một hoặc nhiều đoạn văn bản mới (trong trường hợp này là 1), và trả về kết quả đã được chuyển đổi từ các token trở lại văn bản dễ đọc.

A screen shot of a computer

Description automatically generated

Định nghĩa hàm truy xuất từ database trả về văn bản của tài liệu ban đầu

A screen shot of a computer program

Description automatically generated

Gửi và nhận phản hồi từ model sau đó tạo sinh thông tin theo ý muốn bằng kỹ thuật prompt

A screenshot of a computer program

Description automatically generated

Định nghĩa hàm RAG tổng hợp lại các quy trình trên và chạy thử với một query

**A screenshot of a computer program

Description automatically generated**

1. Thách thức

Sử dụng các mô hình ngôn ngữ lớn (LLMs) để trích xuất thông tin có thể mang lại nhiều lợi ích, nhưng cũng đối mặt với một số thách thức đáng lưu ý:

**Chất lượng dữ liệu đầu vào**

* **Đầu vào không rõ ràng**: Nếu dữ liệu đầu vào không chính xác, thiếu cấu trúc, hoặc có lỗi, mô hình có thể trích xuất thông tin sai lệch hoặc không đầy đủ.
* **Tiền xử lý dữ liệu**: Cần phải đảm bảo rằng dữ liệu được chuẩn bị và tiền xử lý đúng cách để mô hình có thể hiểu và xử lý chính xác.

**Chi phí và tài nguyên**

* **Chi phí tính toán**: Các mô hình ngôn ngữ lớn yêu cầu tài nguyên tính toán đáng kể, làm tăng chi phí vận hành và triển khai.
* **Tối ưu hóa**: Cần phải tối ưu hóa mô hình để đảm bảo hiệu suất và giảm thiểu các vấn đề về tài nguyên.

**Vấn Đề Về Thời Gian**

* **Thời gian phản hồi**: Các mô hình ngôn ngữ lớn thường yêu cầu thời gian xử lý dài hơn để tạo ra kết quả, đặc biệt khi làm việc với văn bản dài hoặc phức tạp. Điều này có thể gây trì hoãn trong các ứng dụng yêu cầu phản hồi nhanh chóng.
* **Chi phí đào tạo và triển khai**: Đào tạo và tinh chỉnh các mô hình ngôn ngữ lớn có thể mất nhiều thời gian và tài nguyên. Việc này có thể dẫn đến sự chậm trễ trong việc triển khai mô hình mới hoặc cập nhật mô hình hiện tại.
* **Tối ưu hóa thời gian xử lý**: Để cải thiện thời gian xử lý, cần phải tối ưu hóa mô hình và thuật toán, nhưng điều này có thể đòi hỏi các kỹ thuật phức tạp và công việc tối ưu hóa liên tục. Việc tối ưu hóa cũng có thể làm giảm độ chính xác của mô hình trong một số trường hợp.
* **Quản lý khối lượng dữ liệu lớn**: Khi xử lý khối lượng dữ liệu lớn, thời gian để đọc, lưu trữ, và phân tích dữ liệu có thể trở thành một vấn đề. Điều này yêu cầu các giải pháp lưu trữ hiệu quả và kỹ thuật phân tích dữ liệu mạnh mẽ để giảm thời gian xử lý.

1. Kết luận

Sử dụng LLMs trong lĩnh vuẹc tríhc xuất thong tin có nhiều triển vọng như: tự động hóa quá trình trích xuất thông tin, cải thiện độ chính xác, có thể xử lý các dữ liệu phi cấu trúc dài và phức tạp cải thiện quản lý tài nguyên. Tuy nhiên, vẫn còn gặp phải một số khó khăn về mặt kỹ thuật như tốc độ và ngữ nghĩa. Đây sẽ là một giải pháp triển vọng trong tương lai.