BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG
BỘ MÔN MẠNG MÁY TÍNH & TRUYỀN THÔNG

NIÊN LUẬN CƠ SỞ
NGÀNH CÔNG NGHỆ THÔNG TIN

Đề tài

ỨNG DỤNG RETRIEVAL-AUGMENTED GENERATION XÂY DỰNG CHATBOT TƯ VẤN TUYỂN SINH VÀ TÍCH HỢP VÀO FACEBOOK MESSENGER

Sinh viên: Nguyễn Minh Nhựt
Mã số: B2205896
Khóa: K48

Cần Thơ, 04/2025

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG
BỘ MÔN MẠNG MÁY TÍNH & TRUYỀN THÔNG

NIÊN LUẬN CƠ SỞ
NGÀNH CÔNG NGHỆ THÔNG TIN

Đề tài

ỨNG DỤNG RETRIEVAL-AUGMENTED GENERATION XÂY DỰNG CHATBOT TƯ VẤN TUYỂN SINH VÀ TÍCH HỢP VÀO FACEBOOK MESSENGER

Người hướng dẫn
TS.Phạm Thế Phi

Sinh viên thực hiện
Nguyễn Minh Nhựt
Mã số: B2205896
Khóa: K48

Cần Thơ, 04/2025

LỜI CẢM ƠN

Trước hết, em bày tỏ lòng biết ơn chân thành tới Trường Công nghệ thông tin và Truyền thông vì sự tận tình trong việc giảng dạy, giúp em có những kiến thức cơ bản vững chắc để hoàn thành Niên luận cơ sở Ngành Công nghệ thông tin.

Em đặc biệt muốn gửi lời cảm ơn sâu sắc tới Thầy Phạm Thế Phi, người đã nhiệt tình chia sẻ những kiến thức và góp ý quý giá, hỗ trợ em trong suốt quá trình làm niên luận. Những góp ý của thầy đã đem đến kiến thức và thúc đẩy tinh thần học hỏi, say mê nghiên cứu của em, giúp em có động lực cố gắng hết sức để hoàn thành niên luận cơ sở. Những kiến thức này sẽ là nền tảng cho sự phát triển sau này trong lĩnh vực Công nghệ thông tin.

Tuy nhiên, với kiến thức và kinh nghiệm hạn chế, em nhận thức được rằng bài báo cáo có thể còn nhiều thiếu sót. Em kính mong nhận được sự đánh giá và góp ý từ thầy để khắc phục và cải thiện.

Em xin chân thành cảm ơn!

Cần Thơ, ngày 14 tháng 4 năm 2025

Sinh viên thực hiện

Nguyễn Minh Nhựt

MỤC LỤC

DANH MỤC HÌNH

TÓM TẮT

Niên luận tập trung vào việc xây dựng chatbot tư vấn tuyển sinh ứng dụng mô hình Retrieval-Augmented Generation (RAG) kết hợp với mô hình ngôn ngữ lớn (LLM) để sinh phản hồi tự động. Chatbot có khả năng truy xuất thông tin từ tài liệu tuyển sinh chính thức và tạo câu trả lời chính xác, hỗ trợ thí sinh và phụ huynh tra cứu điểm chuẩn, ngành học, điều kiện xét tuyển, hồ sơ nhập học,... một cách nhanh chóng. Hệ thống đã được tích hợp vào Facebook Messenger, và có thể hoạt động mà không cần sự can thiệp liên tục của con người. Kết quả nghiên cứu cho thấy Chatbot giúp tối ưu hóa nguồn lực tư vấn, giảm tải công việc lặp lại và nâng cao trải nghiệm người dùng, mở ra hướng ứng dụng trí tuệ nhân tạo trong lĩnh vực giáo dục.

ABSTRACT

This thesis focuses on developing an admissions consulting chatbot using the Retrieval-Augmented Generation (RAG) model combined with Large Language Model (LLM) to generate automated responses. The chatbot can retrieve information from official admissions documents and generate accurate answers, supporting students and parents to quickly access details about scores, academic programs, eligibility criteria, application procedures, and more. The system has been integrated into Facebook Messenger and can operate without continuous human intervention. The research findings indicate that the chatbot optimizes consulting resources, reduces repetitive workloads, and enhances user experience, paving the way for the application of artificial intelligence in education.

PHẦN GIỚI THIỆU

1. Đặt vấn đề

Trong những năm gần đây, quy chế tuyển sinh của các trường đại học liên tục được cập nhật và đổi mới, kéo theo đó là nhu cầu tìm hiểu thông tin ngày càng cao từ học sinh và phụ huynh. Tuy nhiên, việc tiếp cận đầy đủ và chính xác các thông tin quan trọng như ngành học, điểm chuẩn, chương trình đào tạo, quy trình tuyển sinh, điều kiện nhập học, học phí,… không phải lúc nào cũng dễ dàng.

Mặt khác, các trường đại học cũng gặp thách thức trong việc xây dựng và duy trì một đội ngũ tư vấn viên đủ lớn và nắm chắc quy chế tuyển sinh để hỗ trợ kịp thời cho số lượng lớn thí sinh và phụ huynh, đặc biệt vào mùa tuyển sinh. Ngoài ra đội ngũ tư vấn sẽ phải giải đáp các thắc mắc tương tự nhau lập lại nhiều lần cho nhiều người khác nhau. Hơn thế, các phương thức tư vấn truyền thống như điện thoại, email hay gặp mặt trực tiếp thường mất nhiều thời gian, nguồn lực và không còn đủ linh hoạt để đáp ứng nhu cầu ngày càng cao.

Trong khi đó, thí sinh và phụ huynh mong muốn một kênh hỗ trợ nhanh chóng, chính xác, tiện lợi và có thể hoạt động 24/7 để giải đáp mọi thắc mắc một cách hiệu quả. Trước thực trạng này, việc ứng dụng công nghệ để xây dựng một hệ thống tư vấn tự động, thông minh và kịp thời là một giải pháp cần thiết nhằm nâng cao chất lượng tư vấn, tối ưu hóa nguồn lực và cải thiện trải nghiệm của người dùng là nhu cầu cấp thiết các trường Đại học.

2. Mục tiêu đề tài

Xây dựng một chatbot tư vấn tuyển sinh có thể tích hợp vào Facebook hoặc website của trường, giúp thí sinh và phụ huynh dễ dàng tra cứu thông tin. Chatbot có khả năng trả lời chính xác các câu hỏi về điểm chuẩn, ngành học, thời gian nhận hồ sơ, quy trình tuyển sinh, điều kiện nhập học,… đồng thời được thiết kế với cơ sở dữ liệu linh hoạt, dễ dàng cập nhật theo từng năm. Giao diện chat đơn giản, thân thiện với người dùng, không yêu cầu cài đặt phức tạp và có thể tự động hóa quy trình tư vấn, hoạt động 24/7 để hỗ trợ thí sinh một cách hiệu quả.

3. Đối tượng nghiên cứu và phạm vi nghiên cứu

Mô hình chatbot tư vấn tuyển sinh: nghiên cứu các phương pháp, kỹ thuật xây dựng chatbot có khả năng truy xuất và tổng hợp thông tin tuyển sinh từ cơ sở dữ liệu quy chế tuyển sinh của một trường, đảm bảo cập nhật dễ dàng theo từng năm.

Tích hợp và triển khai: nghiên cứu phương pháp tích hợp chatbot vào Facebook Messenger hoặc website, đảm bảo hoạt động ổn định và giao diện thân thiện.

4. Phương pháp nghiên cứu

Thu thập và xử lý dữ liệu: tổng hợp các văn bản chứa thông tin về quy chế tuyển sinh của trường Đại học. Nghiên cứu các phương pháp tiền xử lý dữ liệu, bao gồm làm sạch, chuẩn hóa, tổ chức và định dạng dữ liệu để phù hợp với hệ thống chatbot, đảm bảo tính chính xác, khả năng truy xuất linh hoạt và nâng cấp dễ dàng.

Nghiên cứu và phát triển mô hình: khảo sát các mô hình chatbot phổ biến được áp dụng trong tư vấn và dịch vụ khách hàng, đặc biệt là những mô hình có khả năng truy xuất thông tin và tạo câu trả lời tự động. Phân tích và lựa chọn mô hình phù hợp với yêu cầu của đề tài.

Tích hợp và triển khai: Nghiên cứu các phương thức tích hợp chatbot vào nền tảng Facebook Messenger, website của trường, giao diện trực quan và dễ sử dụng. Đánh giá hiệu quả chatbot thông qua kiểm tra khả năng phản hồi chính xác, duy trì ngữ cảnh trò chuyện và đáp ứng nhu cầu tư vấn tuyển sinh.

5. Bố cục niên luận cơ sở

Nội dung của quyển niên luận cơ sở gồm các phần sau đây:

- Phần giới thiệu

Phần này trình bày các vấn đề của đề tài, mục tiêu đề tài, những nghiên cứu được thực hiện trong lúc thực hiện đề tài.

- Phần nội dung

Phần này trình bày chi tiết bài toán, thiết kế và cài đặt hệ thống, đồng thời nêu lên quy trình kiểm thử, đánh giá phần mềm. Bao gồm các phần:

Chương 1: Đặc tả yêu cầu

Chương 2: Cơ sở lý thuyết

Chương 3: Cài đặt giải pháp.

Chương 4: Đánh giá kiểm thử

Phần kết luận: Phần này trình bày kết quả đạt được của đề tài cũng như những hạn chế mà đề tài chưa thực hiện được, ngoài ra cũng đưa ra hướng phát triển sau này.

PHẦN NỘI DUNG

Chương 1. Đặc tả yêu cầu

1.1. Mô tả chi tiết hệ thống

Trong bối cảnh tuyển sinh đại học ngày càng có nhiều đổi mới và thay đổi nhanh chóng, thí sinh và phụ huynh gặp nhiều khó khăn trong việc tiếp cận thông tin kịp thời, chính xác. Đội ngũ tư vấn của các trường thường xuyên chịu áp lực lớn vì phải giải đáp hàng trăm, thậm chí hàng nghìn câu hỏi, hầu hết có nội dung lặp lại, trong khi những kênh hỗ trợ truyền thống như email, điện thoại hay gặp mặt trực tiếp tỏ ra thiếu linh hoạt.

Trước thực trạng đó, đề tài tập trung xây dựng một hệ thống chatbot tư vấn tuyển sinh nhằm tự động hóa quy trình giải đáp và tối ưu hóa trải nghiệm người dùng. Chatbot được thiết kế có khả năng hiểu ngôn ngữ tự nhiên tiếng Việt, truy xuất thông tin từ kho dữ liệu tuyển sinh của trường và tổng hợp phản hồi dưới dạng hội thoại. Nhờ vậy, thí sinh và phụ huynh có thể nhanh chóng tìm được thông tin về điểm chuẩn, điều kiện xét tuyển, hồ sơ đăng ký, học phí hay chương trình đào tạo mới nhất theo nhu cầu.

Hệ thống chatbot tư vấn tuyển sinh này được thiết kế để cung cấp thông tin một cách độc lập cho từng câu hỏi mà không ghi nhớ ngữ cảnh hội thoại trước đó. Mỗi truy vấn từ người dùng sẽ được xử lý riêng lẻ, đảm bảo phản hồi nhanh chóng và chính xác mà không bị ảnh hưởng bởi các câu hỏi trước đó. Điều này giúp hệ thống duy trì sự ổn định, đơn giản và dễ triển khai, đồng thời phù hợp với mục đích chính là cung cấp thông tin tuyển sinh một cách hiệu quả, thay vì duy trì các cuộc hội thoại phức tạp.

Hệ thống cho phép cập nhật dữ liệu khi có thay đổi về quy chế tuyển sinh. Trước khi đưa vào sử dụng, dữ liệu này sẽ trải qua quá trình tiền xử lý nhằm loại bỏ thông tin dư thừa, chuẩn hóa định dạng và chuyển đổi thành dạng vector để chatbot có thể tìm kiếm nhanh chóng khi nhận truy vấn từ người dùng. Sau khi truy xuất được các đoạn văn bản liên quan từ kho dữ liệu, hệ thống sẽ sử dụng mô hình ngôn ngữ lớn (LLM) để tổng hợp nội dung, đảm bảo câu trả lời mạch lạc, dễ hiểu và sát với yêu cầu của người dùng trước khi gửi phản hồi. Đây là một bước tiện lợi, giúp cập nhật thông tin mới mà không cần huấn luyện lại mô hình, đảm bảo chatbot luôn cung cấp câu trả lời chính xác và kịp thời.

Nhờ kiến trúc linh hoạt và khả năng tùy biến, chatbot có thể triển khai trên nhiều kênh như website chính thức hoặc mạng xã hội. Hệ thống sẽ hoạt động hoàn toàn tự động 24/7, giúp giảm tải công việc lặp lại cho nhân viên tư vấn, đồng thời cung cấp cho thí sinh và phụ huynh một kênh hỗ trợ tiện lợi, có thể truy cập mọi lúc, mọi nơi.

1.2. Vấn đề và giải pháp liên quan đến bài toán

1.2.1. Vấn đề trong việc xây dựng chatbot tư vấn tuyển sinh

Chatbot tư vấn tuyển sinh là một hệ thống đòi hỏi khả năng xử lý ngôn ngữ tự nhiên, truy xuất thông tin chính xác và đảm bảo phản hồi mạch lạc cho người dùng. Tuy nhiên, trong quá trình phát triển, có nhiều thách thức đặt ra liên quan đến độ chính xác của mô hình, chất lượng dữ liệu và khả năng mở rộng của hệ thống.

Một trong những phương pháp phổ biến để xây dựng chatbot là sử dụng các kịch bản cố định, trong đó chatbot phản hồi dựa trên các câu trả lời được lập trình sẵn. Phương pháp này có ưu điểm là dễ triển khai nhưng thiếu linh hoạt, vì nó chỉ có thể xử lý một tập hợp các truy vấn giới hạn. Khi người dùng đặt câu hỏi ngoài phạm vi được lập trình, chatbot dễ dàng đưa ra câu trả lời không liên quan hoặc từ chối phản hồi. Điều này ảnh hưởng lớn đến trải nghiệm của người dùng, đặc biệt trong lĩnh vực tuyển sinh, nơi thông tin thay đổi thường xuyên và yêu cầu khả năng cập nhật linh hoạt.

Vấn đề tiếp theo là khả năng xử lý ngôn ngữ tự nhiên. Đối với chatbot tư vấn tuyển sinh, việc hiểu đúng câu hỏi và ý định của người dùng là yếu tố quan trọng để cung cấp phản hồi chính xác. Tuy nhiên, tiếng Việt là một ngôn ngữ có cấu trúc phức tạp với nhiều biến thể, dấu câu, ngữ cảnh khác nhau và đặc biệt là các cách diễn đạt không cố định. Điều này khiến cho chatbot dựa trên mô hình học máy hoặc các phương pháp truyền thống gặp khó khăn trong việc xử lý chính xác truy vấn của người dùng. Nếu mô hình không đủ mạnh, chatbot có thể hiểu sai ý định hoặc trả lời không đúng trọng tâm, làm giảm hiệu quả tư vấn.

Một thách thức lớn khác trong việc phát triển chatbot tư vấn tuyển sinh là vấn đề dữ liệu. Để chatbot hoạt động hiệu quả, cần có một kho dữ liệu hỏi đáp phong phú, đa dạng và được tổ chức tốt. Tuy nhiên, dữ liệu chuyên ngành tuyển sinh thường không có sẵn ở dạng chuẩn hóa, mà rải rác trong các tài liệu quy chế, trang web hoặc văn bản nội bộ. Việc thu thập, làm sạch và gán nhãn cho dữ liệu này đòi hỏi nhiều thời gian và công sức. Hơn nữa, dữ liệu cần được cập nhật liên tục để phản ánh chính xác các thay đổi về quy chế tuyển sinh, điểm chuẩn, chỉ tiêu ngành học và các điều kiện xét tuyển. Nếu dữ liệu không được cập nhật kịp thời, chatbot có thể cung cấp thông tin lỗi thời, gây ảnh hưởng tiêu cực đến quyết định của thí sinh và phụ huynh.

Ngoài ra, hiệu suất và tính khả dụng của chatbot cũng là một vấn đề quan trọng. Trong mùa tuyển sinh, số lượng người truy cập và đặt câu hỏi có thể tăng đột biến, yêu cầu hệ thống phải có khả năng xử lý nhiều truy vấn đồng thời mà không làm chậm thời gian phản hồi. Việc tối ưu hóa tốc độ truy xuất dữ liệu và sinh câu trả lời là một thách thức lớn, đặc biệt khi hệ thống phải xử lý các tập dữ liệu lớn trong thời gian thực. Nếu chatbot phản hồi chậm hoặc không chính xác, người dùng có thể mất niềm tin vào hệ thống.

Phương pháp fine-tuning mô hình ngôn ngữ lớn (LLM) để cải thiện chất lượng phản hồi cũng đặt ra nhiều vấn đề. Mặc dù fine-tuning giúp chatbot học được ngữ cảnh chuyên biệt hơn, nhưng nếu dữ liệu huấn luyện không đủ lớn hoặc không bao phủ đầy đủ các tình huống thực tế, chatbot có thể gặp khó khăn trong việc tổng quát hóa kiến thức. Hơn nữa, việc tinh chỉnh mô hình yêu cầu tài nguyên tính toán lớn, thời gian huấn luyện dài và khó khăn trong việc cập nhật dữ liệu mới. Khi có thay đổi về chính sách tuyển sinh, nếu chatbot dựa vào mô hình đã fine-tuned trước đó, hệ thống sẽ phải huấn luyện lại hoàn toàn để phản ánh thông tin mới, gây ra sự trì hoãn và tăng chi phí vận hành.

Những vấn đề trên cho thấy rằng, để xây dựng một chatbot tư vấn tuyển sinh hiệu quả, cần một phương pháp tiếp cận linh hoạt hơn, có khả năng truy xuất thông tin nhanh chóng mà không phụ thuộc quá nhiều vào việc huấn luyện lại mô hình. Đây là lý do mà phương pháp Retrieval-Augmented Generation (RAG) trở thành một lựa chọn tối ưu cho bài toán này.

1.2.2. Giải pháp cho việc phát triển chatbot tư vấn tuyển sinh:

Qua quá trình nghiên cứu và đánh giá các phương pháp xây dựng chatbot, RAG (Retrieval-Augmented Generation) được lựa chọn làm giải pháp chính nhờ vào những đặc trưng nổi bật phù hợp với bối cảnh tư vấn tuyển sinh. Thay vì dựa hoàn toàn vào mô hình ngôn ngữ đã được huấn luyện sẵn, RAG kết hợp giữa truy xuất thông tin nội bộ và sinh phản hồi, giúp chatbot vừa có khả năng tìm kiếm nội dung liên quan từ kho dữ liệu tuyển sinh của trường, vừa có thể tổng hợp và diễn đạt thông tin một cách tự nhiên, mạch lạc.

Một ưu điểm quan trọng của RAG so với fine-tuning là khả năng cập nhật dữ liệu một cách linh hoạt mà không cần huấn luyện lại toàn bộ mô hình. Khi có thay đổi về quy chế tuyển sinh, điểm chuẩn hoặc thông tin ngành học, đội ngũ quản trị chỉ cần bổ sung tài liệu mới vào kho dữ liệu mà không ảnh hưởng đến hoạt động của Chatbot. Điều này giúp đảm bảo rằng hệ thống luôn cung cấp thông tin chính xác mà không gặp phải độ trễ do quá trình huấn luyện lại mô hình như trong phương pháp fine-tuning.

Bên cạnh đó, việc áp dụng RAG giúp tối ưu hóa hiệu suất của chatbot. Nhờ vào cơ chế truy xuất thông tin theo dạng vector, hệ thống có thể nhanh chóng tìm kiếm nội dung phù hợp từ kho tri thức, giảm đáng kể thời gian xử lý so với việc chatbot tự sinh phản hồi hoàn toàn bằng mô hình ngôn ngữ lớn. Điều này đặc biệt quan trọng trong mùa tuyển sinh, khi số lượng người truy vấn tăng cao và yêu cầu tốc độ phản hồi nhanh để đảm bảo trải nghiệm người dùng.

Ngoài ra, việc sử dụng RAG cũng giúp cải thiện tính minh bạch và độ tin cậy của chatbot. Vì chatbot không tự động suy đoán câu trả lời mà dựa trên dữ liệu thực tế được truy xuất từ tài liệu chính thức của trường, hệ thống có thể cung cấp thông tin có độ chính xác cao, giảm thiểu nguy cơ trả lời sai hoặc không phù hợp. Điều này tạo

sự tin tưởng cho thí sinh và phụ huynh khi sử dụng chatbot để tra cứu thông tin tuyển sinh.

Tóm lại, phương pháp RAG không chỉ giúp chatbot tư vấn tuyển sinh phản hồi linh hoạt, chính xác mà còn giải quyết được các hạn chế của những phương pháp truyền thống như chatbot kịch bản cố định hoặc mô hình fine-tuning.

Chương 2. Cơ sở lý thuyết

2.1. Retrieval-Augmented Generation (RAG):

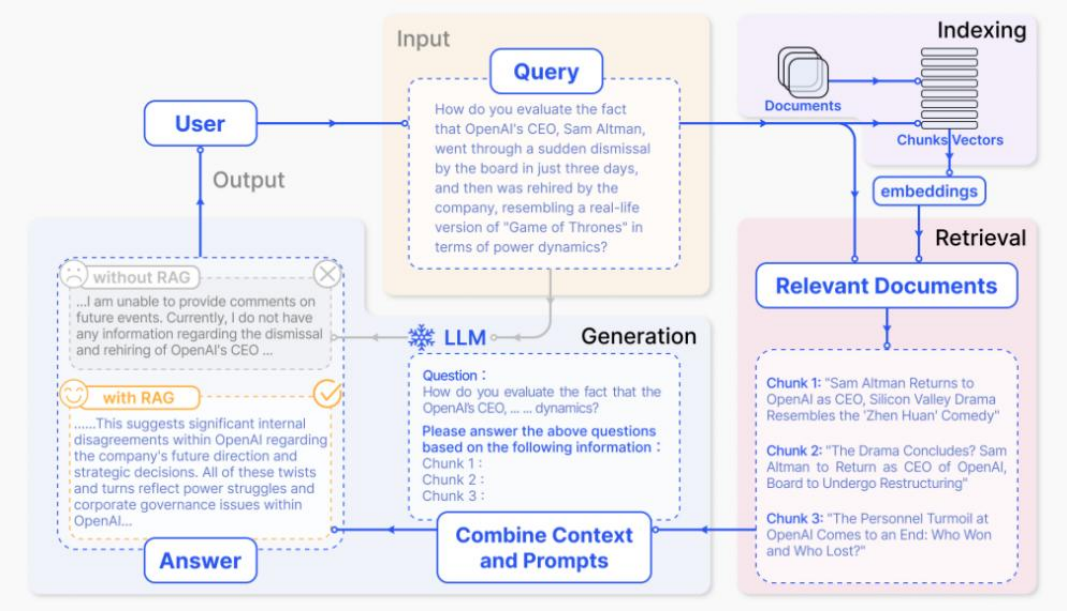
2.1.1. Tổng quan về Retrieval-Augmented Generation:

Retrieval-Augmented Generation (RAG) là một phương pháp tiên tiến trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP), giúp cải thiện hiệu quả của các mô hình sinh văn bản bằng cách tích hợp cơ chế truy xuất thông tin. Thay vì chỉ dựa vào dữ liệu tĩnh đã được huấn luyện trước, RAG cho phép mô hình tham khảo một tập hợp tài liệu bên ngoài để lấy thông tin liên quan đến truy vấn của người dùng, từ đó sinh ra phản hồi phù hợp hơn.

Cách tiếp cận này mang lại nhiều lợi ích so với các mô hình ngôn ngữ truyền thống. Các mô hình chỉ sử dụng dữ liệu huấn luyện cố định có xu hướng bị hạn chế bởi phạm vi thông tin được cung cấp trong quá trình đào tạo, dẫn đến nguy cơ sinh ra câu trả lời lỗi thời hoặc không chính xác. RAG khắc phục nhược điểm này bằng cách kết hợp hai bước: trước tiên, hệ thống truy xuất các tài liệu phù hợp từ cơ sở dữ liệu có sẵn, sau đó sử dụng mô hình ngôn ngữ lớn (LLM) để tạo câu trả lời dựa trên thông tin tìm được. Điều này đảm bảo rằng chatbot có thể cung cấp thông tin chính xác mà không cần phải huấn luyện lại mô hình mỗi khi có sự thay đổi trong dữ liệu đầu vào.

Với ứng dụng trong chatbot tư vấn tuyển sinh, RAG giúp hệ thống cung cấp phản hồi chính xác dựa trên tài liệu tuyển sinh mới nhất. Khi có sự thay đổi về quy chế hoặc chính sách tuyển sinh, chỉ cần cập nhật kho dữ liệu mà không cần huấn luyện lại toàn bộ mô hình, giúp tiết kiệm đáng kể tài nguyên tính toán và thời gian cập nhật.

2.1.2. Quy trình RAG



Hình 1: Kiến trúc của một mô hình RAG[1]

- Chỉ mục hóa (Indexing) là bước đầu tiên trong quy trình, trong đó các tài liệu liên quan được chuyển đổi thành dạng nhúng vector (vector embeddings) và lưu trữ trong một cơ sở dữ liệu tìm kiếm nhanh. Việc ánh xạ tài liệu thành vector giúp hệ thống có thể nhanh chóng xác định mức độ tương đồng giữa nội dung truy vấn của người dùng và các đoạn văn bản trong cơ sở dữ liệu.

- Truy xuất thông tin (Retrieval) là quá trình tìm kiếm và trích xuất các tài liệu liên quan nhất từ cơ sở dữ liệu vector dựa trên truy vấn của người dùng. Khi một câu hỏi được đặt ra, hệ thống sẽ sử dụng cơ chế so khớp vector để xác định những đoạn văn bản có độ tương đồng cao với truy vấn đầu vào.

- Tăng cường thông tin (Combine Context and Prompts) là giai đoạn mà các kết quả truy xuất được kết hợp với truy vấn ban đầu để cung cấp bối cảnh cho mô hình ngôn ngữ lớn. Bước này giúp đảm bảo rằng phản hồi sinh ra dựa trên dữ liệu có thật, thay vì chỉ dựa vào kiến thức được mô hình học từ trước.

- Tạo sinh phản hồi (Generation) là bước cuối cùng, trong đó mô hình ngôn ngữ lớn sử dụng cả truy vấn ban đầu và thông tin được truy xuất để tạo ra câu trả lời hoàn chỉnh. Nhờ vào bước này, chatbot có thể phản hồi linh hoạt và chính xác theo ngữ cảnh của người dùng, trong khi vẫn đảm bảo thông tin dựa trên các nguồn dữ liệu có sẵn.

Cấu trúc này giúp chatbot tư vấn tuyển sinh cung cấp thông tin đáng tin cậy mà không yêu cầu cập nhật thủ công mô hình AI. Thay vì phải tinh chỉnh mô hình ngôn ngữ mỗi khi có thay đổi về điểm chuẩn hoặc chính sách xét tuyển, chỉ cần cập nhật kho dữ liệu, từ đó giúp hệ thống hoạt động linh hoạt và có khả năng mở rộng tốt hơn.

2.1.3. Lợi ích của RAG:

Việc áp dụng RAG trong chatbot tư vấn tuyển sinh mang lại nhiều lợi ích thiết thực. Trước hết, RAG giúp chatbot có thể truy xuất thông tin từ nguồn tài liệu chính thống, đảm bảo rằng câu trả lời luôn dựa trên dữ liệu thực tế và không bị lỗi thời. Điều này đặc biệt quan trọng trong môi trường tuyển sinh, nơi các quy định, điểm chuẩn và chính sách xét tuyển có thể thay đổi hằng năm.

Ngoài ra, RAG cải thiện độ tin cậy của chatbot bằng cách giảm thiểu hiện tượng "ảo giác thông tin" (hallucination), một vấn đề phổ biến ở các mô hình ngôn ngữ lớn khi chúng tự tạo ra nội dung không có thực. Nhờ vào cơ chế truy xuất tài liệu, chatbot có thể cung cấp phản hồi kèm theo nguồn tham khảo, giúp thí sinh và phụ huynh dễ dàng xác minh thông tin.

Hơn nữa, RAG tối ưu hóa tài nguyên bằng cách cho phép hệ thống sử dụng dữ liệu có sẵn thay vì đòi hỏi một lượng lớn dữ liệu để huấn luyện. Điều này giúp giảm đáng kể chi phí vận hành và phát triển, đồng thời tăng tính linh hoạt khi triển khai chatbot trên các nền tảng khác nhau.

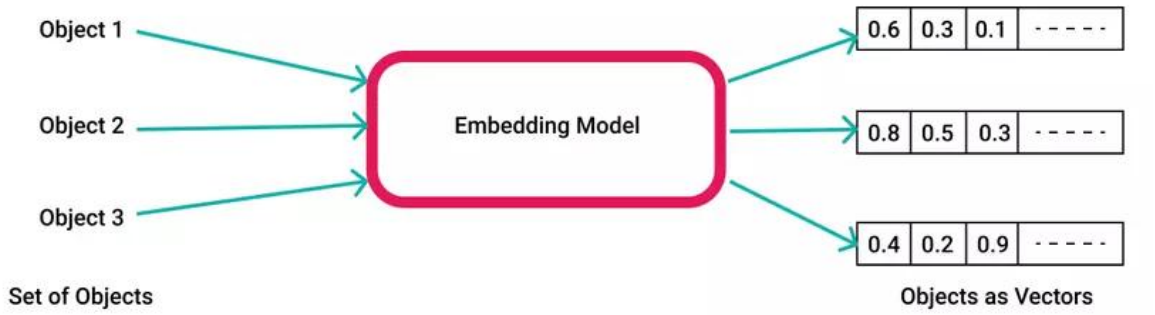
Tóm lại, RAG không chỉ nâng cao chất lượng câu trả lời của chatbot mà còn giúp hệ thống cập nhật dữ liệu dễ dàng, duy trì tính chính xác và cung cấp trải nghiệm người dùng tốt hơn trong quá trình tư vấn tuyển sinh.

2.2. Document Embedding (Statement Embedding)

2.2.1. Tổng quan về Document Embedding

Document Embedding là kỹ thuật ánh xạ văn bản thành dạng biểu diễn số trong không gian vector, giúp hệ thống có thể xử lý, tìm kiếm và so sánh nội dung dựa trên ý nghĩa thay vì chỉ dựa vào từ khóa. Quá trình này cho phép chatbot tư vấn tuyển sinh nhận diện mối liên hệ giữa các câu hỏi của người dùng và nội dung tài liệu đã lưu trữ, từ đó cải thiện độ chính xác khi tìm kiếm thông tin.

Một lợi thế quan trọng của Document Embedding là khả năng biểu diễn ngữ nghĩa của văn bản một cách ngắn gọn nhưng đầy đủ. Các câu có ý nghĩa tương tự nhau sẽ có vector embedding gần nhau trong không gian số, giúp hệ thống dễ dàng xác định và trích xuất các đoạn văn bản phù hợp với truy vấn của người dùng. Điều này đặc biệt quan trọng trong chatbot tư vấn tuyển sinh, nơi mà câu hỏi của thí sinh có thể được diễn đạt theo nhiều cách khác nhau nhưng vẫn hướng đến cùng một nội dung thông tin. Có thể đơn giản hình dung cách embedding vector được sinh ra như một black box dưới đây:

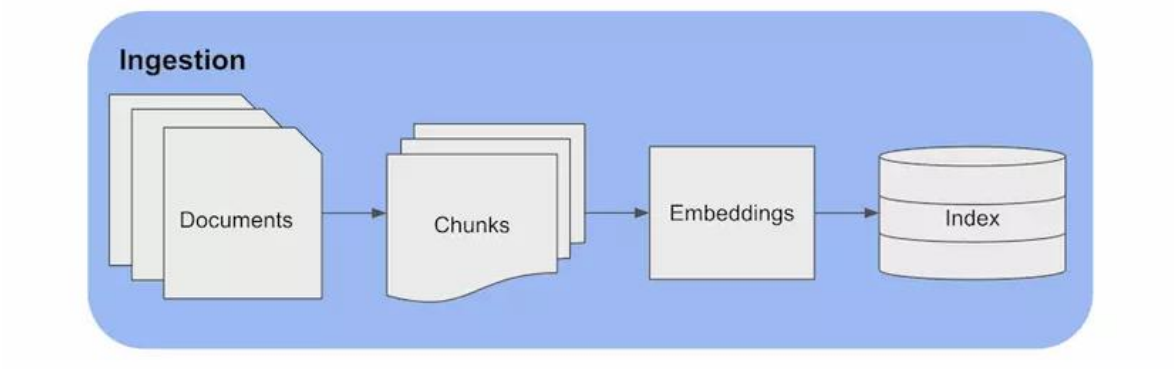


Hình 2: Minh họa hoạt động của mô hình nhúng văn bản[2]

Trong chatbot ứng dụng RAG, embedding đóng vai trò quan trọng trong việc giúp hệ thống tìm kiếm các đoạn văn hoặc câu có liên quan nhất đến câu hỏi của người dùng, đồng thời so khớp truy vấn với các câu trả lời tiềm năng trong cơ sở dữ liệu. Ngoài ra, embedding còn hỗ trợ tóm tắt hoặc suy luận thông tin dựa trên dữ liệu đã được nhúng. Vì embedding vector biểu diễn đặc trưng của một đối tượng đầu vào, nên các đối tượng có đặc điểm tương đồng hoặc cùng thể loại sẽ có vector gần nhau trong không gian nhúng, giúp chatbot hoạt động hiệu quả hơn trong việc truy xuất thông tin.

2.2.2. Quy trình tạo Document Embedding

Document Embedding trải qua nhiều bước quan trọng để chuyển đổi tài liệu từ dạng văn bản thô sang biểu diễn vector.



Hình 3: Quy trình nhúng đoạn vặn bản

Trước tiên, dữ liệu đầu vào cần được tiền xử lý để loại bỏ các ký tự không cần thiết, chuẩn hóa dấu câu và tách từ chính xác. Đối với tiếng Việt, quá trình tokenization phải đảm bảo rằng các cụm từ quan trọng không bị cắt nhỏ, giúp mô hình embedding có thể học được mối quan hệ giữa các từ trong câu.

Sau khi dữ liệu được làm sạch, bước tiếp theo là chia nhỏ văn bản thành các đoạn ngắn hơn (chunking). Trong hệ thống chatbot tư vấn tuyển sinh, các tài liệu như quy chế tuyển sinh, đề án tuyển sinh hay tài liệu hướng dẫn thường có độ dài lớn, không thể xử lý toàn bộ trong một lần. Vì vậy, việc chia văn bản thành các đoạn ngắn hơn giúp hệ thống dễ dàng tìm kiếm và trích xuất thông tin chính xác hơn.

Mỗi đoạn văn sau đó được ánh xạ thành một vector embedding và lưu trữ trong cơ sở dữ liệu vector. Khi người dùng nhập truy vấn, hệ thống sẽ so sánh vector của truy vấn với các vector tài liệu đã lưu để tìm ra nội dung phù hợp nhất.

2.2.3. Document Embedding trong RAG

Trong mô hình Retrieval-Augmented Generation (RAG), Document Embedding đóng vai trò không thể thiếu:

Truy xuất tài liệu liên quan: Khi người dùng đặt câu hỏi, hệ thống sẽ sinh vector embedding cho truy vấn, rồi so sánh với các vector embedding đã được lưu trong cơ sở dữ liệu. Những đoạn văn có độ tương đồng ngữ nghĩa cao nhất sẽ được chọn để tổng hợp thành câu trả lời cuối cùng.

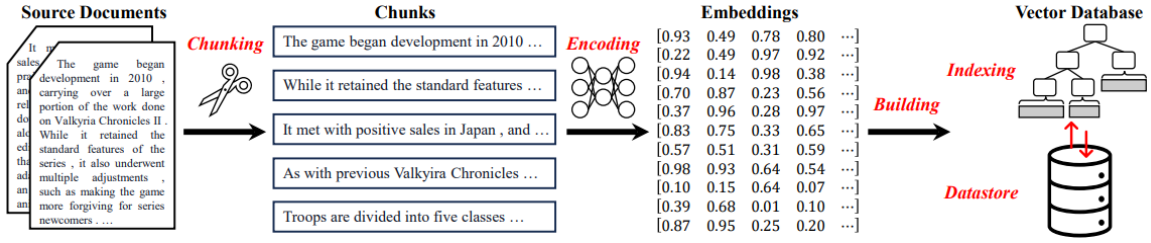
Giảm thiểu "ảo giác thông tin": Khi mô hình ngôn ngữ lớn (LLM) xây dựng câu trả lời, việc tham chiếu đến các vector embedding của tài liệu gốc giúp cung cấp bối cảnh và dữ liệu chính xác, hạn chế tình trạng mô hình tự "suy diễn" thông tin không có thực.

Dễ dàng cập nhật dữ liệu: Khi có sự thay đổi hoặc bổ sung thông tin mới liên quan đến tuyển sinh, đội ngũ phát triển chỉ cần nhúng lại (re-embed) những nội dung đã chỉnh sửa và thêm các vector này vào cơ sở dữ liệu vector. Nhờ vậy, hệ thống luôn duy trì khả năng truy xuất và trả lời nhất quán, không bị gián đoạn.

2.3. Cơ sở dữ liệu vector (Vector Database)

2.3.1. Tổng quan về cơ sở dữ liệu vector

Cơ sở dữ liệu vector là hệ thống lưu trữ và quản lý dữ liệu dưới dạng vector nhúng, tương tự như cách các cơ sở dữ liệu truyền thống lưu trữ chuỗi ký tự hoặc dữ liệu dạng bảng. Điểm khác biệt nằm ở chỗ, thay vì chỉ lưu trữ văn bản thô hoặc các thuộc tính dạng số thông thường, một cơ sở dữ liệu vector sẽ tiếp nhận các đại diện số học (vector embeddings) của văn bản hoặc hình ảnh, âm thanh. Mỗi vector nhúng là một biểu diễn số học của dữ liệu, được tạo ra thông qua các mô hình học sâu để thể hiện mối quan hệ ngữ nghĩa giữa các đoạn văn bản hoặc dữ liệu gốc. Nhờ có khả năng này, cơ sở dữ liệu vector cho phép hệ thống tìm kiếm và đối sánh tài liệu dựa trên mức độ tương đồng về mặt ý nghĩa, thay vì chỉ dừng lại ở mức độ trùng khớp ký tự như các phương pháp truyền thống.

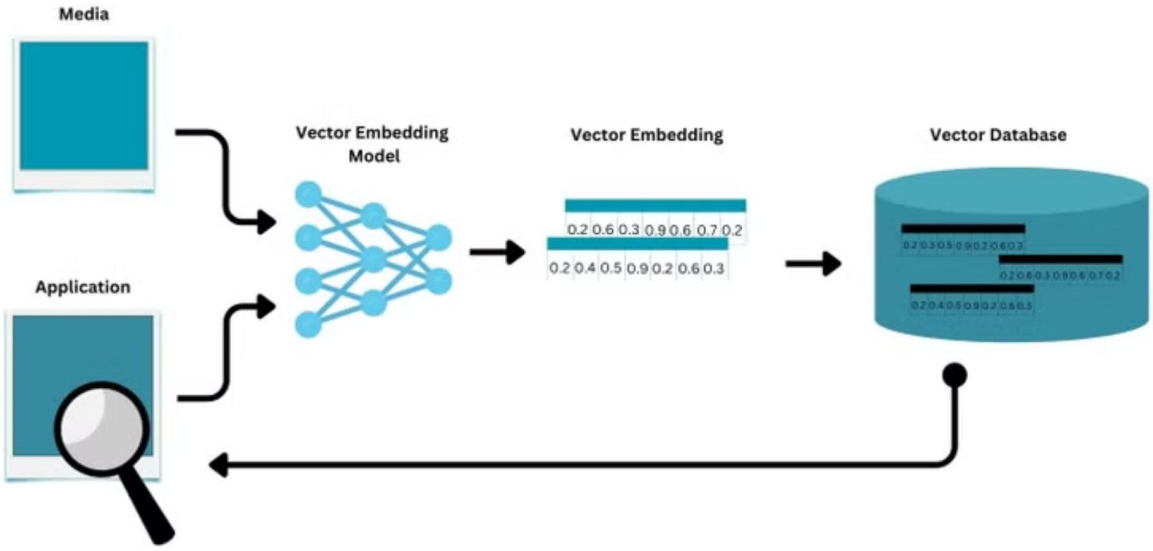


Hình 4: Cơ sở dữ liệu vector[4]

2.3.2. Cấu trúc và nguyên lý hoạt động

Cơ sở dữ liệu vector hoạt động dựa trên việc tạo chỉ mục cho các vector nhúng. Sau khi dữ liệu được tiền xử lý và chia thành những đoạn văn bản ngắn, mỗi đoạn sẽ được đưa qua mô hình embedding để chuyển thành vector số. Các vector này thường có chiều (dimension) cố định, ví dụ 384 chiều hoặc 768 chiều, tùy thuộc vào mô hình được sử dụng. Tiếp đó, cơ sở dữ liệu vector sẽ áp dụng các kỹ thuật lập chỉ mục nhằm hỗ trợ việc tìm kiếm nhanh chóng trong không gian nhiều chiều. Cơ sở dữ liệu vector sử dụng các thuật toán tìm kiếm lân cận gần nhất (Nearest Neighbour Search – NNS) để tìm ra các vector có khoảng cách nhỏ nhất so với vector truy vấn. Quá trình tìm kiếm có thể thực hiện theo hướng chính xác tuyệt đối hoặc gần đúng (Approximate Nearest Neighbor – ANN), tùy vào yêu cầu về tốc độ và độ chính xác của hệ thống.

Ở quy mô nhỏ, phương pháp tìm kiếm chính xác tuyệt đối có thể áp dụng, nhưng khi số lượng vector trong cơ sở dữ liệu tăng lên hàng triệu hoặc hàng chục triệu, việc quét toàn bộ vector để tìm kết quả phù hợp nhất trở nên không khả thi do tiêu tốn nhiều tài nguyên. Để giải quyết vấn đề này, các thuật toán tìm kiếm lân cận gần đúng được sử dụng nhằm tối ưu tốc độ truy vấn mà vẫn duy trì độ chính xác cao.



Hình 5: Cách database vecotr hoạt động

Các thuật toán phổ biến trong cơ sở dữ liệu vector bao gồm:

Hierarchical Navigable Small World (HNSW): HNSW sử dụng cấu trúc đồ thị phân cấp, cho phép hệ thống tìm kiếm theo cách duyệt qua các lớp nút liên kết với nhau. Cách tiếp cận này giúp rút ngắn khoảng cách tìm kiếm và tăng tốc độ truy vấn, đặc biệt hữu ích khi số lượng vector rất lớn.

Annoy (Approximate Nearest Neighbors Oh Yeah): Được phát triển bởi Spotify, Annoy sử dụng cấu trúc rừng cây (forest of trees) để phân chia không gian vector thành nhiều vùng nhỏ hơn, giúp giảm số lượng phép so sánh khi tìm kiếm. Điều này giúp tăng tốc độ truy xuất dữ liệu trong các hệ thống có quy mô lớn.

Inverted File Index (IVF): IVF là một kỹ thuật phổ biến trong thư viện FAISS của Facebook AI Research. Nó sử dụng phương pháp phân cụm để lập chỉ mục, giúp tìm kiếm nhanh hơn bằng cách chỉ so sánh truy vấn với một tập hợp con các vector thay vì toàn bộ dữ liệu. Khi kết hợp với lượng tử hóa vector, IVF có thể tối ưu hóa hiệu suất mà vẫn giữ được mức chính xác cao.

2.3.3. Ưu điểm khi sử dụng cơ sở dữ liệu vector

Khi văn bản đã được chuyển thành vector embedding, hệ thống có thể truy xuất theo ngữ nghĩa thay vì chỉ so sánh từ khóa, giúp cải thiện độ chính xác khi tìm kiếm. Cơ chế indexing và phân mảnh dữ liệu trong cơ sở dữ liệu vector còn cho phép chatbot phản hồi nhanh hơn, duy trì chất lượng tư vấn trong điều kiện số lượng truy vấn tăng cao vào giai đoạn cao điểm tuyển sinh. Một lợi thế khác nằm ở tính linh hoạt khi cập nhật dữ liệu, bởi đội ngũ vận hành chỉ cần nhúng lại những văn bản mới hoặc đã chỉnh sửa, thay vì phải cấu hình lại toàn bộ hệ thống.

Nhờ khả năng tối ưu về tốc độ và biểu diễn ngữ nghĩa vượt trội, cơ sở dữ liệu vector ngày càng trở thành thành phần cốt lõi trong các hệ thống RAG nói chung và trong giải pháp chatbot tư vấn tuyển sinh nói riêng. Hệ thống có thể dễ dàng tích hợp và mở rộng, cho phép đội ngũ quản trị cập nhật thông tin mới, thêm hoặc bớt tài liệu cần thiết. Từ đó, chatbot luôn bảo đảm nguồn thông tin chính xác, cải thiện trải nghiệm cho người dùng, đồng thời giúp nhà trường tiết kiệm thời gian và nguồn lực trong suốt quá trình cung cấp dịch vụ tư vấn. Ngoài ra, cơ sở dữ liệu vector còn hỗ trợ kiến trúc serverless, giúp tối ưu hóa chi phí lưu trữ và vận hành.

2.4. Mô hình Ngôn ngữ Lớn (LLM)

2.4.1. Tổng quan về Xử lý Ngôn ngữ Tự nhiên (NLP)

Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP) là lĩnh vực nghiên cứu giúp máy tính có thể hiểu, phân tích và tạo sinh văn bản một cách tự động. NLP kết hợp các phương pháp từ học máy, mạng nơ-ron và thống kê để xử lý các bài toán liên quan đến ngôn ngữ như phân tích cú pháp, trích xuất thông tin, nhận diện thực thể có tên, dịch máy và tổng hợp văn bản.

Trong chatbot tư vấn tuyển sinh, NLP đóng vai trò quan trọng trong việc giúp hệ thống hiểu đúng câu hỏi của người dùng, xác định ý định (intent) và các thực thể quan trọng (entity) để tạo ra phản hồi phù hợp với ngữ cảnh. Một thách thức lớn khi áp dụng NLP vào chatbot tư vấn tuyển sinh là xử lý tiếng Việt, một ngôn ngữ có cấu trúc phức tạp với nhiều hiện tượng như đồng âm khác nghĩa, biến đổi hình thái và cách diễn đạt phong phú. Hệ thống cần có khả năng hiểu rõ từng câu hỏi theo nhiều cách diễn đạt khác nhau để đảm bảo phản hồi chính xác.

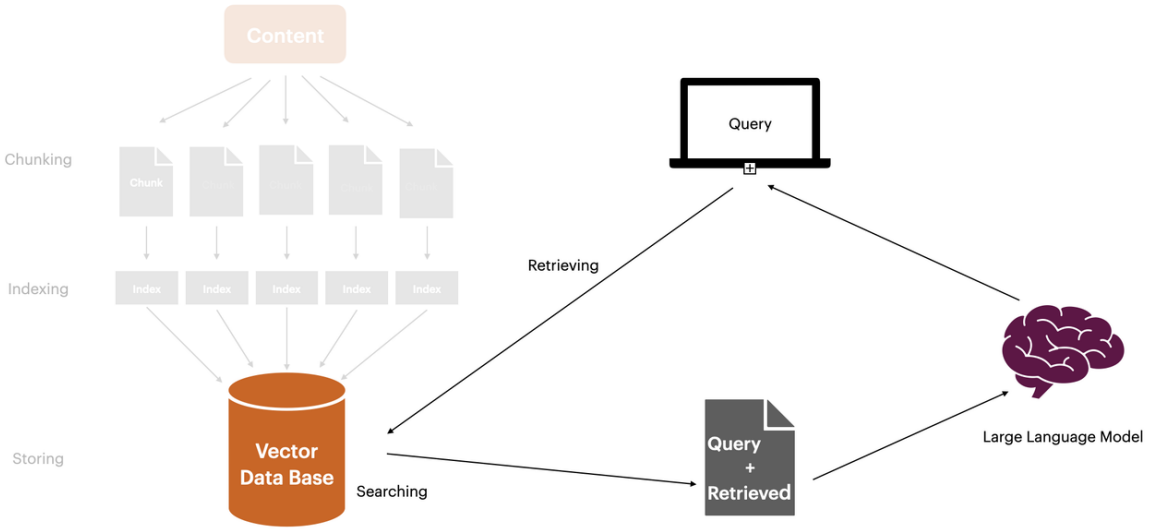
2.4.2. Mô hình Ngôn ngữ Lớn (LLM) trong NLP

Mô hình Ngôn ngữ Lớn (Large Language Model - LLM) là một bước tiến quan trọng trong NLP, giúp máy tính có thể hiểu và sinh văn bản với độ chính xác cao. Các mô hình như GPT-4, BERT hay T5 được huấn luyện trên khối lượng dữ liệu khổng lồ, cho phép chúng dự đoán từ tiếp theo, tạo văn bản mạch lạc và thực hiện các tác vụ ngôn ngữ phức tạp.

Tuy nhiên, các LLM truyền thống gặp một số hạn chế đáng kể. Một trong những vấn đề lớn nhất là hiện tượng "ảo giác thông tin" (hallucination), khi mô hình có thể tự tạo ra câu trả lời không chính xác hoặc không dựa trên dữ liệu thực tế. Ngoài ra, LLM không có khả năng tự động cập nhật thông tin, vì sau khi được huấn luyện, kiến thức của mô hình bị giới hạn trong phạm vi dữ liệu đã được sử dụng. Điều này gây khó khăn khi áp dụng vào chatbot tư vấn tuyển sinh, nơi thông tin liên tục thay đổi theo từng kỳ tuyển sinh. Một nhược điểm khác là các mô hình này không thể cung cấp nguồn tham chiếu cụ thể cho câu trả lời của mình, khiến người dùng khó xác minh tính chính xác của thông tin.

2.4.3. Kết hợp LLM với Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) là một kỹ thuật tiên tiến trong NLP, giúp cải thiện hiệu suất của LLM bằng cách tích hợp khả năng truy xuất thông tin từ các nguồn dữ liệu có cấu trúc như cơ sở dữ liệu vector hoặc kho tài liệu văn bản. Phương pháp này hoạt động theo hai giai đoạn chính:



Hình 6: Luồng hoạt động của RAG và LLM

Đầu tiên, khi nhận được truy vấn từ người dùng, hệ thống sẽ tìm kiếm các đoạn văn bản có liên quan trong cơ sở dữ liệu tuyển sinh bằng cách so khớp vector nhúng. Thay vì chỉ dựa vào kiến thức đã được huấn luyện sẵn, chatbot có thể truy xuất thông tin từ những tài liệu mới nhất, đảm bảo phản hồi luôn cập nhật và chính xác.

Tiếp theo, thông tin truy xuất được kết hợp vào truy vấn gốc và đưa vào mô hình ngôn ngữ lớn để tạo sinh câu trả lời. Nhờ vậy, chatbot có thể phản hồi một cách tự nhiên mà vẫn đảm bảo tính chính xác của dữ liệu. RAG giúp cải thiện đáng kể chất lượng phản hồi, giảm thiểu lỗi "ảo giác thông tin" do mô hình không tự suy đoán mà dựa vào dữ liệu thực tế.

Nhờ vào việc tích hợp RAG, chatbot có thể cung cấp các câu trả lời chi tiết, có căn cứ mà không cần huấn luyện lại toàn bộ mô hình. Khi có thay đổi về quy chế tuyển sinh hoặc ngành học, đội ngũ quản trị chỉ cần cập nhật kho dữ liệu mà không ảnh hưởng đến hiệu suất chatbot. Điều này giúp hệ thống duy trì độ chính xác cao mà không cần tốn nhiều tài nguyên tính toán.

2.5. Prompt Engineering:

2.5.1. Tổng quan về Prompt Engineering

Prompt Engineering là quá trình thiết kế các câu lệnh (prompt) để tối ưu hóa khả năng sinh phản hồi của mô hình ngôn ngữ lớn (LLM). Đây là một yếu tố quan trọng trong quá trình tương tác giữa con người và mô hình AI, đặc biệt trong các hệ thống chatbot tư vấn tuyển sinh, nơi yêu cầu phản hồi phải chính xác, mạch lạc và sát với nhu cầu của người dùng. Prompt không chỉ giúp định hướng đầu ra của mô hình mà còn có thể cải thiện hiệu suất bằng cách giảm thiểu lỗi sai, hạn chế các phản hồi không mong muốn và đảm bảo nội dung phản hồi có tính nhất quán.

Prompt Engineering đã trở thành một lĩnh vực quan trọng trong nghiên cứu và ứng dụng AI. Các phương pháp thiết kế prompt có thể ảnh hưởng đáng kể đến hiệu quả của chatbot, từ việc xử lý truy vấn phức tạp cho đến khả năng duy trì ngữ cảnh trong các cuộc hội thoại kéo dài. Trong chatbot tư vấn tuyển sinh, một prompt được thiết kế tốt sẽ giúp hệ thống hiểu rõ câu hỏi của người dùng, truy xuất thông tin phù hợp và trình bày phản hồi một cách rõ ràng, dễ hiểu.

2.5.2. Các loại Prompt

Có nhiều loại prompt khác nhau, mỗi loại phục vụ một mục đích cụ thể và ảnh hưởng đến cách AI phản hồi. Một trong những loại phổ biến nhất là prompt tường minh (Explicit Prompt). Đây là dạng prompt cung cấp hướng dẫn rõ ràng và trực tiếp, giúp AI hiểu chính xác yêu cầu và tạo ra kết quả mong muốn. Ví dụ, khi yêu cầu dịch thuật, một prompt tường minh có thể là: “Dịch câu sau sang tiếng Pháp: ‘Hôm nay trời đẹp’.” Với hướng dẫn cụ thể như vậy, AI dễ dàng đưa ra câu trả lời chính xác mà không cần suy đoán thêm.

Khác với prompt tường minh, prompt ngầm định (Implicit Prompt) yêu cầu AI phải suy luận dựa trên ngữ cảnh thay vì được hướng dẫn trực tiếp. Loại prompt này thường đưa ra một câu hỏi mở hoặc một câu lệnh không rõ ràng về định dạng mong muốn, buộc AI phải tự quyết định cách trả lời. Ví dụ, một prompt ngầm định có thể là: “Làm thế nào để nói ‘Hôm nay trời đẹp’ bằng tiếng Pháp?” Trong trường hợp này, AI sẽ không chỉ dịch đơn thuần mà còn có thể diễn giải thêm về cách diễn đạt phù hợp với ngữ cảnh.

Cuối cùng, prompt sáng tạo (Creative Prompt) được thiết kế để khuyến khích AI tạo ra nội dung độc đáo và có tính sáng tạo cao. Thay vì chỉ yêu cầu một câu trả lời chính xác, prompt này thường mang tính chất gợi mở, khuyến khích AI phát triển ý tưởng theo nhiều hướng khác nhau. Ví dụ, một prompt sáng tạo có thể là: “Viết một câu chuyện về một thế giới nơi con người có thể thay đổi thời tiết bằng cảm xúc của họ.” Loại prompt này đặc biệt hữu ích trong lĩnh vực viết lách, sáng tác nội dung, hoặc các ứng dụng đòi hỏi tư duy linh hoạt từ AI.

Bằng cách lựa chọn loại prompt phù hợp với mục tiêu sử dụng, người dùng có thể tối ưu hóa hiệu quả của mô hình AI, tạo ra nội dung có chất lượng cao hơn và phù hợp với nhu cầu sử dụng của người dùng.

2.5.3. Kỹ thuật thiết kế Prompt hiệu quả

Để thiết kế một prompt hiệu quả, điều quan trọng nhất là đảm bảo sự rõ ràng và chính xác trong yêu cầu. Một prompt ngắn gọn, không mơ hồ sẽ giúp AI hiểu đúng mục đích của người dùng, tránh tình trạng tạo ra những phản hồi không mong muốn hoặc không liên quan. Nếu câu hỏi quá chung chung hoặc thiếu thông tin, AI có thể hiểu sai ý định và đưa ra kết quả không chính xác.

Việc cung cấp ngữ cảnh là yếu tố quan trọng giúp AI tạo ra câu trả lời phù hợp hơn. Một prompt có thêm thông tin về bối cảnh, như mục đích sử dụng hay phạm vi chủ đề, sẽ giúp AI hiểu rõ yêu cầu và tạo ra phản hồi có tính chính xác cao hơn. Ví dụ, thay vì chỉ yêu cầu "Viết một đoạn văn về du lịch", có thể cải thiện bằng cách bổ sung thêm chi tiết như "Viết một đoạn văn ngắn về trải nghiệm du lịch tại một thành phố ven biển nổi tiếng."

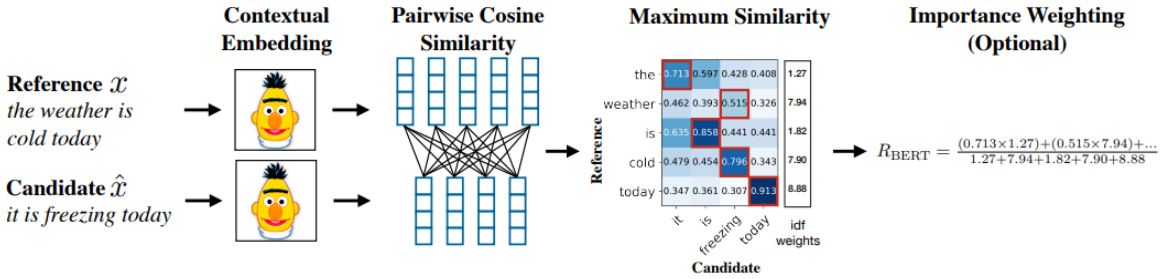
Một kỹ thuật khác giúp tối ưu hóa prompt là chỉ định định dạng mong muốn. Nếu cần AI cung cấp câu trả lời theo dạng danh sách, đoạn văn, bảng biểu hoặc một phong cách viết cụ thể, hãy nêu rõ ngay trong prompt. Điều này giúp hạn chế sự khác biệt giữa kết quả mong muốn và kết quả thực tế mà AI tạo ra. Ví dụ, thay vì hỏi "Kể tên một số địa điểm du lịch nổi tiếng", có thể thay bằng "Liệt kê năm địa điểm du lịch nổi tiếng kèm theo một câu mô tả ngắn cho mỗi địa điểm."

Cuối cùng, thử nghiệm và cải tiến liên tục là một bước không thể thiếu trong quá trình thiết kế prompt. Bởi vì AI có thể phản hồi theo nhiều cách khác nhau, việc kiểm thử nhiều lần sẽ giúp xác định prompt nào hoạt động tốt nhất. Nếu kết quả chưa đạt yêu cầu, người dùng có thể tinh chỉnh câu lệnh, thay đổi cấu trúc hoặc bổ sung chi tiết để hướng dẫn AI theo hướng mong muốn. Quá trình này giúp tối ưu hóa chất lượng đầu ra, đảm bảo mô hình AI hoạt động hiệu quả và tạo ra nội dung phù hợp với mục tiêu sử dụng.

2.6. Tổng quan về phương pháp đánh giá BERTScore

2.6.1. Giới thiệu về BERTScore

BERTScore là một phương pháp đánh giá tự động văn bản được sinh ra bởi các hệ thống xử lý ngôn ngữ tự nhiên (NLP), dựa trên mô hình BERT (Bidirectional Encoder Representations from Transformers). Khác biệt với các kỹ thuật đánh giá truyền thống như BLEU hoặc METEOR, vốn chủ yếu đánh giá dựa trên sự trùng khớp từ ngữ hoặc n-gram, BERTScore tập trung vào việc đo lường độ tương đồng về mặt ngữ nghĩa giữa văn bản tạo ra (candidate) và văn bản tham chiếu (reference) thông qua việc sử dụng các vector biểu diễn ngữ cảnh.



Hình 7: Minh họa các haojt động của BERTScore[6]

2.6.2. Nguyên lý hoạt động

BERTScore thực hiện đánh giá bằng cách tính toán độ tương đồng cosin giữa các embedding ngữ cảnh của từng token trong hai câu đánh giá và tham chiếu. Những embedding này do mô hình BERT cung cấp, phản ánh không chỉ ý nghĩa cục bộ của từ mà còn cả ngữ cảnh tổng thể trong câu. Nhờ khả năng này, BERTScore vượt qua được hạn chế của các phương pháp truyền thống khi không chỉ so sánh bề mặt câu chữ mà còn đánh giá được ý nghĩa thực sự mà câu mang lại.

2.6.3. Ưu điểm của BERTScore

Ưu điểm nổi bật của BERTScore là khả năng hiệu quả trong việc nhận diện và đánh giá các cách diễn đạt khác nhau của cùng một ý nghĩa (paraphrase). Ngoài ra, phương pháp này còn giải quyết được những vấn đề về các phụ thuộc xa và các thay đổi thứ tự từ ngữ có ý nghĩa quan trọng, vốn thường bị bỏ qua hoặc xử lý chưa hiệu quả trong các kỹ thuật đánh giá truyền thống. Các nghiên cứu đã chứng minh rằng BERTScore có mối tương quan cao hơn rõ rệt với các đánh giá từ con người, từ đó cung cấp một phương pháp đánh giá toàn diện và hiệu quả, phản ánh chính xác hơn sự tương đồng ngữ nghĩa giữa văn bản tạo ra và văn bản tham chiếu.

2.6.4. Quy trình đánh giá

-Chuẩn bị dữ liệu: Xây dựng một tập hợp các cặp câu, mỗi cặp gồm câu cần đánh giá và câu tham chiếu tương ứng.

-Biểu diễn bằng mô hình BERT: Chuyển đổi các câu này thành các embedding thông qua mô hình BERT, trong đó mỗi token được biểu diễn bởi một vector ngữ cảnh.

-Tính toán độ tương đồng: Tính toán độ tương đồng cosine giữa embedding của mỗi token trong câu đánh giá với các token trong câu tham chiếu, sử dụng phương pháp ghép cặp tối ưu (greedy matching).

-Tổng hợp kết quả: Tổng hợp các giá trị tương đồng này để tính ra các chỉ số Precision (độ chính xác), Recall (độ phủ), và F1 (điểm tổng hợp), nhằm đưa ra đánh giá cuối cùng về chất lượng của văn bản.

Chương 3. Cài đặt giải pháp

3.1. Môi trường cài đặt

Hệ thống chatbot tư vấn tuyển sinh được xây dựng và triển khai trên nền tảng Node.js, sử dụng ngôn ngữ lập trình JavaScript. Việc lựa chọn Node.js giúp tối ưu hóa khả năng xử lý các yêu cầu bất đồng bộ, phù hợp với các ứng dụng chatbot đòi hỏi thời gian phản hồi nhanh và khả năng mở rộng dễ dàng.

Để tạo ra một chatbot tư vấn tuyển sinh linh hoạt, hiệu quả và dễ quản lý, một số công nghệ và thư viện đã được sử dụng bao gồm:

OpenAI: Được tích hợp thông qua thư viện chính thức của OpenAI để sử dụng mô hình GPT-4o-mini, cung cấp khả năng sinh ra các câu trả lời tự nhiên dựa trên thông tin truy xuất từ cơ sở dữ liệu tuyển sinh.

LangChain và FAISS: LangChain được sử dụng để tạo ra vector embeddings thông qua API của OpenAI, cụ thể là mô hình text-embedding-3-small. Các vector embeddings này được lưu trữ và quản lý bằng FAISS (Facebook AI Similarity Search), một cơ sở dữ liệu vector mạnh mẽ giúp truy xuất thông tin nhanh và chính xác.

Express.js: Framework Express.js được dùng để xây dựng và quản lý các API kết nối chatbot với nền tảng Facebook Messenger. Các API này cho phép chatbot nhận, xử lý, và phản hồi các yêu cầu từ người dùng thông qua webhook.

request: Thư viện này được sử dụng để gửi các HTTP request từ chatbot đến API của Facebook Messenger, cho phép chatbot nhận và trả lời tin nhắn trực tiếp qua nền tảng Messenger.

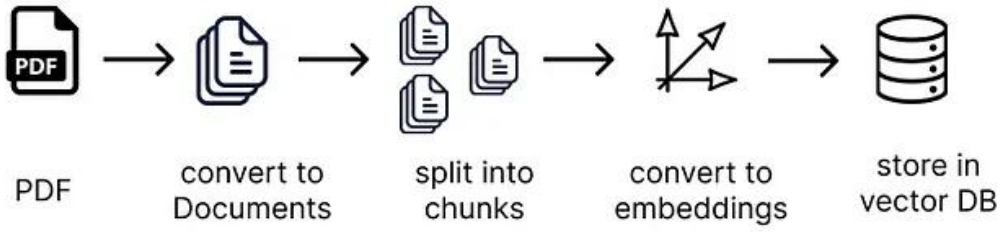
Hệ thống chatbot được triển khai thử nghiệm trên môi trường máy tính cá nhân trước khi được triển khai chính thức lên nền tảng server của Render, một dịch vụ lưu trữ và triển khai ứng dụng web dựa trên cloud nhằm đảm bảo hiệu suất ổn định và khả năng xử lý nhiều yêu cầu đồng thời một cách nhanh chóng và hiệu quả.

3.2. Cài đặt và triển khai hệ thống:

3.2.1. Thu thập và xử lý dữ liệu

3.2.1.1. Thu thập dữ liệu

Để đảm bảo độ chính xác và tính hữu ích của chatbot, dữ liệu cần được thu thập từ nhiều nguồn đáng tin cậy của trường đăng lên. Các nguồn này bao gồm cổng thông tin chính thức của trường trên các phương tiện chuyền thông, các tài liệu liên quan như quy chế tuyển sinh, đề án tuyển sinh, điểm chuẩn. Trong niên luận này tôi sẽ sử dụng dữ liệu tuyển sinh của Trường Đại học Cần Thơ làm dữ liệu cho Chatbot.



Hình 8: Quy trình chuyển dữ liệu thô thành vector lưu trong database[7]

3.2.1.2. Tiền xử lý và làm sạch dữ liệu

Nhằm chuẩn hóa và cải thiện chất lượng các tài liệu cần được chuyển đổi sang định dạng văn bản thuần (plain text) để đảm bảo tính nhất quán. Các yếu tố không liên quan như chữ ký email, biểu đồ phức tạp, hình ảnh và quảng cáo cũng cần được loại bỏ. Ngoài ra, việc xử lý lỗi chính tả, điều chỉnh dấu câu và loại bỏ khoảng trắng thừa giúp nâng cao chất lượng dữ liệu. Một bước quan trọng khác trong quá trình này là phân đoạn dữ liệu (chunking), trong đó nội dung được chia thành các đoạn nhỏ từ 100-300 từ hoặc theo ngữ cảnh logic đồng thời bổ sung ngữ cảnh cho các đoạn thiếu ngữ cảnh. Mỗi đoạn được gán nhãn hoặc tiêu đề để hỗ trợ quá trình truy xuất thông tin một cách hiệu quả.

3.2.2. Xây dựng cơ sở dữ liệu vector từ dữ liệu thô

3.2.2.1. Vector hóa dữ liệu bằng text-embedding-3-small

Các đoạn văn đã xử lý ở bước trước được chuyển đổi thành vector embeddings thông qua API của OpenAI sử dụng mô hình text-embedding-3-small. Mỗi đoạn văn bản được chuyển thành một vector 1536 chiều, giúp biểu diễn đầy đủ ý nghĩa ngữ nghĩa của thông tin tuyển sinh.

3.2.2.2. Lưu trữ dữ liệu bằng FAISS

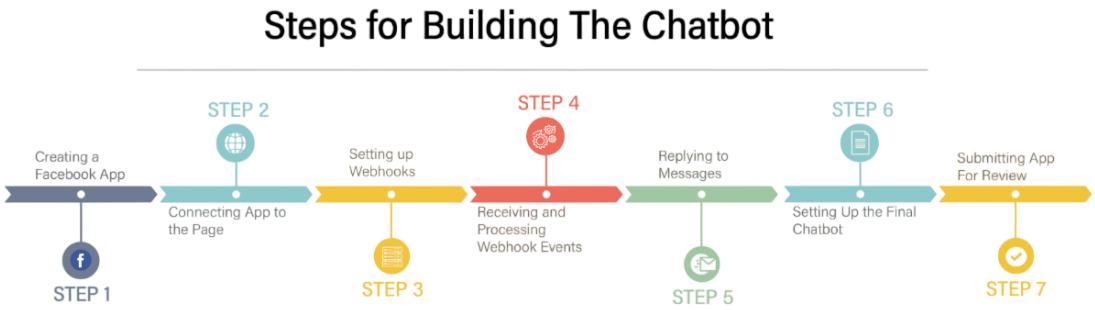
Sau khi tạo embeddings, các vector này được lưu trữ trong cơ sở dữ liệu vector FAISS (Facebook AI Similarity Search). FAISS sử dụng kỹ thuật Inverted File Index (IVF) kết hợp với thuật toán Approximate Nearest Neighbor (ANN) để tăng tốc độ tìm kiếm và truy xuất thông tin. Điều này giúp chatbot nhanh chóng xác định các đoạn văn bản liên quan nhất với truy vấn của người dùng, ngay cả khi số lượng vector tăng lên rất lớn.

3.2.3. Xây dựng hệ thống chatbot sử dụng Retrieval-Augmented Generation (RAG)

Hệ thống chatbot tư vấn tuyển sinh được xây dựng dựa trên kiến trúc Retrieval-Augmented Generation (RAG), tích hợp khả năng truy xuất thông tin từ cơ sở dữ liệu và sinh phản hồi tự nhiên. Khi người dùng đặt câu hỏi, chatbot sẽ chuyển đổi câu hỏi

đó thành vector embedding bằng mô hình text-embedding-3-small. Sau đó, hệ thống so sánh vector embedding này với các vector đã lưu trữ trong cơ sở dữ liệu FAISS nhằm xác định những đoạn văn bản phù hợp nhất liên quan đến nội dung câu hỏi. Những đoạn văn này được tổng hợp và đưa vào một prompt hướng dẫn rõ ràng sau đó sử dụng API để mô hình GPT-4o-mini tạo ra phản hồi phù hợp, đảm bảo tính chính xác và sát với yêu cầu của người dùng.

Triển khai chatbot lên server và tích hợp vào Facebook Messenger, việc thích hợp vào Facebook Messenger sẽ thực hiện theo các bước như bên dưới và mội hướng dẫn tích hợp điều được Facebook Developer hướng dẫn chi tiết [10], sau quá trình kết nối giữa server và Facebook thì đã thành công tích hợp Chatbot vào Facebook Messenger.



Hình 9: Các bước triển khai Chatbot lên Facebook Messenger[8]

Hệ thống chatbot được triển khai thông qua API sử dụng Express.js và được host trên Render để đảm bảo khả năng mở rộng và hoạt động ổn định. API thực hiện nhiệm vụ nhận truy vấn từ người dùng, tìm kiếm thông tin trong FAISS, sau đó sinh phản hồi thông qua mô hình OpenAI GPT-4o-mini.

Luồng xử lý chatbot trên Render:

- Messenger Webhook nhận sự kiện và gửi dữ liệu đến server chatbot trên Render.

- Hệ thống trích xuất câu hỏi từ tin nhắn và chuyển thành vector embeddings.

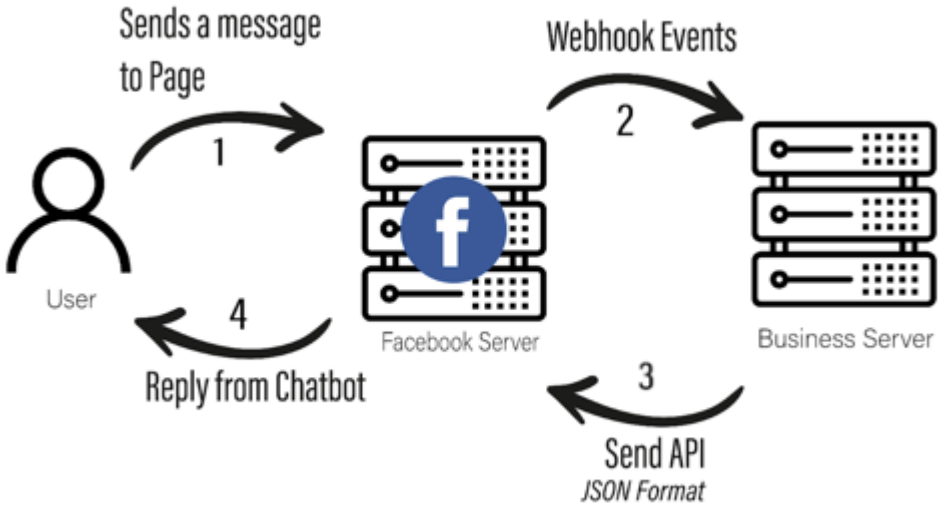
- FAISS thực hiện tìm kiếm trong kho dữ liệu để truy xuất các đoạn văn bản liên quan.

- Mô hình GPT-4o-mini của OpenAI tổng hợp thông tin và tạo câu trả lời dựa trên nội dung đã truy xuất.

- Hệ thống gửi phản hồi trở lại Facebook Messenger, hiển thị nội dung tư vấn cho người dùng.

Server được triển khai trên Render, đảm bảo chatbot hoạt động 24/7 mà không cần quản lý thủ công máy chủ. Render cung cấp tính năng tự động scale khi có nhiều người dùng đồng thời, giúp chatbot duy trì tốc độ phản hồi nhanh mà không bị gián

đoạn. Tích hợp chatbot vào Messenger để nhận tin nhắn từ người dùng gửi đến server triển khai chatbot. Và xây dựng giao diện Messenger nâng cao trải nghiệm người dùng



Hình 10: Quy trình khi nhận tin của người dùng đến khi phản hồi[8]

Quy trình hoạt động:

Người dùng gửi một tin nhắn tới trang Facebook Messenger.

Facebook Server tiếp nhận tin nhắn này và gửi thông tin dưới dạng sự kiện Webhook đến Business Server (máy chủ ứng dụng chatbot).

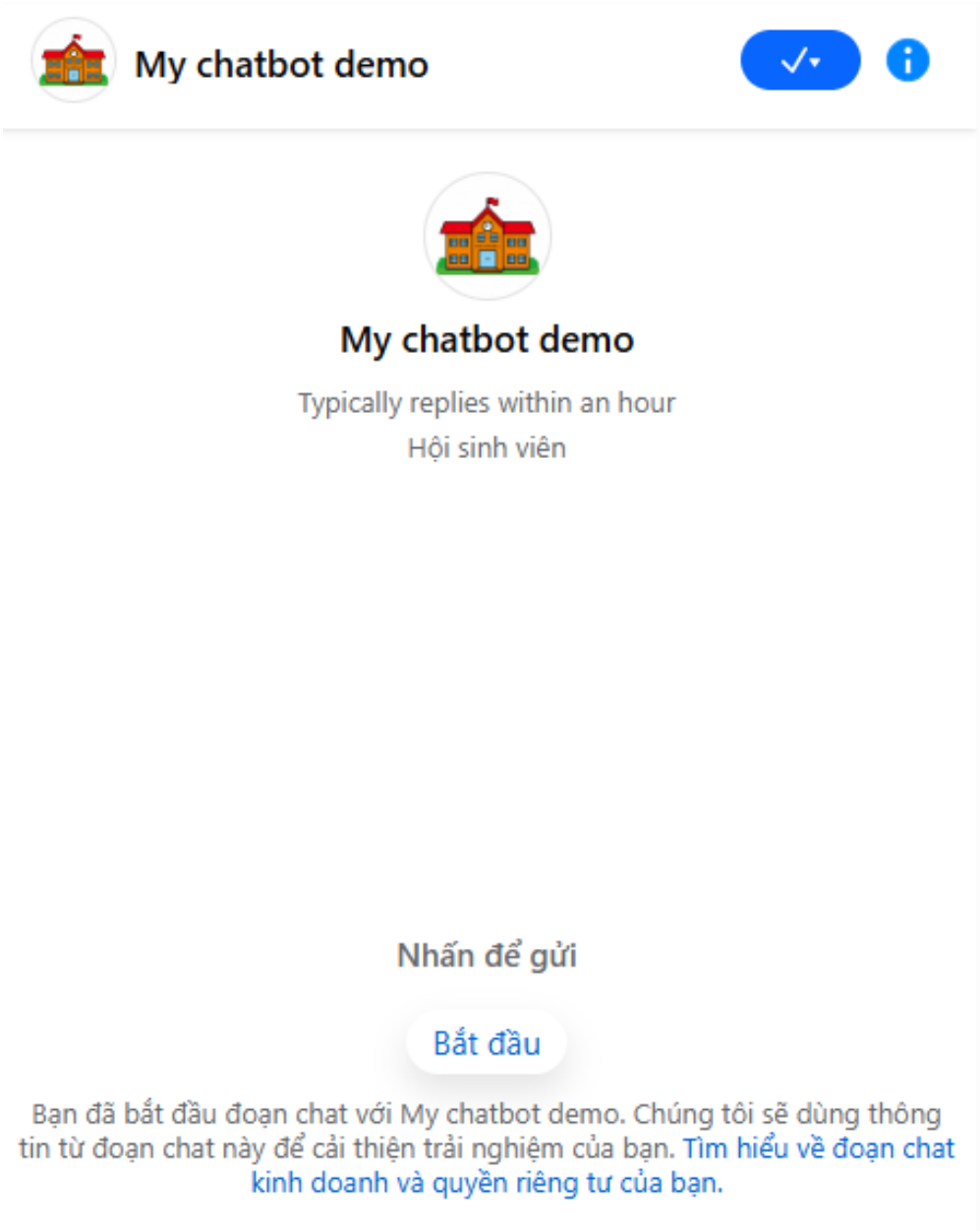
Business Server xử lý sự kiện webhook này, phân tích cú pháp JSON, xử lý dữ liệu cho câu hỏi, trích xuất nội dung và truy vấn thông tin trong FAISS và Mô hình GPT-4o-mini tạo phản hồi dựa trên dữ liệu đã truy xuất.

Business Server gửi phản hồi trở lại Facebook Messenger thông qua API dưới định dạng JSON.

Facebook Server hiển thị phản hồi từ chatbot cho người dùng trong giao diện Messenger.

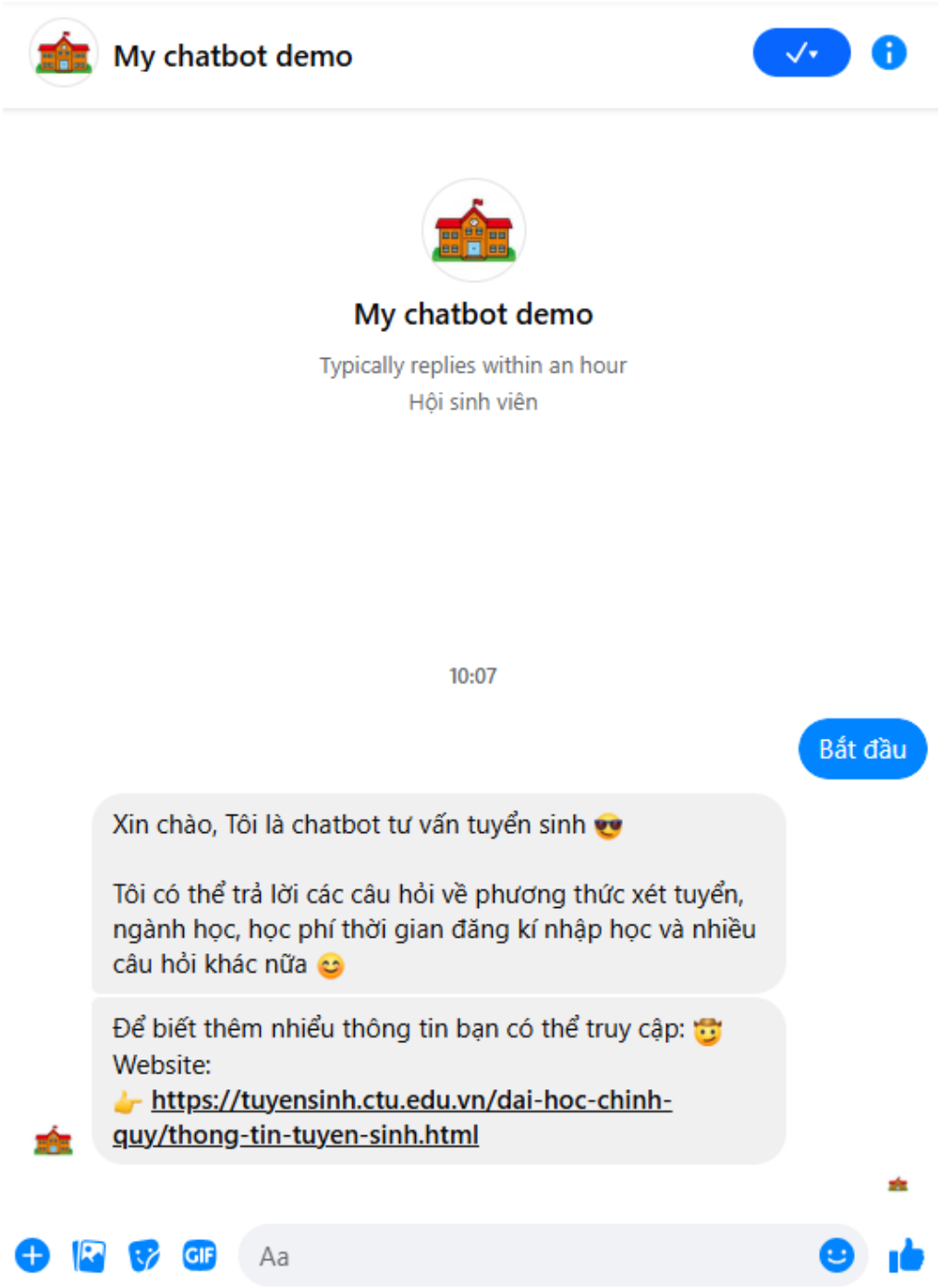
3.3. Kết quả thực hiện

Giao diện của Chatbot sau khi được tích hợp vào Facebook Messenger:



Hình 11: Giao diện messenger ở phía người dùng

Giao diện Messenger của chatbot rất đơn giản chỉ gồm một nút "Bắt đầu" khi người dùng nhấn vào sự kiện cho nút bắt đầu sẽ được kích hoạt.



Hình 12: Phản hồi của Chatbot khi người dùng nhấn nút "Bắt đầu"

Sau khi người dùng nhấn nút bắt đầu Chatbot được thiết lập mật định sẽ trả lời thông tin như trên để giới thiệu về bản thân chatbot và giúp người dùng nhận biết được mình đang tham gia cuộc trò chuyện với chatbot. Đồng thời trong phản hồi này cũng hướng dẫn cách người dùng có thể giao tiếp với chatbot hiệu quả. Sau đó người dùng có thể bắt đầu hỏi các thông tin tuyển sinh với Chatbot.

Chương 4. Đánh giá kiểm thử

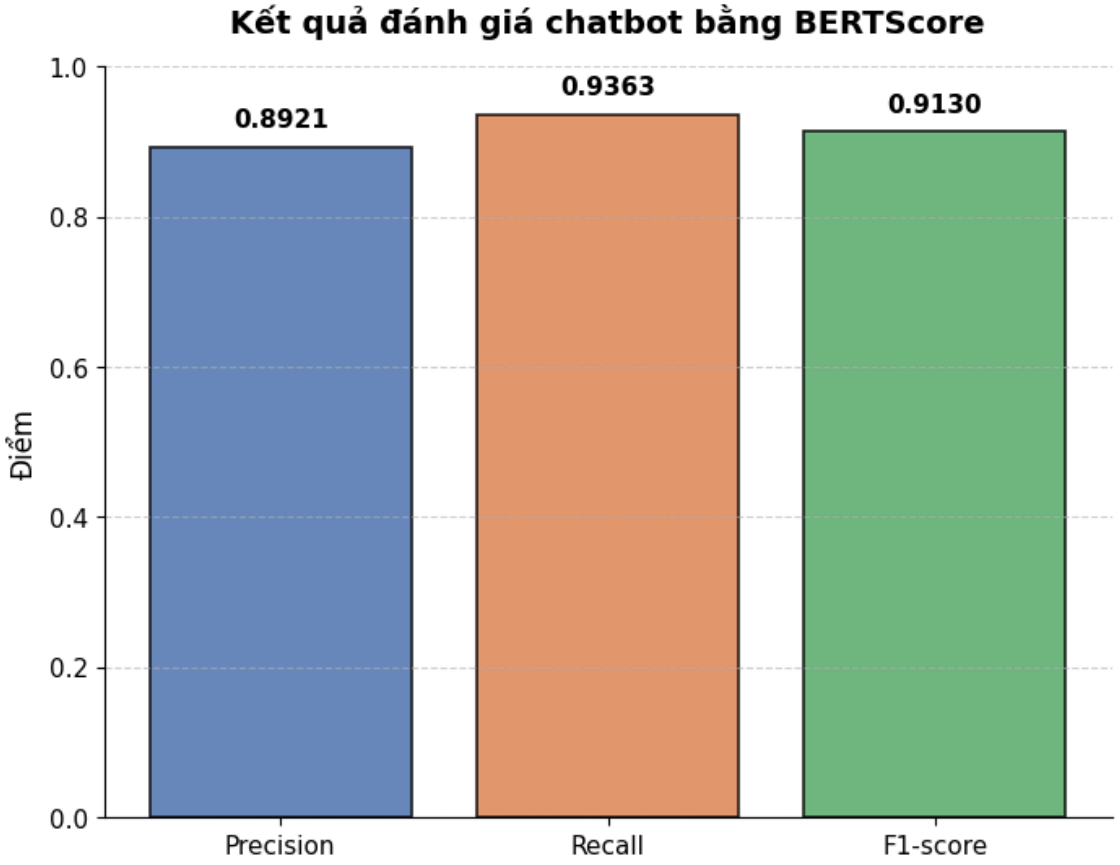
4.1. Kết quả đạt được đánh giá qua BERTScore

Để chuẩn bị cho bước kiểm thử này thì hơn 350 câu hỏi liên quan đến các ngành học, điểm chuẩn, khối xét tuyển và các quy định về phương thức xét tuyển đã được lập ra, sau đó dựa trên dữ liệu quy chế tuyển sinh của trường để tao ra câu trả lời chính xác cho từng câu hỏi để tạo thành bộ dữ liệu tham chiếu chất lượng cao. Kết quả của đánh giá này chỉ cho thấy được độ chính xác trong câu trả lời của Chatbot. Quy trình đánh giá bằng BERTScore

- Thu thập phản hồi từ chatbot: Mỗi câu hỏi trong tập dữ liệu kiểm thử được nhập vào chatbot để thu thập câu trả lời mà chatbot cung cấp.

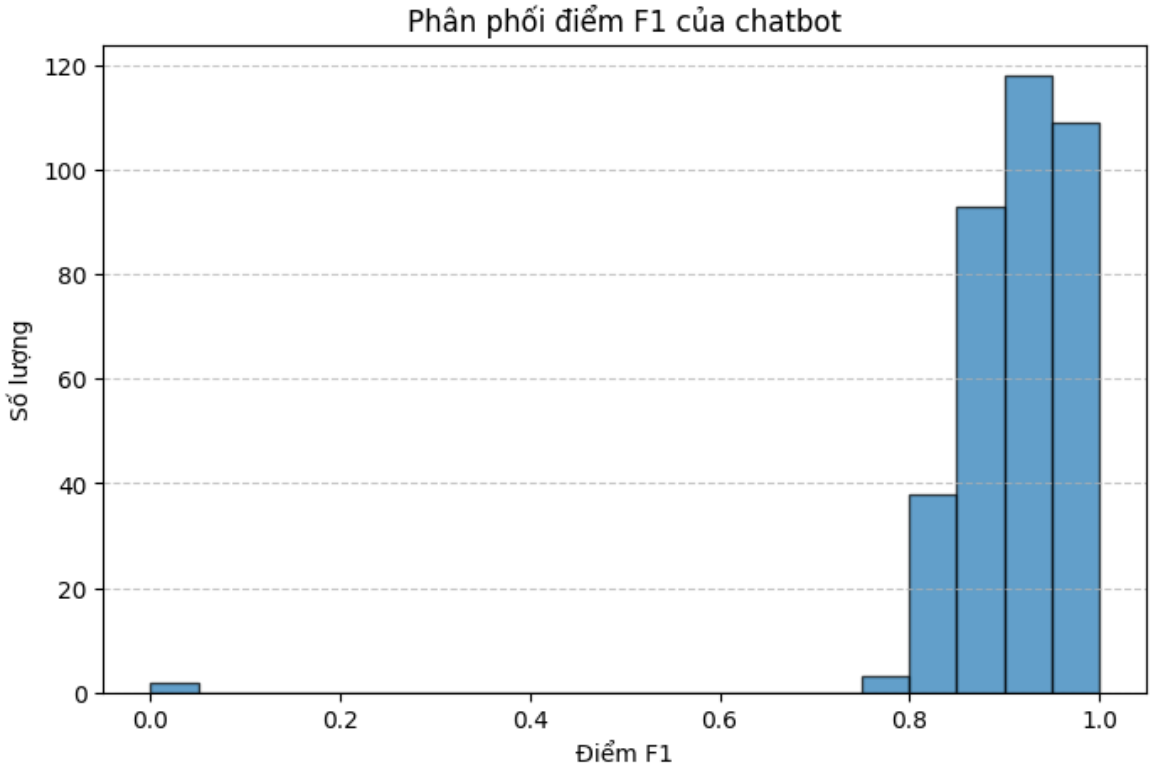
- Tính toán BERTScore: Sử dụng mô hình BERT, chuyển đổi cả hai câu trả lời của chatbot và câu trả lời tham chiếu thành các vector ngữ cảnh. Sau đó, độ tương đồng cosine giữa các vector này được tính toán để xác định mức độ tương đồng ngữ nghĩa giữa hai câu trả lời.

- Phân tích kết quả: Các chỉ số Precision, Recall và F1-score được tính dựa trên cácgiá trị tương đồng này, cung cấp đánh giá chi tiết về hiệu suất của chatbot.



Hình 13: Kết quả đánh giá phản hồi của Chatbot bằng BERTScore

Sau quá trình thử nghiệm và đánh giá chatbot bằng BERTScore, tôi thu được những kết quả đáng chú ý thể hiện qua các chỉ số Precision, Recall và F1-score. Cụ thể, chatbot đạt điểm Precision là **0.8921**, Recall là **0.9363**, và F1-score là **0.9130**. Những con số này cho thấy chatbot có hiệu suất rất cao trong việc tạo ra các phản hồi có ý nghĩa tương đồng với văn bản tham chiếu. Đặc biệt, chỉ số Recall cao hơn một chút so với Precision, điều này cho thấy chatbot có xu hướng tạo ra các phản hồi phong phú, cung cấp nhiều thông tin hơn so với văn bản gốc.

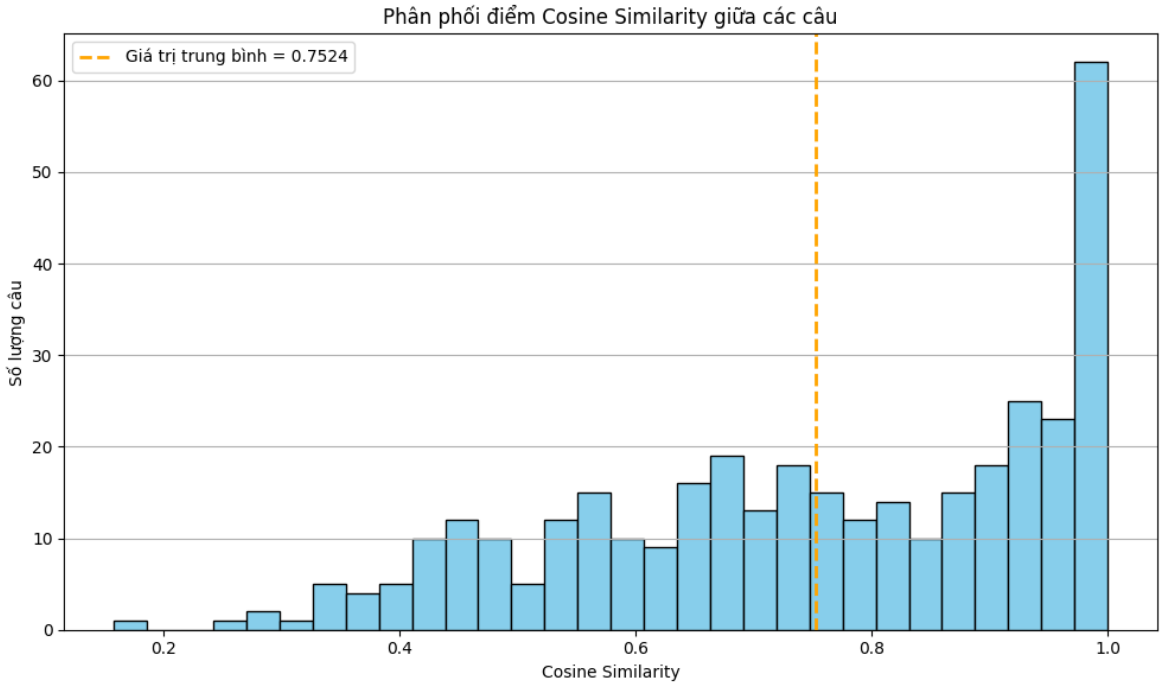


Hình 14: Biểu đồ phân phối của điểm F1

Ngoài ra, để có cái nhìn sâu hơn về chất lượng phản hồi của chatbot, tôi đã phân tích phân phối điểm F1 trên tập dữ liệu thử nghiệm. Biểu đồ phân phối cho thấy phần lớn các câu trả lời đạt điểm cao, đặc biệt tập trung ở mức 0.8 đến 1.0, chứng tỏ chatbot có khả năng tạo ra phản hồi rất sát với văn bản tham chiếu. Tuy nhiên, vẫn có một số câu có điểm F1 thấp hơn, chủ yếu rơi vào khoảng dưới 0.8, phản ánh những trường hợp mà chatbot có thể đã tạo ra các câu trả lời khác biệt đáng kể so với câu tham chiếu.

Nhìn chung, từ kết quả thu được, có thể khẳng định rằng chatbot tư vấn tuyển sinh này có khả năng hỗ trợ tư vấn hiệu quả, đáp ứng tốt các yêu cầu thông tin của người dùng về tuyển sinh, ngành học và các vấn đề liên quan. Về hiệu suất hiện tại Chatbot phụ thuộc nhiều vào API OpenAI và server triển khi Chatbot nên các thông số không được đánh giá chi tiết nhưng trong quá trình kiểm thử cho thầy Chatbot có thể phản hồi gần 500 câu hỏi trong 10 phút.

Bên cạnh việc sử dụng BERTScore, tôi cũng thực hiện đánh giá mức độ tương đồng ngữ nghĩa giữa phản hồi của chatbot và câu trả lời tham chiếu bằng phương pháp Cosine Similarity, sử dụng mô hình distiluse-base-multilingual-cased-v1 từ thư viện SentenceTransformers. Mỗi câu trả lời từ chatbot và câu tham chiếu được mã hóa thành vector ngữ nghĩa, sau đó tính điểm cosine giữa các cặp vector tương ứng để đo lường độ tương đồng của cả câu tham chiếu và câu phản hồi từ Chatbot.



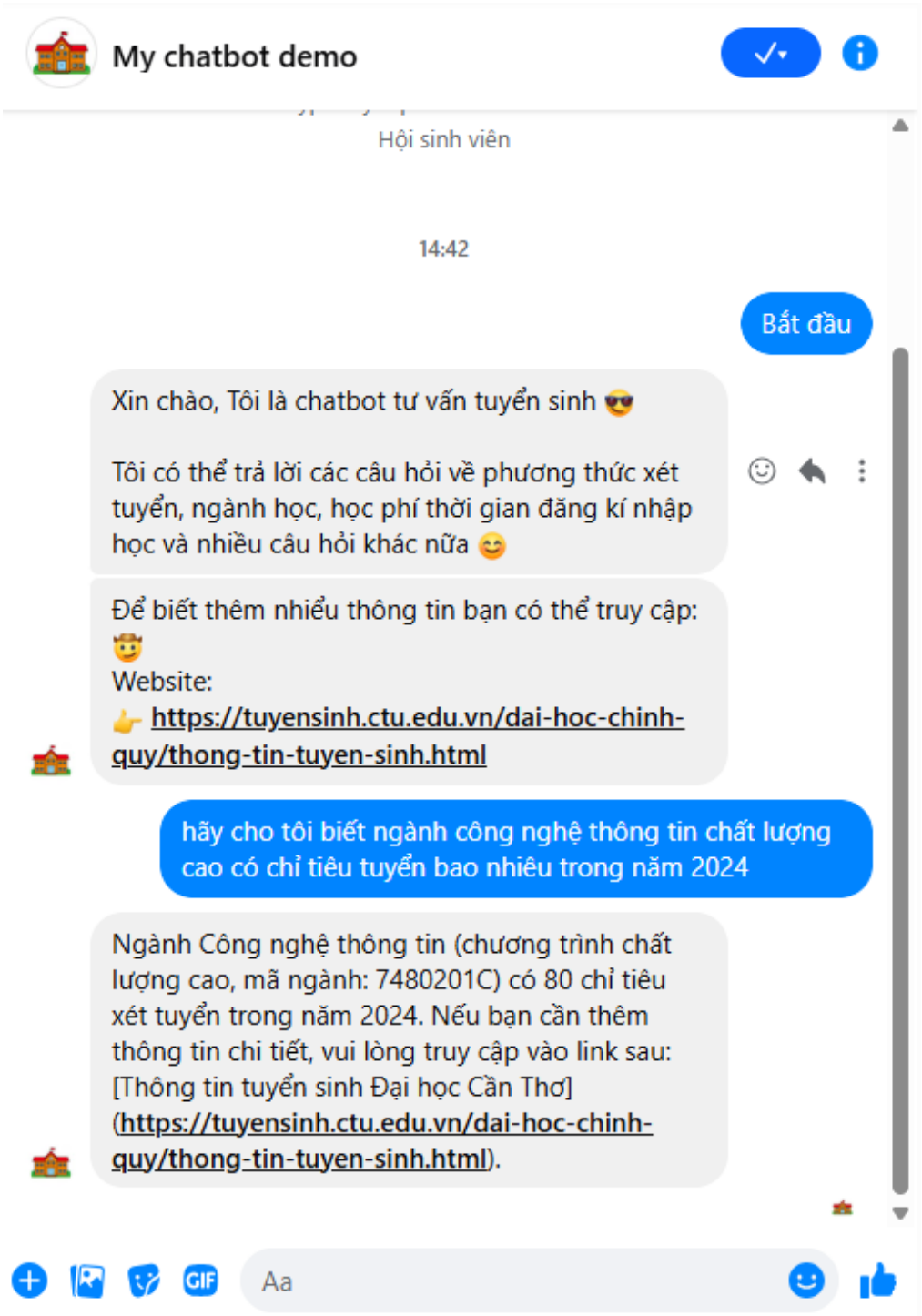
Hình 15: Phân phối điểm Cosine Similarity trên toàn bộ tập dữ liệu

Kết quả được thể hiện trong Hình 15, là biểu đồ phân phối các điểm Cosine Similarity trên toàn bộ tập câu hỏi thử nghiệm. Biểu đồ cho thấy phần lớn các câu trả lời của chatbot đạt điểm cosine cao, đặc biệt tập trung trong khoảng 0.7 đến gần 1.0, cho thấy mức độ tương đồng rất cao về mặt ngữ nghĩa giữa phản hồi của chatbot và dữ liệu tham chiếu.

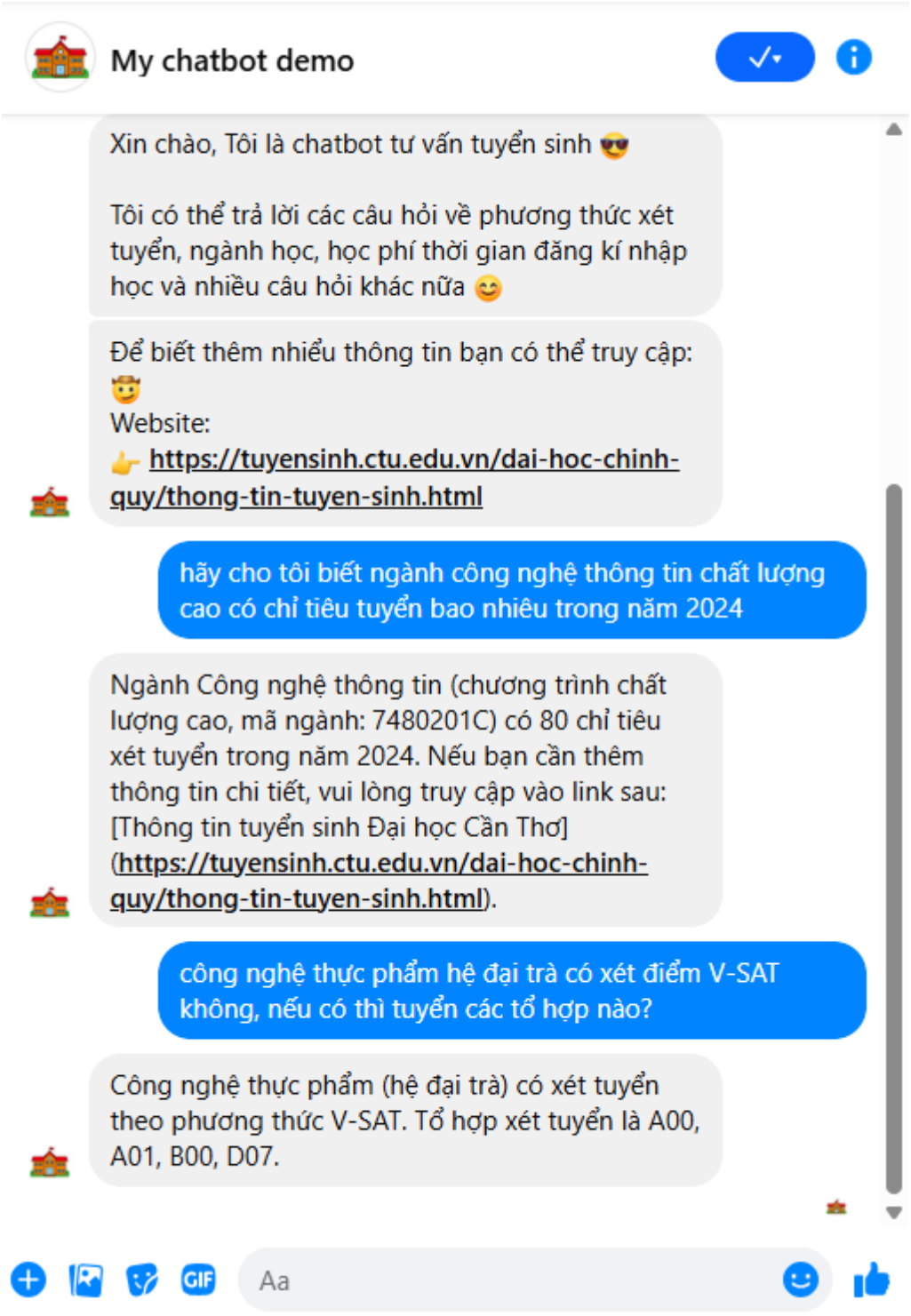
Một thông tin quan trọng khác là giá trị trung bình của toàn bộ tập điểm Cosine Similarity là 0.75. Điều này phản ánh rằng đa số phản hồi của chatbot không chỉ đúng về mặt nội dung mà còn thể hiện sự gần gũi về cách diễn đạt so với các câu trả lời mẫu. Tuy vậy, vẫn tồn tại một tỷ lệ nhỏ các câu có điểm thấp hơn (dưới 0.6), cho thấy trong một số trường hợp nhất định, chatbot có thể tạo ra phản hồi chưa thực sự sát với mong đợi.

4.2. Kết quả phản hồi trên Facebook Messenger

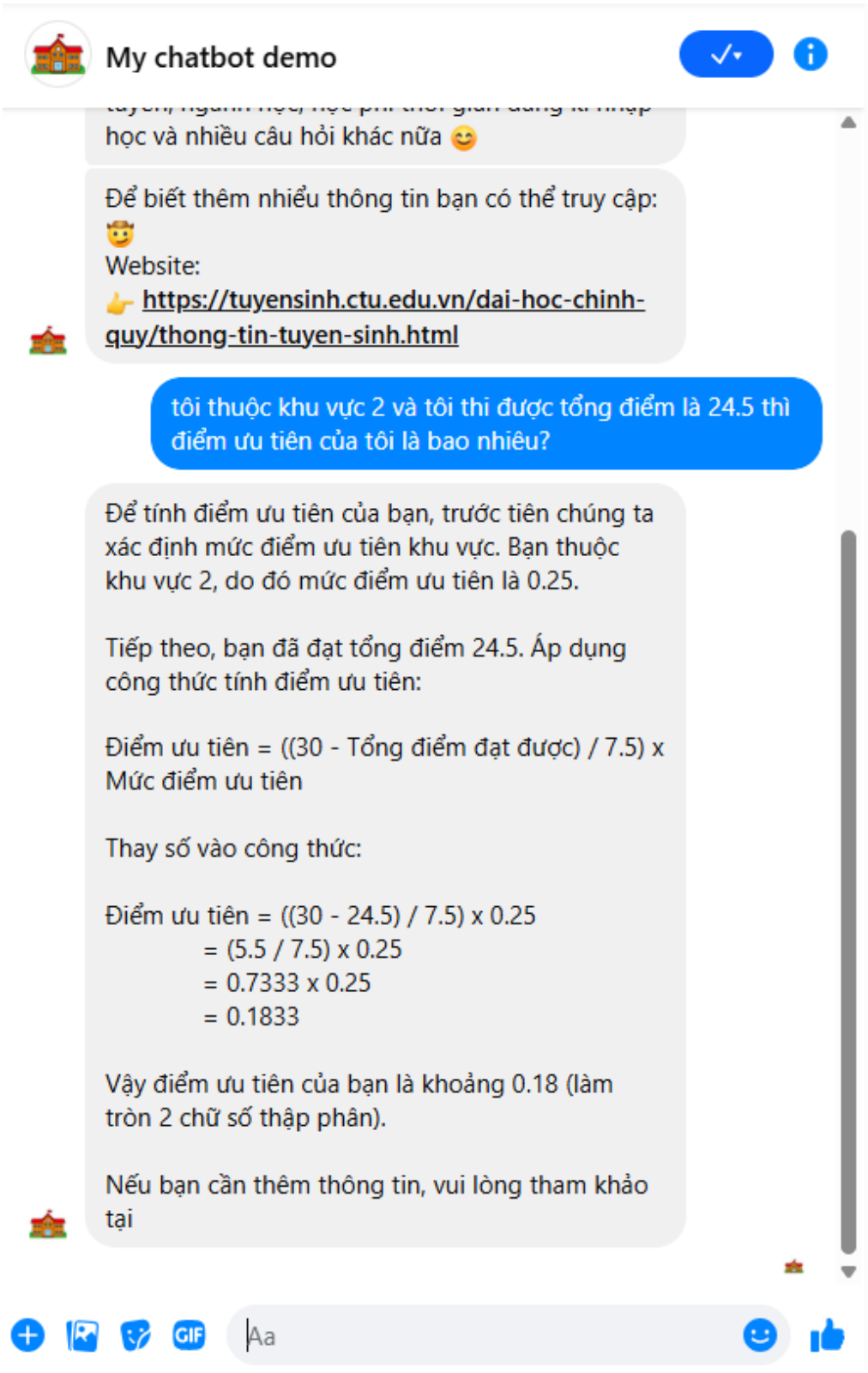
Sau khi được triển khai lên Facebook Messenger thì đây là phản hồi người dùng nhận được khi nhắn tin với chatbot:



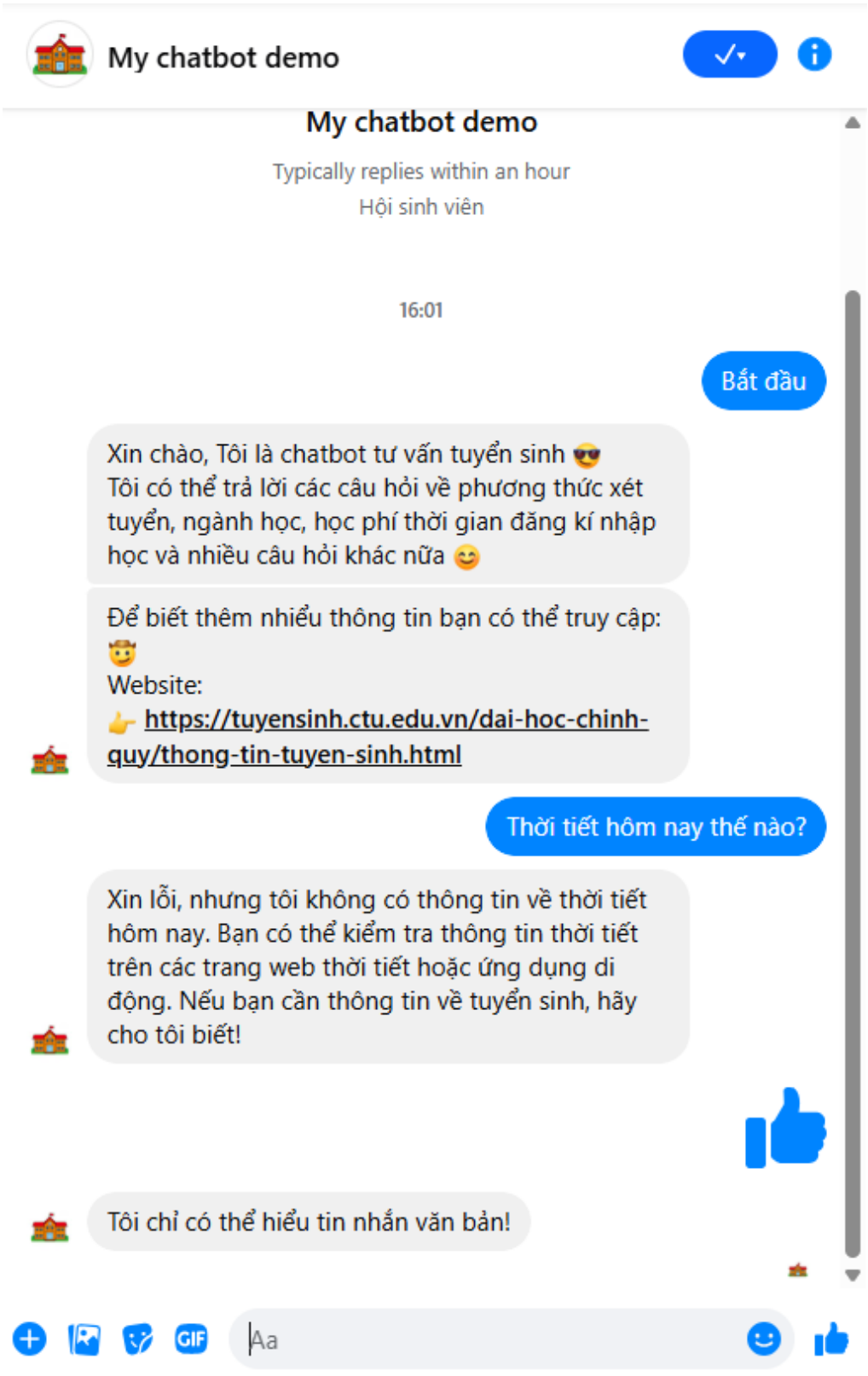
Hình 16: Chatbot phản hồi câu hỏi chỉ tiêu ngành Công nghệ thông tin



Hình 17: Chatbot phản hồi về câu hỏi phương thức và tổ hợp xét tuyển



Hình 18: Chatbot vận dụng công thức để trả lời câu hỏi



Hình 19: Chatbot phản hồi các câu hỏi không liên quan

KẾT LUẬN

1. Kết quả đạt được

Trong quá trình thực hiện niên luận xây dựng hệ thống chatbot tư vấn tuyển sinh đã đạt được một số kết quả, tích hợp mô hình Retrieval-Augmented Generation (RAG) để tối ưu hóa khả năng truy xuất và tạo sinh phản hồi dựa trên tài liệu tuyển sinh chính thức. Hệ thống đã triển khai trên nền tảng Facebook Messenger và có thể phản hồi các câu hỏi một cách chính xác. Một số kết quả quan trọng đạt là Chatbot có khả năng hiểu và trả lời các câu hỏi tuyển sinh một cách linh hoạt, với mức độ chính xác cao dựa trên dữ liệu được lưu trữ trong cơ sở dữ liệu vector. Hệ thống sử dụng mô hình nhúng văn bản (embedding) để cải thiện khả năng tìm kiếm và truy xuất thông tin theo ngữ nghĩa, giúp chatbot phản hồi sát với yêu cầu của người dùng. Đặt biệt hơn hết là khả năng mở rộng và cập nhật dữ liệu linh hoạt: Cơ sở dữ liệu vector cho phép hệ thống cập nhật thông tin nhanh chóng mà không cần huấn luyện lại mô hình, giúp chatbot luôn cung cấp nội dung chính xác và mới nhất.

Mặc dù đạt được những kết quả tích cực, chatbot vẫn còn một số hạn chế cần khắc phục trong các giai đoạn phát triển tiếp theo.

2. Hướng phát triển

Để cải thiện chất lượng và mở rộng phạm vi ứng dụng của chatbot, một số hướng phát triển tiềm năng như nâng cao khả năng xử lý các câu hỏi phức tạp, hiện tại chatbot hoạt động tốt với các câu hỏi trực tiếp về tuyển sinh, nhưng vẫn cần cải thiện để xử lý tốt hơn các câu hỏi mang tính suy luận hoặc yêu cầu kết hợp nhiều nguồn dữ liệu. Mở rộng cơ sở dữ liệu và cải thiện cách đánh chỉ mục (indexing), hiện tại chatbot chỉ hỗ trợ dữ liệu tuyển sinh của một năm tuy nhiên nó vẫn có thể được mở rộng để lưu trữ và truy xuất thông tin từ nhiều năm tuyển sinh khác nhau, giúp người dùng có thể so sánh, tra cứu thông tin từ nhiều giai đoạn khác nhau. Đồng thời, việc áp dụng các phương pháp đánh index tiên tiến hơn sẽ giúp cải thiện tốc độ truy xuất dữ liệu, nâng cao hiệu suất của chatbot. Hiện tại Chatbot đang sử dụng API từ OpenAI để sinh câu hỏi, nếu có hệ thống server đủ mạnh có thể thay thể bằng 1 mô hình LLM mã nguồn mở khác để sinh câu trả lời không cần thông qua dịch vụ bên thứ ba.

Tóm lại, niên luận này đã đạt được mục tiêu ban đầu là xây dựng một chatbot tư vấn tuyển sinh hoạt động hiệu quả. Tuy nhiên, vẫn còn nhiều tiềm năng để nâng cấp và phát triển hệ thống trong tương lai, hướng tới việc tạo ra một nền tảng tư vấn thông minh và tối ưu hóa hơn không chỉ riêng cho tư vấn tuyển sinh mà còn các hệ thông tư vấn khác.

TÀI LIỆU THAM KHẢO

[1] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang, "Retrieval-Augmented Generation,"*Retrieval-Augmented Generation for Large Language Models: A Survey*, 2024.

[2] Pinecone, "Vector Embeddings," Pinecone Learn, 2023. [Online]

[3] MongoDB, "Vector Databases," MongoDB Resources, 2024. [Online].

[4] Shangyu Wu, Ying Xiong, Yufei Cui, Haolun Wu, Can Chen, Ye Yuan, Lianming Huang, Xue Liu, Tei-Wei Kuo, Nan Guan, and Chun Jason Xue, "Retrieval-Augmented Generation for Natural Language Processing: A Survey," 2025.

[5] Pinecone, "Vector Database," Pinecone Learn, 2024. [Online].

[6] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "BERTScore: Evaluating Text Generation with BERT," ICLR Conference Paper, 2020.

[7] Better Programming, "Building a Multi-Document Reader and Chatbot with LangChain and ChatGPT," Medium, 2023. [Online].

[8] Raiyan24r, "Messenger Chatbot Tutorial," GitHub Repository, 2020. [Online].

[9] HaryPhamDev, "Facebook Messenger Restaurant Chat Bot," GitHub Repository, 2020. [Online].

[10] M. Mayo, Mastering Generative AI and Prompt Engineering: A Practical Guide for Data Scientists, Data Science Horizons, 2023.

[11] Facebook, "Messenger Platform Quick Start," Facebook for Developers, 2025. [Online].