

Carolinian Chatbot: End-to-End Question-Answer Pipeline on the University of San Carlos Policies, Procedures, and Guidelines Via Retrieval Augmented Generation

Wayne Matthew A. Dayata

*Department of Computer, Information Sciences, and Mathematics
University of San Carlos
Cebu City, Philippines
wdayata@gmail.com*

Sabrinah Yonell C. Yap

*Department of Computer, Information Sciences, and Mathematics
University of San Carlos
Cebu City, Philippines
sabrinah.yonell@gmail.com*

Abstract—The recent advancements in document embedding and context-based inference models have fueled the development and deployment of question-answering (QA) pipelines. This paper addresses a common challenge faced by universities: student inquiries regarding policies, procedures, and guidelines that often overwhelm staff and lead to delays. To address this, we present a novel End-to-End QA application employing the Retrieval-Augmented Generation (RAG) strategy. Our system leverages the University of San Carlos Student Manual 2023 and supplementary documents, along with the Nomic-Embed-Text-v1.5 embedding model and Meta's powerful LLaMA-3 inference model. Deployed on Streamlit Cloud, the application allows students to access initial responses to their questions. User interactions and retrieved data indicate reliable performance in both document retrieval and inference phases, particularly for queries spanning multiple documents. These results demonstrate the application's potential to handle common student inquiries, thereby reducing workload on university staff and personnel.

Index Terms—Natural Language Processing, Question Answering, Retrieval Augmented Generation, Document Embedding, Vector Search, Large Language Models

I. INTRODUCTION

This section of this paper discusses the background of the study and describes the current context, problems, and relevance that led to the motivation for developing this project.

A. Rationale

People have been frequently experiencing information overload in the digital age, presenting challenges across various domains. In the context of higher education, college students often struggle to navigate the vast amount of text-based information related to their academic journey. University handbooks, enrollment guidelines, and office directories can be overwhelming, especially for students navigating a large campus with numerous departments and resources. Locating the right information quickly and efficiently becomes crucial for completing tasks like enrollment or acquiring specific university services. With the recent increase in student population, officers of student organizations as well as staff from the

offices are also increasingly overwhelmed with the influx of student queries which most of the time were not addressed to the right personnel. The students' lack of adequate knowledge about their university's guidelines and rules results in the lack of timely access to and guidance from their office of concern and the underutilization of the student support services offices and facilities. These facts and issues call for a more efficient approach to information retrieval for students to receive accurate and timely information about their school concerns.

Fortunately, advancements in question-answering (QA) and Natural Language Processing (NLP) offer promising solutions to this information overload problem [1]. QA systems, powered by NLP techniques, can function as intelligent virtual assistants [2]. These systems, trained to understand natural language queries, can retrieve relevant information from college handbooks and university websites. Imagine a chatbot system accessible to students 24/7, capable of answering questions like "What are the requirements for enrollment?" or "Where can I find the scholarship office?" Such a system would encourage students to navigate university procedures independently.

NLP research has blossomed into a dynamic field, constantly pushing the boundaries of human-computer interaction. Beyond question answering, NLP finds applications in various real-world scenarios such as sentiment analysis of customer review data [2]. Machine translation tools, powered by NLP, break down language barriers and foster global communication. Automated document summarization streamlines information processing for researchers and professionals alike.

Looking ahead, the integration of NLP into chatbots and virtual assistants will continue to transform how we interact with technology. As NLP models like LLAMA 3 demonstrate, the ability to process and understand natural language with ever-increasing sophistication opens doors for a future where technology seamlessly integrates into our daily lives, offering intelligent support and guidance [3].

B. Objectives and Scope

With the increased potential and relevance of NLP solutions in real-world contexts, the authors developed the Carolinian Chatbot to generate accurate and sound responses to questions within the context of the University of San Carlos (USC) Student Manual and its supplementary documents. The specific objectives are as follows:

- 1) Preprocess and analyze the relevant texts as inputs for the pipeline.
- 2) Visualize the retrieval of relevant documents for the common questions.
- 3) Build the question-answering chain with the LLaMA-3 model and its accompanying application.
- 4) Deploy the application online with Streamlit.
- 5) Evaluate the overall pipeline.

The development and use of the chatbot are subject to the following conditions and restrictions:

- 1) Only open-source embedding and large language models will be used.
- 2) As this study focuses only on Retrieval Augmented Generation (RAG), no fine-tuning will be performed.
- 3) The answers generated from the pipeline will be strictly within the given context, thus providing info updated only as of the time the texts are retrieved for the model.
- 4) The chatbot cannot be used beyond the scope of the USC related contexts.

II. TECHNICAL BACKGROUND

This section elaborates on the fundamental concepts and technical terms applied throughout this study. The descriptions cover techniques in the field of NLP, specifically on RAG.

A. Comparison of LLM Strategies

There are several methods for adapting large language models for performing specific tasks, whether it may be question-answer, summarization, classification, or any other completion tasks. These methods can be summarized into three general categories - prompt engineering [4], fine-tuning [5], and RAG [6]. RAG is a technique that combines the generative power of language models with the ability to retrieve relevant information from external data sources (online or offline) and compiled by the user. The core idea behind RAG is to leverage the vast amount of knowledge encapsulated in these data sources to enhance the language model's outputs, making them more informative and grounded in the given context while also avoiding hallucinations or the tendency to generate false information.

Fig. 1 presents bar charts comparing the accuracy, cost, implementation complexity, and flexibility [7] of the techniques. Among the techniques, training a model from scratch yields the highest possible accuracy, followed by fine-tuning and RAG, and lastly purely prompt engineering, while for flexibility, this is led with RAG, followed by prompt engineering and fine-tuning, and training from scratch [7]. It is important to note that both fine-tuning and training LLMs from

scratch require a well-curated dataset that can be directly used for additional training, which is rare to be seen and is very computationally expensive. On the other hand, RAG requires much fewer resources for the retrieval and inference phases while still achieving reasonable accuracy levels, which makes it ideal for most use cases including this project.

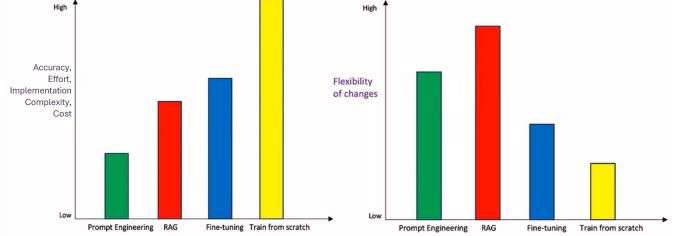


Fig. 1. Accuracy, cost, and flexibility of LLM strategies

B. Text Embedding

A fundamental hurdle in NLP lies in establishing a bridge between the nuanced complexities of human language and the numerical representations computers utilize. Text embedding techniques emerge as a critical solution, transforming textual data – words, sentences, or documents – into numerical vectors. These vectors encapsulate the semantic and syntactic properties of the text, empowering machines to process and analyze language with greater efficacy [8]. Because word embeddings contain several dimensions and can no longer be visualized and spatially represented in a human-compatible and understandable form, dimensionality reduction techniques such as PCA, t-SNE, and UMAP should be performed prior to visualization [9].

Of the available text embedding models, Nomic-embed-text stands out as a popular open-source option designed to address the intricacies of long-context text through its capability of processing sequences stretching up to 8,192 tokens. According to its technical report, it outperformed other closed-source models like OpenAI's text-embedding model on both short and long-context tasks on popular dataset benchmarks such MTEB and LoCo despite having smaller parameter count (137 million) and memory requirements (around 0.51 GB) than most other models need [10] [11]. Fig. 2 presents a comparison of Nomic Embed's performance on the mentioned benchmarks with that of other embedding models.

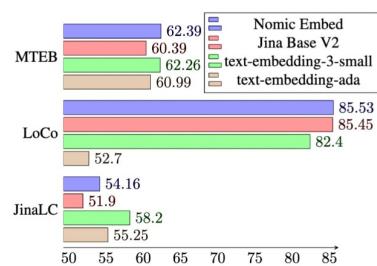


Fig. 2. Comparison of Nomic-Embed with other Embedding Models

C. Vector Databases

Vector databases have become increasingly crucial across various fields due to their exceptional ability to handle high-dimensional data, often referred to as vectors. These specialized databases are designed to efficiently store, index, and facilitate rapid searches within this complex data structure. Each piece of information within a vector database is represented as a vector, residing in a multi-dimensional space. This flexibility allows vectors to capture a diverse range of information, including numerical features, text and image embeddings, and molecular structures [12]. ChromaDB is an open-source vector store designed for storing and retrieving vector embeddings along with various embedding models [13]. Additionally, it can be employed for semantic search engines that operate on text data. This database leverages SQLite and provides users with the option of in-memory storage or local persistence.

D. Document Search and Retrieval

During document retrieval, a query is also transformed into a vector. Similarity search techniques then compare this query vector to the indexed document vectors. By analyzing the distances between these vectors, these techniques can identify documents that share the closest semantic meaning to the query. This enables an efficient and accurate retrieval of highly relevant documents, even for complex or nuanced queries. The four main semantic search techniques can be summarized in Table I [12] and visualized in Fig. 3.

TABLE I
DOCUMENT SEARCH TECHNIQUES

| Metric | Method | Usage |
|--------------------|---|------------------------------|
| Manhattan Distance | Sum of absolute differences between corresponding elements of two vectors while ignoring directionality | Misspelling corrections |
| Euclidean Distance | Straight-line distance between two points in vector space | Document similarity searches |
| Dot product | Directional similarity between two vectors | Similarity of words/phrases |
| Cosine Similarity | Cosine of the angle between two vectors projected in a multi-dimensional space | Document similarity searches |

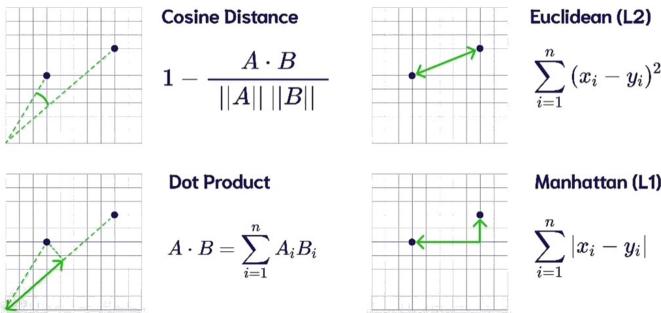


Fig. 3. Distance metrics in vector search

E. Transformer and LLaMA Architecture

The transformer architecture processes language through an encoder-decoder structure [14]. The encoder analyzes the input sequence, breaking it down and understanding relationships between words using self-attention. Self-attention allows the model to focus on relevant parts of the sequence for understanding each element. The decoder then uses the encoded information to generate the output text sequence, one word at a time, again using attention layers to focus on relevant parts of the encoded data for each generated word. This parallel processing and ability to capture long-range dependencies make transformers fast and effective for various NLP tasks.

Recent advancements have introduced LLaMA [15], a powerful contender of the transformer architecture, which revolutionized several NLP tasks. Both architectures rely on encoder-decoder structures, processing input text via encoding and generating output via decoding. However, LLaMA incorporates several architectural improvements upon the transformer. Fig. 4 and Table II show a visual and tabular comparison of the two architectures respectively [16] [17].

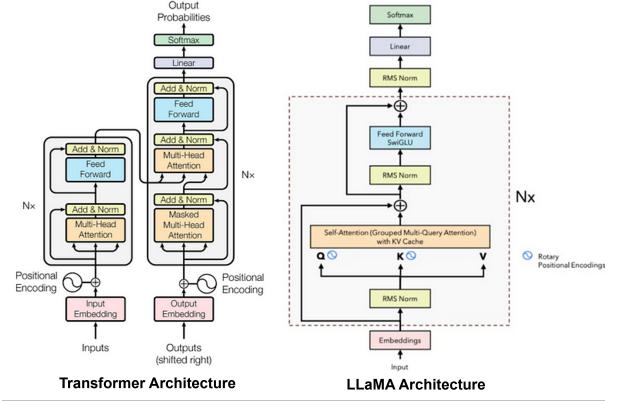


Fig. 4. Transformer architecture vs. LLaMA architecture

TABLE II
CHANGES BETWEEN TRANSFORMER AND LLAMA ARCHITECTURES

| Component | Transformers | LLaMA |
|---------------------|--|--|
| Normalization | Layer normalization after each sub-layer | Root Mean Square (RMS) normalization (better training stability) |
| Activation function | ReLU | SwiGLU (better performance) |
| Positional encoding | Absolute positional encodings | Rotary positional embeddings (reduced parameter usage) |

This project utilizes the 8B parameter variant of the LLaMA-3 model developed by Meta for inferencing, which is trained on a dataset exceeding 15 trillion tokens. This significant advancement enables LLaMA-3 to potentially achieve refined comprehension of language intricacies and propel performance on diverse NLP tasks.

III. METHODOLOGY

This section explains the datasets, tools, and procedures needed to gather and perform to achieve the project objectives. This includes the conceptual framework, a discussion of the pipeline's components, and means of deploying the project.

A. Conceptual Framework

The main goal of this project is to develop an End-to-End question-answer pipeline from the USC resources. Fig. 5 shows the conceptual framework of the study which shows the specific steps involved and the flow of the components, which are elaborated in the remainder of the section.

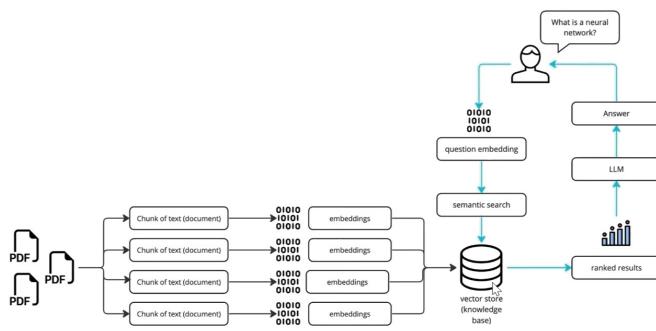


Fig. 5. Conceptual Framework

B. Data Sources and Preprocessing

The texts used for the context of the chatbot are the following:

- 1) USC Undergraduate Students Manual, 2023 Edition [18]
- 2) USC Enrollment Guide AY 2023-2024 [19]
- 3) USC Directory of Student Support Offices and Program Owners [20]
- 4) Guidelines for Motor Vehicle Pass Stickers found in the university's Integrated School Management Information System (ISMIS)
- 5) FAQ posts from the USC Supreme Student Council and USC-Computer and Information Sciences Council Facebook pages highlighting commonly asked enrollment related concerns
- 6) Repository of the undergraduate program brochures found on the USC website's Academics page [21]

Sources (1) and (6) are in PDF formats, source (3) is in Google Slides presentation format, while sources (2), (4), and (5) are in web formats viewable online. However, source (4) can only be visible to students with ISMIS accounts as the viewers are only restricted to official USC students, admin, and staff. Manual cleaning was done through the text to ensure consistency by removing formatting errors, typographical errors, and unsupported characters that libraries could not handle. OCR handled unreadable sections. Standardizations such as punctuation, spacing, and capitalization were applied. Enrollment-related ambiguous sentences were rewritten for

clarification purposes, such as adding missing fee access steps. Abbreviations and their meanings were compiled into a CSV file for model reference. Finally, thematically similar paragraphs were merged and grouped for efficient text chunking.

C. Pipeline Construction

1) Vector Database Creation: At this phase, each preprocessed article is stored in a text file that can be queried for its questions. Note that they were not combined into one big text file to avoid the appearance of paragraphs from multiple topics in a single chunk. For each article or text file, we apply LangChain's [22] RecursiveCharacterTextSplitter() method to divide the article into document chunks with a specified maximum token limit (with the computation of tokens facilitated by tiktoken's "cl100k_base" encoding function) and a specified maximum token overlap limit between chunks.

Next, each document chunk is then fed into the Nomic-Embed-Text embedding model obtainable in HuggingFace [23] to be encoded with vectors of 768 dimensions with each dimension representing a certain semantic meaning to thematically distinguish between the chunks. These encoded vectors, together with the original document source and the metadata (topics) are then saved in a vector store using the ChromaDB library which allows the generated vector store to be persisted and reusable in the upcoming parts of the pipeline. The embeddings can be visualized through Uniform Manifold Approximation and Projection (UMAP) [24], a tool that can accurately map the vectors in a lower dimensional space that is interpretable to humans.

2) Retrieval Augmented Generation: This phase involves the searching of relevant document chunks which will serve as the context for the LLM along with the question for it to generate an appropriate answer based on further analysis of the contexts by the LLM. At first, the question string is fed into the embedding model to create an embedding vector with the same dimensions as that of the other document chunks. This embedding vector is then compared with the rest of the embeddings found in the vector store using the specified vector similarity metrics (refer to Section II-D) such as cosine similarity and Euclidean distance. The documents with the top similarity scores are extracted from the vector store, which are then passed together with the question to the LLaMA-3-8B model by wrapping the context and question in a well-crafted prompt template inside LangChain's [22] RetrievalQAChain pipeline, a class commonly used for employing the RAG approach for question-answering. The model then analyzes the documents, identifies the relevant tokens for the answer, and generates the final answer to be returned to the program, which will finally be displayed along with the metadata of the document chunks indicating the relevant topics.

D. Application Development and Deployment

At this phase, the authors developed an interactive application utilizing the built RAG pipeline through Streamlit, a popular framework for creating interactive data science

applications and can be shared in Streamlit Community Cloud platform [25] to be accessed by other users.

For the authors to evaluate the application based on the prompts made by other users, each question, answer, and source chunk entry was automatically stored in an online spreadsheet via Google Sheets, which is executed by passing the pair of texts through a POST request into a Google Apps Script that handles the data insertion.

IV. RESULTS AND ANALYSIS

This section illustrates the outcomes of the execution of the steps taken to achieve the objectives of the research study. Moreover, it expounds on the different observations of the results obtained as well as interpretations that suggest the effectiveness and improvement points of the project.

A. Exploratory Data Analysis

Before creating the pipeline, the authors first analyzed the composition of the texts such as the distribution of chunks and the relevance from one chunk or topic to the other, as seen in Table III. Here, the number of tokens is computed using the tiktoken library's "cl100k_base" encoding function, which is also the basis of chunk splitting.

TABLE III
DISTRIBUTION OF CHARACTERS, TOKENS, ARTICLES, AND CHUNKS
ACROSS SOURCES

| Source | Characters | Tokens | Articles | Chunks |
|----------------------|----------------|---------------|------------|------------|
| Student Manual 2023 | 141,323 | 28,690 | 36 | 49 |
| Enrollment Guide | 51,493 | 10,521 | 22 | 24 |
| Directory of Offices | 35,048 | 7,700 | 4 | 11 |
| Motor Vehicle Guide | 7,927 | 1,553 | 2 | 2 |
| FAQs from FB Pages | 6,152 | 1,261 | 2 | 2 |
| UG Program Brochures | 112,657 | 20,284 | 42 | 42 |
| Total | 354,600 | 70,009 | 108 | 130 |

The bar plots for the distribution of tokens and articles per source are shown in Fig. 6, while the line plot of the character lengths and token lengths per article is shown in Fig. 7.

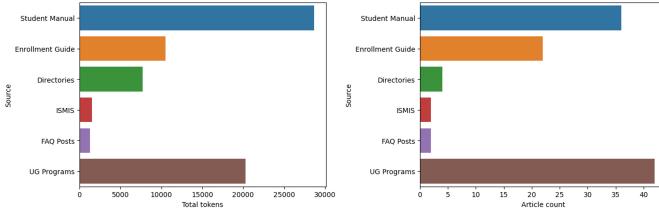


Fig. 6. Token lengths and article counts per source

In total, 95 of the 108 articles contained less than 1000 tokens, 7 of the articles contained between 1001 and 2000

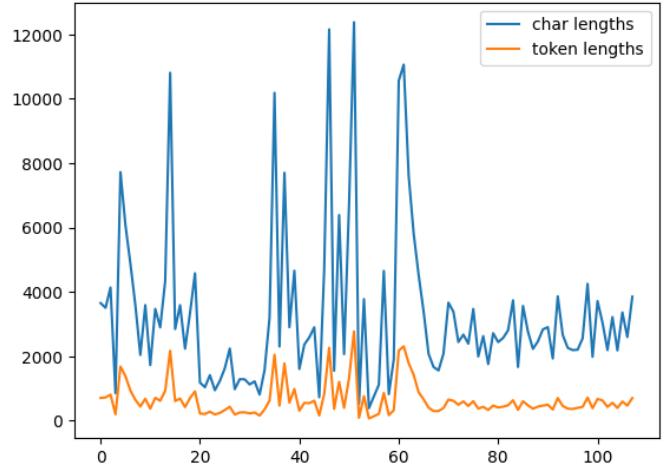


Fig. 7. Distribution of text lengths for all articles

tokens, and 6 articles contained 2000+ tokens. With this, it would be ideal to set the maximum size for chunk splitting to 1000 tokens to preserve the full context of most smaller articles without compromising the coherence of paragraphs between articles. Despite having additional context references, the source with the most articles and tokens was still the Student Manual 2023, keeping it the project's primary focus.

B. Document Embedding

After the splitting of documents into chunks, the chunks were annotated with broader topics to be used as colors and labels in the embedding visualization. For the visualizations, the authors performed dimensional reduction of the vectors via UMAP [24], resulting in the plot shown in Fig. 8.

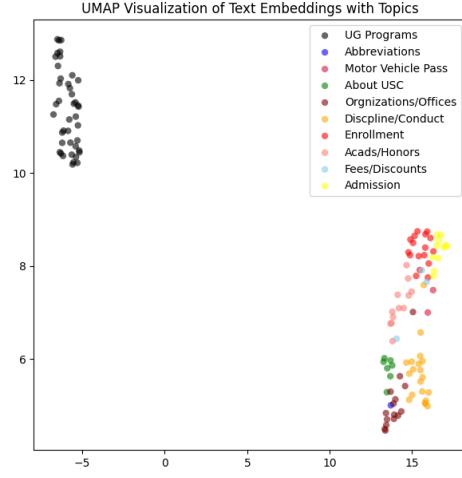


Fig. 8. Embeddings for all articles

In Fig. 8, it can be seen that the embeddings for the undergraduate program chunks are far apart from the embeddings of the other chunks. This tells that the structure of the undergraduate program brochures is completely different from that of the other articles containing procedural and imperative contexts.

Also, the program chunks are mostly just an enumeration of the job positions and courses apart from a technical description which does not involve terms related to the enrollment guide or the student manual at all. The annotated topics of each article were well-grouped together which validated the coherence of the contents of the sources. In addition, the closeness between two particular topic groups also suggested similarities between the different topic categories, such as admission with enrollment (since these are connected topics), About USC with abbreviations and organizations/offices (as the organizations and offices also introduce the image of USC), and enrollment with academics/honors (since part of the enrollment mechanics contains certain academic requirements).

Through understanding the relevance between topics, it is now easier for the readers to trace the search of relevant documents in the retrieval phase due to the context similarity shown within the topic groups.

C. Document Retrieval

This subsection evaluates the relevance of the retrieved documents towards the questions through UMAP visualization of the embeddings and the similarity scores between the embedding of the question and that of the retrieved documents. The authors performed this analysis on four open-ended questions which are all commonly asked by students. Figures 9 to 12 show the plots, relevant documents, and similarity scores per metric for the four questions.

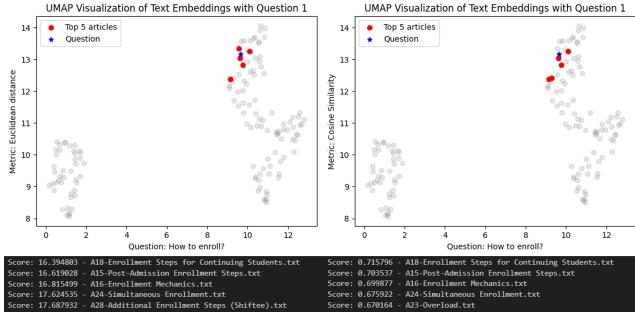


Fig. 9. Retrieved sources with embeddings for question 1 (How to enroll?)

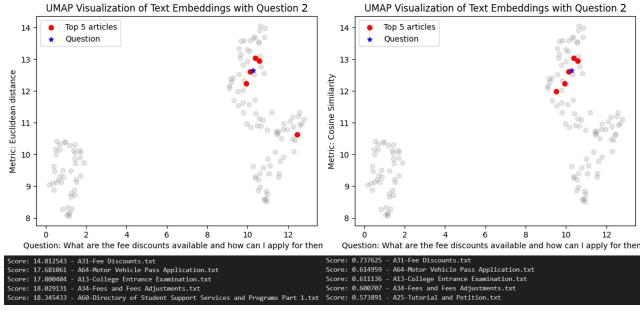


Fig. 10. Retrieved sources with embeddings for question 2 (What are the fee discounts available and how can I apply for them?)

In the plots, the question embedding, retrieved document embeddings, and the other document embeddings are marked

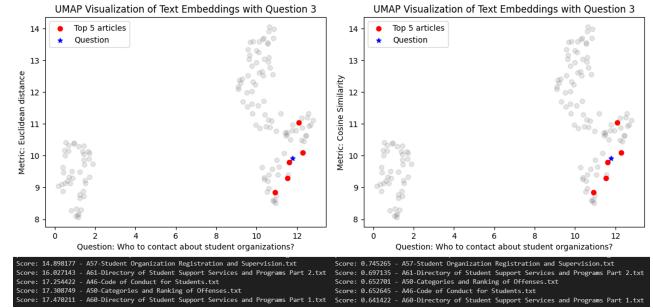


Fig. 11. Retrieved sources with embeddings for question 3 (Who to contact about student organizations?)

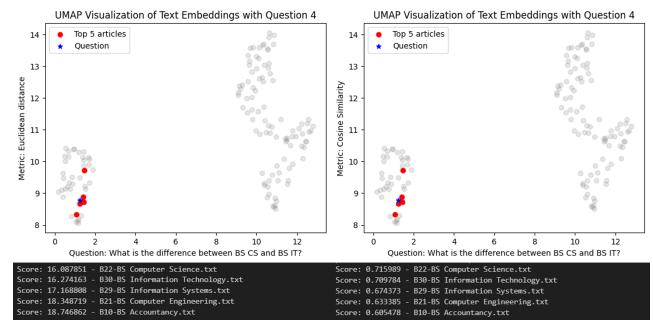


Fig. 12. Retrieved sources with embeddings for question 4 (What is the difference between BS CS and BS IT?)

purple, red, and gray, respectively. The left plot shows the top 5 most similar chunks computed using Euclidean distance with their scores, while the right plot shows the top 5 most similar chunks computed using cosine similarity with their scores.

As seen in the figures, it can be seen that the retrieved questions are significantly close in all directions towards the question embedding, showing a superior ability of the embedding model to store meaningful information from each document as well as generate the proper embeddings for the questions to align with the similar articles. This also shows the importance of choosing a reliable and memory-efficient embedding model for document retrieval tasks.

Another notable observation from the figures is that in all four questions, the top 2 most relevant articles based on Euclidean distance and cosine similarity metrics are the same, and top 5 most relevant articles retrieved for both metrics are present. Lastly, the results also indicate that the document search performed well for complex question types such as comparative questions (Q2 and Q4) which require references from multiple articles, questions with multiple answer variations (Q1, which the model pulled multiple enrollment procedure articles for freshmen and continuing students), and questions from an article split to multiple chunks (Q3) in which only the relevant chunk was retrieved out of three chunks. Factoring in the model's context limits, it can be safe to set the maximum number of retrieved chunks (k) per question in the question-answer chain to 5.

D. Sample Questions and Answers

With the large context window of 8192 tokens in the LLaMA-3 model for storing both the input and output tokens, there is enough room for the model to generate insightful text that addresses the questions asked based on the documents provided. Recall that the token limit per chunk was set to 1000 while the number of chunks for retrieval was set to 5, making the largest possible input token count 5000, or approximately 400-500 words spread across 3 paragraphs.

Fig. 13 shows LLaMA-3's responses to the sample questions which are all complete, accurate, and well-summarized from the sources.

The screenshot shows two panels of a Streamlit application. The left panel contains a sidebar with 'Carolinian Chatbot' and a list of topics like 'Ask anything USC Related here!', 'About USC', 'Core Values', 'Admissions', etc. Below it is a text entry field 'Ask a question...'. The right panel shows a question and its response:

Who to contact about student organizations?

For inquiries and concerns regarding student organizations, you can contact the Office of Student Formation and Activities (OSFA) via landline (032) 2300100 local 102/592, or through the following persons and emails:

- * Student Activities Officer – Ms. Lizzeth Bustillo (ljbustillo@usc.edu.ph)
- * Head of OSFA – Ms. Catherine Algozo (calsagozo@usc.edu.ph)

There are two fee discounts available:

What are the fee discounts available and how can I apply for them?

1. ***Family and Sibling Discount****: This discount is available for siblings enrolled in USC, excluding those in Montessori, Law, and Graduate Programs. The discount is 5% on tuition fees. To apply, click "Student Task - Others", and "Apply Same Family Privilege" in the ISMS portal. Indicate the academic period and year, enter the ID number of the sibling to be included in the privilege, and click "Submit". The application will be forwarded to the VP Finance for approval.
 2. ***Full Payment Discount****: This discount is available for students who pay their tuition fees in full. The discount rates vary depending on the payment method:

- * Offsite Payment Centers and Onsite Cash: 5% discount
- * Onsite Credit Card (Visa or Mastercard): 3% discount
- * Onsite Debit Card: 3.5% discount

To apply for the Full Payment Discount, contact the Accounting Office for verification and necessary computations.

Fig. 13. LLM responses to sample questions

E. Application User Interface

As soon as the parameters were adjusted for the best model performance, the authors developed the application using the Streamlit framework for Python data science applications along with its deployment to Streamlit Community Cloud. Fig. 14 shows the application interface resembling the layouts of most social media messaging and chatbot applications.

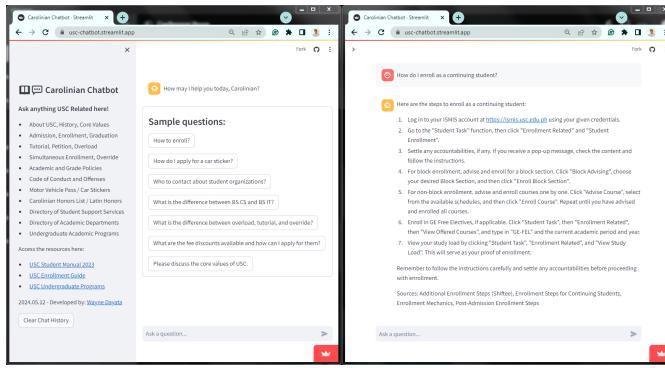


Fig. 14. Deployed chatbot application in Streamlit Cloud

The application initially shows a sidebar listing the related topics and resources, a button to reset chat history, and the main layout displaying buttons of sample questions and a text entry field for users to input their questions. The response from the invoked pipeline, is shown in two parts: The answer generated by the LLaMA-3 model and the article names of the chunks retrieved from the document search phase before the model inference phase.

As seen, the source containing the enrollment process for freshmen students was included in the retrieval but correctly not stated in the response as the prompt specifically asked for the enrollment guidelines only for continuing students.

F. Evaluation

As the app was deployed in Streamlit, several other users managed to access it online and asked questions through the pipeline. Fig. 15 shows the model's responses to some custom questions asked by other users in the system, as logged in the online spreadsheet, showing that the model still performs well on the other questions the authors did not use for sampling.

The screenshot shows a Streamlit application interface with a sidebar and a main content area. The sidebar includes 'Carolinian Chatbot', 'Ask anything USC Related here!', and a list of topics. The main content area shows responses to various user questions:

What are the discounts?

There are two fee discounts available for students during enrollment:

1. Family and Sibling Discount: A 5% discount on tuition fees is available for siblings enrolled in USC, except for those enrolled in Montessori, Law, and Graduate Programs.
2. Full Payment Discount: A discount is granted for students who pay their tuition fees in full, with the discount rate varying depending on the payment method:
 - * 5% discount for Offsite Payment Centers and Onsite Cash payments
 - * 3% discount for Onsite Credit Card (Visa or Mastercard) payments
 - * 3.5% discount for Onsite Debit Card payments

What is the family discount?

The Family and Sibling Discount is a 5% discount on tuition fees for siblings enrolled in USC, excluding those enrolled in Montessori, Law, and Graduate Programs.

What are the required documents for Masters and Doctoral Program International Applicants?

The required documents for Masters and Doctoral Program International Applicants are:

1. Baccalaureate Diploma or its equivalent with remarks of graduation
2. Photocopy of Passport bio-page with at least two-year validity
3. English Proficiency Report, such as TOEFL or at least 60 for TOEFL or as prescribed by the program. Applicants, who graduated from schools that used English as a medium of instruction are required to submit a certificate issued by the head of the institution or its legal representative.
4. Proof of Philippine Citizenship for Dual Citizen Applicants such as Philippine Passport, Identification Certificate (IC), Birth Certificate, and Certificate of Naturalization.

What are the requirements for a student to be officially enrolled?

The provided context does not explicitly state the general requirements for a student to be officially enrolled. It provides specific requirements for different types of students, such as international students, cross-enrollees, special students, and graduates from schools abroad. However, it does mention the requirements for newly accepted applicants to prepare before proceeding to enrollment, which include documents such as Form 138, birth certificate, and a 2x2 colored photo.

Under what circumstances may cross-enrollment be permitted?

Cross-enrollment may be permitted for graduating students with a Deficiency Form, for not more than 6 units during the semester and 3 units during the summer term, inclusive of the total units enrolled in that particular term, and only with the approval of the Department Chair and the School Dean. Additionally, cross-enrollment is not allowed if it is the only remaining course, so as not to prejudice the student's residency in the University.

Until when is withdrawal of course(s) allowed?

Withdrawal of course(s) is allowed until mid-term examinations or until the student has incurred unexcused absences (20% of the total number of required school days).

What might happen to undersized classes during the adjustment period?

According to the text, during the adjustment period, undersized classes might be dissolved, and students may need to consider alternative options.

What should I do if my class gets dissolved?

If a class gets dissolved during the adjustment period, it may be replaced by an extra class group for a particular course. In this case, you can try to enroll in the new class group or look for other available sections of the course. However, the context does not provide a specific step-by-step guide on what to do if your class gets dissolved.

How do I check if my payment has been cleared?

To check if your payment has been cleared, you can follow these steps:

1. Go to "Student Task"
2. Click "Assessment"
3. Click "Student Ledger"

This will show you if your payment has been cleared, meaning the university has received bank confirmation. INC means "Incomplete", which is a grade that indicates that the student has an initial passing grade but has not completed with all the academic requirements or failed to take the final examination for the course due to sickness, emergency cases, or accident. It can also mean a failure to submit a major course requirement.

What does INC mean?

Fig. 15. LLM responses from user questions

However, there are rare instances like in Fig. 16 where the model did not provide a complete answer as it was not sure whether the rules from the context applied to the situation it provided. Nonetheless, the model still performed well in not providing any assumptions, and it shall be the developer who shall adjust that part of the context specifying the coverage of the rule, in the case of Fig. 16, applying for overload during a summer term.

The screenshot shows a Streamlit application interface with a sidebar and a main content area. The sidebar includes 'Carolinian Chatbot', 'Ask anything USC Related here!', and a list of topics. The main content area shows a response to a question:

Do I have to apply for overload for summer classes?

There is no specific mention of summer classes or whether overload applications are required for them. The context primarily discusses overload policies for regular semesters, and it's unclear if the same rules apply to summer classes. I don't know the answer to this question.

Fig. 16. Example response from a complex situation

Another situation where the model did not provide the correct answer is presented in Fig. 17 which is caused by a typographical error by the user in the top row which made the embedding and inference models unable to locate the relevant chunks and answers. This was corrected in the bottom row.

In general, although the USC student manual along with the other USC references are informative in its rules, they have not covered certain extreme conditions that could have been answered by the model had they been also specified in the articles, which can most likely be the kinds of questions that student would ask around other students, officers, or staff as they did in the application.

| | |
|---|--|
| Can you explain more about family privileges? | I apologize, but there is no mention of "family privileges" in the provided context. The topics discussed are related to the Carolinian Honors List, academic policies, code of conduct, and qualifications for the honors list. If you meant to ask about a different topic, please feel free to ask, and I'll do my best to assist you. |
| Can you explain more about family privileges? | The Family and Sibling Discount is a fee discount available for siblings enrolled in USC (University of San Carlos). To apply for this discount, a student can follow these steps: 1. Click "Student Task", "Others", and "Apply Same Family Privilege". 2. Click the home icon and indicate the academic period and year. 3. Indicate the ID number of the sibling to be included in the privilege. 4. Click "Submit". The application will then be forwarded to VP Finance for approval, and the status can be monitored. |
| Can you explain more about family privileges? | The discount is 5% on tuition fees, except for siblings enrolled in Montessori, Law, and Graduate Programs. |

Fig. 17. Variation in responses caused by typographical errors

V. CONCLUSION AND RECOMMENDATIONS

In this project, the authors successfully developed a question-answering pipeline covering essential information from the USC Student Manual 2023 and other supplementary articles. A working application was also built and deployed online for other students to use and evaluate. This eliminates the need for students to perform a manual search across the different articles to obtain the answer nor spend time inquiring about their concerns physically at the other offices. However, in critical conditions or complex situations, it is important that the information generated by the model be verified by students in the relevant areas of the student manual, or they can opt to search for the related office contacts as a workaround to inquire about their queries, which is supported in this application due to the directory of contacts being included in the contexts. The data and source code for this project can be found at https://github.com/20100215/NLP_Student_Manual_Chatbot.

More importantly, the question, answer, and source tuples recorded in the online spreadsheet can be used to determine the accuracy of the answers across different contexts and the most frequently referenced sources from the queries received in the application. With enough user inputs, the data can be further examined to identify parts of the context that should be clarified for better inference. Moreover, this data can further serve as a recommendation to the university administration to conduct reorientations in the most frequently asked questions and spread awareness in the least utilized offices and services.

Finally, the authors noted the following recommendations to improve this project further:

- 1) Incorporating models and algorithms that can handle tabular data, which can be applied in the analysis of program prospectus with their pre-requisite and co-requisite courses.
- 2) Gathering data from other university student manuals and strategically storing them to allow comparison of rules across different universities without mixing up the source universities where the rules apply.
- 3) Performing automated tasks such as notifying the offices of the concerns addressed to them as identified by the model for the staff to reply with additional information.
- 4) Experimenting with fine tuning of inference models with smaller resource requirements for better overall efficiency once an adequate amount of question, answer, and source chunk entries have been reached.

REFERENCES

- [1] M. Caballero, "A brief survey of question answering systems," *International Journal of Artificial Intelligence and Applications*, vol. 12, no. 5, pp. 01–07, Sep. 2021, doi: 10.5121/ijaia.2021.12501.
- [2] M. Sri, "NLP in virtual Assistants," in *Apress eBooks*, 2020, pp. 185–247. doi: 10.1007/978-1-4842-6246-7_5.
- [3] N. T. K. Le, N. Hadiprodjo, H. El-Alfy, A. Kerimzhanov, and A. Teshebaev, "The Recent Large Language Models in NLP," *2023 22nd International Symposium on Communications and Information Technologies (ISCIT)*, Oct. 2023, doi: 10.1109/ISCIT57293.2023.10376050.
- [4] P. Sahoo, A. K. Singh, S. Saha, V. Jain, S. Mondal, and A. Chadha, *A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications*. 2024. doi: 10.48550/arXiv.2402.07927.
- [5] Z. Hu *et al.*, "LLM-Adapters: An Adapter Family for Parameter-Efficient Fine-Tuning of Large Language Models," *arXiv*, Jan. 2023, doi: 10.18653/v1/2023.emnlp-main.319.
- [6] P. S. H. Lewis *et al.*, "Retrieval-Augmented Generation for Knowledge-Intensive NLP tasks." *Neural Information Processing Systems*, vol. 33, pp. 9459–9474, May 2020, [Online]. Available: <https://proceedings.neurips.cc/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf>
- [7] V. Pandey, "Should you Prompt, RAG, Tune, or Train? A Guide to Choose the Right Generative AI Approach," *Medium*, Aug. 19, 2023. [Online]. Available: <https://medium.com/@pandey.vikesh/should-you-prompt-rag-tune-or-train-a-guide-to-choose-the-right-generative-ai-approach-5e264043bd7d>
- [8] R. Patil, S. Boit, V. Gudivada, and J. Nandigam, "A survey of text representation and embedding techniques in NLP," *IEEE Access*, vol. 11, pp. 36120–36146, Jan. 2023, doi: 10.1109/access.2023.3266377.
- [9] R. Egger, "Text Representations and Word Embeddings: Vectorizing Textual Data: Interdisciplinary Approaches, Methodologies, and Applications," in *Applied Data science in tourism*, Springer Nature, 2022. doi: 10.1007/978-3-030-88389-8_16.
- [10] "Introducing Nomic Embed: a truly open embedding model." <https://blog.nomic.ai/posts/nomic-embed-text-v1>
- [11] Z. Nussbaum, J. Morris, B. Duderstadt, and A. Mulyar, "Nomic Embed: Training a Reproducible Long Context Text Embedder," Feb. 2024. doi: 10.48550/arXiv.2402.01613.
- [12] D. Jurafsky and J. H. Martin, "Vector Semantics and Embeddings," in *Speech and Language Processing*, 1st ed., 2024. [Online]. Available: <https://web.stanford.edu/~jurafsky/slp3/6.pdf>
- [13] "Chroma: the AI-native open-source embedding database." <https://www.trychroma.com/>
- [14] X. Amatriain, "Transformer models: an introduction and catalog," *arXiv*, Feb. 2023, doi: 10.48550/arXiv.2302.07730.
- [15] Meta Inc., "Introducing Meta Llama 3: The most capable openly available LLM to date," Meta AI, Apr. 18, 2024. <https://ai.meta.com/blog/meta-llama-3/> (accessed May 15, 2024).
- [16] H. Erz, "The Transformer Architecture: A Visual Guide," [hendrik-erz.de](https://www.hendrik-erz.de/post/the-transformer-architecture-a-visual-guide-pdf-download), Apr. 07, 2024. <https://www.hendrik-erz.de/post/the-transformer-architecture-a-visual-guide-pdf-download> (accessed May 15, 2024).
- [17] P. Khadka, "LLaMA explained! - Pranjal Khadka," *Medium*, Apr. 10, 2024. [Online]. Available: <https://medium.com/@pranjalkhadka/llama-explained-a70e71e706e9>
- [18] University of San Carlos, "University of San Carlos Manual for Undergraduate Students 2023 Edition," Aug. 12, 2023. <https://drive.google.com/file/d/1rFTThhqMrVqMF0k0wMFMOIZuraF4AywYN/view>
- [19] University of San Carlos, "Enrollment Guide for USC Students." <https://enrollmentguide.usc.edu.ph/>
- [20] University of San Carlos, "Enrollment Guide for USC - Contact the Offices." <https://enrollmentguide.usc.edu.ph/contact-the-offices>
- [21] University of San Carlos, "Undergraduate Programs," <https://www.usc.edu.ph/academics/undergraduate-programs>
- [22] "LangChain." <https://www.langchain.com/>
- [23] "Hugging Face – The AI community building the future." <https://huggingface.co/>
- [24] "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction", *umap 0.5 documentation*. <https://umaplearn.readthedocs.io/en/latest/index.html>
- [25] "Streamlit - A faster way to build and share data apps." <https://streamlit.io/cloud>