Transforming Mentorship: An AI-Powered Chatbot Approach to University Guidance

by

Mantaqa Abedin
23241130
MD Faizul Islam Ansari
24141189
Monowar Zamil Abir
21201187
Adib Reza
24141197
Mashrur Rahman
24141214

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
January 2025

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.

2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.

3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.

4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Mantaqa Abedin
23241130

MD Faizul Islam Ansari
24141189

Monowar Zamil Abir
21201187

Adib Reza
24141197

Mashrur Rahman
24141214

Approval

The thesis/project titled "Transforming Mentorship: An AI-Powered Chatbot Approach to University Guidance" submitted by

1. Mantaqa Abedin (23241130)
2. MD Faizul Islam Ansari (24141189)
3. Monowar Zamil Abir (21201187)
4. Adib Reza (24141197)
5. Mashrur Rahman (24141214)

Of FALL, 2024 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 31,2025.

Examining Committee:

Supervisor:
(Member)

Farig Yousuf Sadeque

Dr. Farig Yousuf Sadeque
Associate Professor
Department of Computer Science and Engineering
Brac University

Co-Supervisor:
(Member)

Niloy

Niloy Farhan
Adjunct Lecturer
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam
Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Dr. Sadia Hamid Kazi
Chairperson
Department of Computer Science and Engineering
Brac University

Abstract

Chatbots are agents designed for communicating with humans and mimic the human-human interaction style. They can understand natural language and communicate with users through it. In this complicated digital world, university students face enormous challenges throughout their undergraduate journey. Universities cannot appoint that many mentors to attend to every student's query. Even though there are a notable number of digital tools and platforms to support students, there remains a gap in personalized guidance for newcomers, specifically in the form of chatbots. A digital mentor to meet the specific demands and difficulties faced by new students is what will help them handle academic obstacles more effectively. Filling this crucial gap is essential for the creation of supportive and inclusive educational technologies. Our research aims to design and develop a corpus-based chatbot that will primarily serve as a mentor to students. The interactive chatbot will be able to communicate with the user through informal dialogue generated by natural language processing. Finally, it will conclude by finding and assessing any biases that the chatbot may exhibit. This chatbot can assist first-year students by answering their questions. It can also help students obtain a sense of the topics of a course and plan an effective class routine for their semesters. Thus, it will assist students during their academic life of open-credit universities.

Keywords: Chatbot, Corpus-based, Natural Language Processing (NLP), Personalized guidance, Interactive chatbot, Educational technology, Informal dialogue, Open-credit universities, Bias assessment

Acknowledgement

Firstly, all praise to the great Allah for whom our thesis have been completed without any major interruption. Secondly, to our supervisor Dr. Farig Yousuf Sadeque Sir and co-supervisor Niloy Farhan Sir for their kind support and advice in our work. They helped us whenever we needed help. And finally to our parents without their throughout support it may not be possible. With their kind support and prayer we are now on the verge of our graduation.

Table of Contents

Chapter 1

Introduction

Challenges were found to be significantly present throughout the four years of university education due to the dynamic academic environment, large number of students with smaller number of faculties and staffs, difficult policies and procedures of university programs. Some of these challenges include – being able to get adequate and accurate information about policies, selecting proper courses and having a good class schedule to manage the limited time available for open credit universities, all this while coping with inadequate number of face to face lessons with mentors due to shortage of mentors. Although students are equipped with many resources proffered by the innovations in technology, there still lies a large gap when it comes to on-demand and personal assistance. This is particularly risky for first-year students who are usually challenged by the new environment they find themselves in and might require more help on how best to go about it. We will therefore attempt to fill this gap through providing a corpus-based chatbot that will secondly function as a student's companion. This is because chatbots are being used to offer several benefits, some of which include; the following: providing prompt and personalized assistance (Shawar Atwell, 2007)[4]. These digital agents have capability to understand natural languages and when the user wants to talk to such an agent, then talk in HHI like manner. In this manner, on aggregate, through NLP, the ability of the chatbot will be united to comprehend questions and answers from students suitably to ensure students get the necessary help and direction (Brandtzaeg Følstad, 2017)[6].Therefore the implementation of chatbots educationally has been deemed compulsory all over the world. They can also save the amount of time and energy that the faculty and staff members waste in addressing more basic questions and queries because the bot is capable of responding to it immediately (Winkler Söllner,2018)[11]. For instance, the students can engage the chatbot and ask questions on subjects being undertaken, schedule for class timetables, and receive help on matters concerning general academic endeavors. This automation makes it easy for the tasks that can be solved by algorithms to be solved automatically, freeing the teachers and mentors to attend to tasks that cannot be automated such as the analysis of students behavior, ability to pay attention to lessons and the ability to continue learning making it efficient (Folstad Skjuve, 2019)[12]. Regarding the result of this work I would like to state the following: after the discussion, we will design and build an efficient chatbot to help the first-year students deal with such difficulties in studying more effectively. According to the proposed system's design, the chatbot is supposed to enhance the learning of students by providing them with the right information at the right time regarding their semesters and the topics studied in the respective courses. It will also be beneficial to those whose advice is to be given but it will also relieve the pressure from the faculties and increase the efficiency of advising. Also, it will assess whether the chatbot has any bias in terms of the gender, race, nationality or anything that would make it biased in addressing any of the clientele. Therefore, it is possible to conclude that the use of a specially designed corpuschat as a part of the digital mentoring system as the solution to the challenges linked with the application of new information technologies in universities is perspective. It can minimize the gap for individual attention, to facilitate delivery of technologies in education for better support to individuals. To conclude, this research is aimed at constructing a realistic tool that will facilitate the learning process of the student and reduce the workload of faculty and staff in order to enhance overall effectiveness and efficiency.

1.1 Research Statement

Therefore, in our research, we propose to create and implement a corpus-based chatbot with the objective of helping university students with customized advice using natural language processing. This chatbot will assist students in managing their academic issues and daily time table regarding courses and thus reducing strain of information management among the students and faculties.

1.2 Research Outcome

The objectives of our research will be to design and develop a corpus-based chatbot using natural language processing which users can interact with. We hope to achieve following outcomes after the completion of the research:

1. Chatbot will answer FAQs of students regarding any policy and administrative tasks.
2. It will guide the first-year students throughout their journey.
3. It will suggest course sequences and elective courses according to students' requirements.
4. It will help students during the advising season.
5. It will be able to perform all the tasks in both English and Bengali languages.
6. Finally, the research paper will contribute to the academic community with research findings published in top journals and across the globe to push the pedal of innovative educational technologies.

1.3 Structure of report

Chapter 2 of this paper contains a thorough evaluation of the literature. It contains survey methodology, historical context, and relevant literature. Chapter 3 contains a detailed study work plan illustrated using a Gantt chart. Chapter 4 concludes our pre-thesis 1 report.

Chapter 2

Literature Review

2.1 Survey Methodology

In our research to transform mentorship to chatbot, we have gone through different research papers related to our work to help us know the existing technologies, research gaps, and appropriate research methods. We have surfed through different websites including Google Scholar, ACL Anthology, Papers With Code, ResearchGate and many more to search for related works. Furthermore, we have used different keywords like ‘chatbot’, ‘FAQ chatbot’, ‘chatbot for university’, ‘chatbot for college advising’, ‘Question Answering system’, and ‘AI mentorship’. Thus, we collected almost 50 papers primarily by filtering through paper titles. Following that, we skimmed through all of the 50 papers and selected the top 18 papers for our literature review based on citation count, content relevance, content variation, and publication date.

2.2 Related Works

Adamopoulou and Moussiades (2024) [17] in their paper give comprehensive previews about the transformation of chatbot. In 1950 Alan Turinin ran a test which is called the Turing Test that intended to know if a computer could emulate human conversation. The first chatbot ELIZA developed in 1966 appeared to be a psychotherapist and used pattern matching and templates of responses to interact with users; however, it was not able to go in depth in the conversations that it could engage in. In 1972, PARRY took up the idea of higher quality by implementing a simulation with personality and even emotions of a schizophrenic patient. In 1988 Jabberwocky chatbot was the first chatbot to incorporate the use of AI where it used contextual pattern matching. The term ‘chatterbot’ was first used in 1991 and it was with a TINYMUD player and also without the complexity of the interaction Dr. Sbaitso in 1992 demonstrated sound card characteristics. Then ALICE was developed in 1995.Though it was not intelligent It incorporated AIML for a greater scope and pattern matching. In 2001 further evolution name SmarterChild came which was friendly at performing reasonable tasks through messengers; Nowadays there are many text based and voice recognition based intelligent virtual assistants such as Siri of apple 2010, Watson of IBM 2011, Google assistant 2016, cortana of Microsoft 2014, and alexa of amazon echo 2014. These assistants use natural language processing that enables it to work on natural language and perform various operations that include but are not limited to analyzing them these assistants are however, they have following limitations like no multiple language support and do not know the context of the work they are processing.Since 2016, social media platforms as the primary agents for developing general-purpose and diverse specialized chatbots for various uses have appeared and since then, thousands of chats have been developed. It has a modern iteration in today's social media and voice-activated assistants, including the XiaoIce of Microsoft. Today's chat-bots, which are also significantly advanced from ELIZA, are capable of sustaining much more realistic and para linguistically smart conversations and they use the principles of machine learning algorithms for unsophisticated interactions but are context driven. The use of more than one layer of LSTM in the Seq2Seq has shown greater ability and realistic model of generative chatbots (Csaky, 2017). chatbot builds on more primitive models of ELIZA that only relied on patterns to match and all the way to models such as VHRED by Serban et al. (2016) that incorporates latent variables or hierarchical systems in order to maintain the conversational context and select the best response. Among the recent advancements, one can mention Bidirectional Recurrent Neural Networks with an attention mechanism for processing a great amount of input data (Dhyani Kumar, 2020) and multilingual chatbots based on GRU cells (Sperlí, 2020). The components of chatbot architecture include the user interface, the user message analysis, the dialog management component, the backend component, and the response generation component, and three primary tools for developing chatbots are RASA, Dialogflow, and IBM Watson. Instant messaging apps in the modern world are increasingly adopting the use of chatbots (e. g. , Facebook Messenger, Skype, Twitter, Slack) and websites for executing the following purposes: data-sharing, call-arranging, and ordering. These are some of the issues like; Intent recognition, Toxic content, and Data protection. In conclusion, it is necessary to state that when designing the Chatbot it is necessary to foresee the goal of the conversation, use proper NLP tools, and have constantly changing CXP. One of them is in the sphere of education, where they might provide individual educational assistance, and in healthcare, they might provide detailed information and suggest a treatment. More efforts will be made in the following areas in order to elaborate the future strategies – Natural language processing, privacy concerns, security challenges.

Suta et al. [16] (2020) explore the difficulties in creating chatbots that are intelligent, advanced, and able to converse like humans in the article. With the help of natural language queries from users, these chatbots are capable of understanding and producing responses that are seemingly human. Despite extensive AI development, current chatbots are not as advanced as the fully automated chatbots we hope to use in the future. The main problem is chatbots are weak in natural language processing skills for this reason they struggle to keep up with user's questions and often they provide users with irrelevant and wrong information. To improve the situation, the researchers reviewed a large number of papers published between 1998 and 2018. Papers from Google Scholar, IEEE Xplore, ACM, and Web of Science were discussed and used as a database to understand and analyze the current state of chatbots, as well as to keep up with chatbot technology trends. The models mentioned in the paper primarily involve machine learning(ML) and NLP techniques.

The paper primarily focuses on machine learning and natural language processing techniques. Support vector machines (SVM) models are used for question answering, Long Short-Term Memory (LSTM) networks are used for providing answers to queries, and sequence-to-sequence (seq2seq) models are used for encoding and decoding the input-output sequence, which use two Recurrent Neural Networks (RNNs) in this work. The Word2vec technique is used for vector representation of words, and they also used Neural personalized response generation for domain adaptation, which is an AI technique for creating chatbot responses to specific topics that the user requests. The paper's mind map demonstrated how the visual representation of these various models could function, particularly with regard to a chatbot setup. Finally, because the chatbots are trained using various NLP and ML techniques, they are able to effortlessly carry out predefined tasks and identify particular sentence structures. They struggle to understand context and come up with prompt answers. In the future, advanced NLP and ML techniques could aid in enhancing chatbot performance. The ultimate goal could be to create a highly potent chatbot that can converse like a real person, facilitate human-to-human communication, and pass the Turing test in order to create a system of communication that is on the same level with humans.

Ranoliya, Raghuwanshi, Singh (2017) in this paper [8] explore chatbot development for a learning environment to improve customer service while reducing costs. The main goal is to create a chatbot that can provide accurate and quick answers, increasing customer satisfaction. The review of the literature sheds light on the use of Artificial Intelligence Markup Language (AIML) and Latent Semantic Analysis (LSA) as enhancers of response accuracy. Here ELIZA is introduced, which mimics speech using keyword matching and traces the evolutionary trend of chatbots. However a major limitation of ELIZA is, it had problems understanding the context and could not understand the question properly in many cases. For this reason, it gave repetitious and irrelevant responses often. Observing these, ALICE came forward which utilized AIML and XML-based language and pattern matching to organize knowledge and simplify conversational flows. Although AIML can work well in template-based situations, it shows its limitations when faced with complex or situation-specific questions. Here LSA comes into light. Latent Semantic Analysis is a method that examines the connection between a collection of documents and the terms or words they contain. By doing this, it can understand and react to the complex queries and intent of the user. This research suggests that to create a precise, intuitive, and effective chatbot, we need to propose a hybrid approach by combining AIML and LSA with the already existing chatbots like ELIZA and ALICE.

Nguyen (2019) [13] talks about the question-answering system in the biomedical domain in this paper. Understanding the methods is crucial for utilizing the same in terms of an educational environment. Biomedical QA relied on information retrieval methods such as tf-idf ranking and entity extraction tools such as MetaMap by fetching candidate responses from biomedical databases. It also used feature extraction and machine learning approaches like logistic regression. However, it failed to capture complex lexicons of clinical terms. It was limited by their reliance on exact term matching and basic bag-of-words techniques. After this, neural techniques were used where semantic and contextual processing was enhanced. The semantic understanding between question and answer was understood better when methods like the Position-Aware Convolutional Recurrent Relevance Matching and Deep Relevance Matching Model were introduced. Despite all these improvements, challenges such as clinical phrase ambiguity and the need for effective disambiguation tools persisted. Sometimes biomedical QA systems face queries that are not present in open-domain QA and are unique. The complex inquiries require specialized knowledge to answer. On top of that, clinical language has acronyms and specialized terms in it. As there is a shortage of high-quality publicly available data, solving this issue is difficult. Another challenge is that the datasets need to be labeled. These datasets are difficult to obtain as they require expert annotation and ethical consideration. Synthetic datasets are also a game breaker as they perform worse than handcrafted ones. To make biological terminologies accessible, researchers focused on developing clinical ambiguity resolution and summarization tools. It is also important to use neural models to handle complex answers and have common sense reasoning. Variational autoencoders and crowd-sourcing techniques are being explored to handle neural approaches. So if we can combine high-quality labeled datasets, our QA system's performance will improve.

Hwang and Chang (2021) in their paper [18] give previews about the opportunities and challenges of chatbots in education.With the increasing popularity of the use of Mobile devices and artificial intelligence in teaching, chatbots are proving to be helpful in the teaching process and these virtual assistance are found to be highly effective as they are interactive, convenient, and real-time assistants while teaching, they also help in enhancing the communication skills and they provide individual feedback which is very important during interacting process between the teacher and the learner. The flexibility and availability of the chatbots shall be utilized to support the conventional modes of instructing and increasing the learning model. In spite of the numerous opportunities present in the use of chatbots.While chatbots would be highly useful when implemented in the education system, there is little evidence focused on the effectiveness and applicability of the concept, where most existing studies are concentrated on the application of chatbots in the health care system. With regards to the used statistical tools, more so, the ones that have applied structural equation modeling, ANOVA, and t-tests, there will be indications of positive impacts of the chatbots on learning outcomes. The statistical tools mentioned above will be used to establish whether there are significant differences in the performance between groups that utilized chatbots and groups that did not use them. Firstly, to date, there aren't any reviews that can completely describe the usage along with its benefits. Second, the majority of the research is quantitative and thus there is little data, and certainly no qualitative data on other ways through which the use of chatbots can improve students' participation, enthusiasm and learning. Third, some previous works have been designed in only certain subjects; in consequence, its result is not so final and versatile as in typical educational environments.Another major concern pertains to the issue of privacy: Here most chatbots accumulate and analyze the user's information, which in this case is regulated by data protection principles. This, therefore, requires considering the following challenges and possibilities, which would otherwise be realized in the case of exploiting the use of chatbots in learning. It involves formulating a unique and effective plan for the implementation of chatbots in all categories of learning institutions including the basic, secondary and tertiary education institutions and learning institutions that offer training. Therefore, efforts should be made by the researchers to incorporate both the qualitative and quantitative research methodologies in a bid to bolster the empirical findings and conclusions. Moreover, there are few works investigating the difference in achievement or learning rate that is needed to improve the design of the changes to be made after the usage of chatbots in learning. Therefore, by addressing the above- mentioned research gaps, and consequently addressing the explored challenges, both educators and technologists will be able to get a much better understanding of how to improve the tangible implementation of chatbots in this field.

This research [14] by Swanson et al. (2019), talks about the introduction of RNN in conversational retrieval chatbot models. With the advancement of the conversational retrieval chatbot models, a major problem was faced when the models could not understand the context of the questions and give appropriate relevant answers. To solve this problem, RNN was introduced. To find the ideal response, RNN stored the conversational context and potential responses. And by calculating these, it determined the optimal answer. On this foundation, Serban et al. (2015) and Zhou et al. (2016) developed a hierarchical architecture that used RNN at both word and utterance levels. It helps to encode the conversational context which improves the contextually accurate responses. After this, Wu et al. (2017) refined the model by building a network that matched responses to specific context questions. Even after all these advancements, multiple challenges remained. It was difficult to deploy these models in real-world production environments effectively, especially in terms of inference speed and scalability. However, these issues were overcome by Lei et al. (2018) who proposed dual encoder architecture. It used a vast whitelist to ensure swift response selection and resulted in an improvement in inference time over conventional LSTM-based models by 4.1 times. However, a major problem still exists as creating and evaluating a response whitelist is challenging. On one side, a clustering-based whitelist offered greater recall whereas a frequency-based whitelist provided higher coverage. We need a combination of both. The most reliable approach remained the human evaluation method as it can identify qualitative factors and overlook the automated metrics. In particular, research shows that human evaluators favor simpler, commonly used solutions and rank the responses from frequency-based whitelists higher in real-world circumstances. Although conversational retrieval models have advanced a lot, challenges remain in scalability, inference speed optimization, and thorough assessment.

This paper [20] by Atmauswan and Abdullahi (2022), talks about the development, application, and user interaction of educational chatbots, highlighting key studies and achievements. We have come a long way from the early chatbots like ELIZA and ALICE which were designed for simple interactions. Now, chatbots use AI and

NLP for sophisticated functionalities. The development of mobile phones and the internet increased the popularity of chatbots. It is now used to find information and also for customer service. It can handle a lot of complex tasks as well such as administrative tasks, offering personalized information, and whatnot. One example is a chatbot called LISA which helps students with both general and customized inquiries. Creating a chatbot needs several considerations. It needs to have a distinct personality which will ease user engagement and enhance user experience. It must ensure smooth interactions by handling misspellings, local idioms, and different natural language variations. We need to focus on user needs and make sure to give high priority to User-centered design. By doing so, we can ensure to provide relevant information and services. User input and surveys shape the chatbot's functionality and interaction style. However, maintaining data privacy, response accuracy, and collaborating these to ensure a smooth system is very challenging in terms of implementing chatbots in educational contexts. We also need to make sure that it mimics human conversation which seems natural and avoids user discomfort. In the future, chatbots should have much more AI capabilities to handle more complex queries. Expanding functions to include proactive help, such as reminders and notifications, may increase their usefulness. Overall, the design and deployment of chatbots in university environments promise to improve student support and administrative efficiency. Chatbots can become vital tools for improving the campus experience as technology improves if they are designed with the user in mind and address educational concerns.

Cui et al. (2017) introduce SuperAgent, a customer service chatbot designed for e-commerce websites in this article [7]. In contrast to traditional chatbots that rely on human communication, SuperAgent uses massive amounts of openly accessible e-commerce data to deliver customer service. To respond to user inquiries, the chatbot pulls information from product descriptions, customer reviews, and customer questions and answers. SuperAgent can easily handle various routine queries using this data, as a result it frees up human support professionals to focus on other hard questions. Text question answering, opinion mining, fact question answering, FAQ search, and chit-chat interaction modeling are some sub engines that are used to deliver precise answers to user queries various modern machine NLP and machine learning approaches are used by these engines. The authors used the SemEval-2016 Task 1 dataset to assess their model. This dataset contained various publicly available e-commerce data that was specially created for question-answering activities.. Product Information (PI), Questions amp; Answers (QA), and Customer Reviews (CR) from a variety of e-commerce websites, including Amazon.com, Ebay.com, and JD.com, are the data sources included in the collection. Overall, the study's dataset offered broad details on several product features from the viewpoint of users, making it appropriate for the conversation engine's evaluation and training. The model uses a number of sub-engines, including an opinion-oriented text QA engine for extracting review information, a regression forest model for FAQ searches, an attention-based LSTM seq2seq model for casual chat, and a DSSM model for fact-based queries. These models achieve state-of-the-art performance: the chit-chat engine maintains coherence with a perplexity of 16.9, and the FAQ search model achieves a mean accuracy of 0.78455. SuperAgent's usability studies show how important and useful it is for enhancing the online buying experience. Through the integration of product information, customer reviews, and customer QA sessions, SuperAgent creates a single, easily available knowledge library for users. With everything considered, SuperAgent improves consumer pleasure and convenience by simplifying the process of learning about products on e-commerce websites.

Sweidan et al. (2021) introduce an Android application designed to assist students at the University of Jordan in this research [19]. It provides tools including campus maps, reminders for lectures and exams, access to personal information and course schedules, and a bilingual chatbot (Arabic and English) that can respond to a range of questions. The University of Jordan students' daily and academic queries made up the dataset used in the creation of the SIAAA-C application. Visiting various university faculties and completing an online survey over the course of a month were the two steps in the data collection procedure. The primary dataset contains anticipated inquiries about the office hours of teachers, their contact details, course study materials, course summaries, suggested semester timetables, and campus facility locations. After the questions were gathered, they were categorized, duplicates were eliminated, and responses were created and kept in a database. In order to provide relevant responses, the chatbot's primary functionality depends on pattern matching and keyword recognition algorithms. The model first determines the language of the input text by determining whether it contains Arabic or English characters. It then pulls the appropriate response from a predefined set of responses kept in a Firebase Realtime Database. The chatbot matches the user's query against these stored items in the database to determine the most accurate response. During the COVID-19 epidemic, 102 University of Jordan students tried the SIAAA-C application, and the results were positive. The chatbot answered questions in both Arabic and English with a 92% accuracy rate. The use of the app was well-received as 88.23% of users acknowledged its increasing value throughout the pandemic, and over 90% reported no difficulties. The study comes to the conclusion that SIAAA-C, particularly in times of crisis, promotes students' academic management effectively.

In this article, the system El-Ashmawi et al.[21] (2023) have developed to create a chatbot for college enquiries is quite similar to what we want to achieve in our research, the authors use a technique called ALICE which is an award winning open-source regular language artificial intelligence visit robot that responds to queries using AIML.They also use RASA framework which uses LSTM to save memory. Getting all of the data into one interface without having to deal with the hassle of filling out numerous forms and windows can be almost impossible. The goal of the college chatbot is to make this unnecessary by offering a consistent and intuitive interface. This interface provides educators and students with natural language responses to their questions. This chatbot was created by AI algorithms that analyze various user requests and understand the messages. In order to give the user an answer, the system assesses the query and does additional analysis. When asked a question, the technology answers it as though it were coming from a real person. They designed it mainly to make websites more user-friendly and interactive, also to lower the student search time. The dataset was collected by the team members from various departments of the university and also the university website manually. It contained different information like fees, course structure and other queries that a student may face. The system was implemented using Keras, Tensorflow and other natural language processing units. The three-layered Multilayer Perceptron (MLP) Neural Network model—that is, an input layer, an output layer, and a hidden layer—was used to construct the model. Tokenizing and lemmatization are two NLP techniques used to handle the dataset. A feed-forward artificial neural network (ANN) and the Keras sequential models are then used to train the data. In the result section we can find that whenever a user asks any query about the university, the chatbot response is pretty accurate and the model accuracy is around 87%.

In this article [23], Formica et al.(2023) describes how the answers to complex NL questions can be converted into SPARQL queries, which can be used on knowledge bases (KBs) like DBpedia or Wikidata. The dataset used for this problem is a Complex-Sequential Question Answer (CSQA) dataset consisting of over 200,000 dialogues with 1.6 million questions and answers sourced from Wikidata. Three various kinds of questions were used: comparative, quantitative, and logical. A semi-automatic template based approach was proposed by the authors to solve the problem. The method involved human expertise by creating query templates and classifying techniques to match the NL question to the matched template. The best query template was predicted using a combination of classification techniques, including Gradient-Boosted Decision Trees (GBDT), Naive Bayes, and Support Vector Machines (SVM), which focused on syntactic features(Parts of Speech tagging) and semantic features(Discriminative words). The system required a number of steps to function, including feature extraction, question classification to match the ideal template, named entity recognition and relation linking, and SPARQL query creation. Syntactic and semantic features were combined to create a feature matrix, which was then extracted using the CountVectorizer function. After that, the classifiers picked the best template for the respective NL questions which are related with the KB sources. Next, using the resources from the chosen query template, the actual SPARQL query got created. The result of these methods worked very well as the questions matched with the query templates nicely as the result was 65.44% where the fully automatic system had 10.52% less than this. Beside having 0.65 precision, for logical questions the model had a recall of 0.79 and F-score of 0.69. Ultimately, these findings demonstrate how well the semi-automatic template-based method performs in delivering accurate responses to challenging natural language queries across knowledge bases.

In this article, Nguyen et al.[26] (2023) found the problem ‘summermelt’ where 22-40% of students do not get admitted to a college even though they got accepted. The reason behind this is the few academic advisors allocated for proportionally larger amounts of students. The scarcity of social-emotional support and information causes the students to drop out. To resolve this problem, a chatbot called ‘Lilo’ is introduced and they study the interaction process of the users with Lilo. The article uses four primary datasets which are daily user surveys, semi -structured interviews, daily diary entries and user logs of chatbot. These datasets are used to capture various aspects of user experience. Different NLP platforms are used to build the chatbot. It is built on 'RASA' for understanding students' FAQs and generating proper responses. 'GPT-2' is used to make the training data more robust. Furthermore, to achieve the social-emotional supporting goal, Lilo uses 'VADER' which helps to understand the sentiment of the students' text. The tasks for Lilo can be categorized into 3 divisions which are reminders, reflections and social relationships and Lilo excelled in all the tasks. As a reminder tool, it informed participants about which task should be completed when and it also helped to schedule their works. 66.67% participants were pleased with the reminders by Lilo. As a tool for reflection, it acts like an advisor and answers different questions of the participants. Five out of nine participants expressed how Lilo helped them in their interviews and prepared them for other opportunities. As for the social relationship role, Lilo works as the broker between the student and the human advisor. Interacting with Lilo, helps the students to better formulate their thoughts before approaching the actual human advisor which improves their relationship with their advisor. Along with the benefits, there are few limitations mentioned in the article. The feedback was taken from a few participants who can not represent the majority of the students and the participants interacted with the chatbot with enthusiasm which might not be similar to other students. To sum up, the study describes design and development of a chatbot for college advising which is analyzed over data collected from a four week pilot study of four primary data sources.

According to Dinh and Tran[22] (2023), chatbots have the ability to transform analog services into digital and provide 24/7 assistance. With the help of AI and enough resources it can enhance customer satisfaction and decrease the workload of employees. Different domains like medicine, e-commerce, and education are already using chatbots due to its ability of automation, immediate response, and personalization. 'DialoGPT', 'Meena' and 'DALL-E' are a few examples of chatbots. Technologies like chatgpt are already providing generalized services but to ensure accurate responses in smaller domains it is necessary to provide domain-specific data to the model. This research tried to build an effective model for answering university related queries and tried to see if combinating various approaches can supersede the deep learning approach for low-resource languages like Vietnamese. To achieve its goal it introduces a new chatbot called 'EduChat' for the university domain. This chatbot model can assist students with university queries, admission formalities, academic details and also queries related to student life. The dataset for the model was created by surveying six academic advisors and two staff members of student affairs of HUFLIT university. That resulted in 6250 text messages which got categorized into 25 classes. They also used TF-IDF, tokenizer, and stop word removal for data preprocessing. Most interestingly this AI model uses 3 steps to understand the queries which are rule-based approach, machine learning approach and chatgpt approach. Firstly, the model understands the query using the NLP module and tries to categorize the query into one of the 25 categories mentioned in the table of rule-based approach. They used Definite Clause Grammar (DCG) as the rule based system. If the query is out of the scope of the rule based approach then the model uses the machine learning module to generate the response. The paper used Improved Random Forest (IRF), KNN, SVM, Decision Tree, Neural Net, Naive Bayes, LSTM and BiLSTM for the machine learning module. Surprisingly, IRF gets the best accuracy (97.69%) followed by BiLSTM (96.98%). Finally, if any query falls outside the threshold, the model passes the query to ChatGPT. Furthermore, 50 messages were passed to the NLP module where it successfully classified all the messages and assigned 1 out of the 25 categories to the text. A survey was conducted with 40 freshers of HUFLIT University students and around 90% of them marked the chatbot as convenient and useful. To conclude, most of the existing chatbots fail to present up-to-date information which the paper tried to resolve using EduChat using NLP and machine learning modules. The paper also intends to enrich the model with larger dataset and more categories for the rule-defined table to enhance the accuracy of the model.

Mukherjee et al.[25] (2023), addresses the complex problem of generating polite responses in chatbots, which is difficult to combine regular and polite sentences in a dataset. In this article, the researchers discussed a three-step solution: first, train a model to convert sentences into polite ones, then generate synthetic polite dialogue pairs, and finally train these two to further develop the dialogue model. In the first step, the authors used the BART model for making the neural sentences turn into polite ones which actually makes the model a politeness transfer model. Then this model helps to generate synthetic dialogue pairs which carry the contexts and their corresponding polite answers. After this for training the chatbot these synthetics pairs get used and later it makes polite responses based on the questions given. The dataset for this research is the DailyDialog dataset. It is an open-domain dataset which has 13,118 daily life based human-human talks. Then they start deleting the politeness markers from the polite sentences in the Enron email dataset, the markers are added to the output sentences, and the politeness transfer model is trained using the recently generated synthetic dataset. Subsequently, by merging the tagging and generation processes, they used the BART model to reduce the complexity of the earlier approaches to a single step. Following that, researchers used these artificial polite dialogue pairs to fine-tune a variety of pre-trained models, including GPT-2, DialoGPT, and Blender Bot. The results of these models were produced using both automated and human evaluations. The automatic metrics content similarity, BLEU score, and polite score indicate how well the model produces polite, cogent, and content-preserving responses. Experts were used to assign an overall rating inside five to the model's output regarding politeness, fluency, and coherence. The overall result from the proposed methods beats the performance of baseline models in generating more polite and meaningful coherent responses. The new model achieves a higher polite score, better content similarity and also the fine-tuned model responded with correct answers with politeness. BlenderBot was found to be the most optimally tuned model among all the models, exhibiting the most courteous and fluid responses. The overall result from the proposed methods beats the performance of baseline models in generating more polite and meaningful coherent responses. Finally, this paper shows a new approach to increase the politeness of a chatbot and it can be achieved by a straightforward end-to-end model and by not using complex methods. This research suggests that we can have very natural human-like responses in the future by making further new methods from the chatbots. It also highlights that politeness in HCI is very important and it helps to find practical solutions to make chatbots respond more politely.

Lee et al.[24](2023) built and reviewed a chatbot named PEEP-Talk. PEEP-Talk chatbot is aimed at improving the learning of English by providing practical consummate dialogues. While there are other chatbots including the likes to to Andy, Mondly, Speak Now, Duolingo, and Babbel and so on, the effectiveness of most of these chatbots is rendered limited by their inability to offer real free and genuine interactions to their users PEEP-Talk therefore uses the SITUATION-CHAT database for various situational chaos. It incorporates three key modules: A DialoGPT for natural verbal conversation, a context detector (CD) module using XLNet for feedback on the appropriateness of the user's conversation, and the grammar error correction (GEC) module with deep learning-based single model semblant to Seq to Seq. This system is on the web and it possesses an interface where learners can engage in conversation, provide feedback, require correction concerning grammars, and descriptions of certain situations. Incorporating with the aforementioned objectives, PEEPS-Talk's effectiveness may improve the learner's communication skills by enhancing the language practice. The SITUATION-CHAT context uses real-life life-like situational dialogues for the teaching of English as a foreign language.It includes broad domain categories (e.g., education, business and finance, arts and entertainment, etc.). g. ; videos , sciences, lectures, news) and limited segments (e. g. With well-defined contexts such as "hospitalization" or "auto accident" which can be simulated in the context of an organization (office, school). " This dataset is filtered for only English dialogues and collected from a Korean English parallel corpus. From a pool of 2,779, a demonstrated professional English teacher assisting chose 330 situations, adding up to 73,192 dialogue histories. To these, annotators elaborated guidelines that include limiting the description to a few words, and writing it in the first person. Training set: 248 situations, 16,298 dialogues, 65,192 utterances; validation set: 42 situations, 1,000 dialogues; 4,000 utterances, test set: 40 situations, 1,000 dialogues, 4,000 utterances. It ranges from traveling, shopping, the airport, hospital, etc. The models used include ConvAI2's baseline or Profile Memory while other enhanced models used include TransferTransfo and Lost In Conversation. Training included the use of a GeForce RTX 8000 GPU, a two-epoch 18-hour length cycle, and the Adam optimizer learning rate adjusted to specific values. The models, which had been set having 117M, 255M, 502M, and 774M parameters, incorporated beam search during decoding. The Context Detector (CD) was trained based on MRPC and CoLA datasets.

In order to transform customer service, Pandya and Holia [27](2023) introduce Sahaay, an open-source framework that makes use of LangChain, a unique Large Language Model (LLM). It suggests swapping your standard FAQs with chatbots—which are responsive and context-aware—that offer individualized, real-time service. The research illustrates the improved performance of the Flan T5 XXL model and shows how useful the chatbot is in a variety of applications, especially in educational institutions. It does this by using web scraping for data collection and complex embeddings for context retention. By using effective, automated interactions, the objective is to increase customer happiness and loyalty. The Birla Vishvakarma Mahavidyalaya (BVM) Engineering College website provided the dataset utilized in the paper. Here, Publicly accessible data is included, such as chat logs, product manuals, help forums, customer service FAQs, and details on associated companies. By using web scraping techniques, BeautifulSoup collected data that provided the necessary context for training and optimizing the Large Language Model (LLM) to provide accurate and contextually relevant responses. This study analyzes three Flan T5 language model versions developed by Google which are Flan T5 XXL, Flan T5 Base, and Flan T5 Small. Accuracy and context retention were best achieved by the biggest and most powerful model, Flan T5 XXL. Flan T5 Base offered a balance between performance and efficiency, but Flan T5 Small, which needed the fewest resources, had the lowest accuracy. Moreover, It was found that the Flan T5 XXL model satisfied the needs of the chatbot the best. The study produced "Sahaay," an automated customer service chatbot that utilizes Flan T5 XXL and LangChain models. Its open-source structure also makes it easier to integrate into a variety of customer support platforms, which improves client interactions' efficiency and personalization. Overall, the study supports the usefulness of complex language models for automating customer support, with Sahaay demonstrating visible benefits in raising customer happiness and improving operations.

Shahriar et al.[28] (2023) highlight the scarcity of rich datasets for Question-Answering (QA) in low resource languages like Bengali. This has motivated to create the first Bengali Question-Answer Generation (QAG) pipeline which consists of a question generation model, extraction model for answer span and bidirectional consistency filtering which will deal with inconsistency of the QA pairs. The paper tries to reduce human participation in building labeled datasets. The past initiatives to create Bengali QA dataset were using machine translation. Bengali-SQuAD and SQuAD_Bn are two examples of machine translation approaches but these datasets often fail to detect correct answer span based on the context due to their direct translation approach from high resource language to low resource language. To train the QAG model, the paper takes help from different datasets. SQuAD1.1 and SQuAD2.0 are translated where SQuAD1.1 adds over one lakh question answer pairs and SQuAD2.0 adds over half a lakh unanswered questions. 200 articles are taken for training and the validation set is used as testing data from the Bengali_SQuAD dataset. A portion of BARD (Bengali article classifier) and SQuAD_Bn is also used to enrich the 'BanglaQA' dataset. The context is first tokenized using NLTK tokenizer and then context, questions and answers are translated independently using the NLLB model. After that, the answer span is tokenized using UToken tokenizer and aligned using the fastText for correct vector representation. After all of preprocessing, the answer span extraction model and question generation model are trained by fine tuning BanglaT5, and bidirectional consistency filtering is done by fine tuning BanglaBert. To evaluate the created dataset BanglaQA, 2100 QA pairs are selected randomly and assessed by a group of 7 expert students on different criterias where BanglaQA achieves 98% on grammatical accuracy, 97% accuracy on context relevance, and 96% accuracy on consistency of QA pairs. BanglaBERT, XLM-RoBERTs, and mBERT are used on the BanglaQA dataset and BanglaBERT outperforms all by achieving 75.70 EM points and 86.14 F1 score.

Campese et al.[29] (2024) discuss the challenges of enhancing question-answering (QA) systems by improving the retrieval and ranking of semantically matched questions without generating new answers for the question. Solving these problems will be helpful for applications like FAQs, forum services, and QA caching systems. A novel unsupervised pre-training(PT) method is used for knowledge distillation from a basic question retrieval model.To teach how to rank questions based on their relevance to the query, a new PT task is created, and this method generates an unsupervised dataset of 18 million examples using a question-answer database (QADBS) and also modifies the rank to simulate the ranking objective. The dataset utilized in the paper consists of Quora-Match, which has 200,000 question-question-answer triplets for binary classification; SemEval 2016, a question ranking dataset for community QA; and QRC, a question ranking resource with 15,000 queries each associated with 30 question-answer pairs. On the QRC dataset, various transformer-based architectures, such as the MiniLm-12L-v2 and Deberta-v3-base models, were used for fine-tuning. The paper also compares a variety of PT techniques, including distillation methods, sentence-level objectives, Random Token Swap (RTS), and Masked Language Model (MLM). The techniques achieve excellent results on QRC and Quora-matched datasets, with the method improving Precision by 1.05%, Map by 0.43%, and MRR by 0.75%, indicating a significant improvement while establishing effectiveness and stability. Combining PT reranking and retrieval models results in synergistic benefits and, over time, improves model performance. Hence, combining different distillation methods with new pre-training(PT) methods we see a very promising result from the article. For instance, on the Quora-match dataset the ROC-AUC score increases by 0.28%. For improving the performance of the Question answering system the specialized pre-training task works as a power house. It works as a strong foundation for future work in improving system ranking and retrieval of questions.

Chapter 3

Work Plan

Pre-Thesis 1, Pre-Thesis 2, and the final thesis comprise the three stages of the thesis project. In November 2023, team formation was our first assignment. We were always searching for others interested in artificial intelligence and natural language processing, and we came across one another. We began searching for our supervisor in December 2023 and waited for prerequisites to be finished before beginning Pre-Thesis 1. We decided on our topic after consulting with our esteemed supervisor in January and February of 2024, and we subsequently registered for Pre-Thesis 1. Our whole thesis journey is illustrated in the summary diagram below, which includes the steps we took and the deadlines for each task:

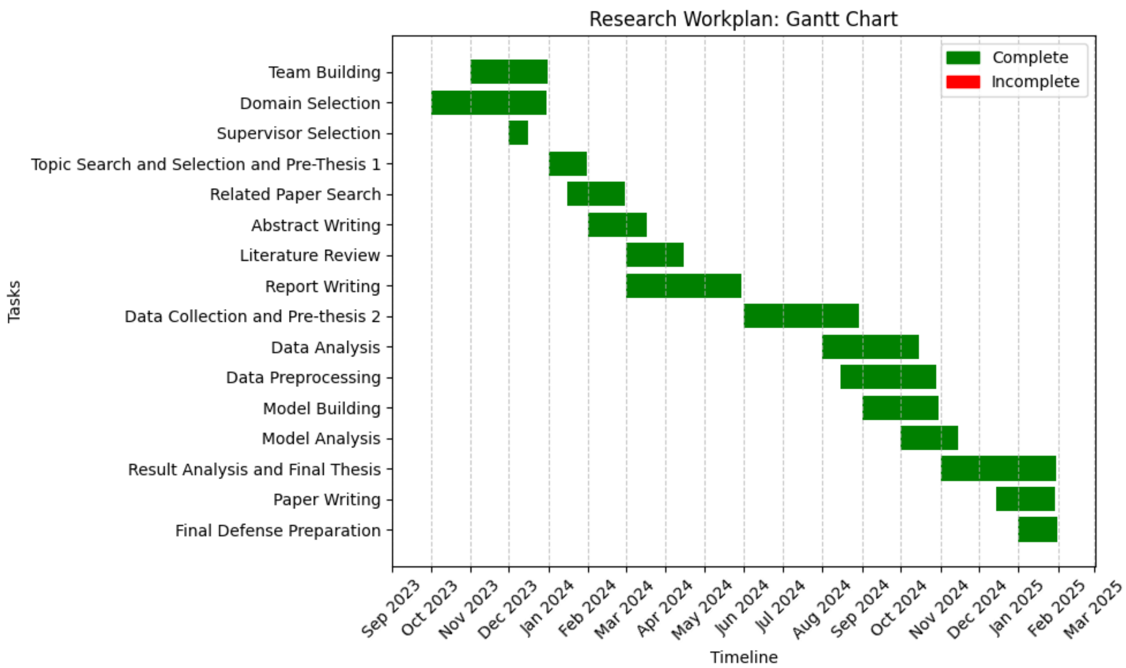


Figure 3.1: Gantt Chart

After registering for Pre-Thesis 1, we started getting ready for work by reading articles related to our topic and the field of natural language processing. Our research paper's abstract was later created. After submission, we moved on to the literature review and report, which is how Pre-Thesis 1 came to an end. We had only begun working on Pre-Thesis 2 when we turned in the report on June 7, 2024. A number of tasks were completed, including data collection, preprocessing, analysis, model evaluation, training, and testing. Additionally, a paper was sent to the thesis committee. By November 15, 2024, all Pre-Thesis 2-related activities were concluded. We started working on our Final Thesis after finishing Pre-Thesis 2, where we evaluated and analyzed the information we had gathered. We have completed the paper and now we are ready to present our final thesis.

Chapter 4

Dataset

4.1 Dataset Description and Collection

For this study, we have created the question-answer corpus for training the Question-Answering Language Model targeting BRAC University students. The dataset contains a total 1866 data points typical to students' topics and questions. The focus was to address all the questions a student might have during his whole academic journey from first day to the final day including thesis/internship related queries. All the data points are in the English language. The data points were collected from different sources focusing on common queries of a student's daily academic life. The data points were carefully curated from three primary sources:

Google Sheet: Every semester BRACU provides a google sheet for CSE routine. We can find different information regarding section wise course routines including faculty of a section,class day and hour, class room no etc. Additionally, faculty list and coordinator information can also be found from that sheet. These data points are very much important for students in their daily routine. Section wise course information provided a total 529 data points. Faculty list provided 232 and the coordinator sheet provided 76 data points in our corpus. Another sheet provided 35 data points regarding information about pre-requisites of different courses.

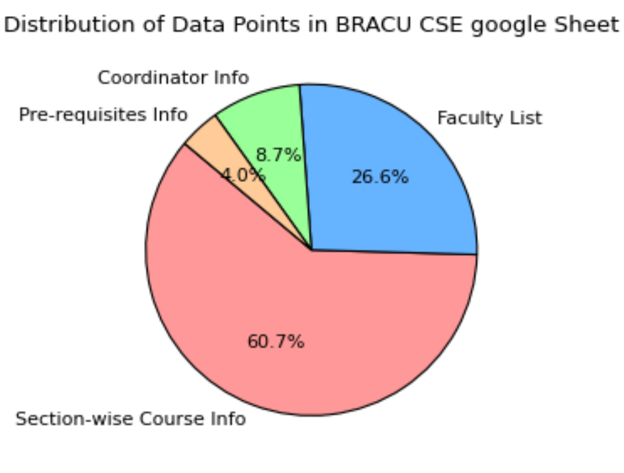


Figure 4.1: Pie chart showing distribution of data points from google sheet

BRAC University website: Information on general and specific common concerns regarding life on the campus, learning facilities, and other student-related services were obtained from official web sites. An example can be:

"Question": "What is the grading scale utilized by BRAC University?":,

"Catergory": "Exams and Grading",

"Answer": "BRAC University uses a grading scale based on the Grade Point Average (GPA) system, where letter grades correspond to specific grade points. The typical grading scale is as follows: A: 4.0 (Excellent), A-: 3.7, B+: 3.3, B: 3.0 (Good), B-: 2.7, C+: 2.3, C: 2.0 (Satisfactory), C-: 1.7, D+: 1.3, D: 1.0 (Pass), F: 0.0 (Fail)"

Total 562 data points were collected using this approach.

Facebook groups: Online forums, in which current students or alumni engage and discuss university experiences were used to identify frequent asked and answered questions. These groups offered valuable information on actual student concerns as well as your everyday comings and goings. This approach contributed 432 data points to our corpus.

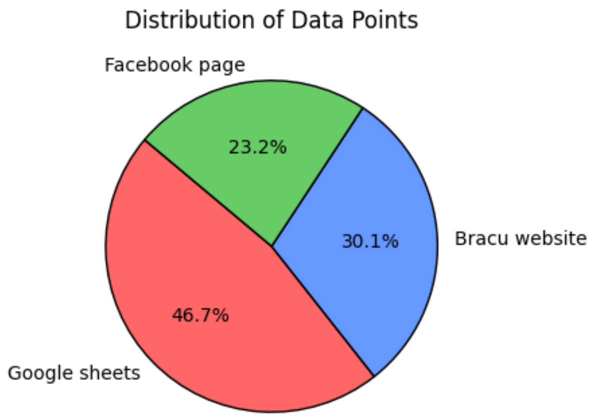


Figure 4.2: Pie chart showing distribution of data point sources

The dataset collected from facebook groups is divided into 14 different categories that reflect major spheres of students' learning process and their programs' services. Such categories include advising, academic progress, exams grading, payment financial aid, library, international student, technical support, policies procedures, TARC, campus, extra curricular activities, admission, transportation, and thesis internship. All the categories are supposed to correspond to a particular domain of students' interests, making the model usable in terms of any aspect of a university life.

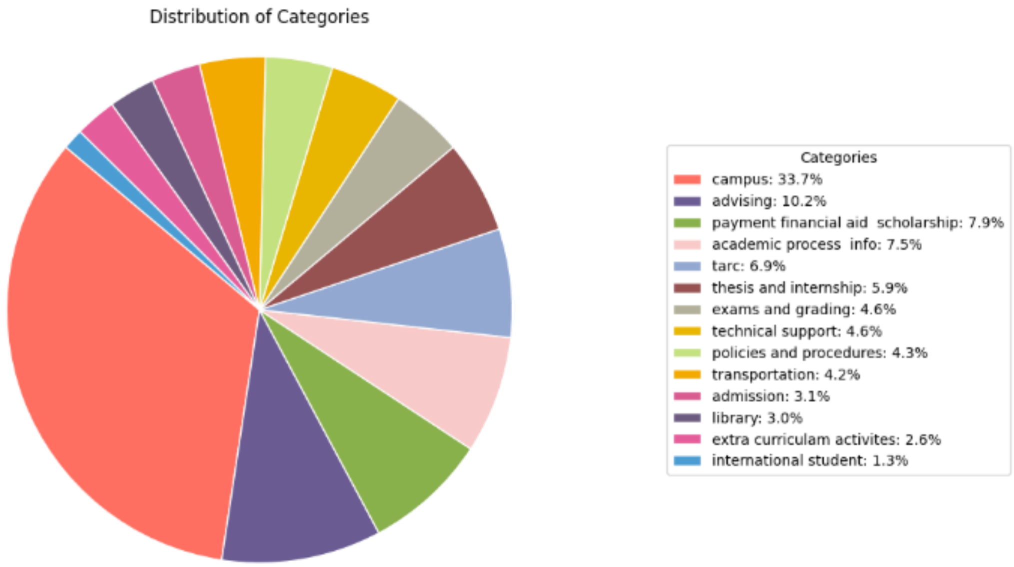


Figure 4.3: Pie chart showing the distribution of categories

The figure 4.2 shows that most of the queries are related to campus and second most is about advising.

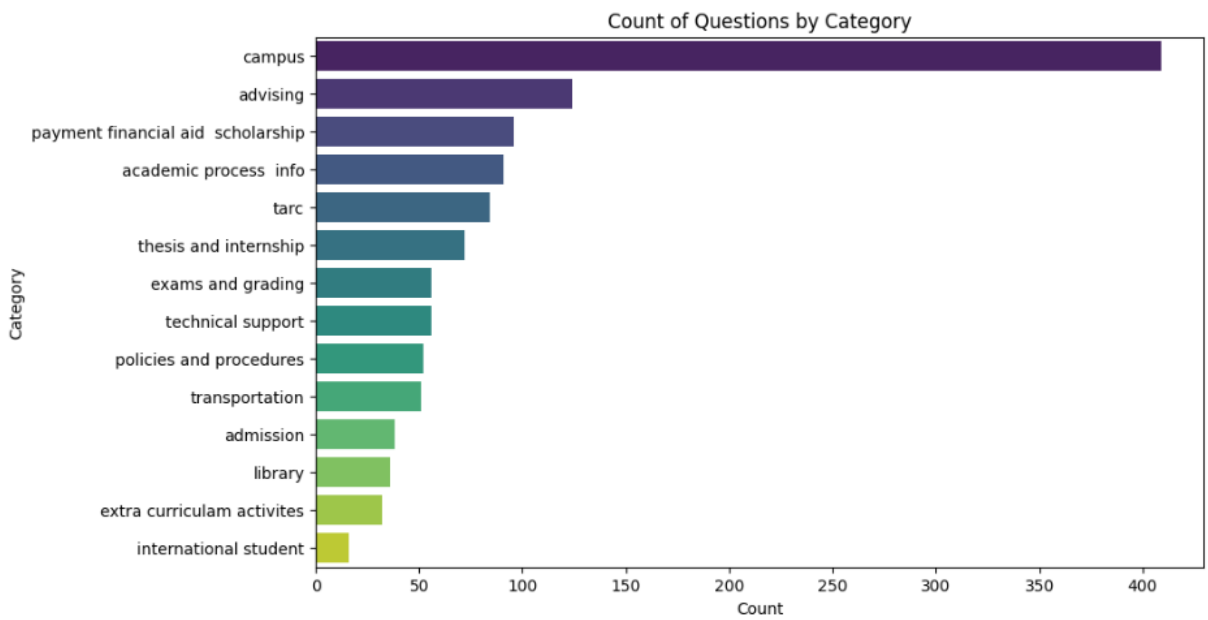


Figure 4.4: Horizontal bar chart showing the number of questions in each category

The number of questions in each category are represented in the figure 4.3. The "campus" category receives the largest number of nearly 400 questions and is ranked first as the most preferred topic by students regarding various aspects of campus including location and facilities. Other popular topics include 'advising' and 'payment financial aid scholarship', identifying that the students need information on the academic advising procedures and types of scholarships and financial assistance. Other related subtopics include; "academic process info", "TARC", "thesis and internship" confirming the conclusion that students frequently seek information concerning an academic process, the TARC campus and requirements for internships or thesis. Farther down the chart with fewer questions, there are categories like 'library,' 'extracurricular activities,' and the 'international student.' They possibly attract less questions because they are more niche or there are fewer students that are curious about the topic.

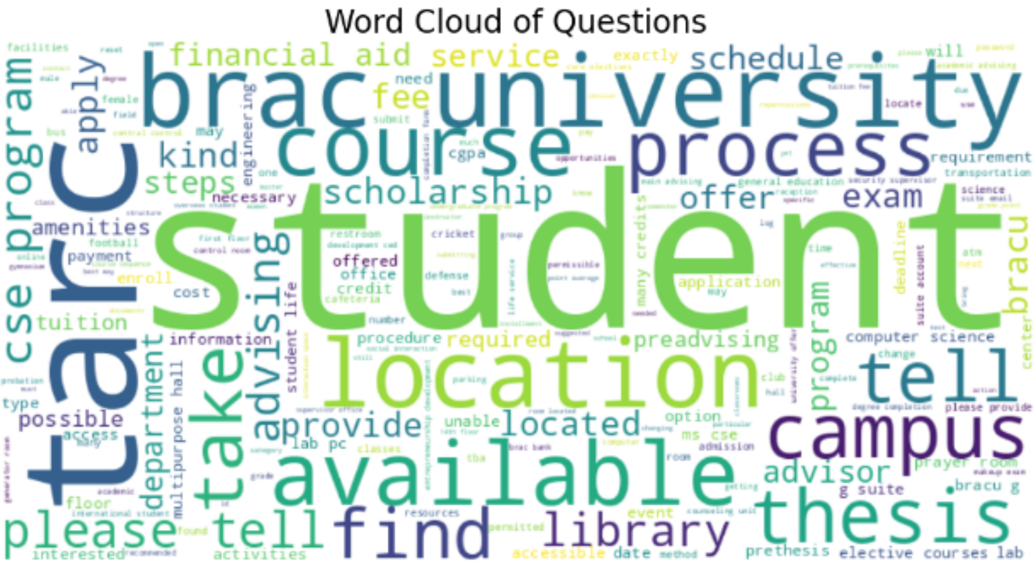


Figure 4.5: Word Cloud of Questions

The following 4.4 figure represents the most used terms in the questions dataset. The size of each of the words reflects the frequency of the word used in the questions. Everyday terms like 'student,' 'tarc,' 'university,' 'course,' 'location' and 'thesis' confirm that a substantial number of the questions are directed at getting information pertaining to student issues, the residential campus, details of the course, and processes of the university. That is why using such terms as "financial aid," "scholarship," "library," "process" indicates that students also often seek information about procedures, facilities, and funding.

Two types of questions are included in our dataset which are static and dynamic. The answer of the static type questions does not change usually. For example, total credits required for degree completion will not change frequently. On the other hand, dynamic type questions have different answers every semester like academic schedule, offered electives etc. For this phase of our thesis, we have mixed the two types of questions for comprehensive analysis.

The questions and answers given range in length and in difficulty level; some of them involve factual information being shared, others share posts-experience advice. For instance, questions vary from basic factual ones as in "what eating places are there to be found in TARC?" from descriptive to exploratory problem-solving questions like "How should students time allocate themselves at BRACU?" Answers offer short but detailed responses to the question posed, and involve giving directions or running descriptions in accordance with the kind of question under discussion.

4.2 Data Preprocessing

The data preprocessing steps contain a number of techniques which are cleaning, tokenizing, lemmatizing and transforming question and answers. As you already know, our dataset consists of pairs of questions and answers and our target is to make sure that our data is not unstandardized and ready to get input into machine learning models. To avoid potential inconsistencies and noise in the raw textual data we designed the workflow.

4.2.1 Dropping Null Values

For dropping the null and unnecessary data firstly, we loaded our datasets into Pandas DataFrame. It helped us to make changes easier and also made easier to analyze. Then, we checked if there's any null values in our question-answer pairs collected from facebook group. If there were any missing values like "questions" or "answers" we removed it. By doing this the data entry was fully completed. After dropping null values, the question-answer pairs had 423 pairs remaining out of total 432. It was very important to clean up the data because incomplete data would have caused biases and errors during the model training process.

4.2.2 Text Normalization (Lowercasing and Removing Punctuation)

After dropping null values, we started our text normalization. In the normalization part, we converted all the text of "questions" and "category" columns into lowercase to make sure variation in capitalization does not lead to different tokens for the same word. Example: "Chatbot" and "chatbot" will have the same context after applying it. We also deleted the punctuation marks as they are not crucial for our tasks and could create inconsistencies if left unprocessed.

4.2.3 Tokenization

Tokenization method was applied to split the questions into word tokens. Tokenization helps to break down the raw text into smaller components so that it is easier to analyze. It is also a key part of Natural Language Processing(NLP). So, we turned each question into a list of words, which will be processed in our further steps.

4.2.4 Lemmatization

After tokenization, we used "WordNetLemmatizer". It was applied for lemmatization. Lemmatization is used to bring back the words in their actual form and it helps to bring consistency across various variations of words. For instance, if words like "Adding" and "Added" are there in the dataset, they both would be reduced to "Add". We used this step to decrease the vocabulary size and to ensure that semantically similar words are handled the same in the machine learning models.

4.2.5 Numerical Transformation (Using Keras' Tokenizer)

Once tokenized and lemmatized, we stepped our foot into numerical transformation. Using this method, the words in the questions are mapped to unique integer values. Machine learning models need numerical values to train, they cannot process raw data text directly. So, we are turning them into their numerical representations. The tokenizer assigns each word an distinct integer after it gets fitted on the text. Later it converts all the textual data into sequences of numbers. The model could take these sequences as inputs.

4.2.6 Sequence Padding

Machine learning models require uniform length input means they need input data of the same length. For that, sequences are padded to a fixed length of 50 words. Padding guarantees that all input sequences are of the same number of elements. Padding not only ensures the same number of elements but also enables efficient batch processing and avoids error at the time of model training. Zeros are added to the end of a sequence when it has fewer than 50 words because it has to reach the mandatory length. It also ensures all inputs are standardized and analysisable.

4.2.7 Exploratory Data Analysis (EDA) and Word Class Visualization

The structure and distribution of the data are understandable by this preprocessing technique named exploratory data analysis. The below figure shows the distribution of different question categories as it is providing an overview of the data composition. Furthermore, we created a histogram to analyze the frequency of questions in each category. Another visualization method word cloud is also used to highlight the most frequently occurring words in the dataset, which show a high level overview of key terms in the questions. This is helping to point out the most common topics of the dataset.

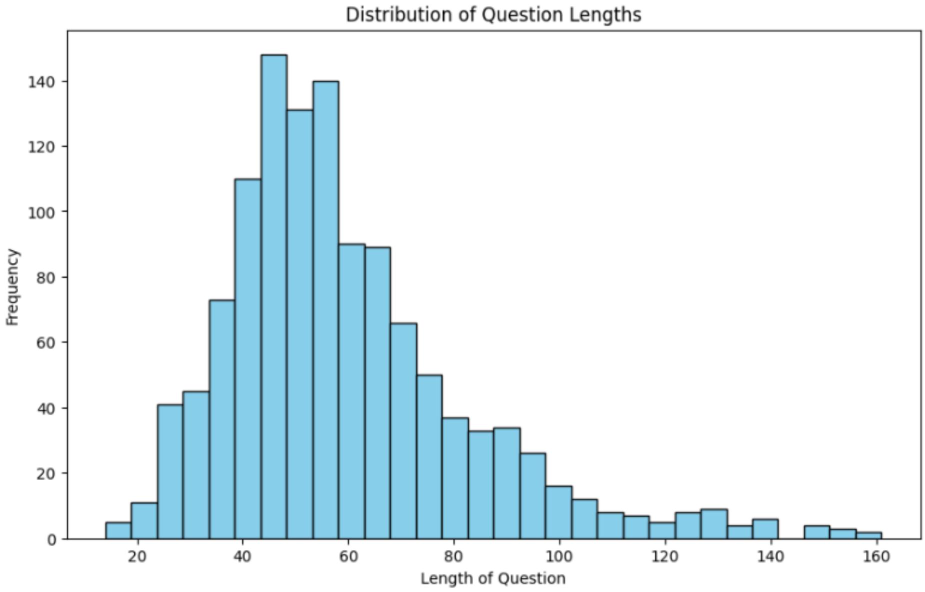


Figure 4.6: Histogram showing distribution of question lengths

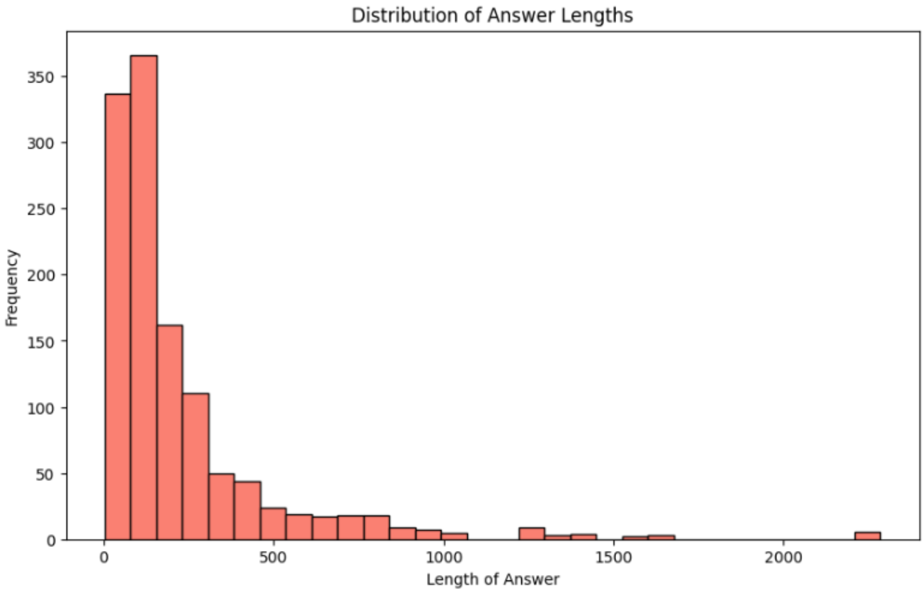


Figure 4.7: Histogram showing distribution of answer lengths

These 4.5 and 4.6 figures show that most of the questions and answer lengths are relatively short but few are relatively more descriptive than the majority. While most questions range between 40 to 60 words, most answers fall between 0 to 500 words. This indicates that the answers are mostly short and precise, although for some descriptive questions there are answers length of more than 2000 words.

4.2.8 RecursiveCharacterTextSplitter

The RecursiveCharacterTextSplitter is a tool used to divide large files into smaller more manageable bites, commonly characterized by characters, paragraphs, or sentences. In our case, it splits documents into 1000 characters with 200 character overlap, which leads to the ability to have context information from the previous chunks as well. This preprocessing step is very important because documents may be so large that they act as a nuisance to the actual retrieval by the model. It also allows for easier storing of the text in a vector database as the text can be segmented and retrieved and processed in smaller magnitudes enhancing a drastic enhancement of the entire retrieval and generation pipeline.

4.2.9 Google Generative AI Embeddings (embedding-001)

Google Generative AI Embeddings is one of the generated models that are used in transforming textual data to vectors. Facilitates semantic search by converting text into vectors providing the best material for comparative analysis for the given text type with large sets of documents.

How It Works

The Google Generative AI Embeddings model operates within a way that parses text from PDFs, and other documents, into something known as embedding. These embeddings are quantized representations of meaning and context of texts which allow the system to find the 'similarity' between different texts. This process facilitates the scanning process of the system in that it enables it to identify the most appropriate document segments to respond to a given query by a user. By adopting these embeddings, the model ensures that only the most relevant information is given for a particular response.

Why Should It Be Used for the BRAC University Chatbot?

- **Efficient Document Retrieval:** Since the university deals with several documents, for instance, course catalogs, and policies incorporating these documents makes it easier and effective to retrieve them each time students pose certain questions.

- **Contextual Relevance:** The embeddings enable the chatbot to find the nearest semantically related data, thereby always providing the correct data from the official sources of the university in its responses.

4.2.10 sentence-transformers/all-MiniLM-L6-v2

The transformers based model sentence-transformers/all-MiniLM-L6-v2 is used for creating vector embeddings from textual documents. This model is highly useful to generate embeddings which is required for finding out semantic similarities between differently structured sentences. In our case, it can help to create embeddings for both user query and dataset so that the semantic similarity between them can be used during the retrieval process.

4.2.11 Chroma DB

Chroma DB is a vector database which can store vector embedding of large dimensions. It is similar to FAISS but it can store metadata and it can dynamically update embeddings. This metadata can be sent to the LLM to provide rich context and enhance relevant answer generation.

4.2.12 Facebook AI Similarity Search

FAISS is used to store document chunks as vector embeddings that allow for its similarity search. Following the slicing of the text into segments, Google Generative AI Embeddings are used to generate vector embeddings. Such embeddings are then used by FAISS to construct an index to enable efficient approximate nearest neighbor (ANN) search. When a user enters a question, FAISS finds a similar vector embedding of the question and then goes through the documents and pulls the most semantically related chunks. These chunks are then fed to the model to produce an answer which renders FAISS as crucial for efficient retrieval in large datasets.

4.3 Data Ingestion Pipeline

This section talks about the design and implementation of the data ingestion pipeline which is used to extract data from multiple CSV files and webpages. Finally, load them into a relational database and eventually into a vector database. The figure 4.10 displays the whole workflow of this section.

4.3.1 Overview

The purpose of this data ingestion pipeline is two-fold:

Structured Data Storage: A relational database provides structured data storage for raw tabular information including questions and answers along with course schedules and faculty data and many more. It is specially useful to support database updates and modifications and maintain auditability.

Semantic Search and Retrieval: Repurposing the stored records with vector embeddings creates possibilities for Chroma to provide advanced searching and question-answering capabilities.

The system achieves compatibility with both standard SQL database queries along with complex natural-language queries by dividing database management from vector embedding operations.

4.3.2 Database Creation and Population

CSV: The database(SQLite) creation step includes creating tables for each CSV file. This is because every file have different structures (e.g. the CSV file containing informations about faculty list contains 6 columns which are Initial, Name, Designation, Status, Room and Email on the other hand the CSV file containing information about course prerequisites contains 3 columns about Course, Pre-Requisite and Full Chain Pre-Requisite) and due to this the different tables also have different schema. While populating the tables, each row stores the time so that during data ingestion we do not need to ingest the same data every time. Data ingestion timestamps being added automatically to new rows during record insertion. The timestamps added to each record become essential markers needed for subsequent stages to recognize new and updated records.

id	Course	Prerequisite	Full Chain Prerequisite	Timestamp
1	CSE111	CSE110 (HP)	CSE111–CSE110	2025-01-26 14:32:30
2	CSE220	CSE111 (HP), CSE230 (HP)	CSE230–CSE111–CSE110	2025-01-26 14:32:30
3	CSE221	CSE220 (HP)	CSE220–CSE111–CSE110	2025-01-26 14:32:30
4	CSE250	PHY112 (SP)	None	2025-01-26 14:32:30
5	CSE251	CSE250 (HP)	CSE250	2025-01-26 14:32:30
6	CSE260	CSE251 (HP)	CSE251–CSE250	2025-01-26 14:32:30
7	CSE310	CSE370 (HP)	CSE370–CSE221–CSE220–CSE111–CSE110	2025-01-26 14:32:30
8	CSE321	CSE221 (HP)	CSE221–CSE220–CSE111–CSE110	2025-01-26 14:32:30
9	CSE330	MAT216 (HP)	MAT120–MAT110	2025-01-26 14:32:30
10	CSE331	CSE221 (HP)	CSE221–CSE220–CSE111–CSE110	2025-01-26 14:32:30

Table 4.1: Example of populated table in SQLite dataset

Webpages: The system starts by doing URL retrieval from a text file while validating file formats. The system uses trafilatura to parse HTML responses and extract text content by discarding scripts and styles from each URL request that happens asynchronously. SHA-256 hashing acts as the tracking mechanism for modified extracted content. The system performs hash comparison against its stored records to check for updates. The system does not require a new entry creation once it determines that the content remains unchanged to avoid duplication. A new entry is created while maintaining historical version data for analysis. The program utilizes a local SQLite database to maintain scraped material and linked URLs to supply quick retrieval and a structured file system. Every version of web page content has its own individual storage space which enables future access for analysis along with maintaining a versioned dataset. A data-ingestion script requires execution through an alert to process newly inserted content before it can be indexed embedded or analyzed semantically. The combined method produces data acquisition which seamlessly integrates change identification with effective storage delivering uninterrupted monitoring along with information renewal.

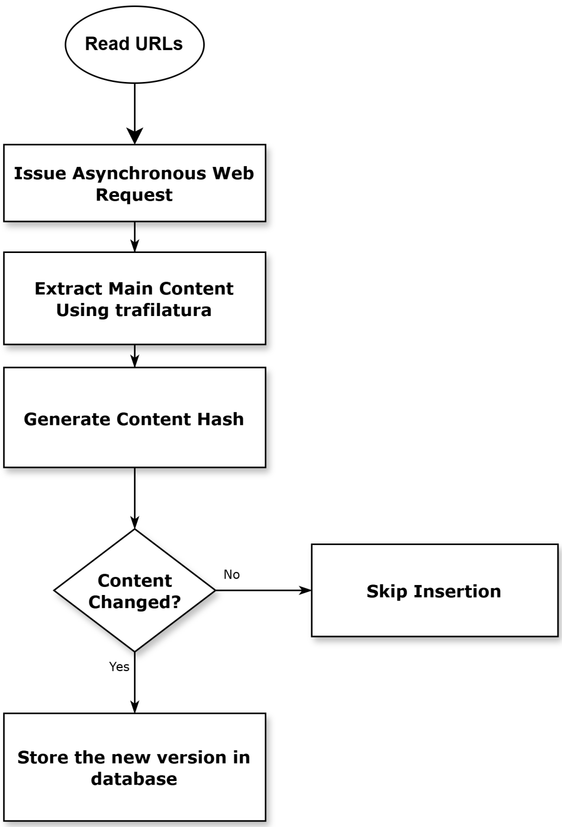


Figure 4.8: Workflow of fetching webpages

4.3.3 Extracting from database and store in vector database

After populating the database, the next is to ingest the data into the knowledgebase. We have used Chroma DB as our vector database. Steps include:

Accessing the Database: A new SQLite connection opens to access the same database. The database process selects specific fields from each table database. For example, with each query the EnglishQA table automatically retrieves the Question field in conjunction with the Answer field and the automatically generated Timestamp field.

Incremental Ingestion Using Timestamps: The script maintains a dictionary which tracks timestamp values associated with each table. For previously ingested tables the pipeline only retrieves data points when the Timestamp value exceeds the last ingested timestamp threshold. Otherwise, it ingests all rows. The optimization system avoids duplicate embedded records and makes possible incremental updates of information.

Text Generation from Records and Chunking: The script generates textual summaries which process each entry. The vector embedding models function using textual inputs. The system generates several short documents from each document entry to store distinct pieces of information data (such as faculty and scheduling information and email communications). By using smaller segments the system provides enhanced granularity in embedding which leads to superior specific query search accuracy.We use RecursiveCharacterTextSplitter to make chunks for any large document. This is necessary since large documents are memory intensive and chunking can help to make processing of these chunks faster and it also facilitates retrieval performance.

Embedding Each Document: Each textual segment is sent to the embedding model in order to embed them through custom embedding methods which utilize the SentenceTransformer model. A sequential processing system transforms text documents into numerical vectors. Embeddings maintain semantic consistency which makes later similarity computation possible.

Storing in Chroma (Vector DB): Vector embeddings and metadata labels such as document ID, table names are stored in a Chroma collection. The ingestion script conducts a pre-insertion check for documents with identical IDs then executes entry deletion for outdated material followed by new updates. The vector database contains only one data representation of every unique record.

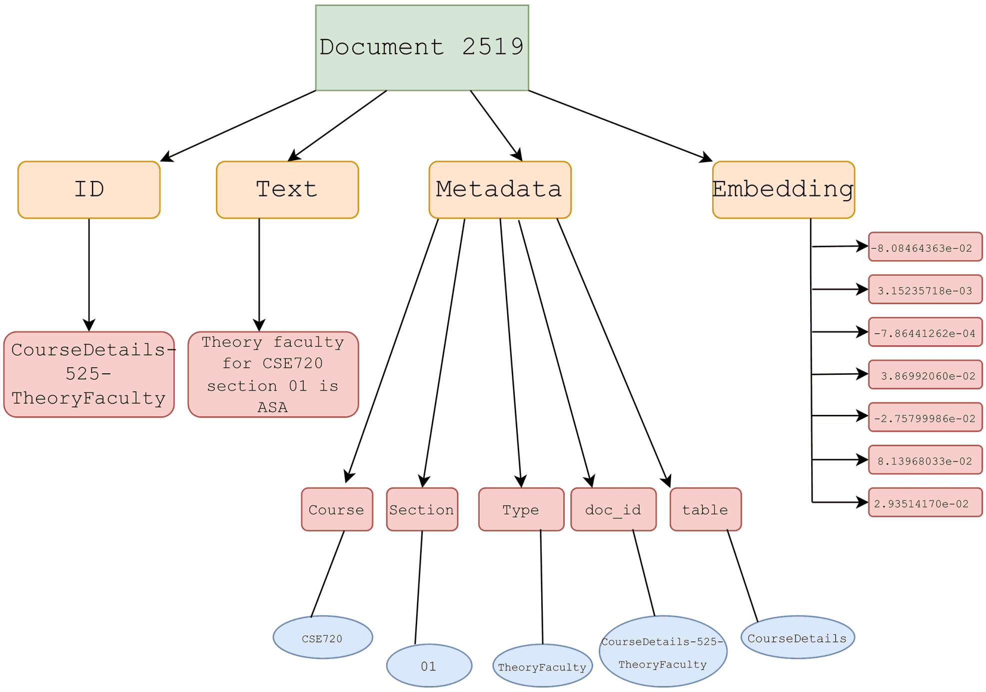


Figure 4.9: Representation of unique record stored in Chroma DB

4.3.4 Advantages of this Data Ingestion Pipeline

Incremental Updates: The pipeline takes timestamp updates to limit its data consumption only to new or modified data that decreases computation time and prevents repeated embedding generation.

Granular Embedding: Data splitting at the record level results in improved search precision through separated textual segments. Users can input demands regarding record subsections (example: "theory faculty for CSE101") to obtain the system's most suitable data segments.

Maintaining a Single Source of Truth: The SQLite database receives data from users through validated structures yet users benefit from Chroma's flexible semantic search system which omits repetitive detail storage.

Scalability: Extra CSV files or new tables merge with the current system after straightforward adjustments to processing. The vector database retains its fast retrieval speeds after scaling or sharding operations as your data volume grows.

4.3.5 Workflow of Database Population

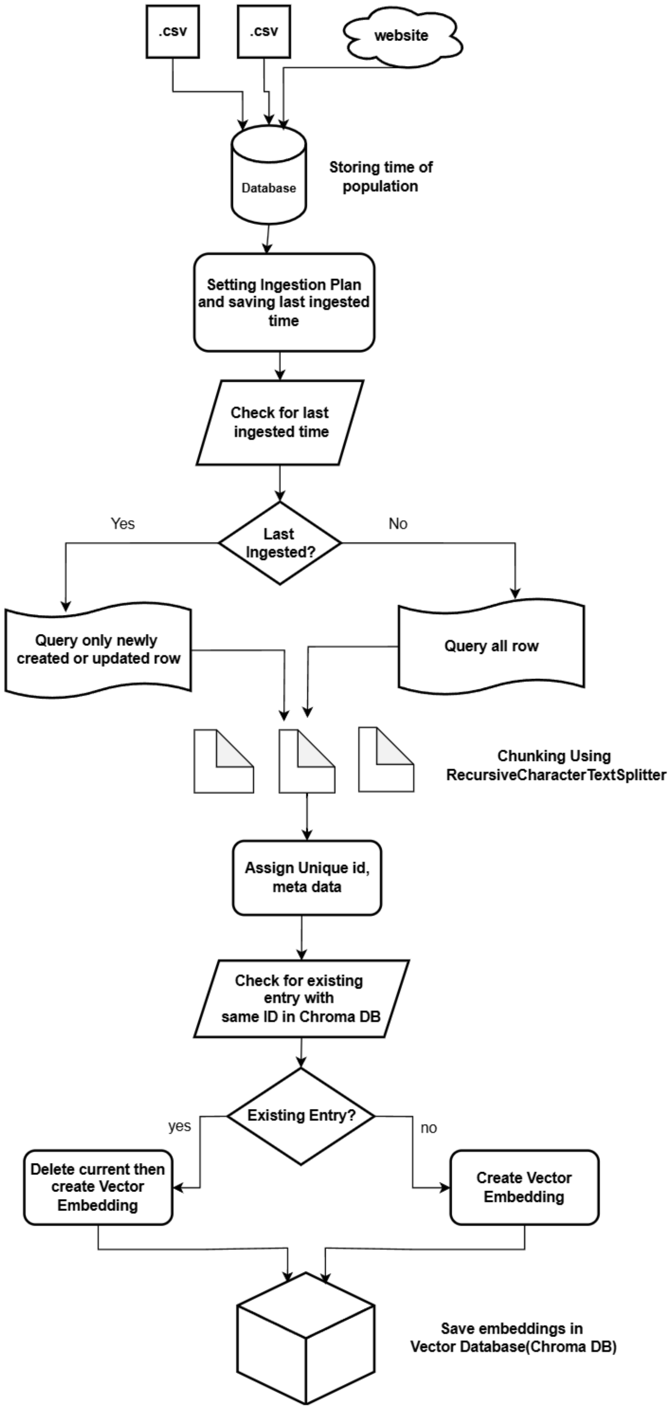


Figure 4.10: Workflow of Database Population

4.3.6 Workflow of Pre-Thesis 02

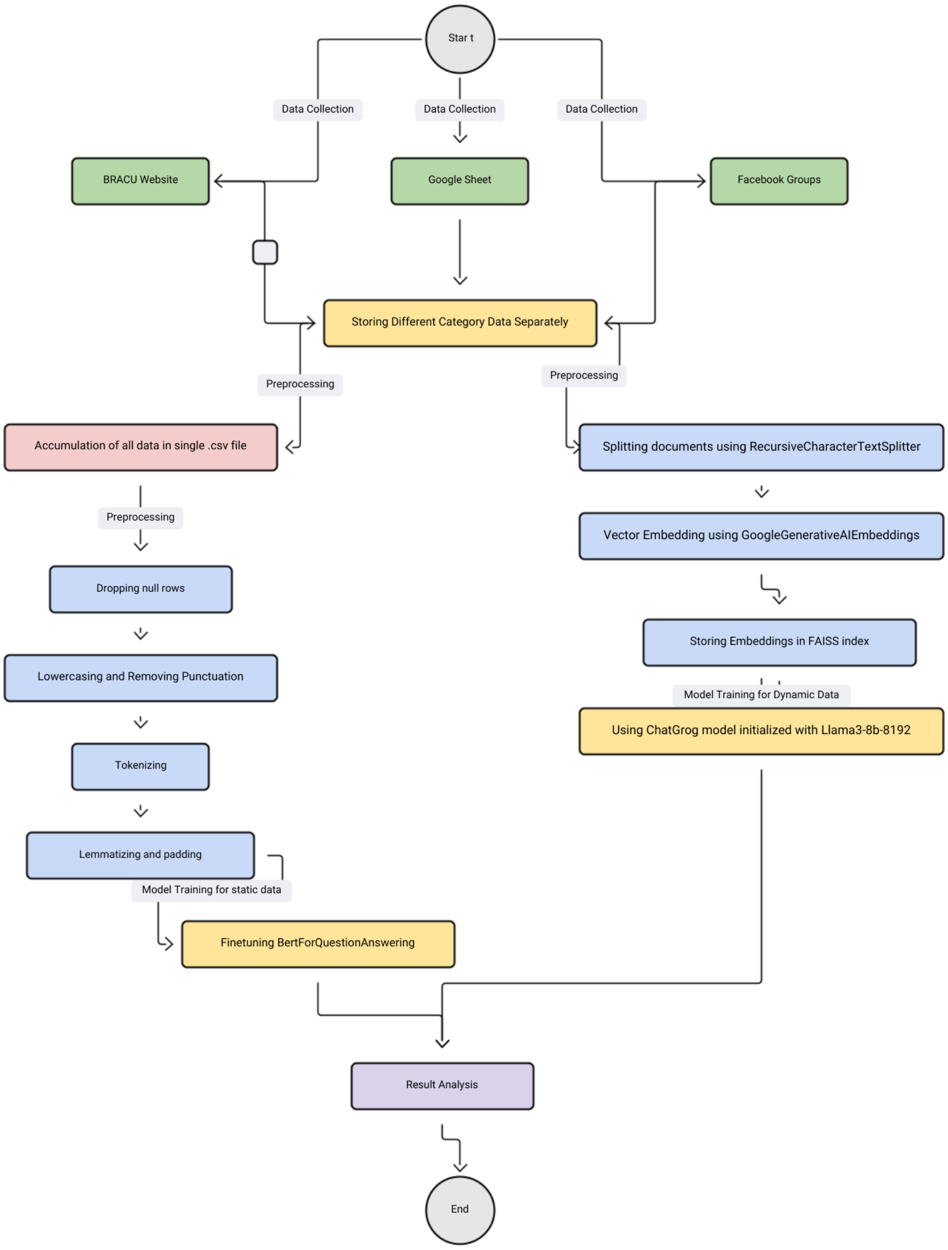


Figure 4.11: Workflow of Pre-Thesis 02

Chapter 5

Model

5.1 BERT

5.1.1 Description of BERT

As part of the research, we used a fine-tuned BERT base model to make the system more accurate and better at doing certain jobs. In natural language processing (NLP), BERT (Bidirectional Encoder Representations from Transformers) is a very important model. As per the main objectives, it uses both the context of a word from the left side and the right side simultaneously of all the layers. At 12 transformer layers, 768 hidden units, and 12 attention heads, BERT base is the small and light version of BERT compared to the BERT large. But still performs a great job on many NLP tasks.

For our research, we fine-tuned the BERT base model on a specific dataset to enhance its performance. Fine-tuning involves taking a pre-trained model and training it for a few additional epochs so it can adapt to the characteristics of the new dataset. This approach allows the model to leverage the vast knowledge it has gained from its pre-training on general corpora, such as books, Wikipedia articles, and other large text collections, while fine-tuning optimizes it for the target domain. In this case, fine-tuning BERT on a specialized dataset allowed the model to better understand the nuances of the task we were addressing.

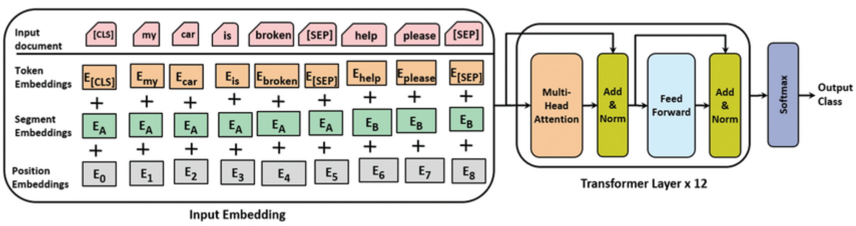


Figure 5.1: A representation of the BERT model architecture.

One of the best things about using BERT base for fine-tuning is that it can understand the meaning of words in connection to the text around them very well because it can capture the two-way context of words. It is this very important for such jobs as the answering of questions as this involves contextual understanding and the intricate connections between concepts. As we modified BERT using our dataset slightly, we obtained improved task-oriented outcomes and assisted the model in gaining a better understanding of its environment. The given model was helpful within this system because it could tackle deep language challenges and was finetuned based on the given topic data.

5.1.2 Architecture of BERT

The design of this BERT based question answering model starts with the First Layer Input where the input of the text such as question or Statement is converted to Token ID, Attention Mask and Token Type. The tokens are then given to the embedding layers where the token IDs are converted into dense vector representations. Positional encodings are incorporated to explain the manner and position of the words within the sequence.

The subsequent one is the Transformer Encoder containing the 12 BERT Layers. Each layer performs multi-head self-attention to get an understanding of contextual relation between words and this layer is followed by a feed-forward network using layer normalization and residual connections. There is the final encoder layer, which goes directly to the Question-Answering Head: In this module, it is required to apply a linear layer to predict the start and end samples of the answer in the given input. Last, depending on the phase of the program, if the phase is training the model provides Cross-Entropy Loss of the model weights to improve the model and if the phase is inference, the program provides the model's Start/End Token Positions that contain the desired answer.

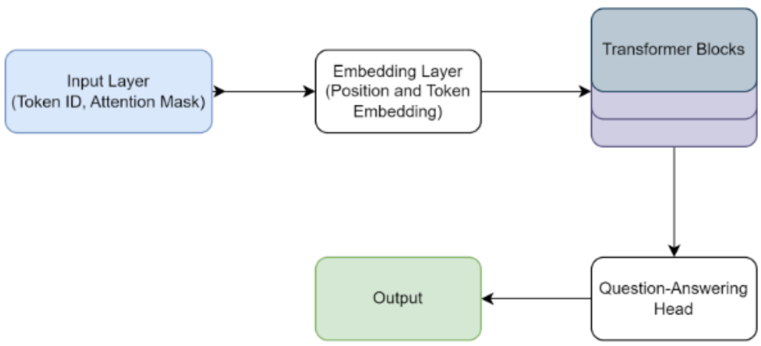


Figure 5.2: Layers of the BERT model architecture.

5.2 Llama

5.2.1 Description of Llama

Llama3-8b-8192 is a variation and it is a large language model that aims at providing responses similar to that of humans to queries made in natural language. It is perfect for handling tons of text-heavy data, and its responses are intelligent, coordinating with the context which makes it suitable for applications that demand natural language understanding and interfaces.

How It Works

Llama is used within applications through the Groq API which allows for using the language understanding of the model at its core. When a user posts a question, the system utilizes a template, the ChatPromptTemplate, to inform the model regarding the input it has received. That question is stated within this template to assist the model concerning key components. The system then finds the documents from embedded texts using similarity search with FAISS to obtain interesting chunks. These are the numerical mappings which indicate the meaning of the document to enable the model in computing a right response. Based on the input question and the collected document portions, the model generates a logical continuation that resembles a human response; thus, it is suitable to provide substantive answers with maximum reference to the context.

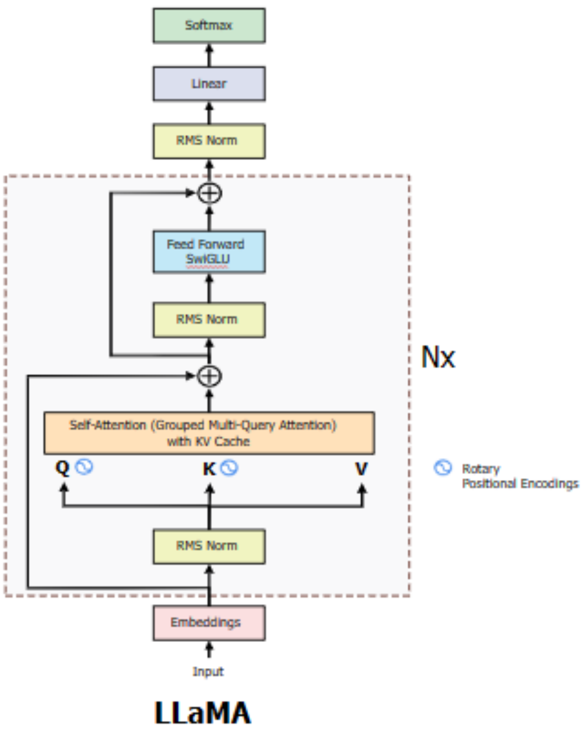


Figure 5.3: A representation of the Llama architecture.

Why Should It Be Used for the BRAC University Chatbot?

- Natural Interaction: Llama improves the overall functionality of the chatbot thus making it possible for the chatbot to respond to as many questions which a student may wish to ask including admission details, scholarship information and even the policies of the universities.

- Contextual Understanding: Since it deals with particular document portions, the model guarantees that answers are semantically correct; in this case, the answer stems from university-related documents instead of plausible but unrelated results.

5.2.2 Retrieval-Augmented Generation (RAG) Architecture

In our study, both the Retrieval-Augmented Generation, and the LLaMA 3 8B models were employed to improve the model's performance of generating accurate contextualized responses. What RAG does is it incorporates the ability of language models that have already been trained with an outside retrieval model. This lets the model dynamically get relevant information from a big collection of documents before it makes a response. This two-part system lets the model not only write text that makes sense, but also answer questions that are factually true and useful by using outside information about the real world.

The design is made up of two main parts: the generator and the retriever. Since, there are usually two parts in this kind of model, namely Retriever, for this reason, has to go and look for papers in some other database of outside knowledge and pull out the papers that match the query that was sent. High level of retrieval techniques are employed to search for the right papers to supply the generator with, which is supplied with papers most relevant to help it in its work. The generator in this case is the LLaMA 3 8B model which combined the information from the papers presented and arrived at the final result. The papers that were retrieved are used to help the generation process ensure that the outputs that are given out are not only accurate but also pertain to the case scenario.

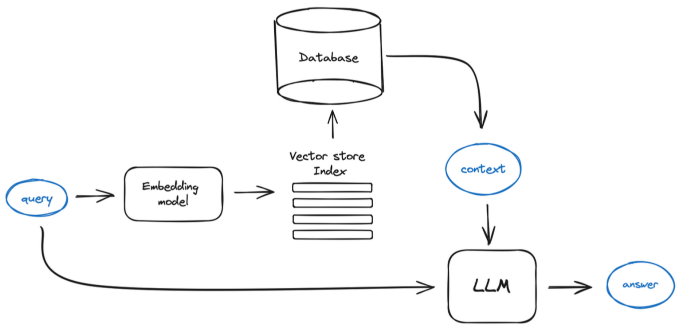


Figure 5.4: A representation of the RAG architecture.

RAG is a great tool for making facts more consistent if used in some of its forms. It is regrettable that traditional language models can sometimes generate a piece of information which may be quite logical but misleading. However, RAG makes a correction to the issue by responding to factual, intelligible papers. It makes it possible to avoid wrong or "hallucinated" content production, which is also possible thanks to such restrictions. RAG is also a very scalable and flexible design that allows you to inject large external datasets or specific knowledge corpus. This means that it can be applied to a number of problems. When we distilled this LLaMA 3 8B model to conduct a specific experiment for this work, we used RAG to make the results that we obtained more reliable. It also enabled the model to generate sensible answers as well as to check all the derived responses with accurate outside data.

5.2.3 Workflow Summary of RAG (Pre-Thesis 2)

When a user inputs a question through the Streamlit interface, the system begins by loading and processing documents from a directory using PyPDFDirectoryLoader. These documents are then split into smaller chunks by the Text Splitter. These chunks are transformed into vector embeddings using Google Generative AI Embeddings while these embeddings are indexed in the FAISS vector store. When a user enters a question, it is encoded and the FAISS vector store conducts a similarity search to obtain vectors most closely related to the user's question. These retrieved documents are passed to the LLM (Groq Cloud's Llama3-8b-8192), which processes the user question and the relevant documents using a prompt template. Finally, the LLM generates an answer based on the provided context and returns the answer to the user through the Streamlit interface. The FAISS vector store thus indirectly interacts with the user input by retrieving the most relevant document vectors based on the query's vector representation.

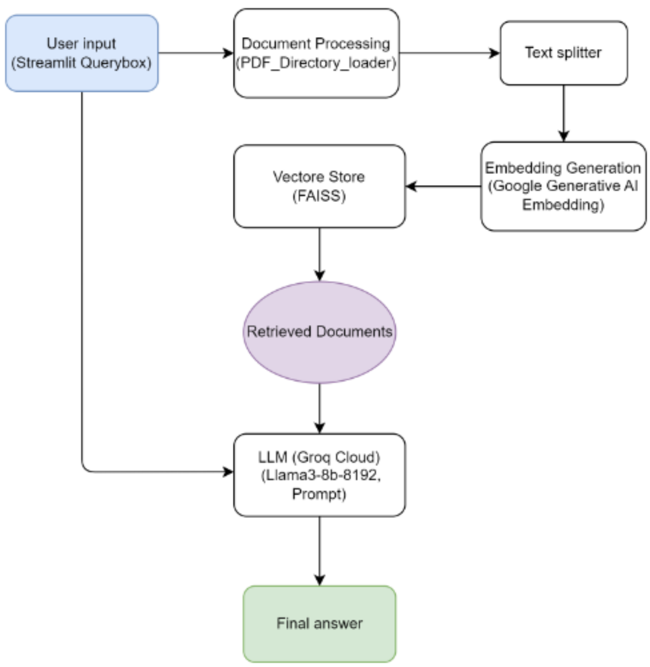


Figure 5.5: Workflow summary of the RAG architecture.

5.3 Hybrid Retrieval Mechanism

The retrieval mechanism functions as the system's key operational element that enables users to perform effective queries into stored knowledge. Modern retrieval operates through semantic embeddings to decode query meanings rather than traditional models which work with exact keyword matches. However, both methods have their limitations:

Structured text retrieval using BM25 search methods provides fast results while achieving effective matching but breaks down when document and query words have different expressions.

Although Semantic search detects broader concepts through Vector Embeddings with ChromaDB it loses efficiency when finding specific sector vocabulary terms or precise textual counterparts.

This investigation uses a Hybrid Retrieval Model that integrates BM25 lexical ranking techniques with ChromaDB vector retrieval to optimize information retrieval. The approach provides correct search outcomes independent of whether users type literal terms or conceptual equivalents. Once the user sends a query, the query gets tokenized and gets ready for later steps.

5.3.1 BM25 Scoring

The Best Matching 25 ranking function identified as BM25 serves information retrieval systems by measuring document relevance to search queries. The Basic Matching model represents an upgraded version of TF-IDF (Term Frequency-Inverse Document Frequency) standards which function extensively throughout search engine applications. After the initialization of the retriever with an instance of knowledge base and a list of documents containing id, text and metadata, the corpus for BM25 is made out of the extracted texts of the documents. Then the total corpus is also tokenized to match with the tokenized query. This tokenized corpus is used to initialize the BM25 model. The retriever then query the BM25 model with the tokenized query to get the relevance scores for each document in corpus and take the highest top 10 scores in our case. Lastly, the top 10 results are stored in a combined document including the scores.

$$\mathrm{BM25}(D,Q) = \sum_{t \in Q} \mathrm{IDF}(t) \cdot \frac{f(t,D)\left(k_1+1\right)}{f(t,D) + k_1\left(1-b + b \cdot \frac{|D|}{\mathrm{avgDL}}\right)}$$

$$\mathrm{IDF}(t) = \log\left(\frac{N - n(t) + 0.5}{n(t) + 0.5}\right) \qquad ([5])$$

5.3.2 Chroma DB distance Calculation

The HybridRetriever system applies both BM25 to retrieve documents alongside using ChromaDB to enhance semantic query matching. ChromaDB is responsible for:

Vector Representations: All documents that compose the corpus have their own embedding vector. As queries arrive at ChromaDB the system creates embedding vectors for queries and uses them in calculation.

Similarity Computation: When performing comparisons ChromaDB employs cosine similarity along with alternative vector similarity calculations for every document vector in its knowledge base.

Distance-Based Ranking: ChromaDB positions documents according to their comparative vector distances towards the query input. The level of similarity directly correlates with the distance between query and document.

The HybridRetriever system applies both BM25 to retrieve documents alongside using ChromaDB to enhance semantic query matching. ChromaDB is responsible for:

Vector Representations: All documents that compose the corpus have their own embedding vector. As queries arrive at ChromaDB the system creates embedding vectors for queries and uses them in calculation.

Similarity Computation: When performing comparisons ChromaDB employs cosine similarity along with alternative vector similarity calculations for every document vector in its knowledge base.

$$\text{CosineSimilarity}(A, B) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}} \qquad ([10])$$

Distance-Based Ranking: ChromaDB positions documents according to their comparative vector distances towards the query input. The level of similarity directly correlates with the distance between query and document.

Top-K Selection: Like BM25, ChromaDB returns the top k results but here less distance indicates more similarity. The system retrieves 10 documents that possess similar vectors. During semantic retrieval the system identifies relevant documents which demonstrate conceptual or contextual relationships with the query despite lacking direct lexical matches because such associations remain vital for discovering more abstract topic-oriented content.

Again if there is any common document in top 10 of BM25 and Chroma DB then it means there is already an entry for that document in the combined docs list. So we just update the place holder for the distance key to the document's distance. If any document of Chroma DB is absent in BM25's result then a new entry is registered into the combined doc where the value of BM25 is set to zero:

Like BM25, ChromaDB returns the top k results but here less distance indicates more similarity. The system retrieves 10 documents that possess similar vectors. During semantic retrieval the system identifies relevant documents which demonstrate conceptual or contextual relationships with the query despite lacking direct lexical matches because such associations remain vital for discovering more abstract topic-oriented content.

Again if there is any common document in top 10 of BM25 and Chroma DB then it means there is already an entry for that document in the combined docs list. So we just update the place holder for the distance key to the document's distance. If any

document of Chroma DB is absent in BM25's result then a new entry is registered into the combined doc where the value of BM25 is set to zero.

5.3.3 Normalizing and Combined Scoring

To produce a single combined score, the retrieval system must handle two very different measures—BM25 scores and vector distances:

The system performs BM25 Normalization through score multiplication by the candidate set maximum divisor to generate values between 0 to 1. These scores are stored in the **bm25_normalized** array. The system adjusts vector distance values so that closer correspondences transform into higher numerical results. The function **distance_similarity = 1/(1+distance)** provides simple normalization techniques...

Weighted Combination: The final document ranking system uses weighted sums of normalized BM25 scores with normalized vector similarities to compute its final score. For example, the combined score formula incorporates:

$$\text{CombinedScore}(D, Q) = \lambda \cdot \text{BM25}(D, Q) \; + \; \big(1 - \lambda\big) \cdot \text{Similarity}(D, Q).$$

Here w(BM25) and w(Chroma) are the weights that can be adjusted to customize the combined score which enables users to adjust lexical and semantic weight contributions. We have kept both the weights to 0.5 which gives equal preference to both lexical and semantic values.

5.3.4 Final Ranking

Once the combined score for each document is calculated:

Sorting: documents receive sorting with descending arrangement based on combination scores so relevant documents rating highest by both lexical and semantic evaluation rise to the top position.

Top-K Results: As the last step the retriever supplies the top k documents to users or downstream programs. The retrieval system provides ranked lists of documents which users or downstream applications need for data processing purposes or screen display needs.

By balancing BM25 and ChromaDB scores, the hybrid retriever capitalizes on the strengths of each method:

Through BM25 the search precision remains high for queries requiring exact word matches. Through ChromaDB users can draw out semantic information with contextual understanding which simple lexico-semantic matching often overlooks. This methodological integration delivers a strong flexible retrieval system suitable for diverse application scenarios ranging from basic keyword retrieval to deep semantic investigations.

5.3.5 Workflow of Hybrid Retriever

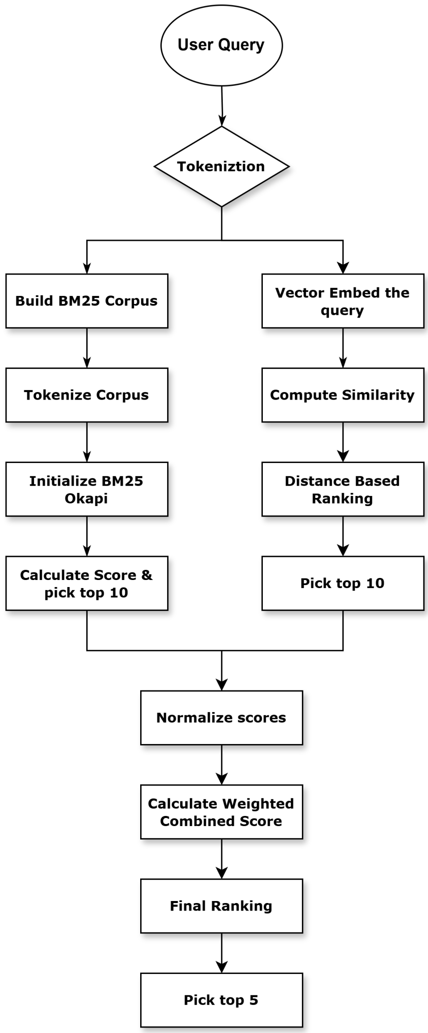


Figure 5.6: Workflow of Hybrid Retriever

5.3.6 Response Generation

The automated response system uses multiple stages that guarantee accurate interactions while delivering contextually relevant and properly formatted results. These steps include:

1. User Query Processing

The user query processor operates through Streamlit as part of the chatbot interface to provide real-time query handling. The chatbot processes a user's input then keeps the information in its session state. The system allows the bot to track messages history along with prior conversations which stands as a vital requirement for contextual response generation. By storing session history the bot maintains seamless conversations that flow naturally instead of delivering random or split responses.

The chatbot tool performs querying classification to establish which kind of response should be generated: contextual retrieval or general information or action execution. Through its structured query processing strategy the chatbot produces more efficient and accurate responses.

- **Real-time Processing:** The chatbot completes user input processing instantly to stay responsive to users.
- **Session State Management:** The system keeps an inventory of past chats that allows it to maintain contextual understanding in its responses.
- **Query Categorization:** The system uses query categorization to understand question types in order to pick proper response strategies.

2. Hybrid Retrieval for Contextual Information

The hybrid retrieval system serves to improve the chatbot's response knowledge base. The retrieval system combines keyword- and semantic-search technologies to deliver accurate results from a context-based perspective. The chatbot utilizes:

- **BM25 Algorithm:** The document scoring system uses a probabilistic ranking function which evaluates document relevance through term frequency determination along with inverse document frequency analysis (TF-IDF).
- **ChromaDB Embeddings:** Semantic search utilizes vector similarity to retrieve documents that preserve contextual meaning through its retrieval process.

The hybrid system maintains retrieval balance by using keyword-based BM25 to deliver exact term matches alongside embedding-based retrieval that detects related meanings within text. The retrieval process produces rankings of relevant documents that limit response generation to top-ranked contexts.

3. Contextual Representation for Response Generation

The system arranges retrieved documents that show the highest relevance into an organized structure. The constructed response uses a combination of user query and hybrid retriever context alongside corresponding references from the knowledge base. The organized format of this representation guarantees both valid and source-based backing for every response.The systemized method for arranging collected data enables the chatbot to produce responses which match user queries while maintaining factual accuracy. Additional security comes from the chatbot referencing its source material that builds user trust together with credibility. Users who access source references in the chatbot system can check the validity of generated responses to enhance the accuracy of chatbot education outcomes.

4. LLM-Based Response Generation

Through GROQ LLaMA-3.3-70B - a modern language model - the system develops natural language responses. Consistency and accuracy in the chatbot initiation phase require the definition of a system message which defines the essential foundation of an interaction by explaining how the assistant functions while also determining response types and behavior boundaries.The system message includes the following directives:

- The chatbot conducts its initial introduction during each new dialogue.
- The system keeps "context" out of its verbal responses.
- The system keeps responses direct and polite while also keeping them brief and easy to understand.
- The system ends all dialogue with users who display improper behavior.

The concluded chat dialogue alongside its retrieved contextual elements get passed through LLaMA-3.3-70B to GROQ API for creating a response. Through this process the chatbot produces responses with structured organization that specifically matches users' previous inputs. The formulas used by LLaMA-3.3-70B:

$$\text{Attention}(Q, K, V) = \text{softmax}\Big(\frac{Q\,K^T}{\sqrt{d_k}}\Big)\,V$$

$$\text{FeedForwardNetwork}(x) = \text{ReLU}\big(x\,W_1 + b_1\big)\,W_2 \;+\; b_2 \qquad ([9])$$

5. Response Delivery and Conversation Management

Following its response generation process the LLM requires various post-processing operations to create final user-ready content. Users see the response produced by the LLaMA-3.3-70B model when it appears in the chat interface for their interaction. The session history stores the response which maintains dialogue coherence and creates an ongoing context for following interactions. The chatbot performs termination conditions assessment on responses to detect inappropriate behavior before displaying '¡¿' to end the conversation. The controlled delivery process preserves the chatbot's ability to keep track of context and answer effectively to multiple queries while respecting ethical standards in all conversations.

Error Handling and API Failures

An exception handling system within the chatbot enables the proper management of failures including API outages or connectivity problems. When the LLM encounters an unavailable state it notifies users properly while advising them to attempt again at a later time. Through this proactive measure the system prevents feedback loss in situations of unexpected failures which ensures reliable user experiences.

5.3.7 Workflow of Response Generation

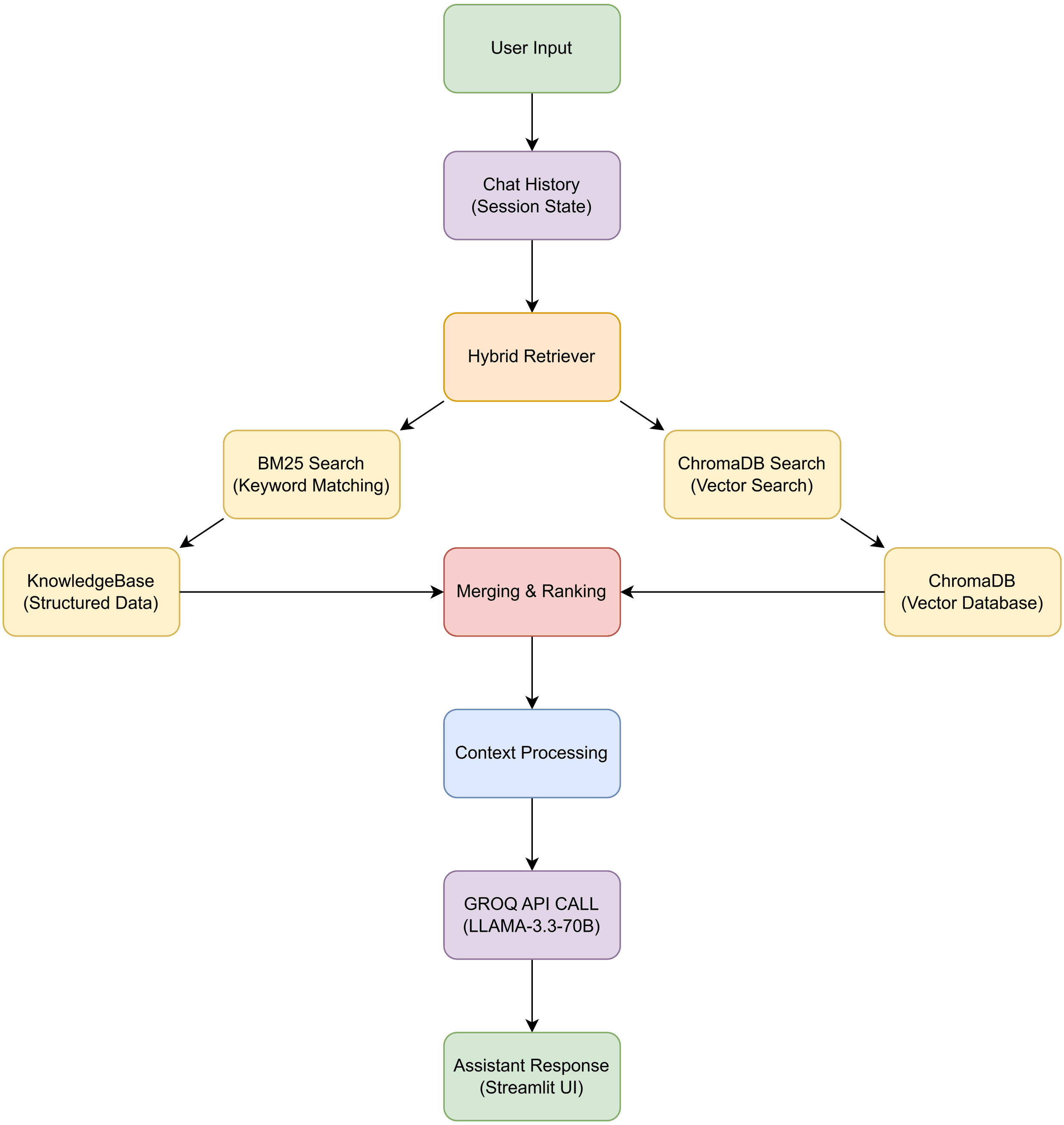


Figure 5.7: Workflow of Response Generation

5.3.8 Demonstration of Chatbot Interaction

what is the prerequisite for cse420

The prerequisites for CSE420 are CSE321, CSE331, and CSE340. The full prerequisite chain is CSE321-CSE331-CSE340-CSE221-CSE220-CSE111-CSE110.

who is the coordinator of cse221?

I don't have information about the coordinator of CSE221.

who is the theory coordinator of cse221

The theory coordinators of CSE221 are FGZ and IBA. Their emails are fakhruddin.gazzali@bracu.ac.bd and imran.azad@bracu.ac.bd.

who is TAW

TAW refers to Mr. Md. Tawhid Anwar, a Senior Lecturer at BRAC University. His email is tawhid.anwar@bracu.ac.bd and his room is 4G03.

what is the total admission fee for CSE?

Hello, I'm BRACU THESIS BOT, a helpful assistant for BRAC University CSE students. The total admission fee for CSE at BRAC University is BDT 33,900.

who is the theory coordinator of cse340?

The theory coordinator of CSE340 is PBK, and the theory coordinator's email is partha.bhoumik@bracu.ac.bd.

what is the full name of MNY

The full name of MNY is Muhammad Nur Yanhaona.

what is the room number of MNY

The room number of MNY, Muhammad Nur Yanhaona, is 4G06.

what is the room number of Zayed arju

The room number of Zayed Arju, also known as ZBH, is 4P.

who is zmd

ZMD is Zaber Mohammad.

what is the prerequisite of cse321

Hello, I'm BRACU THESIS BOT, a helpful assistant for BRAC University CSE students. The prerequisite for CSE321 is CSE221.

can you provide the full chain?

The full prerequisite chain for CSE321 is CSE221-CSE220-CSE111-CSE110.

what is the email of niloy farhan sir?

The email of Niloy Farhan sir is ext.niloy.farhan@bracu.ac.bd.

can i do double major in math?

Yes, you can do a double major in mathematics. The department offers double majors and minors in mathematics, among other subjects.

Chapter 6

Result and Analysis

This section offers a detailed analysis of the model's performance in the question-answering task. We evaluated the fine-tuned BERT model's behavior across different learning rates, its training and validation loss progression, and its ability to handle queries effectively. Additionally, we used BLEU and ROUGE metrics to provide a more comprehensive assessment of the model's performance.

6.1 Model Performance

6.1.1 BERT Fine-tuning

The model was trained on a dataset of 970 examples, while its validation was performed on 243, divided into batches of size 16. Regarding optimization, we considered the usage of AdamW, the most used optimizer when fine-tuning transformer-based models. During training, we tried several learning rates while fixing the weight decay at 0.01 for 30 epochs. The model maintained consistent gains concerning the reduction of training loss during fine-tuning. Similarly, the trends of the validation loss were similar and flattened or, in some instances, increased, showing overfitting. Herein is summarized the training and validation losses for the different learning rates:

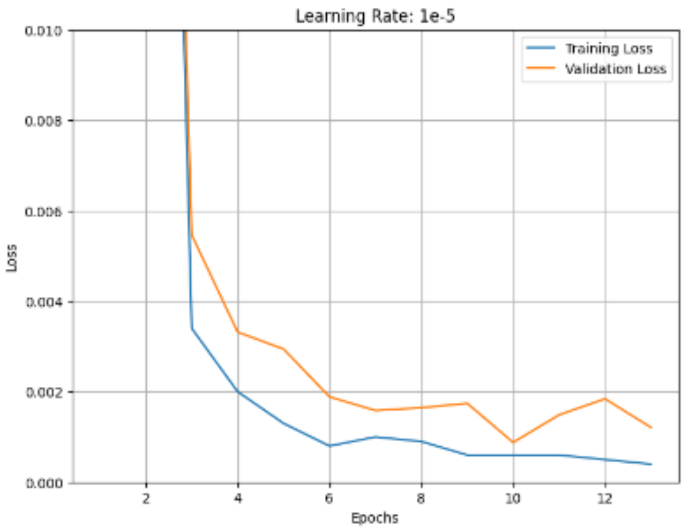
Learning Rate	**Training Loss**	**Validation Loss**
1×10^{-5}	0.00040	0.00120
2×10^{-6}	0.00210	0.00339
3×10^{-6}	0.00090	0.00955

Table 6.1: Training and validation loss for different learning rates

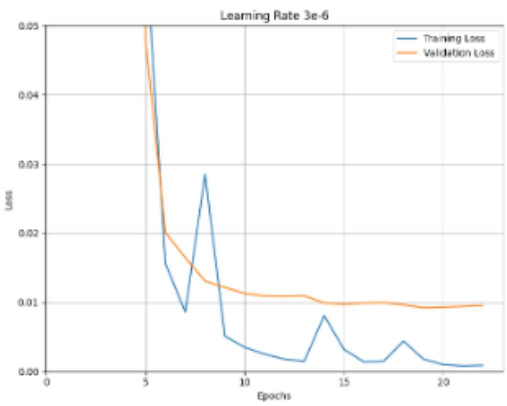
All three learning rates had a very high value of loss initially; these decreased substantially in the early epochs, showing that indeed the model was learning well from the data. The best balance came with a learning rate of 1e-5 since that showed stable convergence and hence reduced overfitting.

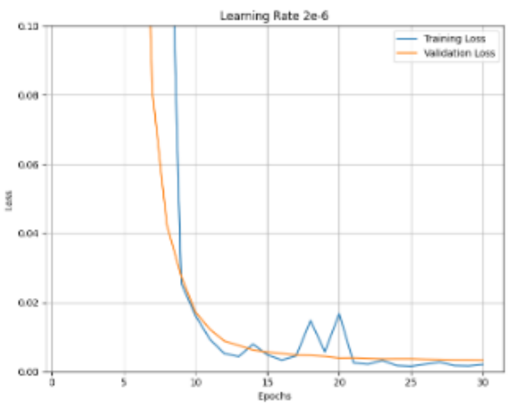
Learning Rate and Optimization

The best performance was obtained with a learning rate of 1e-5 in combination with weight decay set to 0.01. In this case, the training loss decreased smoothly from 4.92 down to 0.0021, while the validation loss decreased from 4.79 down to 0.00339, which means good learning with no significant overfitting. Although training with other learning rates, such as 3e-6, showed more stable training, it failed to show a fast drop of a validation loss; hence, 1e-5 remains the best choice because it offers faster convergence and generalization of the model.



Meanwhile, the learning rate of 3e-6 resulted in more gradual decreases in training and validation loss to eventually yield more stable but slowly improving results. Therefore, 1e-5 proved to be one of the most balanced and efficient overall choices in an attempt to minimize the loss and maximize performance.





Overfitting and Generalization

While the training and the validation losses improved, they had a little gap, which showed that minor overfitting was still present. In the case of the 1×10^{-5} learning rate, the validation loss reached 0.00339 by epoch 30, while the training loss dropped as low as 0.0021. That means the model fitted a bit better on the training data but generalized well with stable validation loss. Meanwhile, the learning rate of 3e-6 resulted in slower convergence and a much higher final validation loss of 0.009556. Early stopping techniques would prevent overtraining when validation loss plateaus during future runs.

6.1.2 Evaluation Metrics Overview

To assess the quality of the generated responses, we evaluated our model using four widely recognized automatic metrics: BLEU, ROUGE-L, BERTScore, and METEOR. The metrics measure distinct aspects of text generation starting from lexical overlap moving through syntactic structure with semantic coherence. Summary charts appear in Figure 6.1 together with supporting detail that follows below.

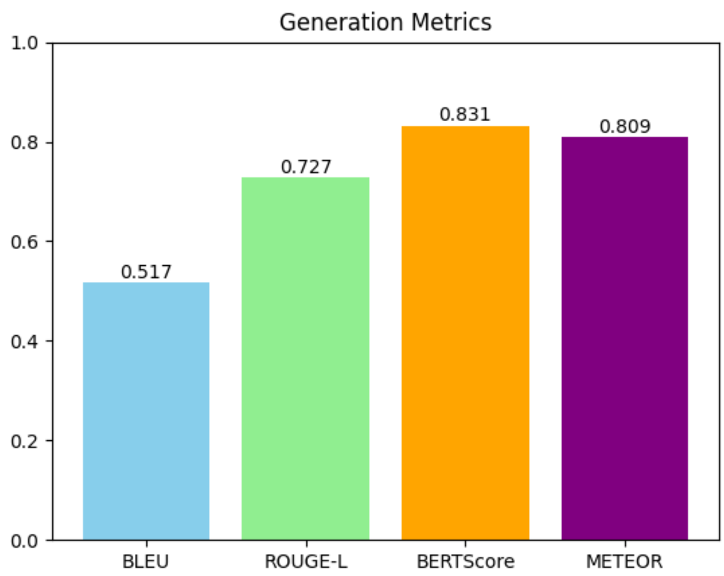


Figure 6.1: Comparison among evaluation metrics

6.1.3 Bilingual Evaluation Understudy

BLEU- Bilingual Evaluation Understudy score serves as the evaluation instrument for machine translation quality and natural language processing model outputs. The metric demonstrates the level of matching between a text and its reference. Under optimal alignment conditions with reference text the BLEU score reaches between 0 and 1.

$$\text{BLEU} = BP \cdot \exp\Bigl(\sum_{n=1}^{N} w_n \, \log p_n\Bigr).$$

$$BP = \begin{cases} 1, & \text{if } c > r, \\ e^{\left(1 - \frac{r}{c}\right)}, & \text{if } c \le r \end{cases} \tag{[1]}$$

How BLEU Score Works:

1. BLEU measures the n-gram matches between generated text and reference material through its methodology.

2. The precision evaluation determines the n-gram level precision by measuring the ratio of matched n-grams compared to the total generated n-grams.

3. BLEU includes brevity penalty measures which penalizes short outputs that significantly deviate from the reference text length.

4. BLEU computes its final score through a weighted geometric mean of n-gram precision scores which applies brevity penalties during the computation.

BLEU Score (0.517)

Our model shows 0.517 performance according to the BLEU evaluation standard which calculates how well the produced text matches against references. Our model showed a BLEU score of 0.517 which demonstrates moderate word pairing alignment. BLEU shows good matching results but since it avoids semantics and synonyms it remains an unreliable tool for language generation systems.

6.1.4 Recall-Oriented Understudy for Gisting Evaluation

One of the most popular metrics for evaluating text generation models operates under the name ROUGE-L (Recall-Oriented Understudy for Gisting Evaluation - Longest Common Subsequence) and works specifically with summarization and dialogue systems. This similarity measure finds the longest common sequence of words that exist between reference and generated text outputs. The n-gram match requirement of BLEU does not measure sentence structure and fluency at the level ROUGE-L does with its LCS evaluation.

$$\text{ROUGE-L} = \frac{\text{LCS}(X, Y)}{|Y|}$$

$$F_1 = \frac{\left(1 + \beta^2\right) R\, P}{R \,+\, \beta^2\, P} \tag{[2]}$$

How ROUGE-L Works:

The Longest Common Subsequence (LCS) algorithm detects the longest series of words which appear identically between the predicted text and reference text although the words do not need to follow each other continuously. The F1-Score calculation in ROUGE-L computes precision and recall along with F1-score through LCS match analysis.

- **Precision:** Precision assesses which portion of the produced text aligns with the reference material.
- **Recall:** An evaluation method which determines what percentage of reference text appears in the produced response.
- **F1-Score:** The F1-Score represents a balanced computation based on precision and recall through their harmonic mean union.

ROUGE-L Score (0.727)

ROUGE-L uses the longest common subsequence or LCS algorithm to measure text structure similarity between generated and reference content. The system scored 0.727 in ROUGE-L F1 which shows that its responses maintain most of the reference answers' basic structure. The results show that the model creates answers with excellent grammatical flow.

6.1.5 BERTScore (0.831)

Using pre-trained transformer embeddings the BERTScore approach calculates token-level semantic similarities beyond traditional metrics BLEU and ROUGE. Within our evaluation our model achieved a BERTScore of 0.831 marking the top result among all metrics. The measurement shows that model responses keep sensible semantic connections with human references despite lacking exact word correspondence. BERTScore demonstrates its capability to understand contextual relationships because it produces natural responses which match reference texts appropriately.

$$\text{BERTScore} = \frac{1}{|X|} \sum_{x \in X} \max_{y \in Y} \cos\big(E(x),\, E(y)\big) \tag{[15]}$$

6.1.6 Metric for Evaluation of Translation with Explicit ORdering)

The evaluation metric METEOR (Metric for Evaluation of Translation with Explicit ORdering) serves to assess machine-generated texts especially within translation and text generation applications. The method improves BLEU by allowing users to measure translation quality through synonym detection and word root analysis in addition to sequence order evaluation with priority placed on recall accuracy. The ability of METEOR to extract semantic meaning and variety in language generates a precise analysis of text quality.

$$F_{\text{mean}} = \frac{P \cdot R}{\alpha\, P \;+\; \big(1-\alpha\big)\, R}.$$

$$\text{Penalty} = \gamma \left(\frac{ch}{m}\right)^{\theta}$$

$$\text{METEOR} = F_{\text{mean}} \times \big(1 \;-\; \text{Penalty}\big) \tag{[3]}$$

How METEOR Works:

1. Exact, Stemmed, and Synonym Matching: METEOR, however, is sensitive to stemmed and base forms (such as 'run' and 'running') as well as synonyms in its attempt to achieve semantic closeness.

2. Alignment and Word Order Penalty: The metric measures how accurate are the words generated, while penalizing the wrong order.

3. Precision, Recall, and F1-Score:

- **Recall-weighted Approach:** Unlike BLEU, which focuses on precision, METEOR assigns more importance to recall, so that important reference content is included.
- **F1-Score Calculation:** The F1-Score represents a balanced computation based on precision and recall through their harmonic mean union.

METEOR Score (0.809)

The METEOR metric measures translation quality through synonymy evaluation and stemmed term analysis and word placement assessment making it more versatile than BLEU. Our system scored 0.780 in METEOR evaluation indicating superior alignment to reference responses with flexibility for different word selection. The results demonstrate that the produced texts maintain semantic relevance while showing diverse linguistic patterns.

3. Comparative Analysis and Insights

A comparative analysis of the four metrics reveals key insights into our model's performance:

High BERTScore (0.831) and METEOR (0.809): Model scores exhibit strong semantic cohesion which demonstrates that it creates contextually appropriate answers despite language variation.

Moderate ROUGE-L Score (0.727): The metrics demonstrate structural alignment between the model's responses and reference text sentences due to structural consistency.

Lower BLEU Score (0.517): Model evaluations based on BLEU score indicate a low match rate which demonstrates its emphasis on semantic meaning delivery through paraphrasing.

Insights and Implications:

The model produces contextually relevant results because its BERTScore and METEOR metrics show high semantic matching although its BLEU and ROUGE-L metrics indicate lower lexical similarity. The NLP metrics BERTScore and METEOR prove more suitable for current NLP applications including abstractive summarization and dialogue generation and paraphrasing because they focus on meaning maintenance over literal word usage. The standard n-gram quantification systems (BLEU, ROUGE) demonstrate their value in checking structural coherency yet they lack full capability to evaluate flexible language production performance of models. The findings demonstrate semantic similarity measures such as BERTScore and METEOR deliver better reliability in assessing text generation quality when compared with the traditional n-gram metrics BLEU and ROUGE-L.

Chapter 7

Limitations and Future Works

7.1 Limitations:

Despite using hybrid retrieval and many other technologies, our chatbot model still exhibits few limitations. This research documented two major problems including Inconsistent Information Retrieval and Alias Recognition Failure.

7.1.1 Contextual Inconsistency:

The system shows variable results depending on the manner in which similar queries are expressed. Different phrasings of requests to retrieve an entity email address present two distinct outcomes where one returns an unavailable status while the other produces correct information. The difference in the response indicates that the data storage format for section was 08 in the database. As a result, trying with only 8 is failing to grab the context. The method struggles to retrieve adequate context for queries which creates differences in obtained information.

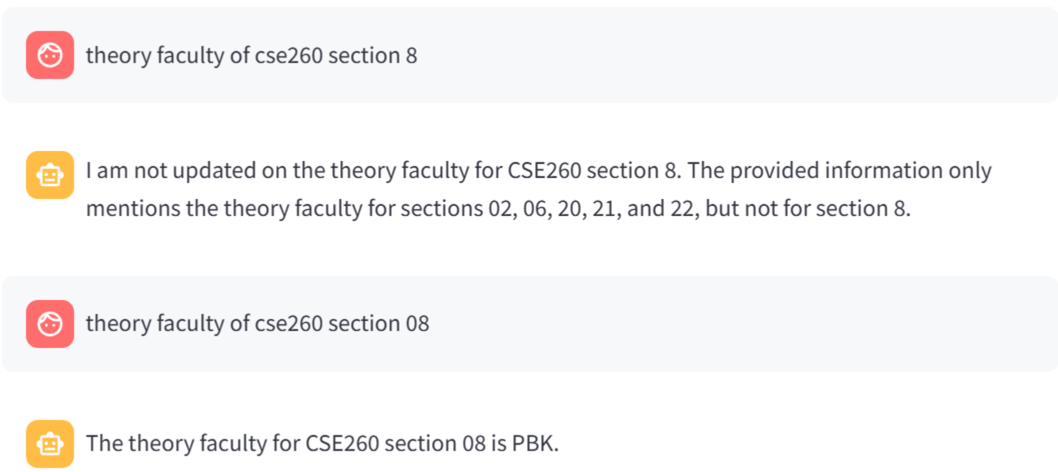


Figure 7.1: Example of Contextual Inconsistency

7.1.2 Entity Recognition Failure

The system does not properly understand the connection between various entity aliases along with their abbreviations resulting in worsened inconsistencies between system responses. The system cannot produce accurate results from full names but will provide effective responses from name acronyms or abbreviations. Entity resolution systems demonstrate poor performance regarding the association of equivalent references across query sets that operate independently. Information retrieval turns unreliable when alias detection systems fail to operate effectively which results in broken or improper output responses.

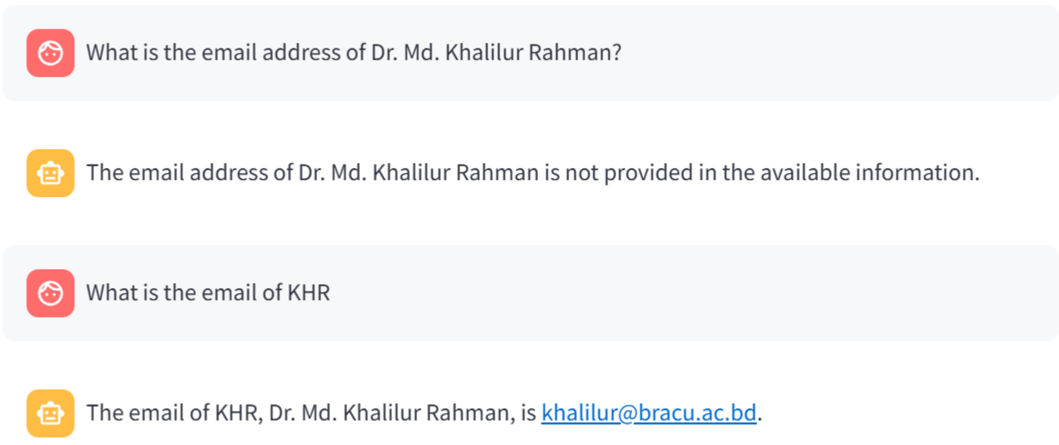


Figure 7.2: Example of Entity Recognition Failure

7.1.3 Limited Scope

The dataset for the provided model is specialized for only the CSE department and this may not address all queries of students. It may not be that much useful for students from other universities if the query is not about BRAC University. This lack of generalizability can not be fulfilled without vast data points from different sources.

7.1.4 Single Language Support

This chatbot model only works for the English language. This lack of multilingual support can cause problems for the students who prefer Bangla or specifically Banglish as the medium of communication.

7.2 Future Works

The observed limitations of the study can be mitigated by following these approaches:

- To address the contextual inconsistency of the study, we can integrate more modern technologies like Agentic RAG where if sufficient context is not retrieved then the model can reorganize the ranking and get adequate context. Another attempt can be paraphrasing the query a few times and retrieving the contexts for each paraphrased version. Retrieval models that use reinforcement learning can also be a good solution for this problem.

- Entity recognition failure can occur in our model since the data points of our models are highly related to each other. The datapoint in our database associated the initials of a faculty with their email address that is why searching with full name is not returning the result. This problem can be addressed by using a knowledge graph. Knowledge graph representation can store all the attributes and relations together which can be very useful to resolve this problem.

- The scope limit can only be resolved by adding new and robust data into our dataset. Using information from other departments and even other universities can make the chatbot a generalized solution for every student's day to day queries.

- Lastly, we can implement multilingual LLM models (e.g. mBert, Multilingual T5, etc) and design our retrieval function to support multi languages. This will make our chatbot capable of multilingual operations.

Chapter 8

Conclusion

This research focused on creating a chatbot for providing students with academic and administrative information at BRAC University. Implementing a hybrid retrieval model, combining keyword-based BM25 ranking with semantic vector search through ChromaDB, helps bridge gaps between simple word matching and an understanding of the intent behind a query posed by a user. Unlike traditional search approaches, such integration helps a lot in terms of effectiveness. A key feature of our system is its organized pipeline of information, allowing for efficient updating and minimizing redundancy in processes. By leveraging modern embedding techniques, including SentenceTransformers, the chatbot effectively identifies student queries with relevant information, allowing for an unobstructed and free flow of conversation. Maintaining a perfect recall and precision balance is a key feature of the chatbot. By combining BM25's lexical search with ChromaDB's vector retrieval mechanism, the system can produce relevant information accurately even when queries are posed in a variety of forms. Integration with GROQ's LLaMA-3.3-70B model helps make the output even more effective, with a spontaneous and friendly conversation during an interaction.

Evaluations of performance with BLEU, ROUGE-L, BERTScore, and METEOR confirm that both meaningful and relevant responses are generated through the chatbot. Despite its semantic comprehension capabilities, several weaknesses, such as variable retrieval performance, alias difficulty, and a narrow CSE department at BRAC University bias, remain present. In addition, the chatbot is at present only operational in English, and therefore, excludes non-English speakers. In future, improvements such as the use of reinforcement learning for retrieval improvement, integration with a knowledge graph for entity improvement, and enlargement of the corpus for use with additional departments and universities will contribute enormously towards enhancing the value of this chatbot. Integration with multilinguality will make it even more inclusive. With such improvements, such a chatbot holds tremendous potential to become an integral part of students' lives worldwide.

Bibliography

[1] K. Papineni, S. Roukos, T. Ward, and W. J. Zhu, "Bleu: A method for automatic evaluation of machine translation," in *ACL*, 2002.

[2] C. Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *ACL*, 2004.

[3] S. Banerjee and A. Lavie, "Meteor: An automatic metric for mt evaluation with improved correlation with human judgments," in *ACL*, 2005.

[4] B. A. Shawar and E. Atwell, "Chatbots: Are they really useful?" *LDV Forum*, vol. 22, no. 1, pp. 29–49, 2007.

[5] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: Bm25 and beyond," *Foundations and Trends in Information Retrieval*, 2009.

[6] P. B. Brandtzaeg and A. Følstad, "Why people use chatbots," in *International Conference on Internet Science*, Cham: Springer, 2017, pp. 377–392.

[7] L. Cui, S. Huang, F. Wei, C. Tan, C. Duan, and M. Zhou, "Superagent: A customer service chatbot for e-commerce websites," in *Proceedings of the 13th International Joint Conference on Natural Language Processing (IJCNLP) and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 97–102. DOI: 10.18653/v1/P17-4017. [Online]. Available: https://doi.org/10.18653/v1/P17-4017.

[8] B. R. Ranoliya, N. Raghuwanshi, and S. Singh, "Chatbot for university-related faqs," in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, IEEE, 2017, pp. 1525–1530. DOI: 10.1109/ICACCI.2017.8126057. [Online]. Available: https://doi.org/10.1109/ICACCI.2017.8126057.

[9] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *NeurIPS*, 2017.

[10] Y. A. Malkov and D. A. Yashunin, "Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[11] R. Winkler and M. Söllner, "Unleashing the potential of chatbots in education: A state-of-the-art analysis," *Academy of Management Annual Meeting Proceedings*, vol. 2018, no. 1, pp. 1–40, 2018.

[12] A. Følstad and M. Skjuve, "Chatbots for customer service: User experience and motivation," in *Proceedings of the 1st International Conference on Conversational User Interfaces*, 2019, pp. 1–9.

[13] V. Nguyen, "Question answering in the biomedical domain," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, Association for Computational Linguistics, 2019, pp. 54–63.

[14] K. Swanson, L. Yu, C. Fox, J. Wohlwend, and T. Lei, "Building a production model for retrieval-based chatbots," in *Proceedings of the First Workshop on NLP for Conversational AI*, Association for Computational Linguistics, 2019, pp. 32–41.

[15] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," in *ICLR*, 2019.

[16] P. Suta, X. Lan, B. Wu, P. Mongkolnam, and J. H. Chan, *An overview of machine learning in chatbots*, 2020. [Online]. Available: https://www.semanticscholar.org/paper/An-Overview-of-Machine-Learning-in-Chatbots-Suta-Lan/d29f43e6b42d9fab625623c84e02909f35118bd8.

[17] E. Adamopoulou and L. Moussiades, "Chatbots: History, technology, and applications," *Journal of Artificial Intelligence Research*, 2021.

[18] G.-J. Hwang and C.-Y. Chang, "A review of opportunities and challenges of chatbots in education," *Interactive Learning Environments*, 2021.

[19] S. Z. Sweidan, S. S. Abu Laban, N. A. Alnaimat, and K. A. Darabkh, "Siaaa-c: A student interactive assistant android application with chatbot during covid-19 pandemic," *Computers & Applications in Engineering Education*, vol. 29, pp. 1718–1742, 2021.

[20] P. S. Atmauswan and A. M. Abdullahi, "Intelligent chatbot for university information system using natural language approach," *Albukhary Social Business Journal (ASBJ)*, vol. 3, no. 2, 2022. DOI: 10.55862/asbjV3I2a007. [Online]. Available: https://doi.org/10.55862/asbjV3I2a007.

[21] W. El-Ashmawi, S. Elbohy, M. Rafik, *et al.*, "An interactive chatbot for college enquiry," *Interactive Learning Environments*, vol. 2, pp. 20–28, 2023.

[22] H. Dinh and T. K. Tran, "Educhat: An ai-based chatbot for university-related information using a hybrid approach," *Applied Sciences*, vol. 13, no. 22, p. 12 446, 2023. DOI: 10.3390/app132212446. [Online]. Available: https://doi.org/10.3390/app132212446.

[23] A. Formica, I. Mele, and F. Taglino, "A template-based approach for question answering over knowledge bases," *Knowledge and Information Systems*, 2023. DOI: 10.1007/s10115-023-01966-8. [Online]. Available: https://doi.org/10.1007/s10115-023-01966-8.

[24] S. Lee, Y. Jang, C. Park, *et al.*, "Peep-talk: A situational dialogue-based chatbot for english education," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Jul. 2023, pp. 190–207.

[25] S. Mukherjee, V. Hudeček, and O. Dušek, "Polite chatbot: A text style transfer application," in *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2023. DOI: 10.18653/v1/2023.eacl-srw.9. [Online]. Available: https://doi.org/10.18653/v1/2023.eacl-srw.9.

[26] H. Nguyen, J. Lopez, B. Homer, A. Ali, and J. Ahn, "Reminders, reflections, and relationships: Insights from the design of a chatbot for college advising," *Information and Learning Sciences*, vol. 124, no. 3/4, pp. 128–146, 2023.

[27] K. Pandya and M. Holia, "Automating customer service using langchain: Building custom open-source gpt chatbot for organizations," *arXiv preprint arXiv:2310.05421*, 2023.

[28] M. S. Shahriar, A. A. F. Chowdhury, M. A. Ehsan, and A. R. Kamal, "Question answer generation in bengali: Mitigating the scarcity of qa datasets in a low-resource language," in *Proceedings of the 13th International Joint Conference on Natural Language Processing (IJCNLP) and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, Nov. 2023, pp. 430–441.

[29] S. Campese, I. Lauriola, and A. Moschitti, "Pre-training methods for question reranking," in *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Mar. 2024. [Online]. Available: https://aclanthology.org/2024.eacl-short.41/.