Personalized Chatbot Service for GSDS Slack Users

Jisoo Kim
Graduate School of Data Sciences,
Seoul National University
jisoo.kim@snu.ac.kr

Suyoung Park
Graduate School of Data Sciences,
Seoul National University
itforspark@snu.ac.kr

Bo-Eun Ahn
Graduate School of Data Sciences,
Seoul National University
ahnbo@snu.ac.kr

Somangchan Park
Graduate School of Data Sciences,
Seoul National University
jrhopefulp@snu.ac.kr

Jiwoo Song
Graduate School of Data Sciences,
Seoul National University
sjiwoo@snu.ac.kr

ABSTRACT

Recent advancements in large-scale language models (LLM) have significantly enhanced the potential for the creation of conversational chatbots. Our objective was to develop an interactive chatbot tailored to the personalized needs of users by leveraging information from Slack, a communication platform widely utilized by members at the Graduate School of Data Science, Seoul National University. To realize this objective, we employed LangChain, an application facilitating Retrieval-augmented generation (RAG)-based LLM. The chatbot system is structured to parse JSON files extracted from Slack, store them in a vector database, assess similarity with a given query, and furnish corresponding answers to queries. Through experimentation, we explored various conditions to enhance chatbot performance and confirmed that improvements can be achieved by optimizing the text split method, database type, and incorporation of metadata.

Reference Format:

Jisoo Kim, Somangchan Park, Suyoung Park, Jiwoo Song, and Bo-Eun Ahn. Personalized Chatbot Service for GSDS Slack Users. BKMS2, XX(X): XXX-XXX, 2023.
doi:XX.XX/XXX.XX

Artifact Availability:

The source code, data, and/or other artifacts have been made available at https://github.com/PomesHouse/SNU-BKMS2-Project.

1 INTRODUCTION

The evolution of chatbots has undergone a significant transformation, driven by advancements in Natural Language Processing (NLP). One significant stride in this domain involves the incorporation of Large Language Models (LLMs), such as OpenAI's GPT (Generative Pre-trained Transformer) series, which has revolutionized the landscape of conversational agents.

At the core of an effective chatbot lies the ability to retrieve information swiftly and accurately. Vector databases play a pivotal role in achieving this by representing data, including text embeddings from LLMs, in a high-dimensional vector space. Each data point is assigned a vector, preserving its semantic relationships with other points. This enables efficient similarity searches, a crucial aspect in enhancing the chatbot's responsiveness and intelligence.

Retrieval-augmented generation (RAG) is the process of incorporating necessary information into LLM model prompts so that AI applications can improve their ability to reason about the data, and LangChain is an application that makes it easy to utilize it [1].

Using the Langchain application, we planned to develop a chatbot as a practical means to efficiently explore information in Slack, which is used as an essential communication channel between members of the Graduate School of Data Science (GSDS) in Seoul National University. The escalating volume of information circulating within Slack channels has posed a significant challenge for users seeking specific information. As a result, there is a pressing need to improve information retrieval processes to address the difficulties users face in locating pertinent data. Also, Users have to spend time searching for essential information. To mitigate this, a more efficient and streamlined approach is required to enhance the user experience and optimize productivity. While the built-in search feature within Slack channels serves its purpose for specific keyword searches, it falls short in facilitating real-time searches for everyday conversations and specific topics. This limitation highlights the necessity for augmenting the search capabilities to better align with users' dynamic information needs. Finally, in a conversation-centric communication environment, the transient nature of discussions poses challenges for retaining important information.

The primary focus of this research is to enhance information retrieval capabilities within the Slack platform through the implementation of a chatbot powered by the RAG model. The goals include the utilization of RAG to optimize the chatbot's performance, the incorporation of a robust keyword-based search system for precise information retrieval based on user-entered terms, and the development of a sophisticated similarity-based search mechanism for nuanced and context-aware results. The key features of this approach encompass empowering users with the ability to input search terms, enabling a similarity search functionality for accurate and relevant content retrieval within Slack, and implementing memory optimization to streamline the user experience by eliminating the need to navigate the entire Slack content for information retrieval. As a result, this research aims to significantly improve the efficiency and effectiveness of information retrieval and communication practices within the Slack platform.

2 Experiments and analysis

The architecture of the chatbot and its interaction with the dataset extracted from Slack are investigated. Additionally, a critical analysis of the architecture's practical deployment is conducted. In the initial part, attention is given to the process of data extraction from Slack, emphasizing the transformation of unstructured conversational data into a structured JSON format. This step is crucial for ensuring the integrity and usability of the dataset in the subsequent architectural components. The architecture's implementation is then scrutinized, with a focus on the integration of key tools such as FAISS, LangChain, and OpenAI GPT3.5. The evaluation is centered around these tools' collective performance in processing complex queries such as their ability to produce accurate and contextually relevant responses.

2.1 Architecture: Datasets

2.1.1 Slack Data Extraction and Structuring

The foundation of the chatbot architecture begins with the extraction and structuring of data from Slack. Slack data, inherently diverse, is initially stored in JSON format separated by channels and sorted by dates. Each JSON file consists of a complex mix of main texts, which are posts and comments, and metadata.

To efficiently manage this data, Slack Dumper is employed [2]. This tool is designed to dump Slack messages, users, files, and emojis, utilizing browser token and cookie for access. It serves various purposes, such as archiving private conversations when application installation is restricted, allowing users to preserve channel messages when Slack does not offer archives, creating Slack Export archives without admin access, and saving favorite emojis. Slack Dumper operates in three modes: listing users/channels, dumping messages and threads, and creating Slack Exports in either Mattermost or Standard modes. For GSDS Slack data like Fig. 1, Slack Exports Standard mode was used to extract data.

Initially, when Slack data was exported from Slack like shown in Fig. 2 through Slack Dumper, it was organized into folders by channels, with each folder containing JSON files sorted by date. However, for enhanced functionality and granularity in the chatbot system, it was necessary to restructure this data further.

Each JSON file, initially containing a mix of multiple posts and comments, was dissected to separate these elements. This process involved isolating each post or comment from the mixed data and then converting them into individual JSON files like shown in Fig. 3. This granular restructuring of the Slack data into distinct JSON files for each post and comment sought to enhance the precision and efficiency of the subsequent search processes in the chatbot architecture.

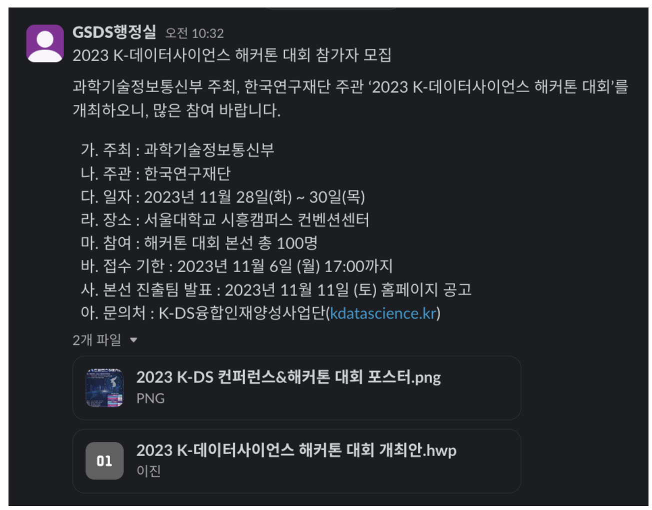
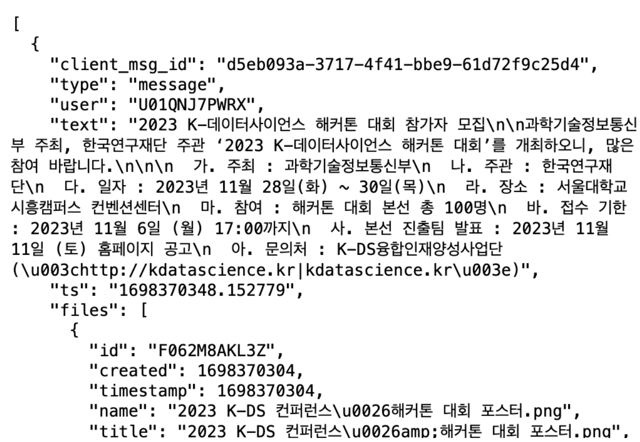


Figure 1: **A screenshot of an original post in GSDS Slack workspace.**



```
[
  {
    "client_msg_id": "d5eb093a-3717-4f41-bbe9-61d72f9c25d4",
    "type": "message",
    "user": "U01QNJ7PWRX",
    "text": "2023 K-데이터사이언스 해커톤 대회 참가자 모집\n\n과학기술정보통신부 주최, 한국연구재단 주관 '2023 K-데이터사이언스 해커톤 대회'를 개최하오니, 많은 참여 바랍니다.\n\n\n  가. 주최 : 과학기술정보통신부\n  나. 주관 : 한국연구재단\n  다. 일자 : 2023년 11월 28일(화) ~ 30일(목)\n  라. 장소 : 서울대학교 시흥캠퍼스 컨벤션센터\n  마. 참여 : 해커톤 대회 본선 총 100명\n  바. 접수 기한 : 2023년 11월 6일 (월) 17:00까지\n  사. 본선 진출팀 발표 : 2023년 11월 11일 (토) 홈페이지 공고\n  아. 문의처 : K-DS융합인재양성사업단(\u003chttp://kdatascience.kr|kdatascience.kr\u003e)",
    "ts": "1698370348.152779",
    "files": [
      {
        "id": "F062M8AKL3Z",
        "created": 1698370304,
        "timestamp": 1698370304,
        "name": "2023 K-DS 컨퍼런스\u0026해커톤 대회 포스터.png",
        "title": "2023 K-DS 컨퍼런스\u0026amp;해커톤 대회 포스터.png",
```


Figure 2: **A screenshot of an JSON file of GSDS Slack workspace right after the export using Slack Dumper**

```
[
  {
    page_content='데이터사이언스대학원 신규 홈페이지 오픈 작업 안내
(4.21.(금))데이터사이언스대학원 신규 홈페이지 오픈을 위하여 다음과 같이
작업을 시행 중에 있습니다. 해당 시간 중 홈페이지 접속이 원활하지 않을 수
있으니 참고하시기 바랍니다.○ 기간: 2023.4.21.(금) 09:00~16:00
(예정) ※ 추가 소요될 수 있음○ 대상: 데이터사이언스대학원 홈페이지
<http://gsds.snu.ac.kr|gsds.snu.ac.kr>  내용: 종전 홈페이지
종료 및 신규 홈페이지 오픈(배포)' metadata={'source': 'general
4', 'seq_num': 1, 'sender_name': 'GSDS행정실', 'date':
'2023-04-21'}
  }
]
```


Figure 3: **A screenshot of a fully processed document for vectorDB. It includes ‘page_content’ which previously corresponds to the “text” from JSON files. It also includes metadata: ‘source’ is the channel from which the message was sent, ‘seq_num’ is the order of the message within the JSON file(the sequence within the same date), ‘sender_name’ is the name of the person who sent the message, and ‘date’ is the date on which the message was sent.**

2.1.2 Data Segmentation Techniques

There were variety of text splitters:

- CharacterTextSplitter: This splitter breaks down text at the character level, ensuring even the minutest details are captured.
- RecursiveCharacterTextSplitter: Building upon the ChacterTextSplitter, this splitter recursively segments the text, delving deeper into the data structure for a more thorough analysis.
- TokenTextSplitter: Operating at the token level, this splitter is particularly adept at processing structured or linguistically nuanced data.

Among these, RecursiveCharacterTextSplitter was chosen and the data were segmented into manageable chunks [3]. The chunk size was 1000, a carefully calibrated figure that balances processing efficiency with the depth of data analysis.

2.2 **Architecture: Structuring and Optimization of the Vector Database**

The organization and optimization of the vector database are essential for the efficiency of data storage and retrieval. The strategies for handling Slack data are explored, from initial extraction and categorization to find storage in the vector database. Also the crucial preprocessing steps taken to refine the data’s clarity and usability, optimizing the chatbot’s performance in data handling and response generation, are discussed.

A. Organization of Extracted Slack Data

The foundational aspect of the vector database involves the meticulous organization of Slack data. Extracted data is systematically categorized into distinct directories based on the respective Slack channels [4]. Each directory comprises chronologically sorted JSON files, each containing messages from specific dates. This organization method plays a critical role in maintaining the integrity and navigability of the data.

B. Strategic Storage in the Vector Database

Within the vector database, each Slack message is individually stored, ensuring that each piece of data is uniquely identifiable and retrievable. To enhance the richness of the data, additional information such as the identity of the message sender, the respective channel, and the date of the message is also stored. This strategic storage approach allows for more nuanced data retrieval and plays a significant role in the chatbot's ability to provide contextually rich responses.

C. Processing and Preprocessing of Slack Messages

A notable aspect of the Slack data processing involves addressing the HTML format in which the messages are presented. Given that the formatting elements (such as bold, italics, and size changes) in the HTML structure might obscure the essential content of the messages, a preprocessing step is undertaken. This step involves the removal of formatting elements and HTML entities like "&gt", "&lt", "\t", and "\n". The rationale behind this decision is to prioritize the content's clarity and relevance over its stylistic presentation, thereby ensuring that the chatbot focuses on the substantive information within the messages.

2.3 **Architecture: Search Efficiency and Accuracy**

2.3.1 Integration of FAISS

To surmount the challenges posed by the high-dimensional nature of the natural language data, Facebook AI Similarity Search (FAISS) was integrated into the architecture [5]. FAISS is renowned for its exceptional performance in managing large–scale datasets showing high quality of performance in similarity search. In the pilot experiment FAISS outperformed other tools like Chroma and pgvector with respect to Slack data. Its advanced indexing and search capabilities enable rapid and precise retrieval of relevant data, which contributes to building an effective chatbot.

2.3.2 Advantages of FAISS

FAISS offers several key advantages

- Scalability: Efficiently handles large and growing datasets.

- Speed: Provides fast search responses, essential for real-time chatbot interactions.
- Accuracy: Delivers high precision in search results, ensuring that the chatbot's responses are relevant and on-pint.

2.4 Architecture: LangChain integration for Enhanced Connectivity

2.4.1 Bridging Data and Language Models

LangChain, recognized as a cutting-edge framework, is utilized as the conduit for connecting structured data with Open AI GPT3.5 turbo model. This integration, critical in transformation of raw data into intelligible, conversational outputs, allows for the efficient flow of information between data repositories and the LLM model. The efficient access, interpretation, and utilization of data by the chatbot are ensured through this process. The full architecture of LangChain is depicted in Fig. 4.

2.4.2 Role of LangChain

In the architecture, several vital roles are played by LangChain:

- Data Retrieval: Relevant information from the preprocessed data is efficiently fetched.
- Information Processing: In tandem with GPT, the processing and interpretation of the retrieved data are conducted.
- Response Generation: Generating coherent, contextually appropriate responses based on the processed information was assisted.

2.4.3 Combining Components for a Cohesive Chatbot System

Seamless Integration

The true power of the architecture is reflected in the seamless integration of its components. A rich knowledge base is provided by the structured data from Slack, which is processed through various splitters and organized in a searchable format via FAISS vector DB. LangChain then serves as the bridge, enabling the LLMs to access and utilize this knowledge. This synergy allows for the chatbot to not only understand and respond to queries accurately but also to do so with sophistication and contextual awareness.

Scalability and Adaptability

Characterized by power, scalability, and adaptability, the architecture allows for easy adjustments to accommodate changes in data or chatbot requirements. The modular nature of the architecture permits the updating or replacement of individual components without disrupting overall functionality.

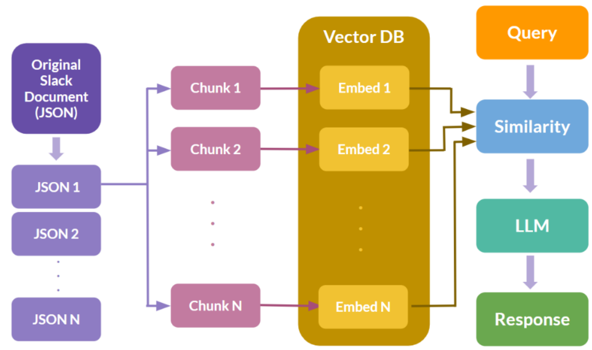


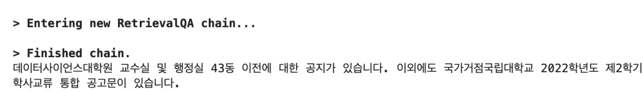
Figure 4: **Diagram illustrating the full architecture of LangChain**

2.5 Experiments: Prompt Engineering

The prompt tuning process in this chatbot architecture is critical so that it directly influences the effectiveness and relevance of the responses generated by the model. Various approaches in prompt tuning are examined, each having distinct influences on the interactions between vector DB and model resulting in distinguishable responses. These methodologies vary from utilizing solely the query to a more comprehensive integration of database content, encompassing both text and metadata, thereby enhancing the depth and precision of the context of the chatbot's responses [6].

A. Implementation of Query-Only Approach

In scenarios where only the query is utilized, the architecture demonstrates a notable limitation in harnessing the database's potential. This approach, while straightforward, often leads to the generation of responses that indicate an inability to access or retrieve relevant information from the database. Such limitations highlight the necessity of integrating database content for a more enriched and informative response system. Fig.5 and Fig.6 show a practical example of this approach.


```
> Entering new RetrievalQA chain...

> Finished chain.
데이터사이언스대학원 교수실 및 행정실 43동 이전에 대한 공지가 있습니다. 이외에도 국가거점국립대학교 2022학년도 제2학기 학사교류 통합 공고문이 있습니다.
```

Figure 5: **Screenshot of results when *prompt template* = *"{query}"*, *Query* =** 대학 관련 공지사항이 무엇이 있나요***?***

```
[Document(page_content='*[데이터사이언스대학원 교수실 및 행정실 43동 이전에 대한 공지 전달]*안녕하세요,오늘 ge
neral  채널에 * 데이터사이언스대학원 교수실 및 행정실 43동 이전 안내 * 공지가 올라왔습니다.자세한 내용은 본 공지를
확인하여 주시고 혹시 이전에 대한 문의사항이 있다면 학생회를 통해 문의해주시기 바랍니다.감사합니다', metadata={'sour
ce': 'students 2023-2', 'seq_num': 2, 'sender_name': 'unknown_user', 'date': '2023-08-0
9'}), Document(page_content='<@U01QNJ7PWRX> 안녕하세요~ 혹시 교수님 면담, 랩실 지원, 배정 결과 공개 등과
관련된 6월 일정을 미리 공지해주실 수 있을까요? 여름방학 계획에 참조하고자 합니다!', metadata={'source': 'rando
m', 'seq_num': 1, 'sender_name': '3기_김소정_94', 'date': '2022-05-18'}), Document(page_conte
nt='국가거점국립대학교 2022학년도 제2학기 학사교류 통합 공고문 안내2022학년도 2학기 국가거점국립대학교 학사교류 일정
을 붙임과 같이 안내하오니 관심있는 분들께서는 붙임을 확인하여 주시기 바랍니다.', metadata={'source': 'general
3', 'seq_num': 2, 'sender_name': 'GSDS행정실', 'date': '2022-07-14'}), Document(page_content
='1기 석사과정 학생들에 해당되는 내용인가요? mysnu에서 졸업신청 메뉴를 못찾겠습니다.', metadata={'source': 'g
eneral 1', 'seq_num': 1, 'sender_name': '3기_박건도', 'date': '2021-09-13'})]
```

Figure 6: **Screenshot of DB contents when *Prompt template = “{query}”, Query =*** 대학 관련 공지사항이 무엇이 있나요***?***

B. Integration of Query with Content-Only Directive

Upon extending the scope to include not just the query but also the content from the database, the chatbot's functionality experiences a significant enhancement. This method involves the utilization of similarity search algorithms to identify and employ data that bears relevance to the query. However, it's crucial to note that this approach confines the chatbot to the boundaries of the database content, excluding any additional contextual information that might be present in the metadata. Such a restriction, while streamlining the response process, limits the depth of the chatbot's understanding and response capabilities. Fig.7 and Fig.8 show a practical example of this approach.

```
> Entering new RetrievalQA chain...

> Finished chain.
제가 찾은 대학 관련 공지사항은 다음과 같습니다:

1. "데이터사이언스대학원 교수실 및 행정실 43동 이전에 대한 공지 전달": 데이터사이언스대학원 교수실 및 행정실이 43동으
로 이전될 예정이며, 자세한 내용은 해당 공지를 확인하시기 바랍니다. 출처는 students 2023-2이며, 보낸 날짜는 2023-
08-09입니다.

2. "6월 일정 관련 문의": 교수님 면담, 랩실 지원, 배정 결과 공개 등과 관련된 6월 일정에 대한 문의가 있었습니다. 출처
는 random이며, 보낸 날짜는 2022-05-18입니다.

위의 공지사항 외에 더 많은 정보가 있을 수 있으니, 필요한 정보가 무엇인지 구체적으로 알려주시면 더 자세한 도움을 드릴 수
있습니다.
```

Figure 7: **Screenshot of results when *Prompt template = “{query} page_content*** 에서 필요한 정보를 찾으세요***.”, Query =*** 대학 관련 공지사항이 무엇이 있나요***?***

```
[Document(page_content='1기 석사과정 학생들에 해당되는 내용인가요? mysnu에서 졸업신청 메뉴를 못찾겠습니다.',
metadata={'source': 'general 1', 'seq_num': 1, 'sender_name': '3기_박건도', 'date': '2021-09-
13'}), Document(page_content='*[데이터사이언스대학원 교수실 및 행정실 43동 이전에 대한 공지 전달]*안녕하세요,
오늘 general  채널에 * 데이터사이언스대학원 교수실 및 행정실 43동 이전 안내 * 공지가 올라왔습니다.자세한 내용은 본
공지를 확인하여 주시고 혹시 이전에 대한 문의사항이 있다면 학생회를 통해 문의해주시기 바랍니다.감사합니다', metadata=
{'source': 'students 2023-2', 'seq_num': 2, 'sender_name': 'unknown_user', 'date': '2023-08
-09'}), Document(page_content='<@U01QNJ7PWRX> 안녕하세요~ 혹시 교수님 면담, 랩실 지원, 배정 결과 공개 등
과 관련된 6월 일정을 미리 공지해주실 수 있을까요? 여름방학 계획에 참조하고자 합니다!', metadata={'source': 'ra
ndom', 'seq_num': 1, 'sender_name': '3기_김소정_94', 'date': '2022-05-18'}), Document(page_co
ntent='안녕하세요 3층 행정실 앞에 앉아있는 성기홍입니다. 다름이 아니라 이번에 데이터사이언스 대학원 홈페이지를 개편하
는데, 학생 전원(1기 2기 모두) 프로필을 홈페이지에 개시할 계획이라 합니다.아마 프로필 공개를 아예 원하지 않는 분들도 계
실거 같고... 아니면 적극적으로 많은 정보를 공개하고 싶은 분도 계실거 같은데요. 우선 아예 프로필 공개를 안하는 옵션은
차교수님 성향상 힘들거 같습니다:sob:그래서 대표단끼리 회의 결과,프로필 필수항목으로 모든 학생분들께사진이름이메일전공비전
다섯가지 정보만 받고, 더 많은 정보 공개를 원하시는 분들은 추가적으로 개인홈페이지, 랩실정보, 직장이력등을 자유롭게 적어
주시면 프로필에 '상세 정보' or 'see more' or '더 보기' 같은 항목을 만들어 사람들에게 엑세스가 가능하게 하려고 합
니다. 정보는 Google docs 파일을 통해 수집할 예정입니다. 1기분들은 작년에 한번 수집했지만, 아마 이번에 새로 수집할
것 같습니다 (업데이트된 내용이 있을 수 있으니).위의 내용에 대해 불만이나 질문사항 있으신 분들은 이 channel에다 자유롭
게 올려주시면 최대한 반영할 수 있도록 할테니 주저하지 마시고 아무얘기나 해주세요! 여기는 학생들 밖에 없는 비밀방이랍니
```

Figure 8: **Screenshot of DB contents *Prompt template = “{query} page_content*** 에서 필요한 정보를 찾으세요***.”, Query =*** 대학 관련 공지사항이 무엇이 있나요***?***

C. Comprehensive Utilization of Query with Full Database Content and Metadata

The most advanced approach in this architecture is the incorporation of both content and metadata from the database, in conjunction with the query. This comprehensive method allows for a more holistic use of the database, enabling the chatbot to conduct a thorough similarity search. Consequently, the chatbot is not only able to access relevant content but also to incorporate additional contextual information into its responses, such as the author of the message, the timestamp, and the specific channel from which the message originated. This enriched approach significantly augments the chatbot's ability to generate more detailed and contextually appropriate responses. Fig.9 and Fig.10 show a practical example of this approach.

```
> Entering new RetrievalQA chain...

> Finished chain.
1. 1기 석사과정 학생들에 해당되는 내용인가요? mysnu에서 졸업신청 메뉴를 못찾겠습니다.
2. 안녕하세요 3층 행정실 앞에 앉아있는 성기홍입니다. 다름이 아니라 이번에 데이터사이언스 대학원 홈페이지를 개편하는데,
학생 전원(1기 2기 모두) 프로필을 홈페이지에 개시할 계획이라 합니다.아마 프로필 공개를 아예 원하지 않는 분들도 계실거
같고... 아니면 적극적으로 많은 정보를 공개하고 싶은 분도 계실거 같은데요. 우선 아예 프로필 공개를 안하는 옵션은 차교수
님 성향상 힘들거 같습니다:sob:그래서 대표단끼리 회의 결과,프로필 필수항목으로 모든 학생분들께사진이름이메일전공비전다섯가
지 정보만 받고, 더 많은 정보 공개를 원하시는 분들은 추가적으로 개인홈페이지, 랩실정보, 직장이력등을 자유롭게 적어주시면
프로필에 '상세 정보' or 'see more' or '더 보기' 같은 항목을 만들어 사람들에게 엑세스가 가능하게 하려고 합니다. 정
보는 Google docs 파일을 통해 수집할 예정입니다. 1기분들은 작년에 한번 수집했지만, 아마 이번에 새로 수집할 것 같습니
다 (업데이트된 내용이 있을 수 있으니).
3. 데이터사이언스대학원 신규 홈페이지 개인 프로필 정보 확인 요청데이터사이언스대학원 신규 홈페이지가 어제 2023.4.24.
오픈되었습니다. 학생회를 통하여 제출하신 이름, 과정명, 연구실, 사진의 정보가 홈페이지(피플-대학원생)에 게재되어 있습니
다. <https://gsds.snu.ac.kr/people/student/>다만, 업체에서 대량의 데이터를 홈페이지에 일괄 입력하다 보니 본
인의 프로필 정보에 이상이 있는지 개별적으로 확인하여 주시기 바랍니다.  또한, 추가로 정보를 업데이트(추가 제출, 사진 추
가 등)하고자 하는 경우에도 행정실로 연락(<mailto:changdae@snu.ac.kr|changdae@snu.ac.kr>) 바랍니다.

위의 세 개의 내용 중에서 대학 관련 공지사항은 3번째 내용입니다.
```

Figure 9: **Screenshot of results when *Prompt template = “{query} page_content*** 에서 필요한 정보를 찾으세요***. metadata*** 정보를 가지고 보낸 사람과 출처를 활용하세요***.*** 해당 정보를 출력할 필요는 없습니다***.”, Query =*** 대학 관련 공지사항이 무엇이 있나요***?***

```
[Document(page_content='1기 석사과정 학생들에 해당되는 내용인가요? mysnu에서 졸업신청 메뉴를 못찾겠습니다.',
metadata={'source': 'general 1', 'seq_num': 1, 'sender_name': '3기_박건도', 'date': '2021-09-
13'}), Document(page_content='안녕하세요 3층 행정실 앞에 앉아있는 성기홍입니다. 다름이 아니라 이번에 데이터사
이언스 대학원 홈페이지를 개편하는데, 학생 전원(1기 2기 모두) 프로필을 홈페이지에 개시할 계획이라 합니다.아마 프로필 공
개를 아예 원하지 않는 분들도 계실거 같고... 아니면 적극적으로 많은 정보를 공개하고 싶은 분도 계실거 같은데요. 우선 아
예 프로필 공개를 안하는 옵션은 차교수님 성향상 힘들거 같습니다:sob:그래서 대표단끼리 회의 결과,프로필 필수항목으로 모든
학생분들께사진이름이메일전공비전다섯가지 정보만 받고, 더 많은 정보 공개를 원하시는 분들은 추가적으로 개인홈페이지, 랩실정
보, 직장이력등을 자유롭게 적어주시면 프로필에 '상세 정보' or 'see more' or '더 보기' 같은 항목을 만들어 사람들에
게 엑세스가 가능하게 하려고 합니다. 정보는 Google docs 파일을 통해 수집할 예정입니다. 1기분들은 작년에 한번 수집했지
만, 아마 이번에 새로 수집할 것 같습니다 (업데이트된 내용이 있을 수 있으니).위의 내용에 대해 불만이나 질문사항 있으신
분들은 이 channel에다 자유롭게 올려주시면 최대한 반영할 수 있도록 할테니 주저하지 마시고 아무얘기나 해주세요! 여기는
학생들 밖에 없는 비밀방이랍니다:partying_face:Ex) 나는 내 사진이 싫어서 그냥 Iphone memoji로 올리고 싶다:rag
e:, 비전을 아무도 못 읽는 러시아어로 써도 되나요?, etc.그럼 여러분의 의견 기다리겠습니다 :i_love_you_hand_sig
n:', metadata={'source': 'students 2021', 'seq_num': 1, 'sender_name': '1기_성기홍', 'date':
'2021-03-18'}), Document(page_content='데이터사이언스대학원 신규 홈페이지 개인 프로필 정보 확인 요청데이터사
이언스대학원 신규 홈페이지가 어제 2023.4.24. 오픈되었습니다. 학생회를 통하여 제출하신 이름, 과정명, 연구실, 사진의
정보가 홈페이지(피플-대학원생)에 게재되어 있습니다. <https://gsds.snu.ac.kr/people/student/>다만, 업체에서
대량의 데이터를 홈페이지에 일괄 입력하다 보니 본인의 프로필 정보에 이상이 있는지 개별적으로 확인하여 주시기 바랍니다.
```

Figure 10: **Screenshot of DB contents when *Prompt template = “{query} page_content*** 에서 필요한 정보를 찾으세요***. metadata*** 정보를 가지고 보낸 사람과 출처를 활용하세요***.*** 해당 정보를 출력할 필요는 없습니다***.”, Query =*** 대학 관련 공지사항이 무엇이 있나요***?***

3 Methodology

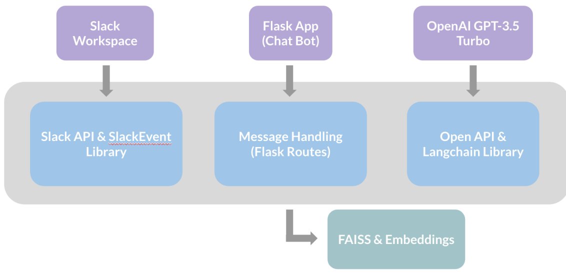


Figure 11: **Diagram illustrating the system architecture, where the Slack API and SlackEventsAPI libraries enable communication between the Flask app (representing the chatbot) and Slack, allowing the app to receive and process Slack events in response to user messages.**

3.1 Chatbot System

The architecture of the chatbot system is designed to seamlessly integrate various components, ensuring a cohesive and efficient conversational experience. The combination of LangChain, Flask, and OpenAI's GPT-3.5 Turbo model forms the backbone of the system. Below is a breakdown of the architecture and how each component contributes to the overall functionality.

LangChain Integration. LangChain acts as a vital intermediary, facilitating the interaction between structured data obtained from Slack and the powerful language model. Its role includes data retrieval, information processing, and response generation. The integration ensures that the chatbot can effectively understand and respond to user queries.

Flask as a Web Framework. Flask is chosen as the web framework due to its simplicity and flexibility. It serves as the glue that binds different components together, enabling seamless communication between Slack, LangChain, and the GPT-3.5 Turbo model. The lightweight nature of Flask makes it an ideal choice for this architecture.

OpenAI GPT-3.5 Turbo for Natural Language Understanding. The chatbot system leverages OpenAI's GPT-3.5 Turbo for natural language understanding and generation. With a carefully tuned temperature parameter and maximum tokens setting, the model enhances the quality of responses. This component is the core engine for generating contextually relevant and coherent replies.

Facebook AI Similarity Search (FAISS) for Information Retrieval. To improve the efficiency of information retrieval, Facebook AI Similarity Search (FAISS) is integrated. FAISS, coupled with LangChain, enables the chatbot to retrieve relevant information from the database. This ensures that user queries are addressed promptly and accurately.

Flask-NGROK for Local Development and Testing. During the development and testing phase, Flask-NGROK is utilized to expose the local Flask application to the internet using ngrok. This allows for real-time testing of the chatbot within the Slack environment. The Slack Event Adapter is configured to receive and handle events from Slack, ensuring seamless integration.

User Interaction and Response Handling. The chatbot interacts with users in the Slack channel. When a user poses a question, the system checks for new queries, processes them, and generates responses using the GPT-3.5 Turbo model. The responses are then posted back to the Slack channel, providing a conversational and interactive user experience.

Continuous Learning and Adaptation. The architecture supports continuous learning and adaptation. The chatbot refines its responses over time by learning from user interactions. LangChain, coupled with the retrieval and generation components, ensures that the chatbot adapts to user preferences and maintains relevancy in its responses.

In summary, the architecture brings together the strengths of LangChain, Flask, GPT-3.5 Turbo, and FAISS to create a powerful and cohesive chatbot system. The integration of these components ensures a smooth and intelligent conversation flow within the Slack platform.

3.2 Code Implementation Details

The provided Python code establishes a Slack bot using the Flask framework and integrates it with the LangChain library. The LangChain library incorporates OpenAI's GPT-3.5 Turbo model for natural language processing and a vector database (FAISS) for efficient information retrieval. The bot responds to user queries posted in a Slack channel.

Slack Integration. The code uses the slack library to interact with the Slack platform. The Flask app is configured with a Slack event adapter to handle incoming events.

Python
```python
import slack
from flask import Flask
from slackeventsapi import SlackEventAdapter
SLACK_TOKEN="<SLACK_TOKEN>"
SIGNING_SECRET="<SIGNING_SECRET>"

app = Flask(__name__)
slack_event_adapter =
SlackEventAdapter(SIGNING_SECRET,
'/slack/events', app)
client = slack.WebClient(token=SLACK_TOKEN)
```

Listing 1: **Integration code in Python using Flask and Slack API for event handling, leveraging SlackEventsAdapter and WebClient with specified Slack token and signing secret.**

LangChain Configuration. LangChain is configured with an OpenAI GPT-3.5 Turbo model and a vector database (FAISS). The RetrievalQA chain is set up to handle question and answering tasks.

```python
from langchain.embeddings import
OpenAIEmbeddings
from langchain.chat_models import
ChatOpenAI
from langchain.chains import RetrievalQA
from langchain import PromptTemplate
from langchain.llms import OpenAI
from langchain.retrievers.self_query.base
import SelfQueryRetriever
from langchain.vectorstores import FAISS

os.environ["OPENAI_API_KEY"] =
"<OPENAI_API_KEY>"
index_base_dir = './faiss_index'

embeddings =
OpenAIEmbeddings(model='text-embedding-ada
-002')
db = FAISS.load_local(index_base_dir,
embeddings)

chatbot_chain =
RetrievalQA.from_chain_type(
    llm=ChatOpenAI(
        temperature=0.5,
model_name='gpt-3.5-turbo',
max_tokens=2000
    ),

retriever=db.as_retriever(search_kwargs={"
k": 3}),
    verbose=True,
    chain_type="stuff",
    chain_type_kwargs={
        'document_prompt': PromptTemplate(

input_variables=["page_content",
"sender_name", "source", "Date"],

template="내용:\n{page_content}\n보낸
사람:{sender_name}\n출처:{source}\n보낸
날짜:{date}"
            )
        },
)
```

Listing 2: **Python code snippet showcasing the integration of LangChain library for building a Retrieval Question Answering (QA) chatbot. It utilizes OpenAI's GPT-3.5 Turbo model, FAISS for vector storage, and a custom chain configuration for document retrieval and response generation.**

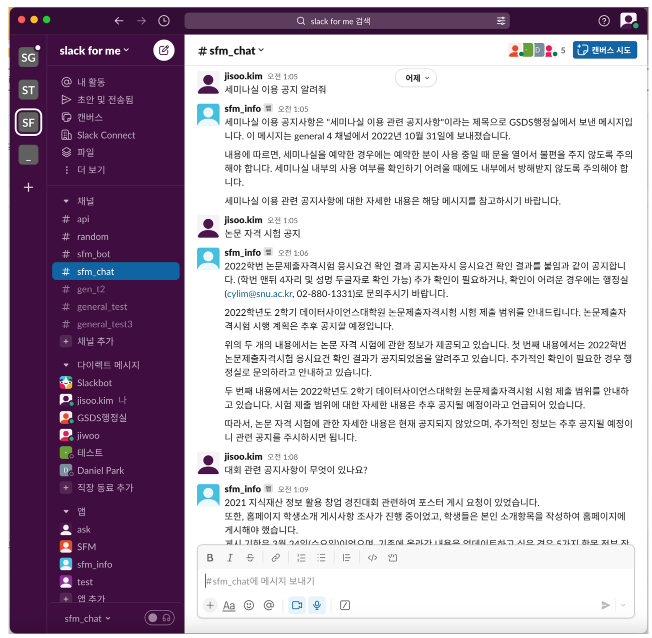


Figure 12: **Screenshot capturing real-time interaction on a Slack channel where a user posts a question, and the chatbot responds dynamically.**

4 Conclusions

Chatbots stand as a fundamental use case for Large Language Models (LLMs). The key functionality of chatbots lies in their ability to engage in prolonged conversations and access information that users seek. In addition to the primary prompt and LLM, memory and search are integral components of a chatbot. In the LangChain official website, various possible uses of LangChain in chatbot services are presented [7].

This research is the first attempt to create a GSDS Slack chatbot, which increases utilization in relation to the actual demand of GSDS community members. We were able to find a lot of reference materials related to chatbot services for English text, but importing and restoring data from Korean-language channels without damage was a challenging task.

Another challenge lies in the limitation of token limit settings, resulting in restricted similarity search configurations. Additionally, fine-tuning parameters like temperature pose another hurdle. Issues such as hallucination and the generation of inaccurate information, which are commonly known challenges in ChatGPT, also to be considered.

When using community channels, there are instances where the information posted in the channel itself is needed, but there may also be cases where one wishes to inquire with the person responsible for handling relevant announcements. In such situations, a problem arose where the chatbot could not retrieve the relevant information even when queried if the name of the responsible person was not present in the text entered into the database. To address this issue, we implemented a system to store metadata, including the author's name, along with the text. Through this enhancement, the chatbot became capable of returning the name and ID of the author who wrote the text containing the keyword, even when the author's name was not explicitly present in the text of the relevant document. While this solution may have its limitations, it proved to be a valuable alternative for overcoming the existing constraints. When searching for information related to specific keywords, the system relied on general data rather than the latest information, resulting in responses that included outdated notices. Although a specific solution for this issue hasn't been implemented yet, we are exploring improvements by broadening the use of metadata. This enhancement aims to enable searches that display results in reverse chronological order.

In conclusion, we conducted various experiments, including text split methods, data processing approaches, and diverse vector databases, to implement a personalized chatbot service for community members using LangChain. Although there are some challenges to address, we have achieved some positive progress (Figure 12). We look forward to discussing further improvements for specialized information exploration in Korean channels and specific communities in the future.

REFERENCES

[1] https://python.langchain.com/docs/use_cases/question_answering/

[2] https://github.com/rusq/slackdump

[3] https://python.langchain.com/docs/modules/data_connection/retrievers/parent_document_retriever

[4] https://python.langchain.com/docs/modules/data_connection/document_loaders/file_directory

[5] https://python.langchain.com/docs/integrations/vectorstores/faiss

[6] https://python.langchain.com/docs/modules/data_connection/document_loaders/json

[7] https://python.langchain.com/docs/use_cases/chatbots