sLLM과 RAG을 활용한 로컬 기반 질의응답 시스템 구현

(Implementation of a Q&A System Based on Local Systems Using sLLM and RAG)

이 상 원*, 서 효 덕, 정 철 우, 권 오 언, 천 승 만
경북IT융합산업기술원
(Sangwon Lee, Hyoduck Seo, Cheolwoo Jung, Oeon Kwon, Seung-Man Chun)
(Gyeongbuk Institute of IT convergence Industry Technology)

Abstract : 본 논문은 Smaller Large Language Model(sLLM)과 RAG(Retrieval-Augmented Generation)을 활용한 로컬 기반 질의응답(Q&A) 시스템의 구현 방법을 다룬다. HuggingFace의 BAAI/bge-m3 모델을 이용해 문서를 임베딩하고 FAISS 벡터 저장소에 저장하여 사용자가 업로드한 파일을 기반으로 질문에 답변하는 방식이다. Streamlit을 통해 사용자 인터페이스를 제공하며, LangChain을 사용해 질문-답변 체인을 구성한다. 본 연구는 로컬 및 저사양 환경에서도 효율적으로 작동하는 sLLM기반 Q&A 시스템을 목표로 하였다.

Keywords : sLLM, RAG, Local Systems, Q&A System, Document Embedding, Vector Search

Ⅰ. 서 론

본 논문은 Smaller Large Language Model(sLLM)과 Retrieval-Augmented Generation(RAG)을 활용해 로컬 및 저사양 환경에서 작동 가능한 질의응답(Q&A) 시스템을 구현하는 방법을 다룬다. 최근 인공지능(AI)과 자연어 처리(NLP) 기술의 발전으로 Q&A 시스템이 다양한 분야에서 주목받고 있으며, 특히 사용자와의 자연스러운 상호작용을 통해 정보 접근성을 높인다. 로컬 시스템은 자원이 제한적이므로 로컬 및 저사양 환경에서 효율적으로 작동하는 Q&A 시스템이 필요하다. 본 연구는 sLLM 모델과 RAG을 이용해 문서 임베딩 및 검색을 구현하여 클라우드 없이도 신속한 질의응답을 제공하는 시스템을 구축하였다[1-2].

Ⅱ. 본 론

1. 시스템 구성

본 연구에서 제안하는 시스템은 크게 세 가지 모듈로 구성된다. 파일 임베딩 및 검색 모듈, 질의응답 인터페이스, 그리고 답변 생성 모듈이다. 이러한 모듈은 Streamlit과 같은 간단한 웹 프레임워크를 통해 통합되며, 사용자는 파일을 업로드하고 질문을 입력하는 방식으로 시스템과 상호작용할 수 있다.

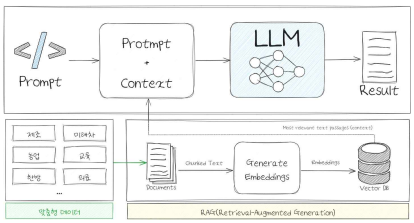

그림 1. RAG 기반 질의응답 시스템 구조
Fig. 1. The structure of the RAG-based Q&A system

1.1 파일 임베딩 및 검색

질의응답 시스템의 핵심은 문서에서 정보를 검색하고 이를 기반으로 답변을 제공하는 것이다. HuggingFace의 BAAI/bge-m3 모델을 사용해 문서를 임베딩하고 FAISS를 통해 벡터 기반 검색을 수행한다. 문서는 작은 단위로 나뉘어 벡터화되어 저장되며 질문과 유사한 벡터를 신속하게 검색할 수 있다. 파일(PDF, TXT, DOCX)은 UnstructuredFileLoader로 로드된 후 RecursiveCharacterTextSplitter로 나뉘어 임베딩되며 FAISS에 저장된다. 이 과정은

*Corresponding Author (swlee@gitc.or.kr)
이상원: (재)경북IT융합산업기술원
※ 본 논문은 2024년 정부(과학기술정보통신부)의 재원으로 한국지능정보사회진흥원의 지원을 받아 수행된 연구임.(2024년 지역특화형 초거대 AI 클라우드 팜 실증 및 확산 환경 조성 사업)

저사양 환경에서도 효율적으로 수행되며 FAISS는 대규모 데이터에서 빠른 검색을 제공하여 응답 시간을 단축한다.

Algorithm 1 Document Embedding and Vector Store Creation

Input: File and embedding model configuration
Output: Vector store for document retrieval
Create a cache directory for embeddings using the file name
Initialize text splitter with:
Chunk size: 500
Chunk overlap: 50
Separators: ["\n\n", "\n", "(?<=\.)", " ", ""]
Length function: len
Load and split the document using UnstructuredFileLoader with the text splitter
Set model name for embeddings as "BAAI/bge-m3"
Set model parameters, such as:
Device: "cuda"
Set encoding parameters, such as:
Normalize embeddings: True
Create HuggingFaceEmbeddings with the model and encoding parameters
Cache the embeddings using CacheBackedEmbeddings with the embedding model and cache directory
Create FAISS vector store from the document chunks using the cached embeddings

그림 2. 문서 임베딩 및 벡터 저장소 생성
Fig. 2. Document embedding and vector store creation

1.2 질의응답 인터페이스

질의응답 시스템의 사용 편의성을 높이기 위해 본 연구에서는 Streamlit 기반 웹 인터페이스를 통해 사용자가 파일을 업로드하고 질의응답을 받을 수 있도록 설계하였다. 파일이 업로드되면 시스템은 자동으로 임베딩과 검색을 수행하여, 사용자가 질문을 입력하면 FAISS를 통해 가장 적합한 문서를 검색하고 답변을 생성한다. 이 인터페이스는 실시간 응답 기능을 제공하여 복잡한 설정 없이도 사용자가 쉽게 시스템을 이용할 수 있게 한다.

Algorithm 2 File Upload and User Input Handling

Input: File and user input
Output: Response to user query
Create file upload interface (allowed file types: pdf, txt, docx)
if a file is uploaded **then**
    Call embed_file(file) to initialize the retriever
end if
Create user input interface
if user provides input **then**
    Call add_history("user", user_input) to record the input
    Display the user input on the screen
end if

그림 3. 파일 업로드 및 사용자 입력 처리
Fig. 3. File upload and user input handling

1.3 답변 생성 및 체인 구성

질의응답 시스템에서 단순 문서 검색뿐만 아니라, 검색된 문서의 내용을 기반으로 정확한 답변을 생성하는 RAG(Retrieval-Augmented Generation) 기법을 도입하였다. RAG은 질문과 검색된 문서를 결합해 문맥을 반영한 답변을 생성하며 LangChain을 사용하여 구현된다. Ollama 통해 sLLM모델을 구동하며 질문 입력 후 업로드된 문서에서 관련 내용을 검색하고 문맥에 맞는 답변을 제공하는 체인이 작동한다. 이 구조는 사용자가 간편하게 질의응답을 수행할 수 있도록 하며, RAG는 문서 기반 질의응답에서 우수한 성능을 발휘해 정확한 답변을 제공한다.

Algorithm 3 RAG Chain Creation and Execution

Input: RAG prompt template, retriever, and user question
Output: AI-generated response based on the retriever context
Initialize the prompt using ChatPromptTemplate with RAG prompt template
Create the RAG chain as follows:
{ "context": Use retriever to format the retrieved documents
"question": Pass the user question using RunnablePassthrough() }
Pass the context and question to the prompt
Send the prompt to the Ollama model
Parse the model's response using StrOutputParser()
The RAG chain now streams the answer to the user's question

그림 4. RAG 체인 생성 및 실행
Fig. 4. RAG chain creation and execution

2. 성능 평가 및 결과

이 시스템은 로컬 및 저사양 환경에서도 효율적으로 작동하며, 문서 임베딩 및 검색 기능을 통해 신속한 질의응답을 가능하게 한다.

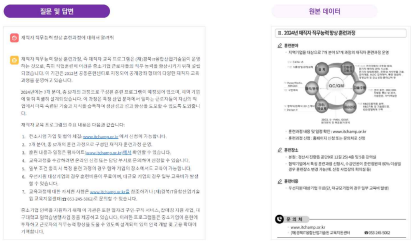

그림 5. 질의응답 시스템 실행 및 결과
Fig. 5. Q&A system execution and results

Ⅲ. 결 론

본 논문에서는 Local sLLM과 RAG을 활용한 로컬 기반 질의응답 시스템을 구현하였다. 향후 연구에서는 더 다양한 파일 형식과 복잡한 데이터셋에 대한 적용 및 시스템 성능 최적화에 집중할 예정이다.

References

[1] Zhao, Wayne Xin, et al. "A survey of large language models." arXiv preprint arXiv:2303.18223 (2023).

[2] Lewis, Patrick, et al. "Retrieval-augmented generation for knowledge-intensive nlp tasks." Advances in Neural Information Processing Systems 33 (2020): 9459-9474.