

학사 데이터 기반의 생성형 AI 챗봇 서비스에 관한 연구

(Research on Generative AI Chatbot Service based on Academic Data)

이준환*

(June-Hwan Lee)

요약

최근 대규모 언어 모델(LLM)의 발전은 텍스트 생성, 번역, 감정 분석 등의 분야에서 우수한 성과를 거두며 대화형 챗봇 시스템에도 큰 변화를 일으켰다. 그러나 LLM 기반의 챗봇은 환각(Hallucination)과 같은 문제로 인해 신뢰성 있는 응답 제공에 한계가 있으며, 특히 학사 데이터와 같이 정확성과 최신성이 중요한 영역에서는 이러한 문제 해결이 더욱 요구된다. 본 연구는 검색 증강 생성(RAG: Retrieval-Augmented Generation) 구조를 활용하여, 챗봇이 내부에 저장된 학습 데이터뿐 아니라 외부의 신뢰할 수 있는 정보 소스로부터 필요한 데이터를 실시간으로 검색하고, 이를 바탕으로 답변을 생성하는 방식으로 학사 데이터를 기반으로 한 생성형 AI 챗봇 시스템의 설계와 구현 방안을 제안한다. 이를 통해 학사 데이터의 특성에 맞춘 정확하고 신뢰성 높은 응답 제공을 목표로 한다. 학사 데이터를 기반으로 한 생성형 AI 챗봇이 비공개 데이터 환경에서도 안전하고 효율적으로 적용될 수 있는 질의응답 시스템의 구현 가능성을 제시하며, 이를 통해 사용자들이 필요로 하는 학사 정보를 신속하게 접근하고 활용할 수 있도록 한다.

■ 중심어 : 생성형 AI ; 챗봇 ; 벡터 임베딩 ; 검색증강생성 ; 환각

Abstract

Recent advances in large-scale language models (LLMs) have brought about significant changes in conversational chatbot systems by achieving excellent results in areas such as text generation, translation, and sentiment analysis. However, LLM-based chatbots have limitations in providing reliable responses due to problems such as hallucination, and this problem needs to be solved even more in areas such as bachelor's data where accuracy and recency are important. This study proposes a design and implementation method of a generative AI chatbot system based on bachelor's data by utilizing the Retrieval-Augmented Generation (RAG) structure, which allows the chatbot to retrieve necessary data in real time from not only internally stored learning data but also external reliable information sources, and generate answers based on them. Through this, the goal is to provide accurate and reliable responses tailored to the characteristics of bachelor's data. It suggests the possibility of implementing a question-answering system that can be safely and efficiently applied even in a non-public data environment by using a generative AI chatbot based on bachelor's data, and through this, allows users to quickly access and utilize the bachelor's information they need.

■ keywords : Generative AI ; Chatbot ; Vector Embedding ; Retrieval Augmented Generation ; Hallucination

I. 서론

최근 자연어 처리(NLP : Natural Language Processing) 기술은 대규모 언어 모델(LLM :

*정회원, 극동대학교 에너지IT공학과

접수일자 : 2024년 11월 18일

게재확정일 : 2024년 11월 28일

Large Language Models)의 발전과 함께 급속히 진보하며, 인공지능의 새로운 가능성을 열어가고 있다[1,2]. 특히 GPT(Generative Pre-trained Transformer)와 같은 생성형 AI 모델들은 텍스트 생성, 번역, 감정 분석 등 다양한 분야에서 우수한 성능을 보이며, 특히 대화형 챗봇 시스템에서 인간과 자연스럽게 상호작용할 수 있는 기반을 제공하고 있다[1,3]. 이로 인해 거대 언어 모델 기반의 챗봇은 고객 상담, 학습 지원, 정보 제공 등에서 빠르게 그 영역을 확장해가며 일상 속의 중요한 도구로 자리잡고 있으나, 모든 상황에서 신뢰성 있는 응답을 제공하는 것은 아니다[4,5]. 특히 학습한 데이터에 기반해 응답을 생성하는 과정에서 발생하는 환각(Hallucination) 문제는 모델이 실제와 다른 비현실적인 정보를 만들어내게 하며, 정보의 신뢰성을 저해하는 요인으로 작용한다[6,7]. 이는 챗봇이 필요한 순간에 적절하고 정확한 정보를 제공하는 데에 한계를 드러내기도 한다. 이러한 한계를 보완하기 위해 최근 검색 증강 생성(RAG: Retrieval-Augmented Generation) 방식이 주목받고 있다. RAG는 AI 모델이 내부에 학습한 지식에만 의존하는 것이 아니라, 외부의 신뢰할 수 있는 정보 소스로부터 필요한 정보를 검색하여 응답을 생성하는 방식이다[6,7]. 이를 통해 챗봇은 더 신뢰성 있는 정보와 출처에 기반해 답변을 제공할 수 있다.

이와 같은 생성형 AI의 발전과 함께, 국내외 여러 기업과 기관들은 생성형 AI를 업무에 도입하려는 움직임을 보이고 있다[8]. 국내의 한 IT 기업이 실시한 조사에 따르면, 국내 기업과 기관 중 많은 기업이 이미 생성형 AI를 도입해 활용하고 있거나, 도입을 고려 중이다. 그러나 범용 LLM을 사용할 때 발생하는 ‘부정확한 정보 제공’과 ‘보안 문제’ 등으로 인해 도입을 망설이는 경우도 상당하다. 이는 교육 공공 부문에서 더욱 큰 이슈로 나타나고 있으며, 부정확한 정보 제공과 기밀 정보의 유출 위험성으로 인해 일부 대기

업들은 ChatGPT 사용을 제한하고, 공공 기관에서는 내부망과 분리된 생성형 AI 구축이 요구되고 있다[9].

본 연구에서는 학사 데이터를 기반으로 한 생성형 AI 챗봇 시스템 구축을 제안한다. 학사 데이터는 실시간으로 변화하며 정확한 정보를 제공하는 것이 중요한 영역이기 때문에, 기존의 범용 AI 모델과는 차별화된 접근이 필요하다. 본 연구는 RAG 구조를 도입해 챗봇이 학사 관련 지식베이스에서 질문에 연관된 정보를 검색한 후, 이를 바탕으로 신뢰성 있는 응답을 제공할 수 있도록 한다. 이를 위해 한국어 문장 임베딩 모델과 생성형 언어 모델을 선정하여 구축하고, 학사 정보의 업데이트와 관리가 효율적으로 이루어질 수 있는 데이터베이스 구조를 설계한다.

따라서 본 논문은 학사 데이터를 기반으로 한 생성형 AI 챗봇이 보다 정확하고 신뢰성 있는 응답을 제공하기 위한 구조적 방안을 제시하고자 하며, 이를 통해 사용자들이 필요한 학사 정보를 신속하고 편리하게 활용할 수 있는 시스템을 구현할 수 있는 기초 자료를 제공하고자 한다. 또한 비공개 데이터 환경에서도 안전하고 효율적으로 활용할 수 있는 AI 기반 질의응답 시스템의 가능성과 실질적 활용 방안을 제시한다.

II. 관련 연구

1. 거대 언어 모델(LLM)

LLM은 인간의 언어를 이해하고 생성하는 데 특화된 대규모 인공지능 신경망으로, 주로 트랜스포머(Transformer) 아키텍처에 기반하여 구축된다[1,2]. 트랜스포머 모델은 순차적 데이터의 학습에 효율적인 구조로서, 대량의 언어 데이터를 학습하여 문맥적 의미와 구조를 정밀하게 파악할 수 있도록 설계되었다[10]. LLM은 주어진 텍스트 입력을 바탕으로 다음 단어 예측, 문장 생성, 텍스트 요약, 번역, 코드 생성 등 다양한 자연어 처리(NLP) 작업에서 뛰어난 성능을

발휘한다[8]. 대표적으로 OpenAI의 ChatGPT, 구글의 Bard, 메타의 LLaMA 등이 있으며, 국내에서는 네이버클라우드의 하이퍼클로바X와 같은 모델들이 활발히 연구되고 있다[11-14].

LLM은 여러 신경망 계층을 통해 텍스트를 이해하고 생성하는 구조로, 임베딩 계층, 피드포워드 계층, 어텐션 계층, 순환 계층 등으로 이루어진다[15]. 각 계층의 역할은 다음과 같다.

표 1. 간판 문자의 인식 정확성 및 누적 정확도

계층	설명
임베딩 계층	텍스트를 수치화하여 모델이 의미를 포착할 수 있는 벡터 형태로 변환하고 이를 통해 모델은 단어의 의미와 문맥을 이해
피드포워드 계층	여러 연결된 계층을 통해 고차원적 의미를 학습하며, 텍스트의 추상적 의미를 인식
어텐션 메커니즘	특정 단어나 구절에 집중하여 문맥적 중요도를 반영하는 방식으로, 텍스트의 중요한 부분을 인식하도록 함
순환 계층	텍스트의 순서와 단어 간 연결 관계를 이해하고 해석하여 문맥을 더욱 정확하게 분석

LLM은 이러한 다양한 계층을 통해 학습한 대규모 데이터에서 의미와 문법을 이해하고 생성함으로써, 다양한 NLP 과제를 수행할 수 있는 언어 처리 모델로 발전했다[13,14]. 그러나 LLM은 대규모 데이터에서 학습하지 않은 정보를 생성하거나 부정확한 정보를 제시하는 환각 현상이라는 문제점이 있다. 환각은 모델이 기존 학습 데이터에 없는 정보를 창출할 때 발생하는 현상으로, 특히 최신 데이터에 대한 응답이나 미학습 영역에서 발생할 수 있다[15,16]. 이를 해결하기 위해 LLM을 보완하는 다양한 연구가 진행되고 있다.

LLM은 텍스트 생성뿐만 아니라 이미지 생성, 음악 생성, 코드 작성과 자동 완성, 음성 생성 등 다양한 분야에서 사용되고 있다[11,14]. 특히 최신의 GPT-3.5와 같은 모델은 대규모 매개변수로 학습하여 사람처럼 대화하는 인공지능으로 발전하였으며, ChatGPT와 같은 상용 서비스로

제공되고 있다[14]. LLM은 인간의 언어 처리 능력을 모방한 인공지능 발전의 상징으로, 그 응용 범위와 성능은 앞으로도 지속적으로 확장될 것으로 예상된다.

2. 벡터 검색(Vector Search)

벡터 검색은 자연어처리(NLP)와 머신러닝 분야에서, 입력된 데이터를 고차원 벡터 형태의 임베딩으로 변환하여 의미 기반의 검색을 수행하는 방식으로, 기존의 키워드 기반 검색의 한계를 보완한다[17]. 이러한 벡터 검색 방식은 특히 RAG 시스템에서 중요한 역할을 하며, 벡터 데이터베이스(Vector Database, VectorDB)와 결합하여 대규모 데이터의 의미론적 관계를 바탕으로 효율적이고 정밀한 정보 검색을 가능하게 한다[6,7]. 벡터 데이터베이스는 대규모 임베딩 벡터를 저장하고 유사도 기반으로 가장 근접한 벡터를 빠르게 찾아내어, LLM이 보다 신뢰성 있는 정보에 기반해 답변을 생성할 수 있도록 한다.

벡터 데이터베이스는 고차원 벡터를 효율적으로 저장, 검색 및 관리하는 기능을 제공하며, 특히 LLM의 한계 중 하나인 장기 기억 부족 문제를 보완하는 데 중요한 역할을 한다[17,18]. 벡터 데이터베이스를 통해 LLM이 문맥을 이해하고 대규모 데이터셋을 활용하여 보다 정밀한 정보를 제공할 수 있다[19]. 기존의 데이터베이스와 달리 벡터 데이터베이스는 수백 개의 차원으로 구성된 고차원 벡터를 다루며, 이러한 벡터의 각 차원은 객체의 특정 속성을 반영한다. 이와 같이 고차원 벡터 데이터는 유사도 기반 검색을 가능하게 하여, 이미지와 텍스트 등의 비정형 데이터뿐만 아니라 반정형 데이터에 대한 검색에도 효과적이다[18,20]. 벡터 데이터베이스와 임베딩 검색 방식을 통합한 RAG 시스템은 고차원 벡터 데이터를 효율적으로 활용하여 AI 모델의 응답 정확도를 높이는 데 기여하며, 텍스트, 이미지 및 기타 비정형 데이터의 의미를 포착해 정교한 검색 결과를 제공한다[7].

RAG 시스템에서 벡터 검색은 특히 임베딩(Embedding)을 활용하여 효율적이고 정확한 검색을 가능하게 하며, 다음과 같은 임베딩 생성 단계, 벡터 인덱싱(Indexing) 및 데이터베이스 구축 단계, 유사성 검색(Querying) 단계, 사후 처리(Post Processing) 및 결과 반환 단계로 구성된다[6,7]. 첫 번째 임베딩 생성 단계에서는 고차원 실수 벡터로, 텍스트의 의미를 수치적으로 표현하는 벡터를 생성하는 과정이다. 이 과정에서 텍스트의 문맥, 어휘 의미, 구문 패턴을 포함한 다양한 정보가 반영된다. 이를 통해 비슷한 의미를 가진 텍스트는 벡터 공간에서 가까운 위치를 차지하게 된다. RAG에서는 이 임베딩을 생성하기 위해 주로 대규모 언어 모델이나 특정 도메인에 맞춰 학습된 모델을 사용하여, 보다 정확하고 의미 기반의 임베딩을 생성한다[21,22]. 두 번째 벡터 인덱싱(Indexing) 및 데이터베이스 구축 단계에서는 생성된 임베딩을 효율적으로 검색하기 위해 벡터 데이터베이스(Vector Database)에 인덱싱한다. 벡터 인덱싱은 대규모 임베딩 데이터에 대한 빠르고 효율적인 검색을 가능하게 하는 구조를 만드는 작업이다. 일반적으로 인덱싱에는 PQ, LSH, HNSW와 같은 알고리즘이 사용된다. 이를 통해 RAG 모델은 방대한 양의 텍스트 데이터를 벡터 형태로 저장하고, 쿼리에 대한 신속한 검색을 지원하는 구조를 갖추게 된다[22,23]. 세 번째 유사성 검색 단계에서는 사용자의 쿼리 임베딩을 데이터베이스의 임베딩과 비교하여 가장 유사한 벡터를 찾는 과정이다. 이 단계에서는 내적, 코사인 유사도, 유클리디안 거리 등의 측정 지표를 사용하여 유사도를 평가한다. RAG에서는 쿼리와 가장 높은 유사도를 기록한 임베딩을 가진 문서들을 검색하여 사용자의 쿼리에 가장 관련성 높은 정보를 가져온다[21-23]. 이때 설정한 수에 따라 상위 유사도를 보인 문서들을 선택하여, 다음 단계에서 생성 과정에 사용한다. 마지막으로 사후 처리 및 결과 반환 단계에서는 선택된 유사 문서들은 최

종 결과로 전달되기 전 추가적인 사후 처리를 거칠 수 있다. 이 과정에서는 임베딩의 유사도 점수를 재조정하거나 특정 기준에 따라 문서 순위를 변경하여 최적의 검색 결과를 반환한다. 이를 통해 RAG 모델은 사용자의 요청에 부합하는 가장 의미 있고 관련성 높은 정보를 생성 응답에 반영하게 된다.

III. 시스템 구현 및 결과

본 논문에서는 PDF 파일을 로드하여 청킹과 임베딩 과정을 거쳐 벡터 데이터베이스에 데이터를 저장한다. 사용자가 질의 하면 벡터 데이터베이스에서 유사한 정보를 찾고, LLM을 통해 정확한 응답을 생성할 수 있도록 그림 1과 같이 시스템을 구성 하였다.

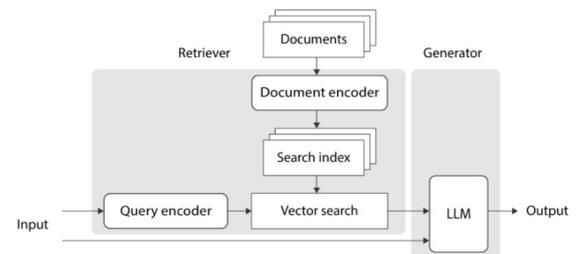


그림 1. 학사 데이터 기반 생성형 AI 챗봇 서비스 구성

1. 문서 로드 및 텍스트 추출

이 단계에서는 챗봇이나 애플리케이션에 사용할 데이터를 준비하고 효율적인 검색을 위해 구조화하는 작업을 수행한다. 다양한 형식의 문서를 로딩하여 텍스트로 변환하며, PDF 문서는 PyPDFLoader를 사용해 로드하고 텍스트로 변환하여 활용한다. PDF 파일의 내용을 페이지별로 추출해 리스트 형태로 저장함으로써, 학사 데이터와 같은 도메인 문서의 모든 내용을 구조화된 데이터로 정리한다. 이렇게 페이지 단위로 구분된 텍스트는 이후 검색 및 응답 생성에서 문맥 정보를 유지할 수 있도록 하며, 임베딩 생성과 유사도 검색의 대상이 된다.

```
# PDF 파일 로드 및 텍스트 추출
def load_and_extract_text(pdf_path):
    loader = PyPDFLoader(pdf_path)
    documents = loader.load()
    # 페이지별 텍스트 추출
    page_texts = [page.page_content for page in documents]
    return page_texts

# 학사 데이터 PDF 로드 예시
pdf_path = "path/to/academic_data.pdf"
page_texts = load_and_extract_text(pdf_path)
print(page_texts) # 페이지별 텍스트 출력
```

그림 2. 문서 로드 및 추출

2. 텍스트 청킹

텍스트 청킹 단계에서는 페이지별로 분할된 텍스트를 모델이 처리할 수 있는 적절한 크기(예: 100-500 토큰)로 나눈다. 이 과정을 통해 생성된 청크는 임베딩 및 유사도 검색에 적합하게 된다. 너무 긴 텍스트는 LLM이 처리하기 어려우므로 적절한 크기로 나누는 작업이 필요하다. 본 연구에서는 청크 크기를 500, 청크 오버랩을 50으로 설정하여 텍스트를 분할하였다. 청크 오버랩을 사용함으로써 문서의 의미적 일관성을 유지하면서도 효율적으로 텍스트를 나눌 수 있다.

텍스트 청킹 방법에는 다양한 옵션이 있으며, 문장 단위로 분할, 특정 기호를 기준으로 분할, 고정 길이로 분할 방법 등이 있다. 본 논문에서는 지정된 길이와 오버랩을 유지하면서 텍스트를 분할하여 청크 간에 중복된 부분을 포함해 문맥을 유지하고 텍스트의 의미 손실을 방지하였다.

```
# 텍스트 청킹 함수
def chunk_text(texts, chunk_size=500, chunk_overlap=50):
    text_splitter = RecursiveCharacterTextSplitter(
        chunk_size=chunk_size, chunk_overlap=chunk_overlap
    )
    chunks = []
    for text in texts:
        chunks.extend(text_splitter.split_text(text))
    return chunks

# 페이지별 텍스트를 청킹
text_chunks = chunk_text(page_texts)
print(text_chunks[:5]) # 첫 5개의 청크 출력
```

그림 3. 텍스트 청킹

3. 임베딩

임베딩은 텍스트의 의미를 고차원 벡터로 변환하는 단계이다. 텍스트가 임베딩되면, 각 텍스트 청크는 벡터 형태로 변환되어 머신러닝 모델에서 계산할 수 있는 수치 데이터가 된다. 벡터는

각 단어 또는 문장의 의미를 수치적으로 표현하고, 유사도 계산을 통해 텍스트 간의 관계를 분석할 수 있게 한다. 임베딩 단계는 RAG 시스템의 중요한 부분으로, 벡터화된 텍스트가 벡터 데이터베이스에 저장되면 유사도 검색을 통한 문서 검색이 효과적으로 수행된다.

```
# 텍스트 청크 임베딩 생성
def generate_embeddings(text_chunks):
    embeddings = []
    for chunk in text_chunks:
        response = openai.Embedding.create(input=chunk, model="text-embedding-ada-002")
        embeddings.append(response['data'][0]['embedding'])
    return embeddings

# 임베딩 생성
text_embeddings = generate_embeddings(text_chunks)
print(text_embeddings[:5]) # 첫 5개 임베딩 벡터 출력
```

그림 4. 임베딩

4. 벡터 저장

생성된 벡터를 벡터 데이터베이스(Vector DB)에 저장하는 단계이다. 벡터 데이터베이스는 임베딩 벡터를 효율적으로 저장하고 관리하며, 빠르게 검색할 수 있도록 인덱싱 기능을 제공한다. 본 논문에서는 벡터 데이터베이스를 사용하여 임베딩 벡터를 저장하고, 이를 통해 대규모 데이터에서도 빠르고 정확하게 유사도 검색을 수행할 수 있도록 하였다. 저장된 벡터들은 이후 검색 단계에서 사용자의 입력 질문과 유사한 벡터를 찾는 데 사용된다.

```
# Pinecone 초기화 및 벡터 데이터베이스 생성
pinecone.init(api_key=os.getenv("PINECONE_API_KEY"), environment="us-west1-gcp")
index = pinecone.Index("academic_data")

# 벡터 삽입
def save_embeddings_to_db(embeddings, text_chunks):
    for i, (embedding, chunk) in enumerate(zip(embeddings, text_chunks)):
        index.upsert([(str(i), embedding, {"text": chunk})])

save_embeddings_to_db(text_embeddings, text_chunks)
print("임베딩 벡터가 벡터 데이터베이스에 저장되었습니다.")
```

그림 5. 벡터 저장

5. 검색

검색 단계는 사용자가 입력한 질문을 처리하여 관련 있는 답변을 제공하는 과정이다. 입력된 질문은 임베딩 모델을 통해 벡터로 변환되고, 이 벡터는 벡터 데이터베이스에서 유사도를 비교하여 가장 가까운 벡터를 가진 청크를 검색한다.

유사성 측정 지표로는 코사인 유사도, 유클리디안 거리 등을 사용할 수 있으며, 검색된 청크들은 GPT 모델에 제공되어 질문에 대한 응답 생성을 위해 사용된다.

```
# 질문에 대한 벡터 임베딩 생성
def generate_question_embedding(question):
    response = openai.Embedding.create(input=question, model="text-embedding-ada-002")
    return response['data'][0]['embedding']

# 질문과 유사한 텍스트 검색
def search_similar_texts(question, top_k=5):
    question_embedding = generate_question_embedding(question)
    results = Index.query(question_embedding, top_k=top_k, include_metadata=True)
    return [match['metadata']['text'] for match in results['matches']]

# 예시 질문 검색
question = "학사 일정에 대해 알려줘"
relevant_texts = search_similar_texts(question)
print("질문과 관련된 텍스트 청크:", relevant_texts)
```

그림 6. 검색

최종적으로, GPT는 검색된 문서 청크를 바탕으로 질의에 대한 답변을 생성하여 사용자에게 제공한다.

5. 실험 결과

본 논문에서는 학사 데이터를 기반으로 생성형 AI 챗봇 시스템을 구현하였다. 실험에 사용한 PDF 파일은 대학교의 입시요강을 파일을 사용하였다.

나. 정시모집

구분	2024학년도		2025학년도	
	모집단위	전형요소 반영비율	전형명	모집단위
수능 위주	디자인학과 민화예매이선학과 연극연기 사원체육	수능 20% 실기 80%	일반 전형	디자인학과 민화예매이선학과 연극연기학과 사원체육학과
	실기고사 방법(유형 및 과제) 사회체육학과 - 윗몸일으키기/ 제자리 멀리뛰기/ 10m 왕복달리기		실기 100%	수능 20% 실기 80%

2 전형 일정

구분	일시	장소	비고
원서접수	2024. 09. 09(월) ~ 09. 13(금) 총 3일 이상	인터넷, 장구	
전형기간	2024. 09. 14(토) ~ 12. 12(목)00:00	본교 지정 고사장	
합격자 발표	2024. 12. 13(금)까지	본교 입학홍보단(http://pskdu.ac.kr) 및 개별통보(중환)	
합격자 문서등록(최초합격자)	2024. 12. 16(월) ~ 12. 18(수)09:00	본교 입학홍보단(http://pskdu.ac.kr)	문서등록(예치금과 동일 효과), 본 등록기간에 반드시 등록금을 납부하여야 함
미등록 중원 합격률보 마감	2024. 12. 26(목) (합격자 발표 18시까지)	출제일자 발표 14시까지 14-18시까지는 개별통보만 가능	
미등록 중원 등록 마감	2024. 12. 27(금) 22시까지		
본 등록	2025. 02. 10(월) ~ 02. 12(수)09:00		

그림 7. 로드된 문서파일 일부(1)

표 2. 실험 결과(질의)

구분	내용
질의1	2025학년도 정시모집 사회체육학과 실기고사 방법 알려주세요.
질의2	2025학년도 수시모집 합격자 발표 및 합격자 등록 일자 알려주세요.

표 3. 실험 결과(ChatGPT응답)

구분	ChatGPT 응답
질의1	극동대학교 사회체육학과 2025학년도 정시모집 실기고사 방법은 다음과 같습니다: 실기 평가 종목 1. 윗몸일으키기 2. 제자리 멀리뛰기 3. 10m 왕복 달리기 전형 방법 1. 실기고사 점수: 80% 2. 수능 점수: 20% 관련 일정 1. 실기고사 날짜: 2025년 1월 25일 (목) 2. 장소 공지: 2025년 1월 10일(수)에 극동대학교 입학처 홈페이지를 통해 확인 가능 더 자세한 정보는 극동대학교 입학처 홈페이지에서 확인하실 수 있습니다.
질의2	극동대학교 2025학년도 수시모집 합격자 발표와 등록 일정은 다음과 같습니다: 1. 합격자 발표: 2024년 11월 13일(수) 오후 2시부터 확인 가능합니다. 2. 합격자 등록: 2024년 12월 16일(월) 오전 9시부터 진행됩니다. 자세한 내용은 극동대학교 입학처 홈페이지에서 확인할 수 있습니다: 극동대학교 입학처.

표 4. 실험 결과(제안한 시스템 응답)

구분	제안한 시스템
질의1	2025학년도 정시모집 사회체육학과 실기고사 방법은 다음과 같습니다:\n\n- 배근력\n- 제자리 멀리뛰기\n- 10m 왕복달리기\n- 메디신볼 던지기\n\n*코로나19 등 감염병 확산 시 변경 가능하며, 소질검사와 기초체력 기록 측정이 포함될 수 있습니다.
질의2	2025학년도 수시모집 합격자 발표는 2024년 12월 13일(금)까지이며, 합격자 등록 기간은 2024년 12월 16일(월)부터 12월 18일(수)까지입니다.

실험결과 ChatGPT만을 사용하였을 경우 실제와 다른 정보를 알려주는 것을 확인하였다. 반면에, 제안한 시스템은 사용자가 질의하는 내용에 대한 정확한 답변을 생성하는 것을 확인하였다.

IV. 결론

본 연구에서는 학사 데이터를 기반으로 한 생성형 AI 챗봇 시스템의 설계와 구현 방안을 제안하였다. 최근 LLM의 발전은 다양한 분야에서 우수한 성과를 거두었지만, 환각 문제로 인해 신뢰성 있는 응답을 제공하는 데 한계가 있다. 특히, 학사 데이터와 같이 정확성과 최신성이 중요한 영역에서는 이러한 문제를 해결하는 것이 더욱 중요하다. 이를 해결하기 위해 본 연구는 RAG 구조를 활용하여, 챗봇이 내부 학습 데이터뿐만 아니라 외부의 신뢰할 수 있는 정보 소스로부터 실시간으로 데이터를 검색하고, 이를 바탕으로 정확하고 신뢰성 높은 답변을 생성하는 방법을 제시하였다. 본 논문에서는 학사 데이터 특성에 맞춘 정확하고 신뢰성 있는 응답을 제공할 수 있는 생성형 AI 챗봇 시스템의 가능성을 보여주었다. 또한, 비공개 데이터 환경에서도 안전하고 효율적으로 적용될 수 있는 질의응답 시스템을 구현하는 방법을 제시하였다. 이를 통해 사용자들은 필요한 학사 정보를 신속하게 접근하고 활용할 수 있게 되어, 학사 정보 서비스의 효율성과 정확성을 높이는 데 기여할 것이다.

REFERENCES

- [1] H. Naveed, A.U. Khan, S. Qiu, et al., "A comprehensive overview of large language models," arXiv preprint, vol. arXiv:2307.06435, pp.1-47, 2023.
- [2] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," OpenAI Blog, vol. 1, no. 8, pp. 9, 2019.
- [3] A. Vaswani, et al., "Attention is all you need," Advances in Neural Information Processing Systems, vol. 30, pp. 1-11, 2017.
- [4] T. Brown, B. Mann, N. Ryder, M. Subbiah, J.D.

- Kaplan, P. Dhariwal, et al., "Language models are few-shot learners," Advances in Neural Information Processing Systems, vol. 33, pp. 1877 - 1901, 2020.
- [5] A. Radford, K. Narasimhan, T. Salimans, & I. Sutskever, "Improving language understanding by generative pre-training," <https://openai.com/index/language-unsupervised/>, pp.1-12, 2018.
- [6] J. Chen, H. Lin, X. Han, et al., "Benchmarking large language models in retrieval-augmented generation," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, no. 16, pp. 17754-17762, 2024.
- [7] P. Zhao, H. Zhang, Q. Yu, et al., "Retrieval-augmented generation for AI-generated content: A survey," arXiv preprint, vol. arXiv:2402.19473, pp.1-22, 2024
- [8] J. Devlin, M.W. Chang, K. Lee, & K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," Proceedings of NAACL-HLT, pp.4171 - 4186, Minnesota, June, 2019.
- [9] Soyeon Kim, Hogeon Seo, "Application based on Generative AI and Prompt Engineering to Improve Children's Literacy," Smart Media Journal, Vol. 13, No.8, pp.26-38, Aug. 2024
- [10] Sophot Ky, Kwangtek Na, Ju-Hong Lee, "Incorporating BERT-based NLP and Transformer for An Ensemble Model and its Application to Personal Credit Prediction," Smart Media Journal, Vol. 13, No.4, pp.9-15, Apr. 2024
- [11] G. Son, H. Lee, S. Kim, S. Kim, N. Muennighoff, T. Choi, ... & S. Biderman, "KMMLU: Measuring Massive Multitask Language Understanding in Korean," arXiv preprint, vol. arXiv:2402.11548, pp.1-31, 2024.
- [12] V. Scotti, L. Sbattella, & R. Tedesco, "A primer on seq2seq models for generative chatbots," ACM Computing Surveys, vol. 56, no. 3, pp. 1-58, 2023.
- [13] P. Zhang, N. Shao, Z. Liu, et al., "Extending Llama-3's context ten-fold overnight," arXiv preprint, vol. arXiv:2404.19553, pp.1-5, 2024.
- [14] M. Hutson, "How does ChatGPT 'think'? Psychology and neuroscience crack open AI large language models," Nature, vol. 629, no. 8014, pp. 986-988, 2024.
- [15] S.H. Lee, & K.S. Song, "Prompt engineering to improve the performance of teaching and learning materials Recommendation of Generative Artificial Intelligence," Journal of the Korea Society of Computer and Information, vol. 28, no. 8, pp. 195-204, 2023.
- [16] W.X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, ... & J.R. Wen, "A survey of large language models," arXiv preprint, vol. arXiv:2303.18223,

- pp.1-140, 2023.
- [17] T. Mikolov, K. Chen, G. Corrado, & J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint, vol. arXiv:1301.3781, pp.1-12, 2013.
- [18] J. Pennington, R. Socher, & C.D. Manning, "GloVe: Global vectors for word representation," Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532-1543, 2014.
- [19] Kyunggho Yu, Juhyeon No, Taekeun Hong, Hyoungju Kim, Pankoo Kim, "A Study on Image Generation from Sentence Embedding Applying Self-Attention," Smart Media Journal Vol.10, No.1, pp. 63-69, 2021
- [20] P. Bojanowski, E. Grave, A. Joulin, & T. Mikolov, "Enriching word vectors with subword information," Transactions of the Association for Computational Linguistics, vol. 5, pp. 135-146, 2017.
- [21] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, ... & H. Wang, "Retrieval-augmented generation for large language models: A survey," arXiv preprint, vol. arXiv:2312.10997, pp.1-21, 2023.
- [22] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, ... & D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," Advances in Neural Information Processing Systems, vol. 33, pp. 9459-9474, 2020.
- [23] D. Lin, "Revolutionizing Retrieval-Augmented Generation with Enhanced PDF Structure Recognition," arXiv preprint, vol. arXiv:2401.12599, pp.1-18, 2024.

저 자 소 개



이준환(정회원)

1994년 단국대학교 전자공학과 학사 졸업.

1996년 단국대학교 전자공학과 석사 졸업.

2001년 단국대학교 전자공학과 박사 졸업.

<주관심분야 : 신호처리, 음성인식, 영상인식>