

LangChain 프레임워크

LLM 애플리케이션 개념 이해하기

충북대학교 의과대학
박 승



충북대학교
CHUNGBUK NATIONAL UNIVERSITY

학습 목표

1. 대규모 언어 모델 기반 애플리케이션의 구성 요소와 장단점을 이해한다.
2. 대규모 언어 모델의 주요 한계점과 이를 극복하기 위한 접근법들을 설명할 수 있다.
3. LangChain 프레임워크의 핵심 요소에 대해 설명할 수 있다.
4. 실습 사례를 통해 실제 문제에 적용 가능한 인터랙티브 LLM 서비스를 개발한다.

기존 소프트웨어 vs. 대규모 언어 모델

■ 기존 소프트웨어 기반 애플리케이션 구성 요소

- 클라이언트 (client)
 - 사용자가 애플리케이션을 사용하는 환경을 의미
 - 사용자는 이 계층을 통해 애플리케이션에 요청을 보내고 결과 수신
- 프론트엔드 (frontend)
 - 사용자가 상호작용하는 애플리케이션의 시각적 부분을 담당
 - 사용자에게 데이터를 표시하고, 백엔드와 통신하여 필요한 데이터를 수신
- 백엔드 (backend)
 - 서버 측에서 애플리케이션의 로직과 데이터 처리를 담당
 - 프론트엔드에서 요청한 데이터를 처리하고, 비즈니스 로직을 수행하며 데이터베이스와 상호 작용
- 데이터베이스 (database)
 - 애플리케이션의 데이터를 저장하고 관리하는 계층
 - 백엔드에서 요청한 데이터를 조회, 추가, 수정, 삭제하는 작업을 수행

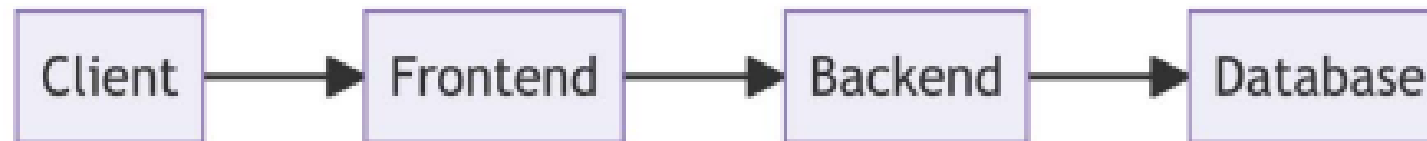


Figure 2.3: A traditional software application

기존 소프트웨어 vs. 대규모 언어 모델

■ 대규모 언어모델 기반 애플리케이션 구성 요소

- 클라이언트 (client)
 - 사용자가 LLM 기반 애플리케이션에 접근하는 환경
 - 웹 브라우저, 애플리케이션 또는 API 요청을 통해 LLM과 상호 작용
- 프롬프트 (prompt)
 - 사용자가 입력한 텍스트를 LLM이 이해할 수 있는 형태로 제공하는 부분
- LLM 백엔드 (LLM backend)
 - LLM이 실제로 작동하는 서버 측의 처리 계층을 의미
 - 프롬프트를 받아 언어 모델을 통해 적절한 응답을 생성함
- 출력 파싱 (output parsing)
 - LLM이 생성한 응답을 사용자가 요구한 형식에 맞게 처리하는 단계
 - LLM의 응답을 단순한 텍스트 외에도 JSON, 표, 코드 등으로 변환하거나 후처리를 통해 정확하고 유용한 정보로 가공

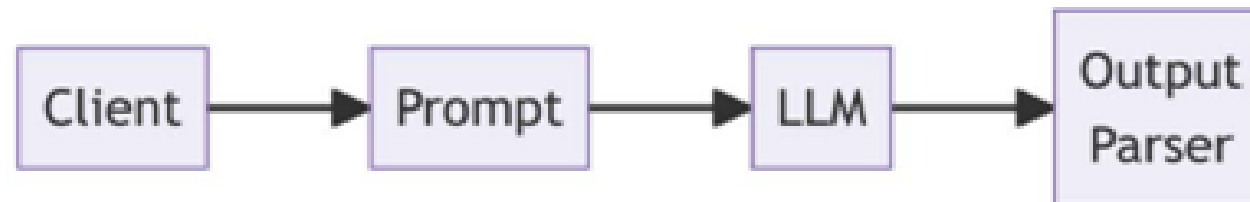


Figure 2.4: A simple LLM application

LLM 기술의 한계를 완화하기 위한 방법

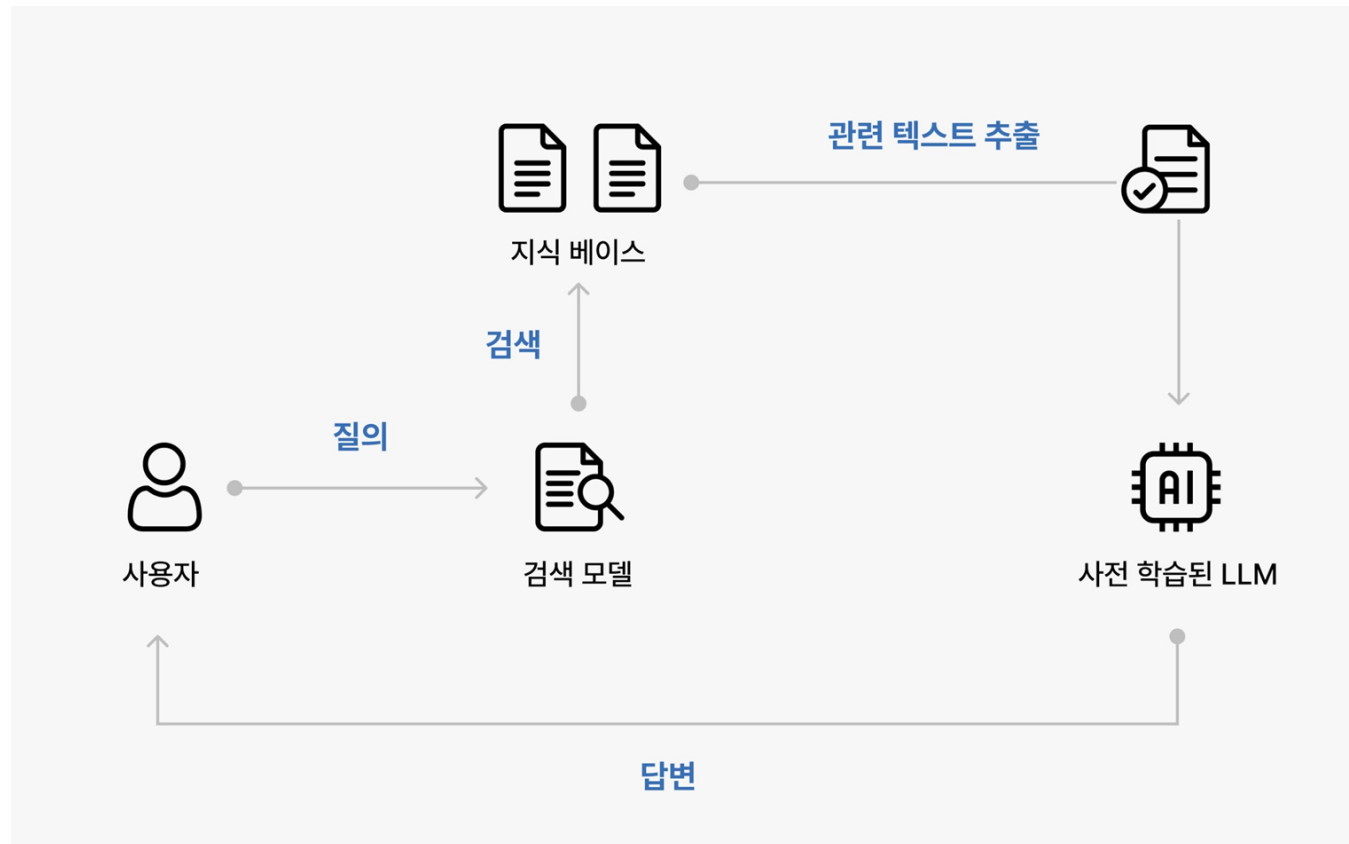
■ 대규모 언어모델의 주요 한계점

한계점	설명
오래된 지식	훈련 데이터에 의존하기 때문에 최신 정보 제공 불가능
맥락 부족	단순한 답변 생성에는 능숙하지만, 세부적인 맥락을 이해하고 설명하기 어려움
환각 위험	특정 주제에 대한 학습이 부족할 경우, 부정확하거나 비논리적인 답변 생성
편향 및 차별	훈련된 데이터가 편향된 경우, 차별적인 응답을 생성할 가능성이 있음
투명성 부족	모델이 생성한 답변의 근거가 불명확하여 해석이 어려움
상호작용 불가능	외부 시스템이나 실시간 환경과 동적으로 상호작용할 수 없음

LLM 기술의 한계를 완화하기 위한 방법

■ (1) 검색 증강 (Retrieval augmentation)

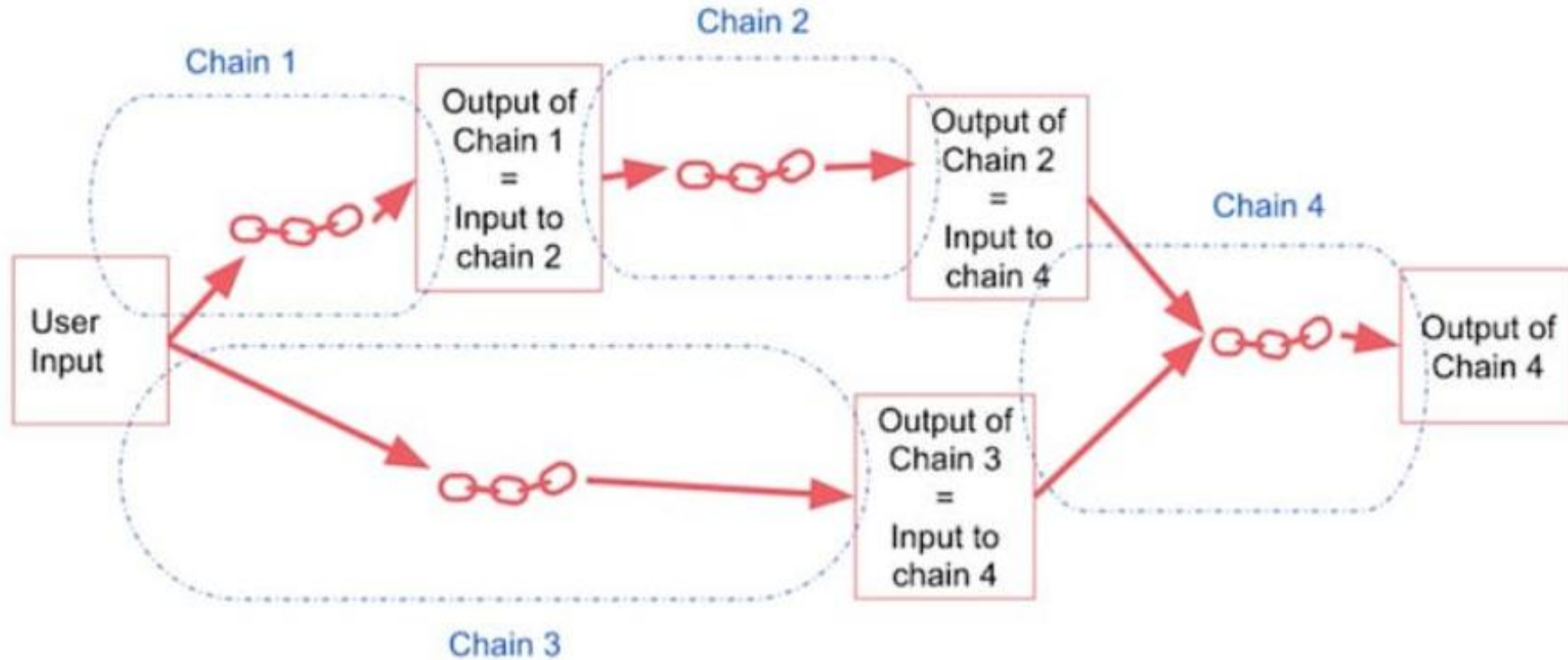
- LLM에 검색 기능을 통합하여 최신 정보에 접근하고, 학습 데이터 외의 지식을 활용



LLM 기술의 한계를 완화하기 위한 방법

■ (2) 체이닝 (Chaining)

- 복잡한 문제를 간단한 단계로 분해하여 여러 모델의 출력을 순차적으로 연결



LLM 기술의 한계를 완화하기 위한 방법

■ (3) 프롬프트 공학 (Prompt engineering)

- 효과적인 프롬프트를 설계하여 LLM이 원하는 결과를 생성하도록 유도

■ (4) 모니터링, 필터링 및 리뷰 (Monitoring)

- LLM의 출력을 모니터링하고 부적절한 내용을 필터링하며, 응답의 품질을 검토

■ (5) 메모리 (Memory)

- 이전 대화나 정보를 기억하여 긴 대화에서도 문맥을 유지하도록 지원

■ (6) 미세 조정 (Fine tuning)

- 특정 작업에 대한 성능 향상을 위해 추가 데이터로 LLM을 재학습

LangChain 프레임워크

- LLM 기반 어플리케이션을 구축하기 위한 오픈소스 파이썬 프레임워크
 - 모듈식 아키텍처를 통해 LLM 어플리케이션을 쉽고 빠르게 개발할 수 있음

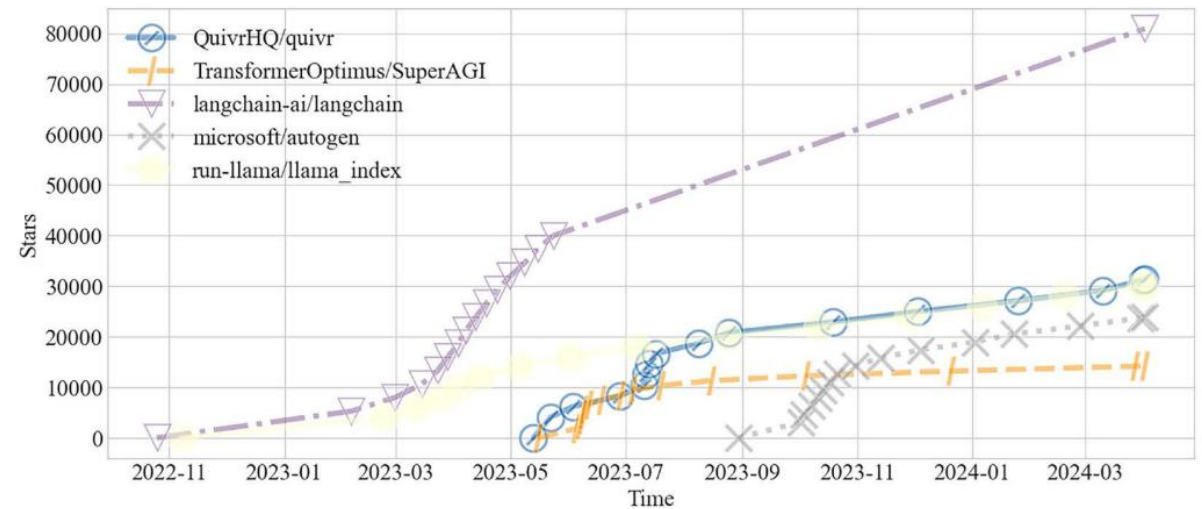
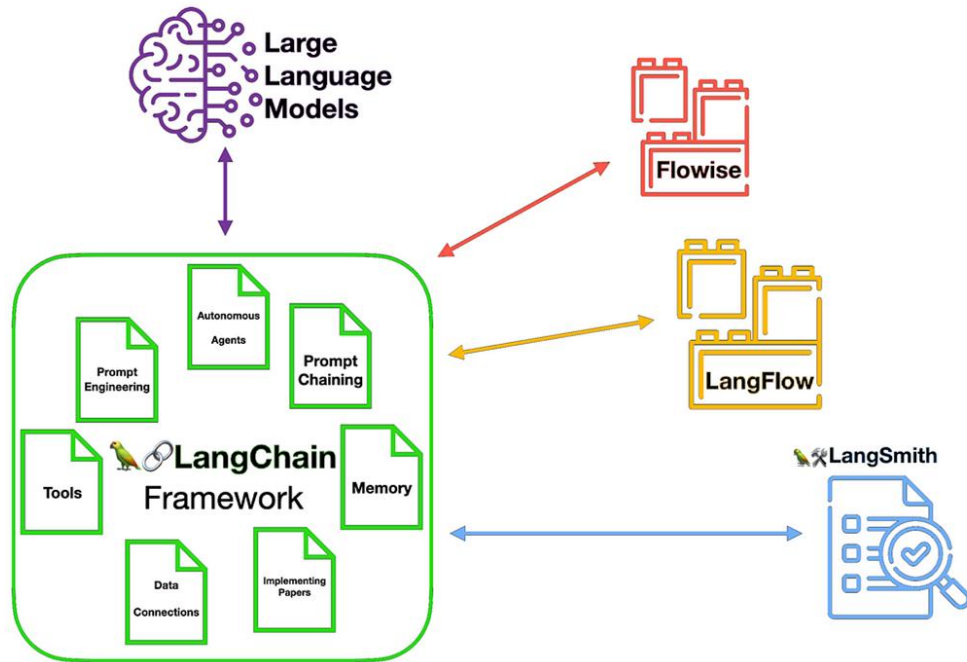
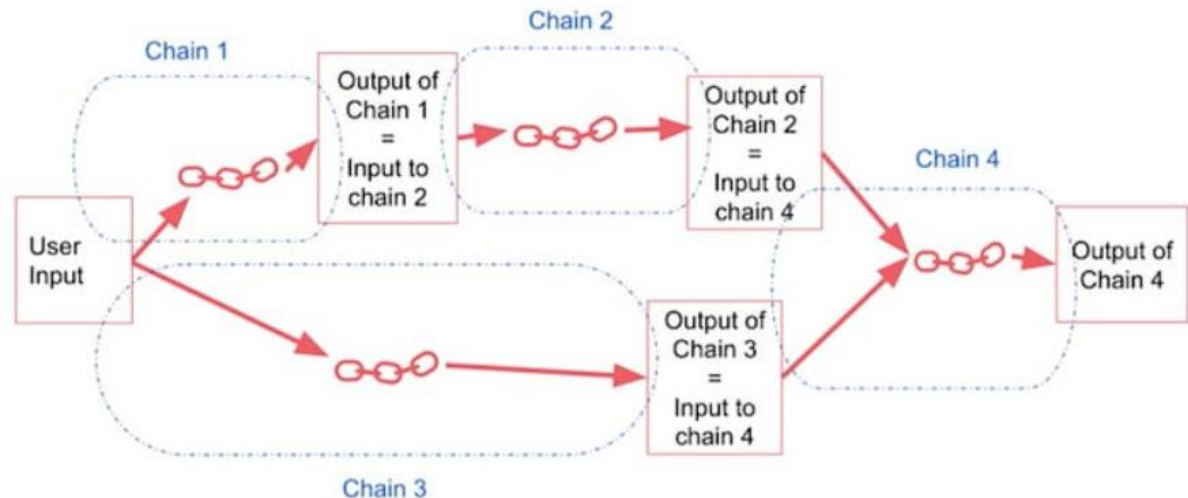


Figure 2.8: Comparison of popularity between different LLM frameworks in Python

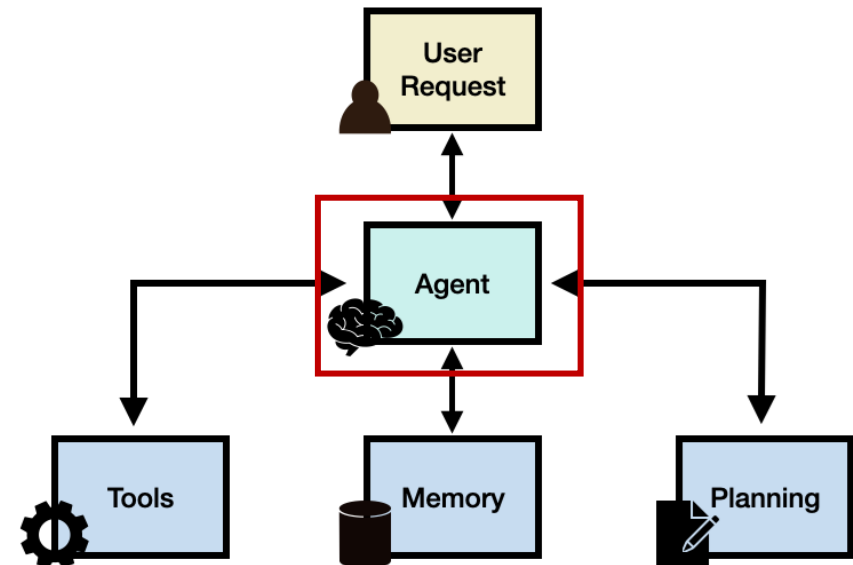
■ (1) 체인 (Chain)

- 모듈형 구성 요소를 재사용 가능한 파이프라인으로 조립하는 개념
- 체인의 핵심 이점
 - 모듈성: 재사용 가능한 구성 요소로 분할 가능
 - 결합성: 각 구성 요소를 유연하게 연결 가능
 - 가독성: 파이프라인의 각 단계를 명확히 표시 가능
 - 유지 보수성: 단계를 추가, 제거, 교체 가능
 - 재사용성: 필요 시 일부 모듈을 재사용 가능
 - 도구 통합: LLM, 데이터베이스, API 등을 쉽게 통합 가능



■ (2) 에이전트 (Agent)

- 목표와 작업을 달성하기 위해 행동할 수 있는 자율적인 소프트웨어 개체
- 주어진 환경을 관찰하고, 그 관찰을 기반으로 실행할 체인을 결정하며, 체인의 지정된 작업을 수행함
- 에이전트의 핵심 이점
 - 목표 지향적 실행: 특정 목표를 달성하기 위한 논리 체인을 설계 가능
 - 동적 응답: 환경의 변화를 관찰하여 반응 가능
 - 상태 유지: 상호 작용을 통해 메모리와 컨텍스트를 유지 가능
 - 견고성: 예외를 처리하고 대안적인 체인을 시도하여 에러에 대응 가능

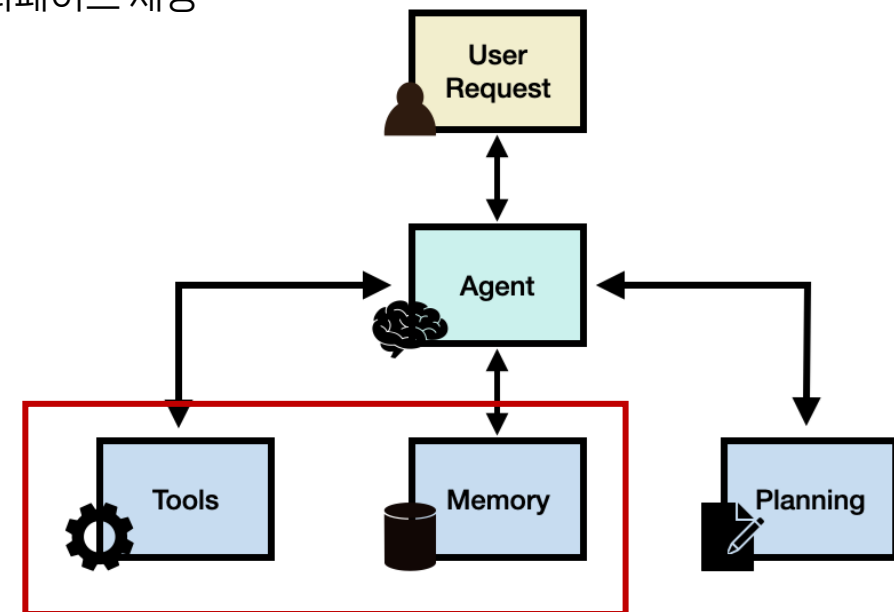


■ (3) 메모리 (Memory)

- 체인 또는 에이전트 실행 간에 지속되는 상태
- 체인은 각 호출마다 대화형 메모리를 모델에 전달해 일관성 제공
- 에이전트는 사실, 관계, 추론 정보를 메모리에 지속시켜 맥락 정보 유지

■ (4) 도구 (Tool)

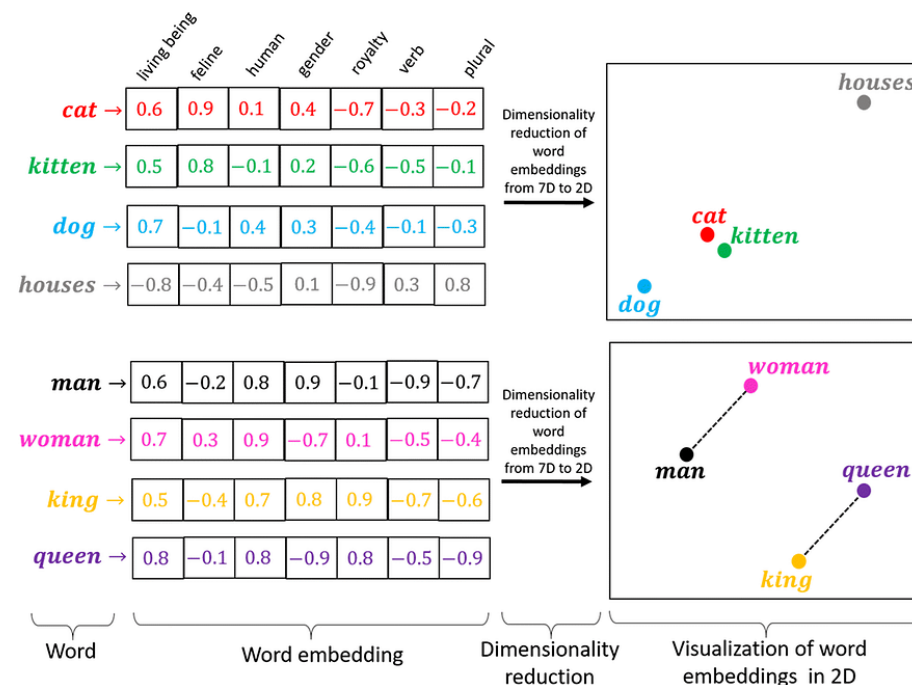
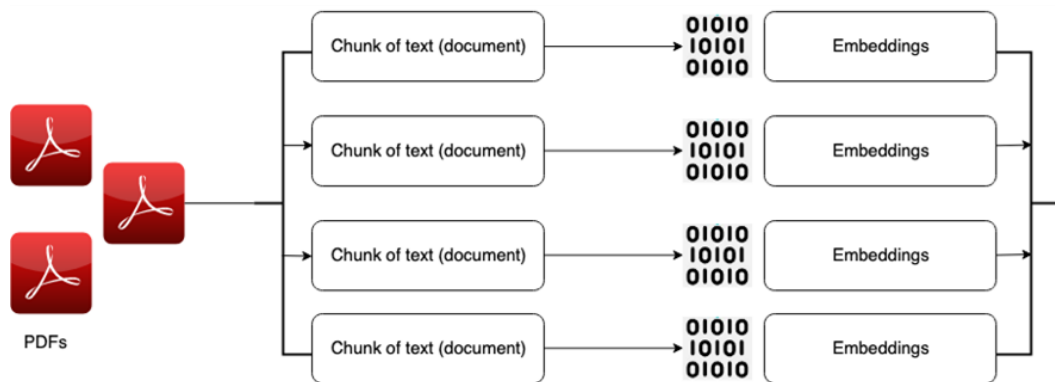
- 에이전트가 DB와 API와 같은 외부 서비스를 통합할 수 있도록 모듈식 인터페이스 제공
- 툴 예시
 - 기계 번역기
 - 계산기
 - 지도
 - 테이블 처리
 - 검색 엔진



LangChain 작동 원리

■ (1) 데이터 준비

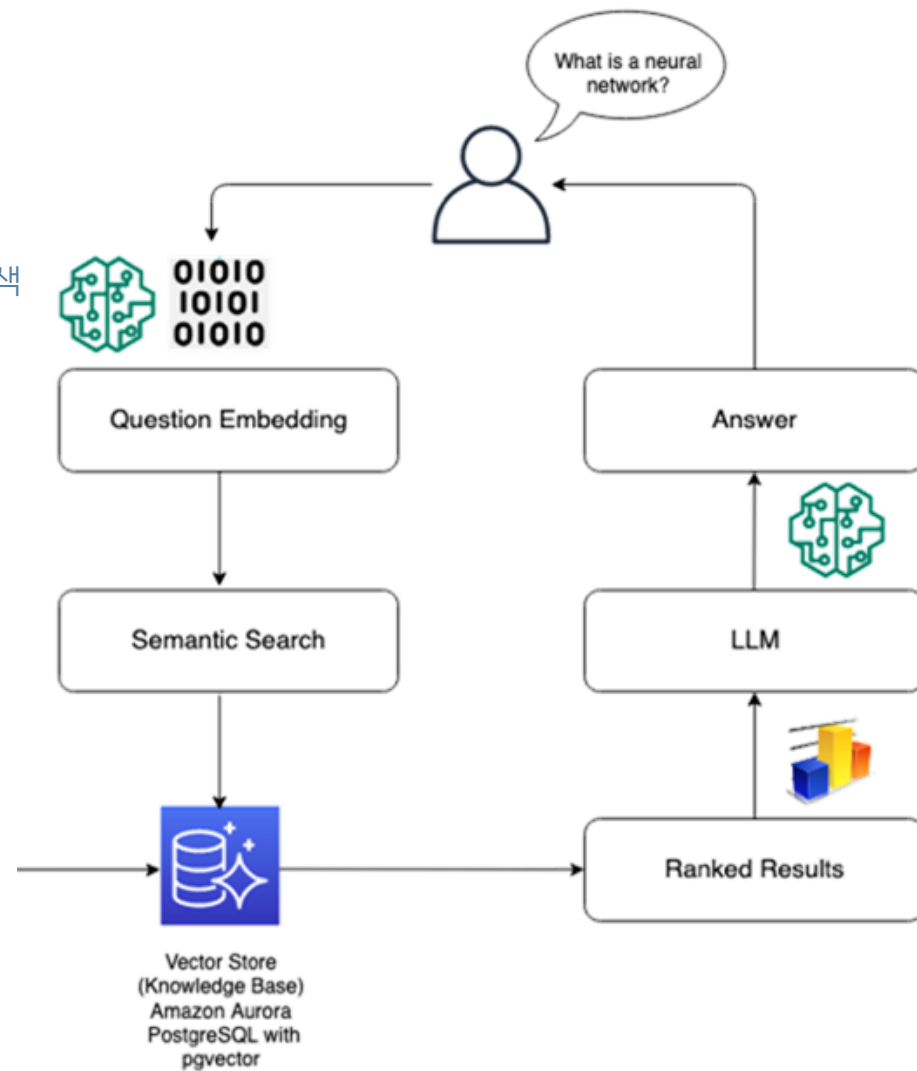
- 문서 로더
 - 텍스트와 메타데이터가 포함된 문서를 로드하여 처리할 수 있도록 준비
- 텍스트 분리
 - 긴 텍스트는 LLM이 한 번에 처리하기 어려우므로 문서를 여러 조각(chunk)으로 분할
 - chunk는 LLM이 더 쉽게 분석할 수 있으며, 문맥을 유지한 상태로 문서를 효율적으로 처리 가능
- 텍스트 임베딩
 - 단어, 문장, 또는 문서와 같은 텍스트 데이터를 벡터로 변환하는 과정
 - 고양이"와 "개"라는 단어는 의미적으로 가깝기 때문에 벡터 공간에서 가까움
 - 반면 "고양이"와 "집" 같은 단어는 서로 의미적으로 멀리 떨어져 있음



LangChain 작동 원리

■ (2) 데이터 검색

- 벡터 저장소
 - 임베딩된 문서 벡터 정보를 저장해두는 데이터베이스
- 의미 기반 검색기
 - 사용자 질문 벡터와 벡터 저장소 내 벡터들 간의 유사성을 계산하여, 의미적으로 가까운 정보 탐색
 - 유사성 계산을 기반으로 가장 관련성이 높은 결과들이 순위가 매겨져 제공
- LLM 모델
 - 검색된 결과는 LLM을 통해 분석되어 최종적인 답변 생성



■ 실습 1) 팩트 체크를 통한 환각 완화

- LLMCheckerChain 라이브러리를 활용한 답변 점검
 - 언어 모델이 만든 응답을 점검하여 정확성을 확인하는 도구로, 질문에 대한 응답이 올바른지 확인
 - (1) 주장 감지: 검증이 필요한 부분 식별
 - (2) 증거 검색: 주장을 지지하거나 반박하는 소스 찾기
 - (3) 판결 예측: 증거를 기반으로 주장의 진위 평가

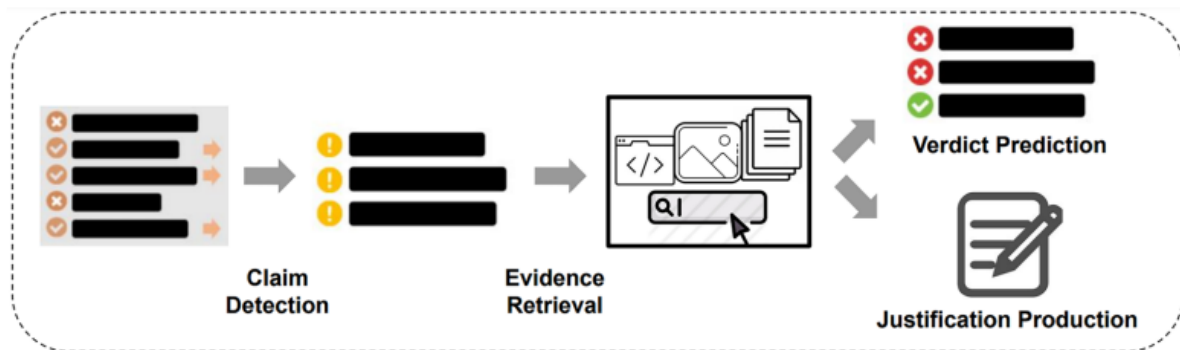


Figure 4.7: Automatic fact-checking pipeline in three stages

Here's a statement: {statement}\nMake a bullet point list of the assumptions you made when producing the above statement.\n

Here is a bullet point list of assertions:
{assertions}
For each assertion, determine whether it is true or false. If it is false, explain why.\n\n

In light of the above facts, how would you answer the question '{question}'

LangChain 활용하기

■ 실습 1) 팩트 체크를 통한 환각 완화

- 1. GPT 모델 구성 및 LLMCheckerChain 연계

> Entering new LLMCheckerChain chain...

> Entering new SequentialChain chain...

> Finished chain.

> Finished chain.

```
{'query': 'What type of mammal lays the biggest eggs?', 'result': 'The Echidna is the type of mammal that lays the biggest eggs.'}
```

```
!pip install langchain
!pip install langchain-openai
```

필요한 라이브러리 가져오기

```
from langchain_openai import ChatOpenAI
from langchain.chains import LLMCheckerChain
```

OpenAI API 키 설정

```
api_key = "sk-proj-zcYjewY8DYjQ4gjIKzBeASFA1ru_McHtN4kA50N-QjVLGEXQx9pivZHXI7ste6bOI7yO1T86"
```

ChatOpenAI로 GPT-4 모델 설정

```
llm = ChatOpenAI(
    model="gpt-4",
    temperature=0.7,
    openai_api_key=api_key
)
```

질문 정의

```
text = "What type of mammal lays the biggest eggs?"
```

LLMCheckerChain 설정

```
checker_chain = LLMCheckerChain.from_llm(llm, verbose=True)
```

질문 실행 및 결과 출력

```
result = checker_chain.invoke(text)
print(result)
```


LangChain 활용하기

■ 실습 2) 정보 추출 및 요약

■ 1. 입력 텍스트 지정 및 GPT 모델 구성

```
# 필요한 라이브러리 가져오기
from langchain_openai import ChatOpenAI

# OpenAI API 키 설정
api_key = "sk-proj-zcYjewY8DYjQ4gjIKzBeASFA1ru_McHtN4kA50N-QjVLGEXQx9pivZHXI7ste6bOI7yO1T8613T3B"

# 텍스트 정의
text = """
Generative AI is artificial intelligence capable of generating text, images, videos, or other data.
Generative AI models learn the patterns and structure of their input training data and then generate new data.
"""

# GPT-4용 프롬프트 정의
prompt = """
Summarize this text in one sentence within 10 words:

{text}
"""

# GPT-4 모델 설정
llm = ChatOpenAI(
    model="gpt-4",
    temperature=0.7,
    openai_api_key=api_key
)

# GPT-4 모델을 사용하여 요약 실행
summary = llm.invoke(prompt.format(text=text))

# 결과 출력
print(summary.content)
```

text = """Generative AI is artificial intelligence capable of generating text, images, videos, or other data using generative models, often in response to prompts. Generative AI models learn the patterns and structure of their input training data and then generate new data that has similar characteristics."""



Generative AI creates data similar to its training input via learned patterns.

LangChain 활용하기

■ 실습 2) 정보 추출 및 요약

- 2. LangChain 표현 언어(LangChain Expression Language)을 활용한 구현
 - 체인형 흐름을 통한 프롬프트 템플릿 구현 가능
 - 비동기 처리, 배치 처리, 스트리밍, 예외 처리, 병렬 처리 등 지원

```
# PromptTemplate 정의
prompt = PromptTemplate.from_template(
    "Summarize this text with 10 words: {text}?"
)

# GPT-4 모델 정의
llm = ChatOpenAI(
    model="gpt-4",
    temperature=0.7,
    openai_api_key=api_key
)

# Runnable을 정의 (파이프라인 방식)
runnable = prompt | llm | StrOutputParser()

# 입력 텍스트
text = """
Generative AI is artificial intelligence capable of generating text, images, videos, or other data using generative models, often in response to prompts. Generative AI models learn the patterns and structure of their input training data and then generate new data that has similar characteristics.
"""

# 파이프라인 실행 및 결과 출력
summary = runnable.invoke({"text": text})
print(summary)
```

text = """Generative AI is artificial intelligence capable of generating text, images, videos, or other data using generative models, often in response to prompts. Generative AI models learn the patterns and structure of their input training data and then generate new data that has similar characteristics."""



Generative AI creates data like text, images, videos from learned patterns.

■ 실습 3) 이력서 요약 서비스

- 1. 라이브러리 설치 및 요약할 정보 정의

```
!pip install pymupdf
```

```
from typing import Optional
import pydantic

class Experience(pydantic.BaseModel):
    start_date: Optional[str]
    end_date: Optional[str]
    description: Optional[str]

class Study(Experience):
    degree: Optional[str]
    university: Optional[str]
    country: Optional[str]
    grade: Optional[str]

class WorkExperience(Experience):
    company: str
    job_title: str

class Resume(pydantic.BaseModel):
    first_name: str
    last_name: str
    linkedin_url: Optional[str]
    email_address: Optional[str]
    nationality: Optional[str]
    skill: Optional[str]
    study: Optional[Study]
    work_experience: Optional[WorkExperience]
    hobby: Optional[str]
```

LangChain 활용하기

■ 실습 3) 이력서 요약 서비스

▪ 2. 데이터 입력 & GPT 모델 및 LangChain 파이프라인 구성

```
import os
from langchain.document_loaders import PyMuPDFLoader
from langchain.output_parsers import PydanticOutputParser
from langchain.output_parsers.json import SimpleJsonOutputParser
from langchain_openai import ChatOpenAI
from langchain.prompts import PromptTemplate

# PDF 파일 로딩
pdf_file_path = os.path.expanduser("openresume-resume.pdf")
pdf_loader = PyMuPDFLoader(pdf_file_path)
docs = pdf_loader.load_and_split()

# OpenAI API 키 설정
api_key = "sk-proj-zcYjewY8DYjQ4gjIKzBeASFA1ru_McHtN4kA50N-QjVLGEXQx9pivZHXI7ste"
```

```
# 출력 파서 설정
parser = PydanticOutputParser(pydantic_object=Resume)

# 프롬프트 템플릿 설정
prompt = PromptTemplate(
    template="Extract information from the provided document.\n{format_instructions}\n{document}\n",
    input_variables=["document"],
    partial_variables={"format_instructions": parser.get_format_instructions()},
)

# GPT-4-turbo 모델 설정
llm = ChatOpenAI(model_name="gpt-4-turbo-preview", openai_api_key=api_key)

# LangChain 파이프라인 구성 (프롬프트 → LLM → JSON 파서)
chain = prompt | llm | SimpleJsonOutputParser()

# PDF에서 로드된 문서를 기반으로 정보 추출 실행
chain.invoke({"document": docs})
```

LangChain 활용하기

■ 실습 3) 이력서 요약 서비스

▪ 3. 출력 결과 확인

John Doe

Software engineer obsessed with building exceptional products that people love

✉ hello@openresume.com 📞 123-456-7890 📍 NYC, NY 🌐 linkedin.com/in/john-doe

WORK EXPERIENCE

ABC Company

Software Engineer

May 2023 - Present

- Lead a cross-functional team of 5 engineers in developing a search bar, which enables thousands of daily active users to search content across the entire platform
- Create stunning home page product demo animations that drives up sign up rate by 20%
- Write clean code that is modular and easy to maintain while ensuring 100% test coverage

DEF Organization

Software Engineer Intern

Summer 2022


- Re-architected the existing content editor to be mobile responsive that led to a 10% increase in mobile user engagement
- Created a progress bar to help users track progress that drove up user retention by 15%
- Discovered and fixed 5 bugs in the existing codebase to enhance user experience

XYZ University

Research Assistant

Summer 2021

- Devised a new NLP algorithm in text classification that results in 10% accuracy increase
- Compiled and presented research findings to a group of 20+ faculty and students



```
{
  'first_name': 'John',
  'last_name': 'Doe',
  'linkedin_url': 'linkedin.com/in/john-doe',
  'email_address': 'hello@openresume.com',
  'nationality': None,
  'skill': 'HTML, TypeScript, CSS, React, Python, C++, React Hooks, GraphQL, Node.js, SQL, Postgres, NoSql, Redis, REST API, Git, Teamwork, Creative Problem Solving, Communication, Learning Mindset, Agile',
  'study': {
    'start_date': 'Sep 2019',
    'end_date': 'May 2023',
    'description': 'Won 1st place in 2022 Education Hackathon, 2nd place in 2023 Health Tech Competition, Teaching Assistant for Programming for the Web (2022 - 2023), Coursework: Object-Oriented Programming (A+), Programming for the Web (A+), Cloud Computing (A), Introduction to Machine Learning (A-), Algorithms Analysis (A-)',
    'degree': 'Bachelor of Science in Computer Science',
    'university': 'XYZ University',
    'country': None,
    'grade': '3.8 GPA'
  },
  'work_experience': {
    'start_date': 'May 2023',
    'end_date': 'Present',
    'description': 'Lead a cross-functional team of 5 engineers in developing a search bar, which enables thousands of daily active users to search content across the entire platform, Create stunning home page product demo animations that drives up sign up rate by 20%, Write clean code that is modular and easy to maintain while ensuring 100% test coverage'
  },
  'company': 'ABC Company',
  'job_title': 'Software Engineer',
  'hobby': None
}
```

감사합니다

Q&A



충북대학교
CHUNGBUK NATIONAL UNIVERSITY