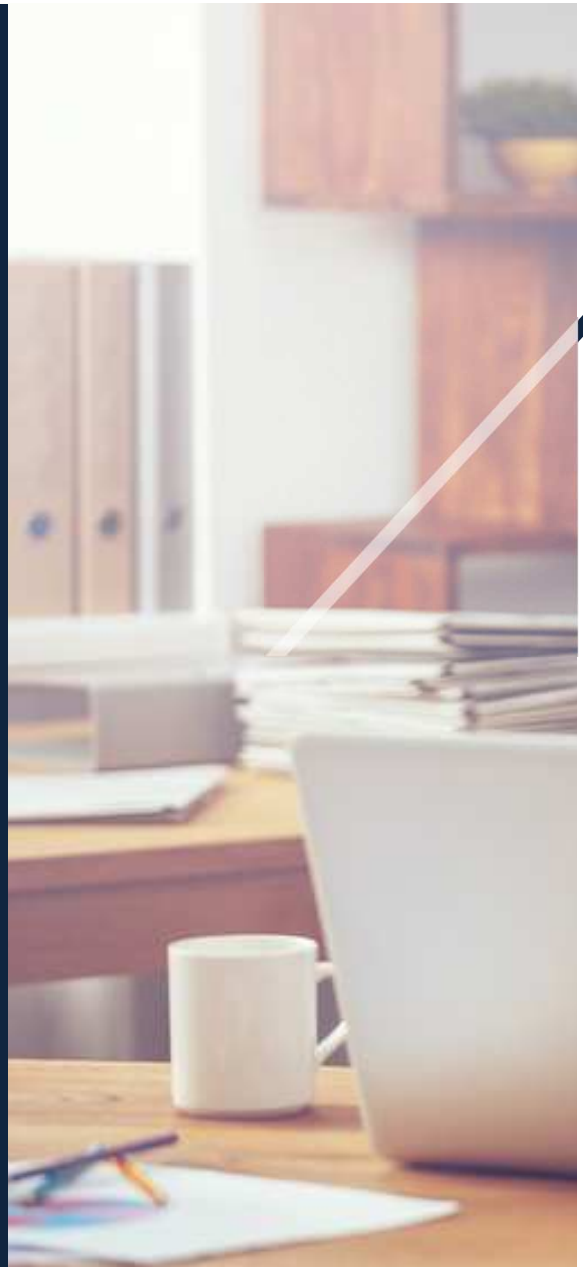
실무에 바로 쓰는

Langchain을 활용한

RAG 서비스 구축

Actionable Content
MASO CAMPUS



NOTICE

- 본 자료는 저작권법에 의거해 허가 받지 않은 복사, 전재, 편집, 재배포 등을 금지함을 알려 드립니다.

- 본 자료는 온/오프라인 또는 동영상 강의를 전제로 제작되었습니다.
  강사의 부가 설명 없이 본 자료의 내용을 **임의 해석할 경우 잘못된 결론**에 이를 수 있음을 유념하십시오.

- 강의를 캡쳐, 녹음, 녹화하는 등의 콘텐츠의 원천 제공 방식 이외의 **저장 행위를 엄격히 금지**하오니
  필요하신 내용은 수업 틈틈이 개별적으로 메모하시기 바랍니다.

Actionable Content
MASO CAMPUS

강의 소개

강의 소개

Actionable Content
MASO CAMPUS

① 강의 개요 및 목적

RAG의 개념에 대해 알아보고, Langchain을 활용하여 RAG 시스템을 구축하는 방법에 대해 배워봅니다.

② 강의 커리큘럼

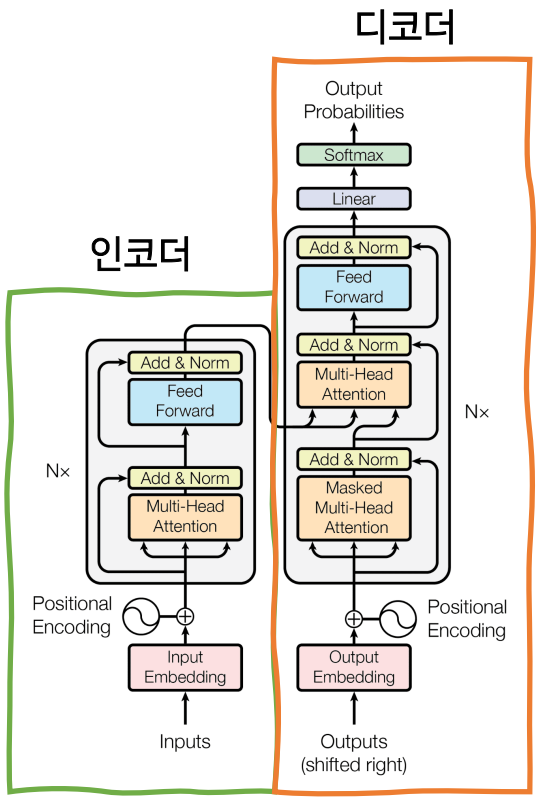
1. RAG 기초와 파이프라인
   - RAG의 개념과 필요성
   - RAG 파이프라인 개요와 핵심 컴포넌트
   - RAG 실제 적용 사례 분석
2. Langchain 프레임워크 소개
   - Langchain 개념과 아키텍처
   - Langchain을 사용한 RAG 구현 워크플로우
3. RAG를 위한 Langchain 핵심 구성요소(실습)
   - Models, Prompt Templates, Document Loaders, Text Splitters, Embeddings, VectorDB, Retrievers
4. Langchain을 더 편하게, LCEL(실습)
   - LCEL 소개 및 기본 구조
   - Runnable의 개념과 핵심 모듈들
   - LCEL을 사용한 RAG 파이프라인 구성
5. 고급 RAG 기법
   - MultiQuery Retriever, Rerank, Context Compression, Question Decomposition
6. Streamlit을 활용한 RAG 챗봇 구현
   - Streamlit 챗봇 기초 구현
   - 대화 이력 관리와 메모리 구현
7. FastAPI를 활용한 챗봇 웹사이트 구축
   - FastAPI 기초와 RESTful API 설계
   - FastAPI와 Langchain RAG 시스템 연동
   - HTML/CSS 기초 – 챗봇 인터페이스 구현
   - JavaScript 기초: 비동기 통신 및 채팅 기능 구현

RAG의 개념과 필요성

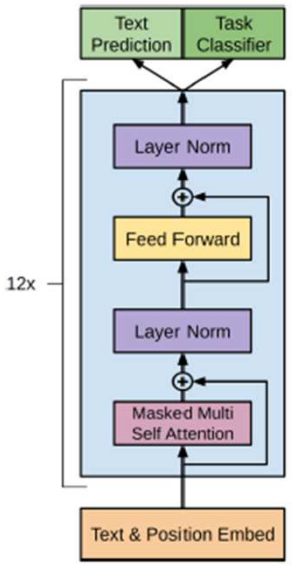
RAG의 필요성 - LLM의 한계

▶ LLM(초거대 언어모델)은 NLP의 정수로써, 디코더에 대규모 데이터를 학습해 완성된 하나의 딥러닝 모델입니다.

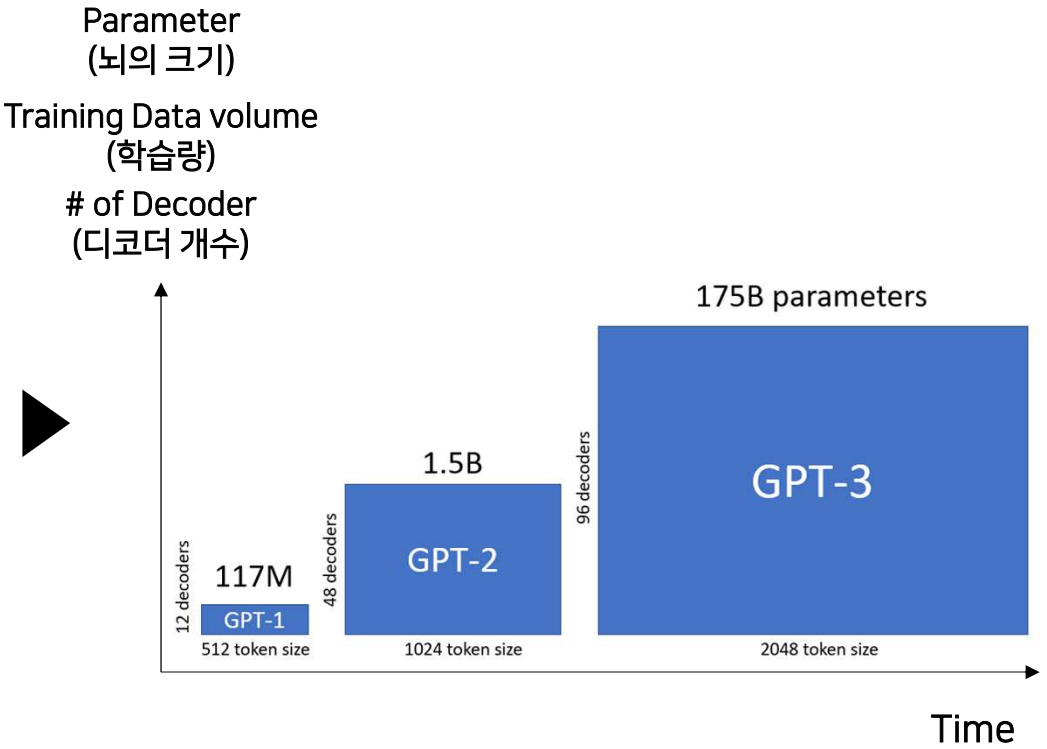
[트랜스포머 아키텍처]



[디코더만 학습한 GPT]

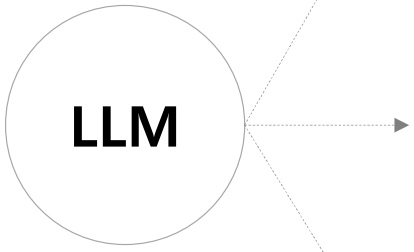


[학습량에 따른 GPT 모델의 발전]



RAG의 필요성 - LLM의 한계

▶ 딥러닝 모델은 긴 길이의 문장을 입력받거나 출력하는 데에 어려움이 있습니다.
▶ 또한 학습된 데이터 이외의 것을 만들어내는 것에 취약하다는 한계점이 있습니다.
▶ 이는 곧 LLM의 한계로 작용합니다.



1. 환각 현상

- 딥러닝 모델은 학습된 데이터 이외의 정보에 취약합니다.
- 오픈AI의 최신 모델인 GPT-4o는 2023년 9월까지, Claude Sonnet 3.5는 2024년 4월까지의 데이터로 학습되어, 이 이후의 정보는 알지 못합니다.

2. 기억 불가

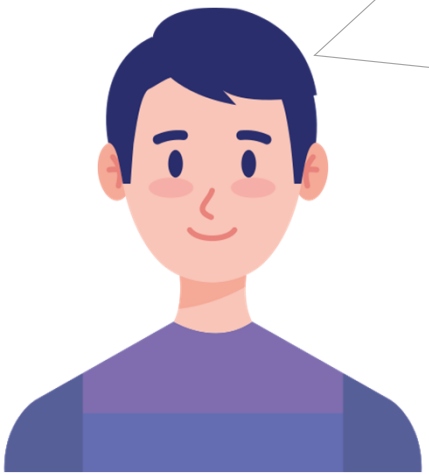
- LLM은 사전학습 시에 받아들인 정보 외의 것은 배우지 못합니다.
- 따라서 오늘 나와 대화하고 있는 LLM은 어제 나와 대화했던 내용을 기억하지 못합니다.

3. 토큰 제한

- LLM은 입력값의 길이가 길어지면 계산량이 크게 늘어나, 대부분의 모델이 길이를 제한하고 있습니다.
- GPT-4o의 입력값 제한은 128,000토큰(문자 단위), Claude sonnet 3.5는 200,000토큰입니다.

RAG의 개념

▶ RAG(Retrieval Augmented Generation)은 LLM의 한계 중 환각 현상 해결에 특효약입니다.
▶ LLM에게 어떤 질문을 할 때, 이에 힌트가 될만한 문장을 함께 넣어 줌으로써, 환각 현상을 방지할 수 있습니다.

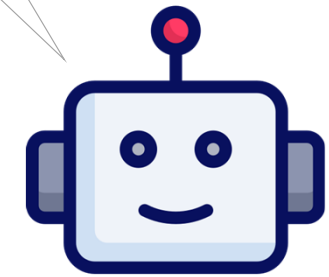


주어진 힌트에 기반하여 질문에 답해주세요.

질문(Question)
현재 갤럭시 Z시리즈의 최신 모델은 뭐야?

힌트(Context) 삼성전자는 7월 10일, 갤럭시 언팩 2024 행사에서 갤럭시 Z 폴드6와 플립6를 공개했다.

갤럭시 Z 폴드6와 플립6 입니다. 이는 갤럭시 언팩 2024 행사에서 공개된 것으로, Z시리즈의 최신 모델입니다.

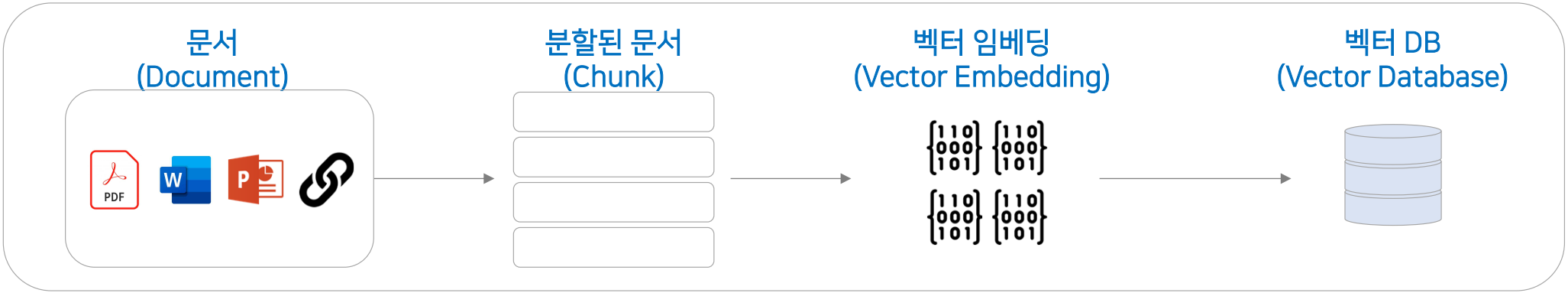


RAG 파이프라인 개요와 핵심 컴포넌트

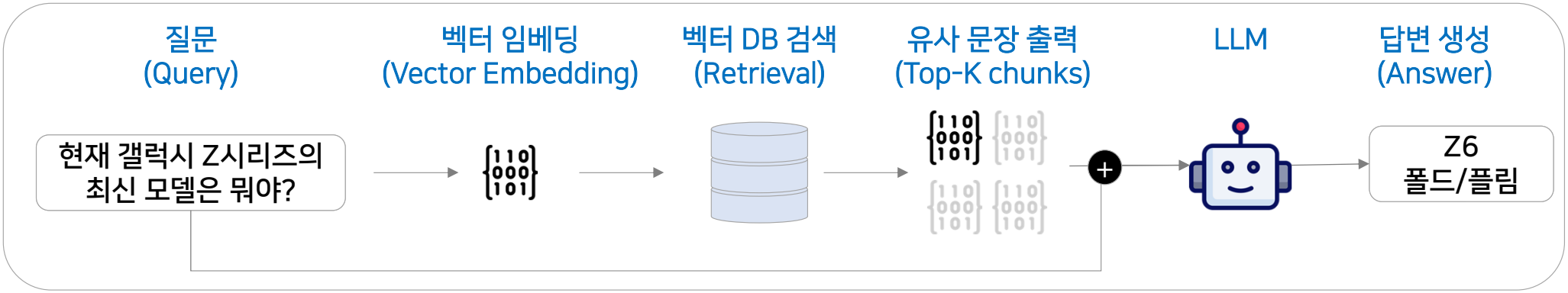
RAG 파이프라인

▶ **RAG는 문서를 숫자로 변환하여 저장하는 인덱싱, 검색하고 답변을 생성하는 검색/생성 두가지 구조로 나뉘어 있습니다.**

데이터 인덱싱

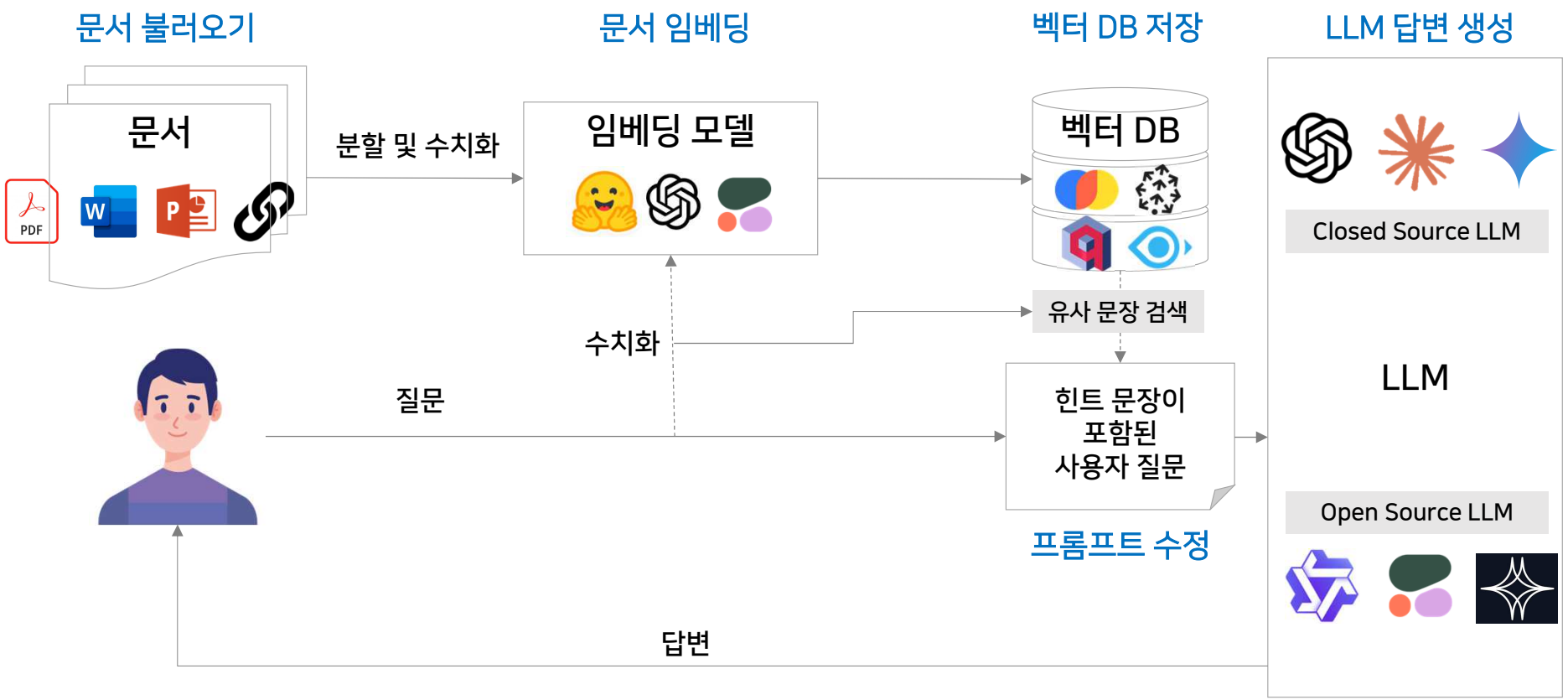


데이터 검색 및 생성



RAG의 핵심 컴포넌트

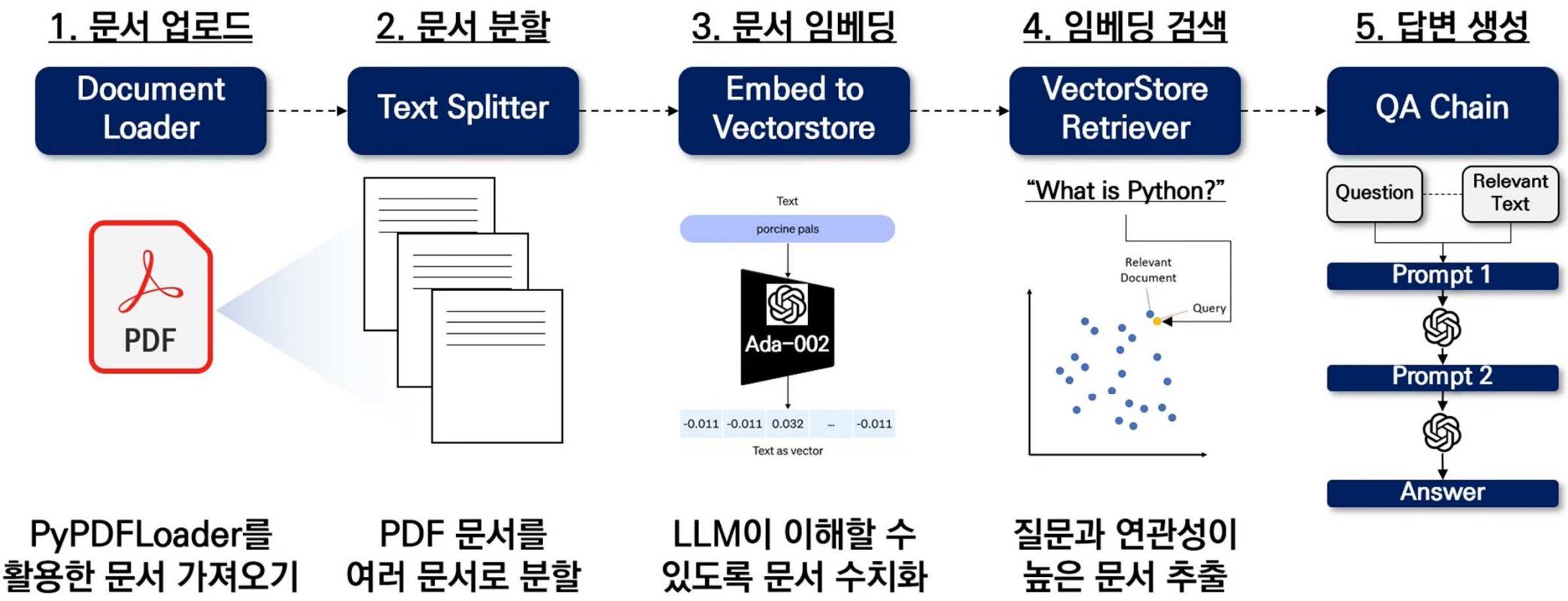
▶ RAG를 구성하기 위해서는 문서 로더, 임베딩 모델, 벡터 DB, LLM 등 다양한 컴포넌트가 필요합니다.



실제 적용 사례 분석

강의 소개

▶ ChatPDF라는 서비스는 PDF 파일을 기반으로 대화할 수 있는 대표적인 RAG 시스템입니다.



Langchain 개념과 아키텍처

Langchain이란?

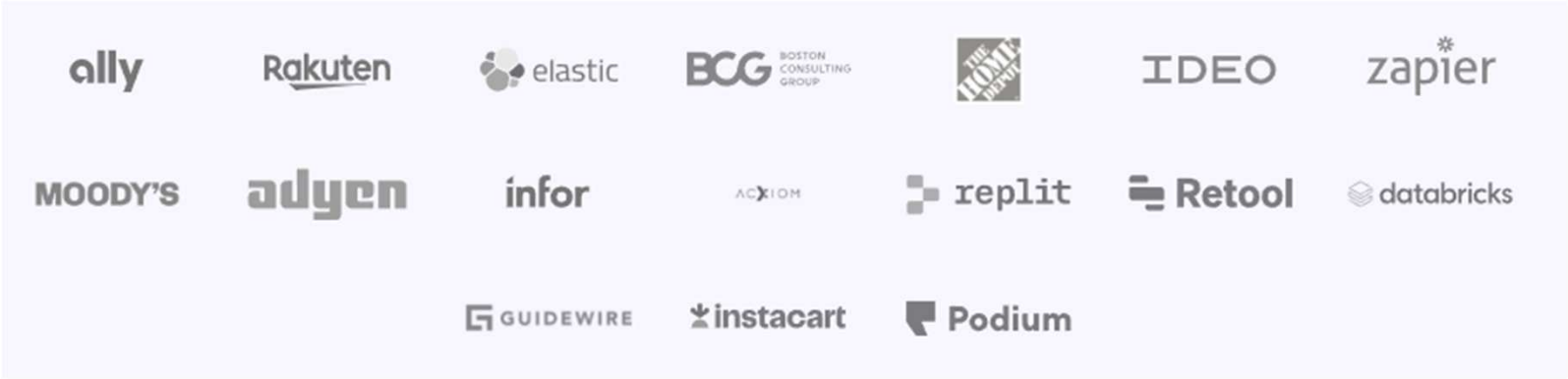
▶ Langchain은 LLM 어플리케이션을 보조하는 프레임워크입니다.

LlamaIndex　LangChain　FlowiseAI

LangChain

LangChain is a framework for developing applications powered by large language models (LLMs).
LangChain은 대규모 언어 모델(LLM)로 구동되는 애플리케이션을 개발하기 위한 프레임워크입니다.

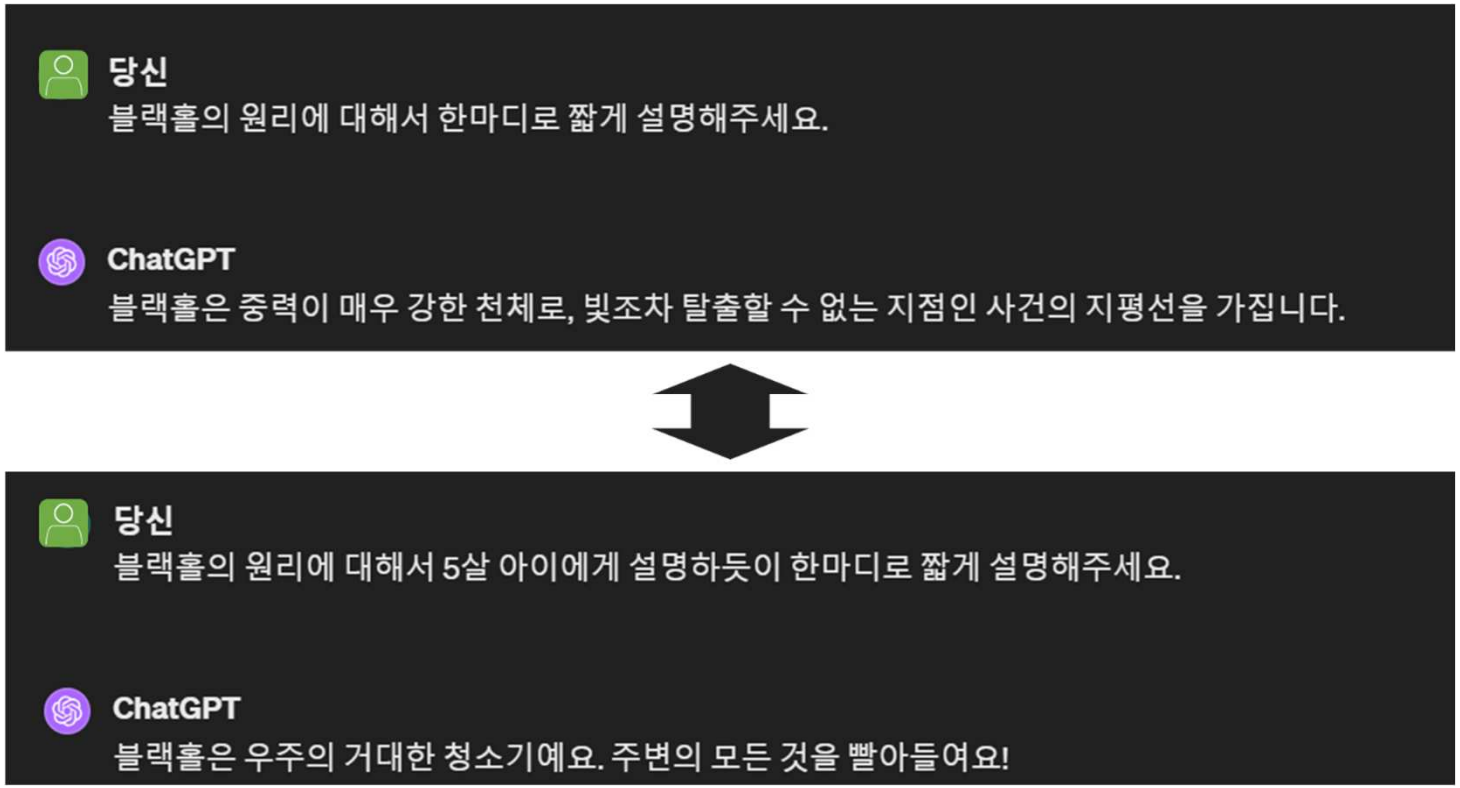
▶ 다양한 언어 모델과 도구를 쉽게 통합할 수 있으며, 유연성과 재사용성이 높아 많은 기업에서 활용하고 있습니다.



Langchain은 왜 필요할까?

▶ **LLM의 발전과 함께 중요해진 프롬프트 엔지니어링을 보조할 수 있습니다.**

LLM의 발전과 함께 LLM에게 어떻게 질문하는가가 중요해졌습니다. 이를 프롬프트 엔지니어링이라 합니다.
Langchain은 프롬프트 엔지니어링의 번거로움을 줄이기 위해 Prompt Template이라는 모듈을 지원합니다.



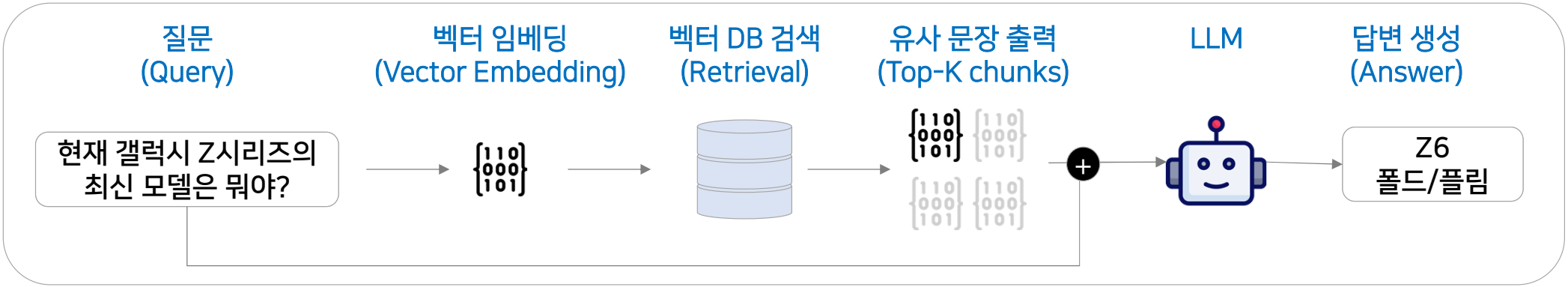
LangChain

Chain

"5살 아이에게 설명하듯 답해줘."
+
사용자의 프롬프트

Langchain은 왜 필요할까?

▶ RAG도 사용자의 질문에 힌트 문장을 합하여 LLM에게 전달하는 일종의 프롬프트 엔지니어링입니다.



질문(Question)
현재 갤럭시 Z시리즈의 최신 모델은 뭐야?

LangChain

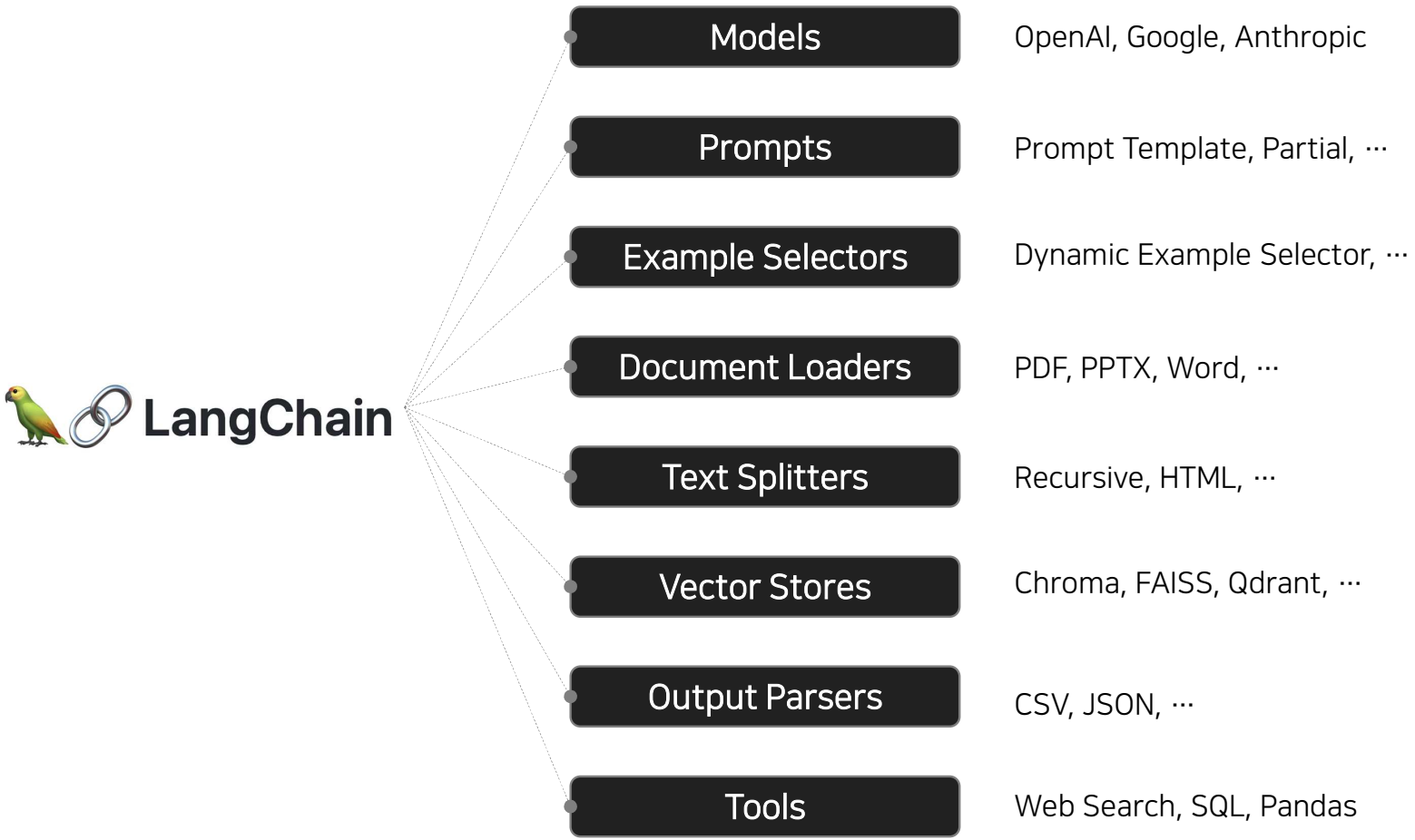
주어진 힌트에 기반하여 질문에 답해주세요.

사용자의 질문:
{Question}

힌트(Context) 삼성전자는 7월 10일, 갤럭시 언팩 2024 행사에서 갤럭시 Z 폴드6와 플립6를 공개했다.

Langchain의 아키텍처

▶ Langchain은 프롬프트 엔지니어링 뿐만 아니라 RAG, Agent 등의 시스템을 만들기 위한 모듈을 모두 갖췄습니다.



Langchain의 아키텍처

▶ Langchain은 프롬프트 엔지니어링 뿐만 아니라 RAG, Agent 등의 시스템을 만들기 위한 모듈을 모두 갖췄습니다.

LLM

초거대 언어모델로, Langchain의 엔진과 같은 역할
예시: GPT-4, Gemini, LlaMa, Solar, …

Prompts

초거대 언어모델에게 지시하는 명령문
요소: Prompt Template, Chat Prompt Template, …

Index

LLM이 문서를 쉽게 탐색할 수 있도록 구조화
요소: Document Loaders, Text Splitters, Vector Store, Retriever,…

Memory

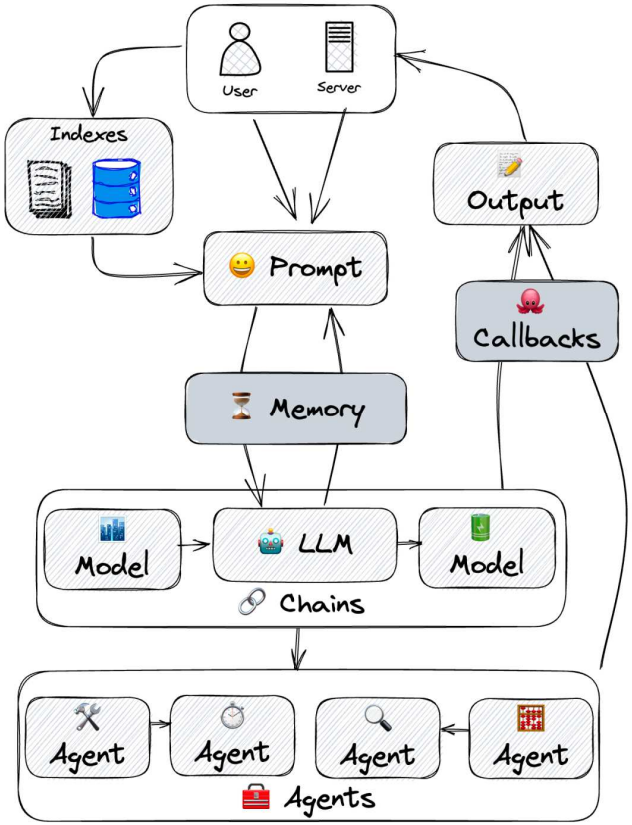
채팅 이력을 저장하여, 이를 기반으로 대화 가능케 함
예시: ConversatinoBufferMemory, Entity Memory, …

Chain

LLM 사슬을 형성하여 연속적인 LLM 호출
예시: LLM Chain, QA Chain, Summarization Chain, …

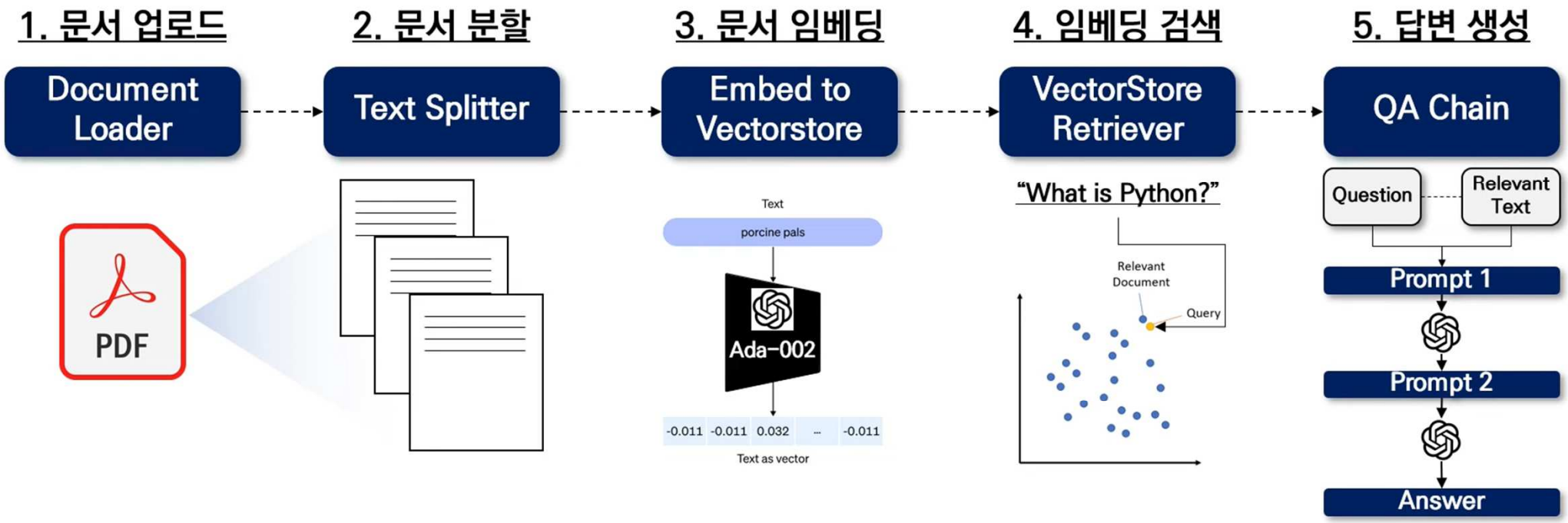
Agents

LLM이 스스로 어떤 작업을 수행할지 계획하고 수행
예시: Websearch Agent, SQL Agent, …



Langchain을 사용한 RAG 구현 워크플로우

▶ ChatPDF라는 서비스는 PDF 파일을 기반으로 대화할 수 있는 대표적인 RAG 시스템입니다.



Models

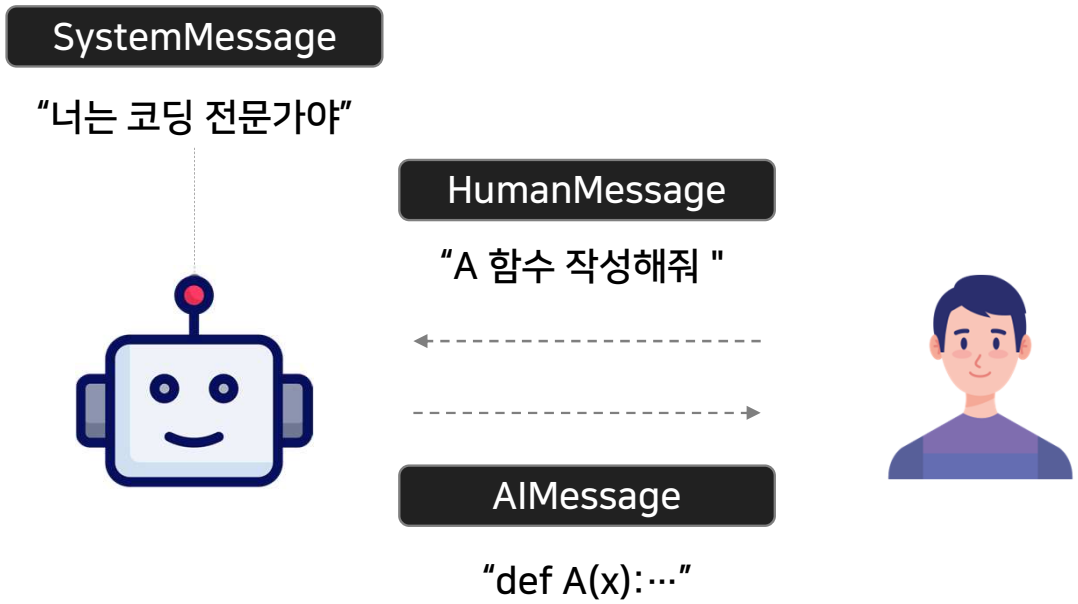
LLM API 활용하기

▶ Langchain을 활용하면 다양한 모델 API를 일관된 형식으로 불러올 수 있습니다.

프롬프트 이해하기

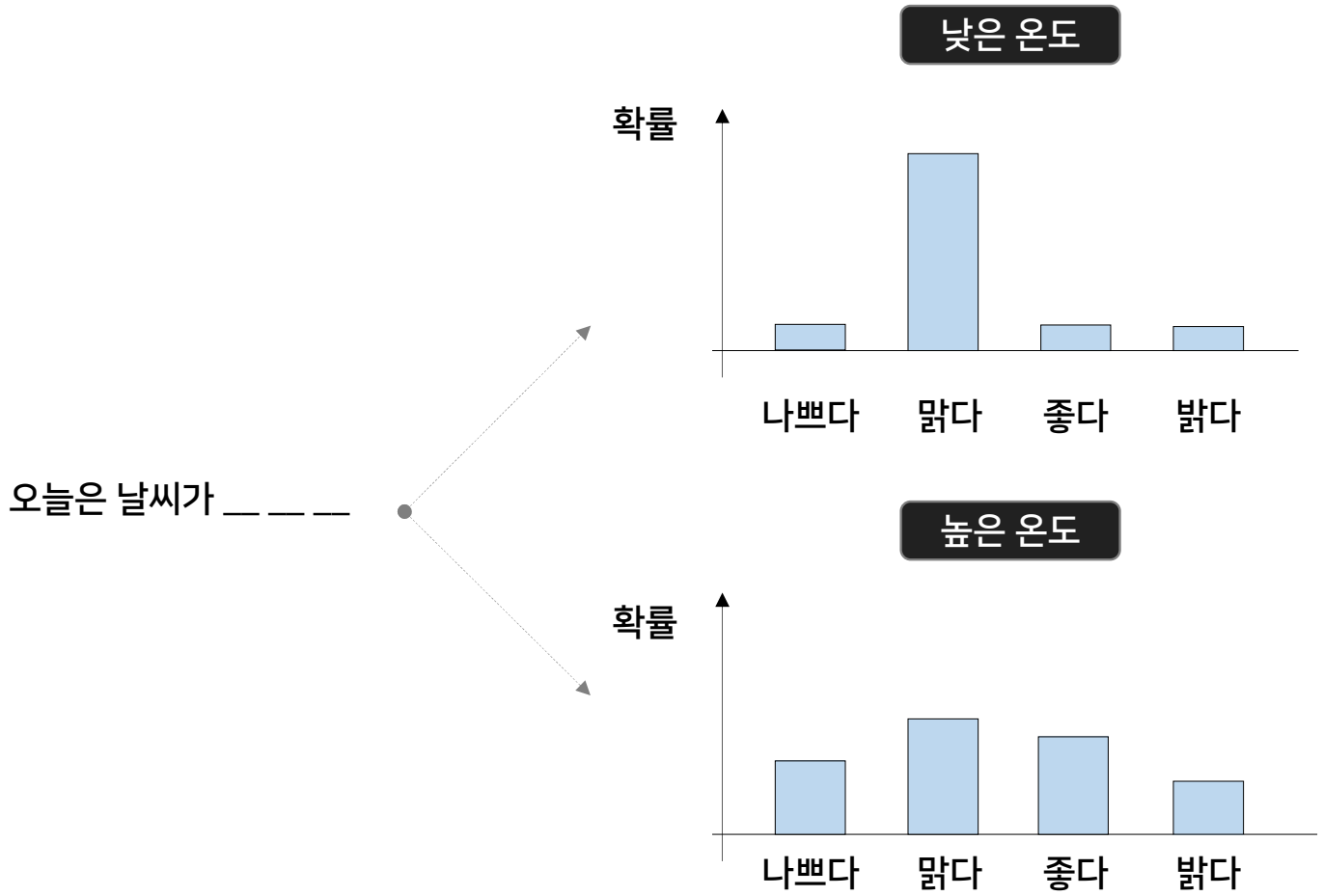
▶ Langchain에서는 LLM에게 보내는 프롬프트의 형식을 크게 3가지로 구분합니다.

프롬프트 종류	역할
SystemMessage	LLM에게 역할을 부여하는 메시지
HumanMessage	LLM에게 전달하는 사용자의 메시지
AIMessage	LLM이 출력한 메시지



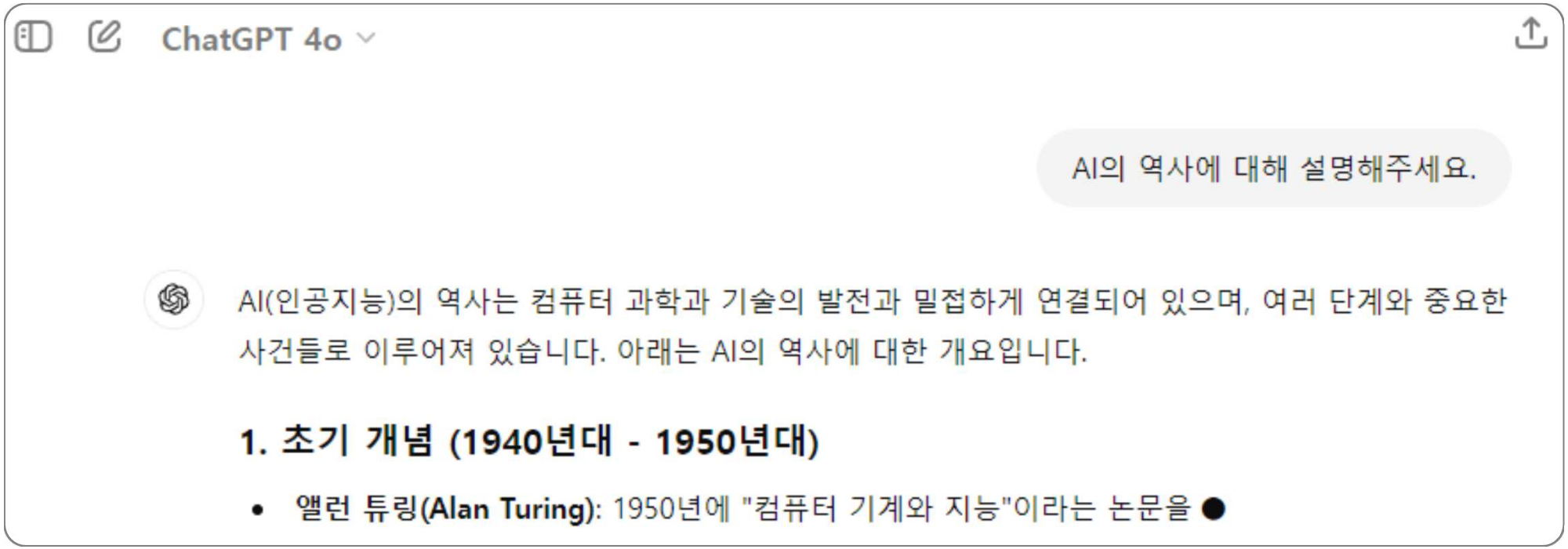
LLM의 Temperature 이해하기

▶ LLM의 매개변수 중 하나인 Temperature는 답변의 일관성을 조정합니다.



LLM 답변 스트리밍하기

▶ Langchain을 활용하면 LLM의 답변을 ChatGPT처럼 스트리밍할 수 있습니다.



LLM 답변 캐싱하기

▶ LLM은 답변을 생성하는 데에 시간과 비용을 소모합니다. 이를 효과적으로 관리하기 위해 같은 답은 캐싱하여 사용합니다.

Prompt Templates

Prompt와 Prompt Template이란 무엇인가?

"**프롬프트**"는 모델에 대한 입력을 의미합니다. 실제 LLM 서비스들의 경우, 사용자가 전부 입력하도록 만들지 않고 Back 단에서 여러 구성요소를 통해 편리한 입력을 지원하도록 합니다.

"**프롬프트 템플릿**"은 이러한 편리한 입력 지원을 위한 모듈입니다. Langchain은 프롬프트를 쉽게 구성하고 작업할 수 있도록 여러 클래스와 함수를 제공합니다.

나는 몹시 배가 고픈 상황!

냉장고를 털어서 할 수 있는 요리의 레시피를
ChatGPT에게 물어봐야겠다!

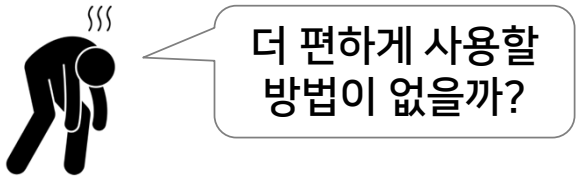
Prompt Template의 개념

[Day 1] 사과와 빵이 들어간 음식을 추천하고 그것의 레시피를 알려줘

[Day 2] 귤과 초콜렛이 들어간 음식을 추천하고, 그것의 레시피를 알려줘

[Day 3] 장어와 고추장이 들어간 음식을 추천하고, 그것의 레시피를 알려줘

⋮



"반복되는 Prompt를 Template으로 구성"

Day n. {음식1}와 {음식2}가 들어간 음식을 추천하고,
그것의 레시피를 알려줘

Prompt Template의 종류

▶ 기본적인 템플릿 설정을 위한 Prompt Templat과 상세 설정이 가능한 ChatPromptTemplate이 있습니다.

PromptTemplate

프롬프트

{매개변수}

```
"""
너는 요리사야. 내가 가진 재료들을 갖고 만들 수 있는 요리를 {개수}추천하고,
그 요리의 레시피를 제시해줘. 내가 가진 재료는 아래와 같아.

<재료>
{재료}
"""
```

ChatPromptTemplate

SystemMessage

{매개변수}

HumanMessage

{매개변수}

AIMessage

{매개변수}

```
SystemMessage(content=
"""
너는 항상 밝은 말투로 대화하는 챗봇이야.
답변의 끝에 이모티콘을 붙여줘.
""")

HumanMessage(content=
"""
오늘은 날씨가 어때?
""")

AIMessage(content=
"""
오늘은 날씨가 아주 좋아요!
""")
```

Few-shot Prompt Template

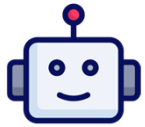
▶ 퓨샷 예제를 제공하면 해당 예제와 유사한 형태의 결과물을 출력합니다. 프롬프트로 표현하기 어려운 경우 사용합니다.

FewShotPromptTemplate

FewShotPromptTemplate 적용 전

질문
엔비디아로 4행시 만들어줘

답변
엔비디아를 통해 그래픽 능력
향상해인공지능을 구동해 머신 러닝을
위해뛰어난 성능으로 세상을
변화시키며혁신과 발전을 이끄는 리더가
돼라

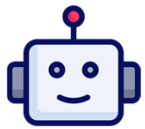


FewShotPromptTemplate 적용 후

질문
엔비디

N행시 예시:
Question: 아이유로 삼행시 만들어줘
Answer:
아: 아이유는
이: 이런 강의를 들을 이
유: 유가 없다.

답변
에: 엔비디아를 알고 나면,
로소 느낄 수 있어요, 그 진가를.
지털 세상을 혁신하는 그들의 힘을,
끼없이 보여주는 기술의 정수를.



Partial Prompt Template

▶ **Prompt Template의 매개변수 중 몇 개만 미리 지정해두는 Partial Prompt를 설정할 수 있습니다.**

매개변수가 2개인 프롬프트

프롬프트

Tell me a {adjective} joke
About the day {date}

오늘 날짜 계산식

```
from datetime import datetime
def _get_datetime():
now = datetime.now()
return now.strftime("%m/%d/%Y, %H:%M:%S")
```

Date 변수는
Partial로 미리 지정

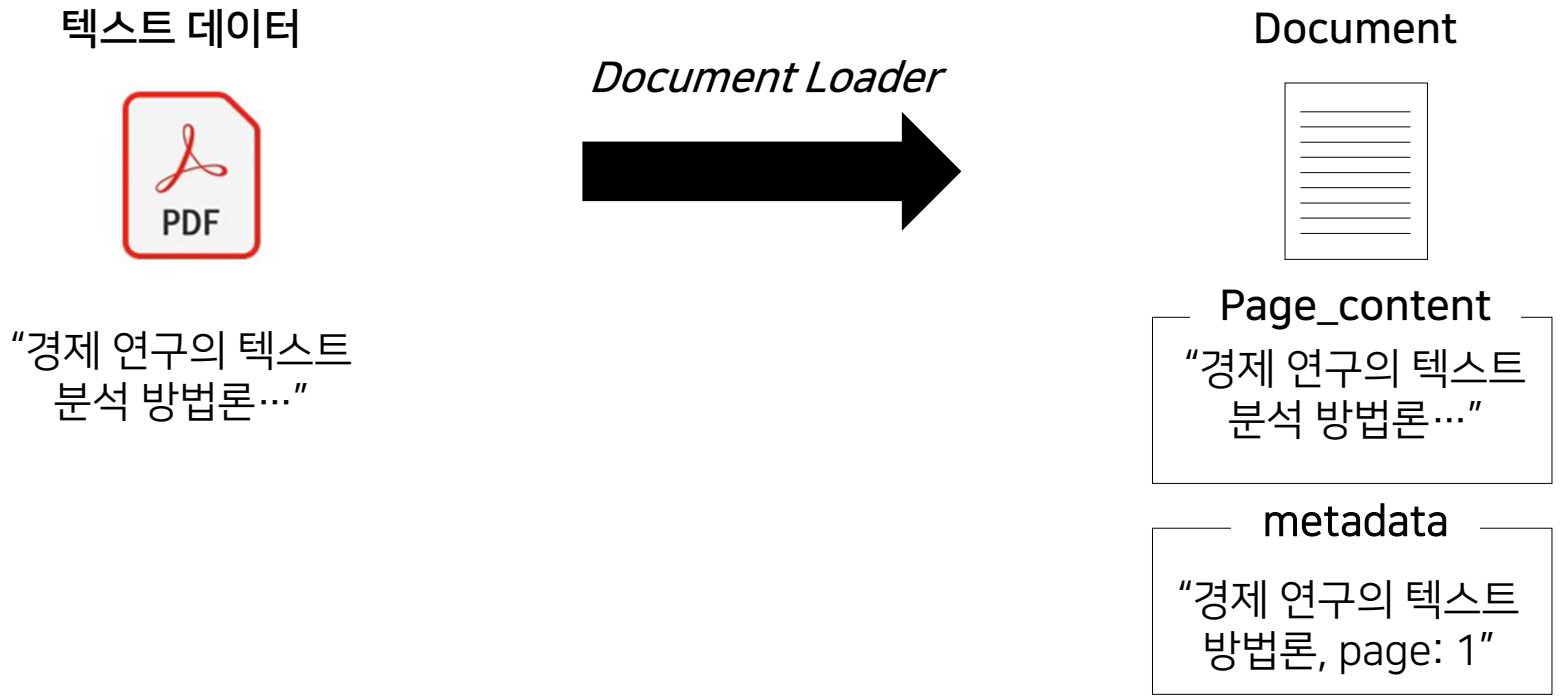
프롬프트

Tell me a {adjective} joke about the day
05/24/2024, 19:44:08

Document Loaders

Document Loader의 개념

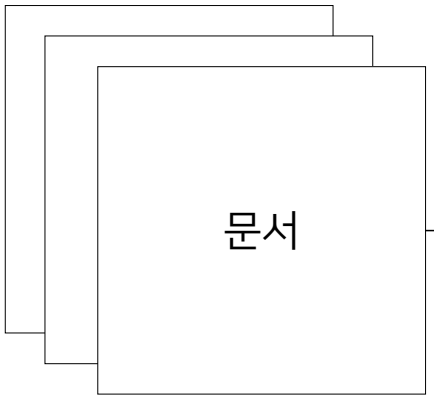
▶ **Document Loader는 주어진 문서를 RAG에서 활용하기 용이한 형태(Document 객체)로 변환하는 역할을 합니다.**



Document Loader의 개념

▶ **Document 객체는 문서의 내용을 담은 Page_content와 메타데이터로 이뤄진 Dictionary입니다.**

Document Loader




```
{'answer': ' The president honored Justice Breyer for his service and mentioned his legacy of
excellence.\n',
 'sources': '31-pl'}
```

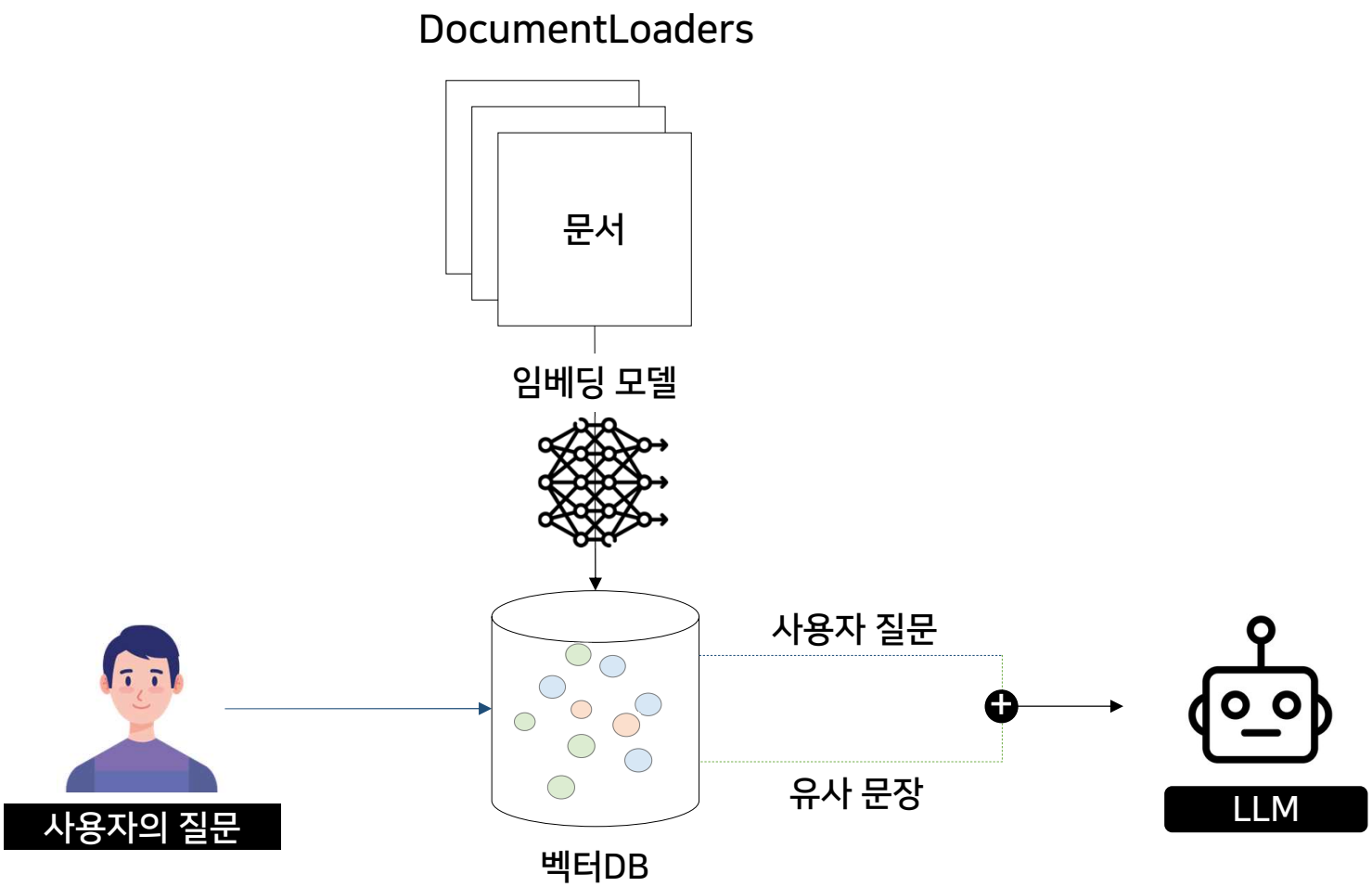
Page_content: 문서의 내용

Metadata: 문서의 위치, 제목, 페이지 넘버 등

Text Splitters

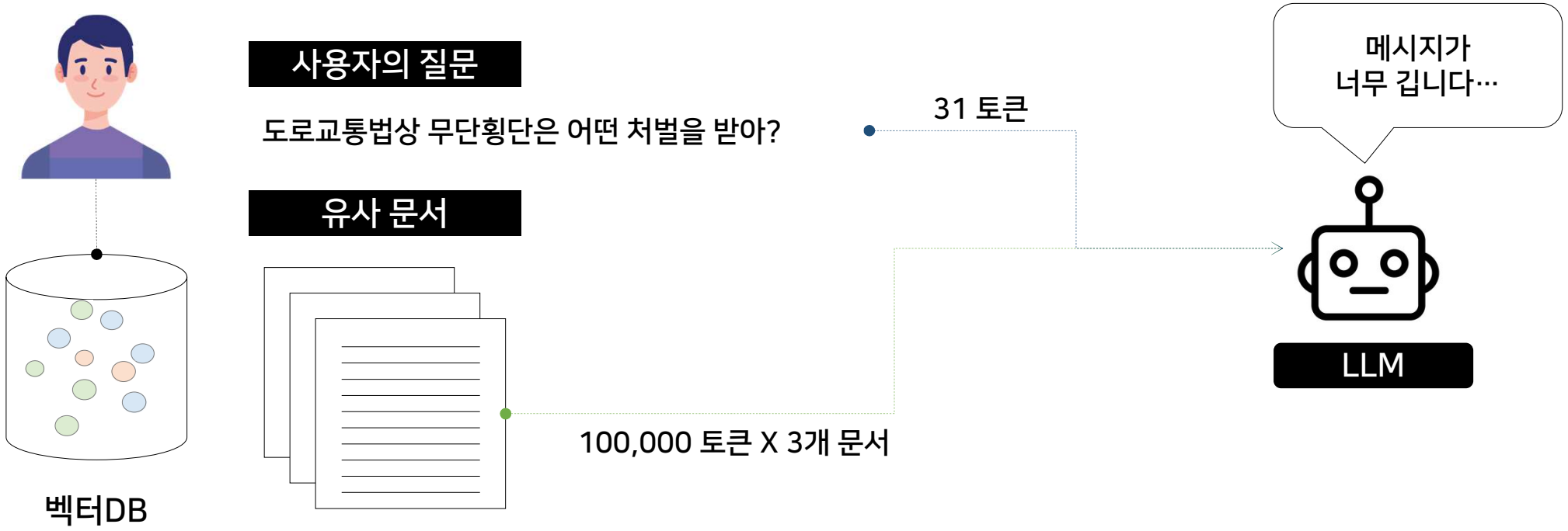
Text Splitter의 개념

▶ RAG는 Document Loader로 불러온 문서를 벡터 임베딩으로 변환하여 벡터DB에 저장하고 이를 활용합니다.



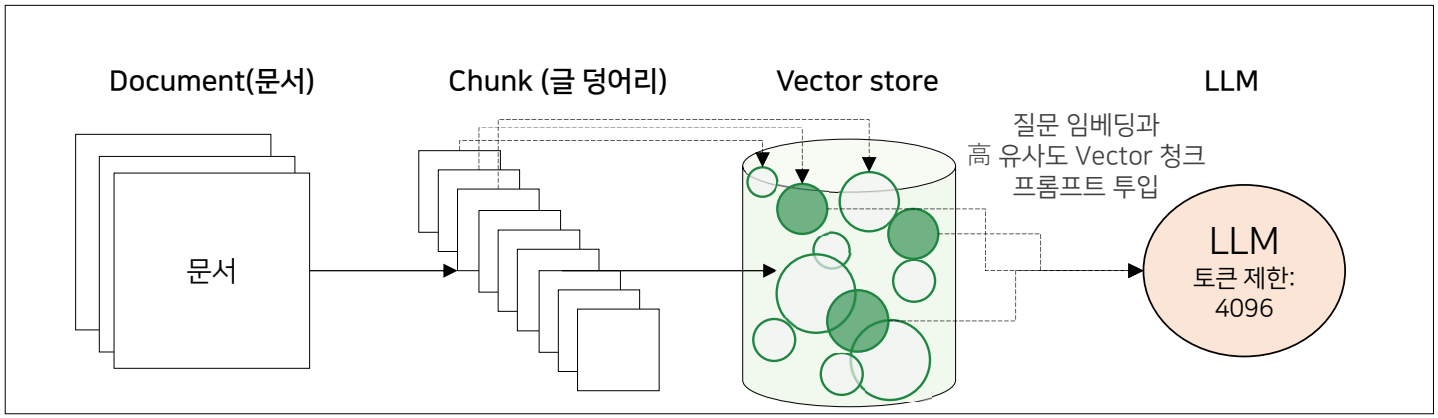
Text Splitter의 개념

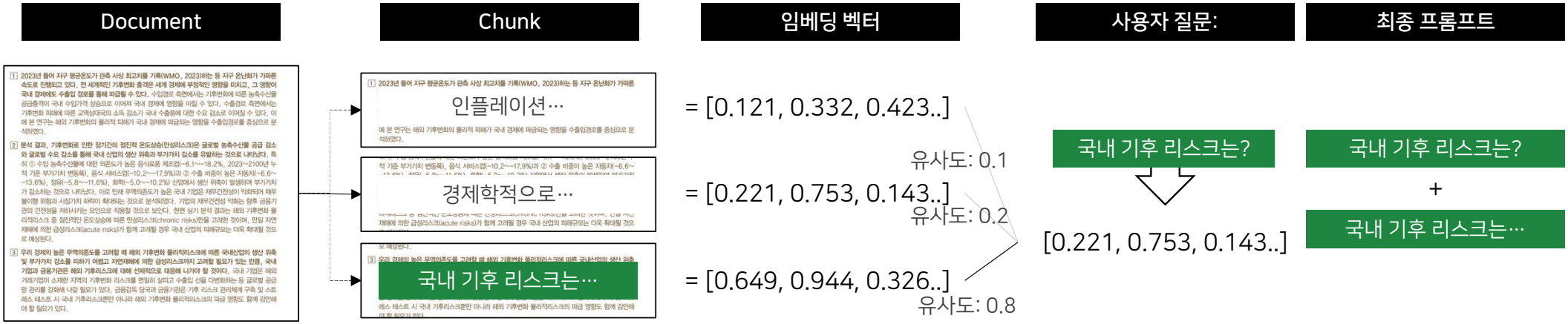
▶ 그런데, LLM에게 문서를 그대로 입력하여 답변하도록 하면 입력값 길이 제한으로 인해 오류가 발생할 수 있습니다.



Text Splitter의 개념

▶ 따라서 문서를 여러 개의 조각(Chunk)로 분할하여 벡터 DB에 저장하고, 이를 RAG 시스템에 활용합니다.





Text Splitter의 종류

▶ Text Splitter에는 기본 Splitter와 Recursive Splitter가 있습니다. 대부분의 경우 Recursive Splitter를 사용합니다.

CharacterTextSplitter

2023년 들어 지구 평균온도가 관측 사상 최고치를 기록(WMO, 2023)하는 등 지구 온난화가 가파른 속도로 진행되고 있다. 전 세계적인 기후변화 충격은 세계 경제에 부정적인 영향을 미치고, 그 영향이 국내 경제에도 수출입 경로를 통해 파급될 수 있다. 수입경로 측면에서는 기후변화에 따른 농축수산물 공급충격이 국내 수입가격 상승으로 이어져 국내 경제에 영향을 미칠 수 있다. 수출경로 측면에서는 기후변화 피해에 따른 교역상대국의 소득 감소가 국내 수출품에 대한 수요 감소로 이어질 수 있다. 이에 본 연구는 해외 기후변화의 물리적 피해가 국내 경제에 파급되는 영향을 수출입경로를 중심으로 분석하였다.

분석 결과, 기후변화로 인한 장기간의 점진적 온도상승(만성리스크)은 글로벌 농축수산물 공급 감소와 글로벌 수요 감소를 통해 국내 산업의 생산 위축과 부가가치 감소를 유발하는 것으로 나타났다. 특히 ① 수입 농축수산물에 대한 의존도가 높은 음식료품 제조업(-6.1~-18.2%, 2023~2100년 누적 기준 부가가치 변동폭), 음식 서비스업(-10.2~-17.9%)과 ② 수출 비중이 높은 자동차(-6.6~-13.6%), 정유(-5.8~-11.6%), 화학(-5.0~-10.2%) 산업에서 생산 위축이 발생하여 부가가치가 감소하는 것으로 나타났다. …

Max_token = 500

2023년 들어 지구 평균온도가 관측 사상 최고치를 기록(WMO, 2023)하는 등 지구 온난화가 가파른
속도로 진행되고 있다. 전 세계적인 기후변화 충격은 세계 경제에 부정적인 영향을 미치고, 그 영향이
국내 경제에도 수출입 경로를 통해 파급될 수 있다. 수입경로 측면에서는 기후변화에 따른 농축수산물
공급충격이 국내 수입가격 상승으로 이어져 국내 경제에 영향을 미칠 수 있다.
수출경로 측면에서는 기후변화 피해에 따른 교역상대국의 소득 감소가 국내 수출품에 대한 수요 감소로 이어질 수 있다. 이에 본 연구는 해외 기후변화의 물리적 피해가 국내 경제에 파급되는 영향을 수출입경로를 중심으로 분석하였다.

**구분자 1개 기준으로 분할하므로,
max_token을 지키지 못하는 경우 발생**

RecursiveCharacterTextSplitter

2023년 들어 지구 평균온도가 관측 사상 최고치를 기록(WMO, 2023)하는 등 지구 온난화가 가파른 속도로 진행되고 있다. 전 세계적인 기후변화 충격은 세계 경제에 부정적인 영향을 미치고, 그 영향이 국내 경제에도 수출입 경로를 통해 파급될 수 있다. 수입경로 측면에서는 기후변화에 따른 농축수산물 공급충격이 국내 수입가격 상승으로 이어져 국내 경제에 영향을 미칠 수 있다. 수출경로 측면에서는 기후변화 피해에 따른 교역상대국의 소득 감소가 국내 수출품에 대한 수요 감소로 이어질 수 있다. 이에 본 연구는 해외 기후변화의 물리적 피해가 국내 경제에 파급되는 영향을 수출입경로를 중심으로 분석하였다.

분석 결과, 기후변화로 인한 장기간의 점진적 온도상승(만성리스크)은 글로벌 농축수산물 공급 감소와 글로벌 수요 감소를 통해 국내 산업의 생산 위축과 부가가치 감소를 유발하는 것으로 나타났다. 특히 ① 수입 농축수산물에 대한 의존도가 높은 음식료품 제조업(-6.1~-18.2%, 2023~2100년 누적 기준 부가가치 변동폭), 음식 서비스업(-10.2~-17.9%)과 ② 수출 비중이 높은 자동차(-6.6~-13.6%), 정유(-5.8~-11.6%), 화학(-5.0~-10.2%) 산업에서 생산 위축이 발생하여 부가가치가 감소하는 것으로 나타났다. …

Max_token = 500

2023년 들어 지구 평균온도가 관측 사상 최고치를 기록(WMO, 2023)하는 등 지구 온난화가 가파른
속도로 진행되고 있다. 전 세계적인 기후변화 충격은 세계 경제에 부정적인 영향을 미치고, 그 영향이
국내 경제에도 수출입 경로를 통해 파급될 수 있다. 수입경로 측면에서는 기후변화에 따른 농축수산물
공급충격이 국내 수입가격 상승으로 이어져 국내 경제에 영향을 미칠 수 있다.

**줄바꿈, 마침표, 쉼표 순으로 재귀적으로 분할하므로,
max_token 지켜 분할**

Text Splitter의 매개변수

▶ **Chunk_overlap이라는 매개변수는 텍스트 분할 시, 앞뒤로 조금씩 겹치게 만들어 문맥을 더 많이 포함하도록 합니다.**

1

2023년 들어 지구 평균온도가 관측 사상 최고치를 기록(WMO, 2023)하는 등 지구 온난화가 가파른 속도로 진행되고 있다. 전 세계적인 기후변화 충격은 세계 경제에 부정적인 영향을 미치고, 그 영향이 국내 경제에도 수출입 경로를 통해 파급될 수 있다. 수입경로 측면에서는 기후변화에 따른 농축수산물 공급충격이 국내 수입가격 상승으로 이어져 국내 경제에 영향을 미칠 수 있다. 수출경로 측면에서는 기후변화 피해에 따른 교역상대국의 소득 감소가 국내 수출품에 대한 수요 감소로 이어질 수 있다. 이에 본 연구는 해외 기후변화의 물리적 피해가 국내 경제에 파급되는 영향을 수출입경로를 중심으로 분석하였다.

2

분석 결과, 기후변화로 인한 장기간의 점진적 온도상승(만성리스크)은 글로벌 농축수산물 공급 감소와 글로벌 수요 감소를 통해 국내 산업의 생산 위축과 부가가치 감소를 유발하는 것으로 나타났다. 특히 ① 수입 농축수산물에 대한 의존도가 높은 음식료품 제조업(-6.1~-18.2%, 2023~2100년 누적 기준 부가가치 변동폭), 음식 서비스업(-10.2~-17.9%)과 ② 수출 비중이 높은 자동차(-6.6~-13.6%), 정유(-5.8~-11.6%), 화학(-5.0~-10.2%) 산업에서 생산 위축이 발생하여 부가가치가 감소하는 것으로 나타났다. …

1

2023년 들어 지구 평균온도가 관측 사상 최고치를 기록(WMO, 2023)하는 등 지구 온난화가 가파른 속도로 진행되고 있다. 전 세계적인 기후변화 충격은 세계 경제에 부정적인 영향을 미치고, 그 영향이 국내 경제에도 수출입 경로를 통해 파급될 수 있다. 수입경로 측면에서는 기후변화에 따른 농축수산물 공급충격이 국내 수입가격 상승으로 이어져 국내 경제에 영향을 미칠 수 있다. 수출경로 측면에서는 기후변화 피해에 따른 교역상대국의 소득 감소가 국내 수출품에 대한 수요 감소로 이어질 수 있다. 이에 본 연구는 해외 기후변화의 물리적 피해가 국내 경제에 파급되는 영향을 수출입경로를 중심으로 분석하였다.

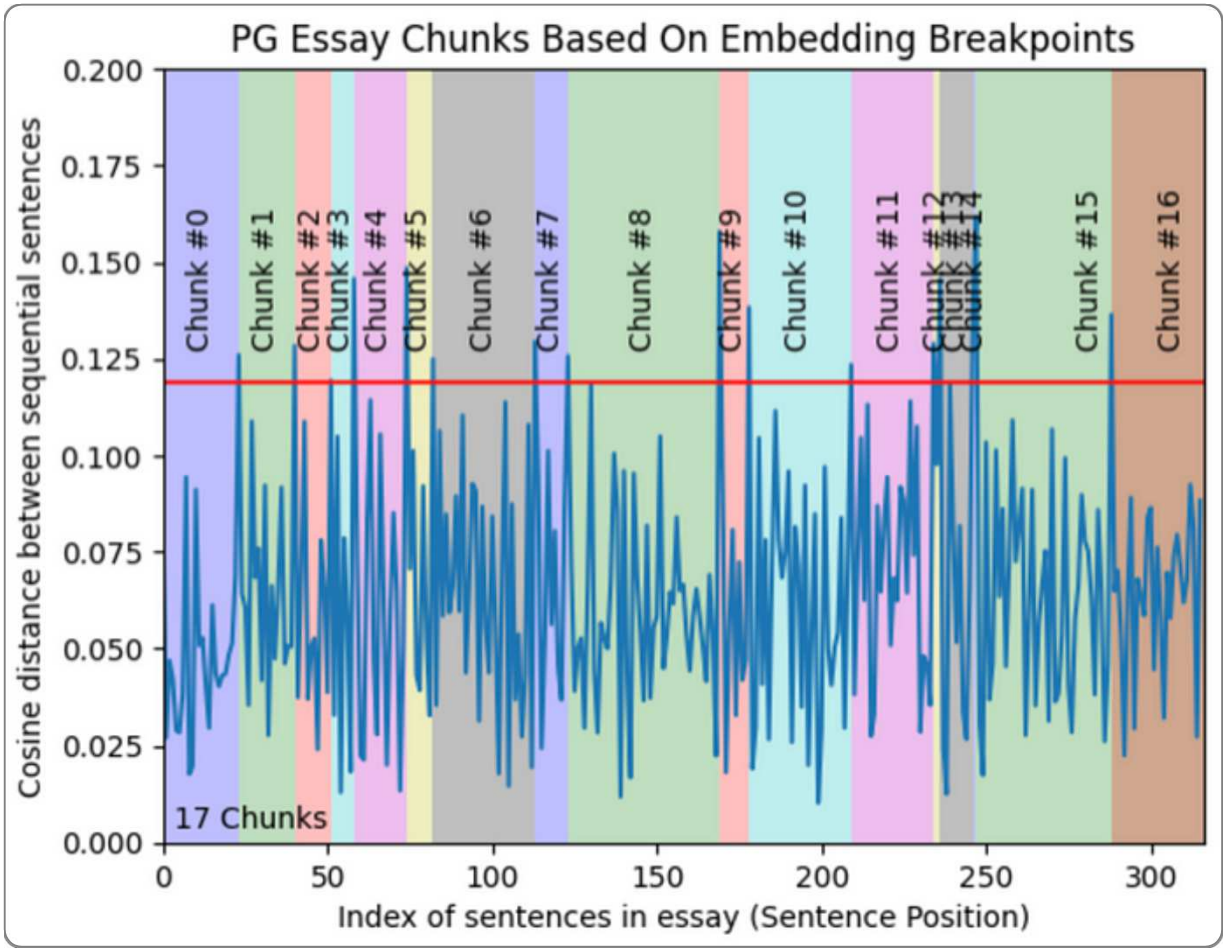
2

chunk_overlap

이에 본 연구는 해외 기후변화의 물리적 피해가 국내 경제에 파급되는 영향을 수출입경로를 중심으로 분석하였다. 분석 결과, 기후변화로 인한 장기간의 점진적 온도상승(만성리스크)은 글로벌 농축수산물 공급 감소와 글로벌 수요 감소를 통해 국내 산업의 생산 위축과 부가가치 감소를 유발하는 것으로 나타났다. 특히 ① 수입 농축수산물에 대한 의존도가 높은 음식료품 제조업(-6.1~-18.2%, 2023~2100년 누적 기준 부가가치 변동폭), 음식 서비스업(-10.2~-17.9%)과 ② 수출 비중이 높은 자동차(-6.6~-13.6%), 정유(-5.8~-11.6%), 화학(-5.0~-10.2%) 산업에서 생산 위축이 발생하여 부가가치가 감소하는 것으로 나타났다. …

Semantic Chunker

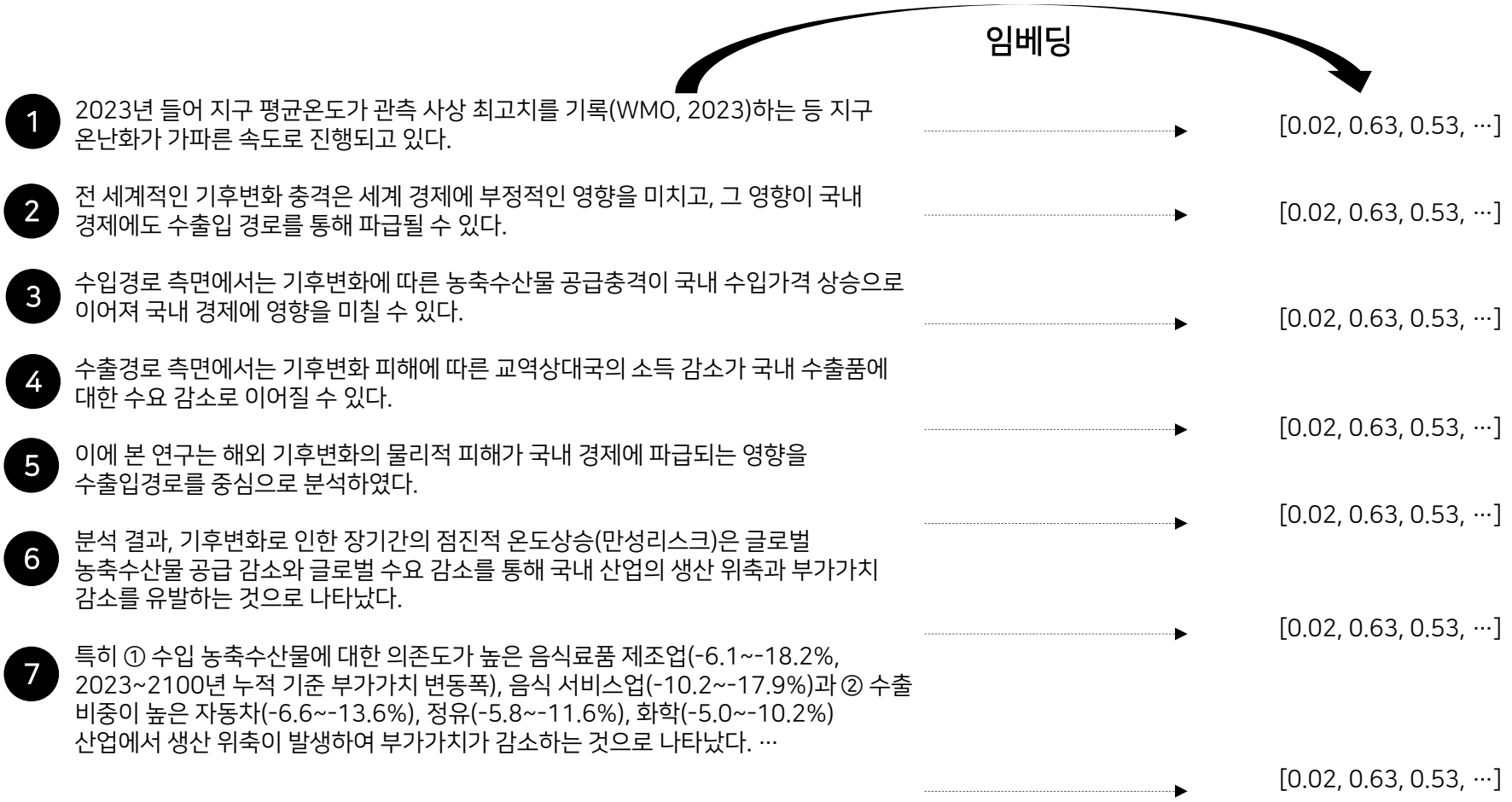
▶ 문자 분할기(Character)나 재귀적 분할기(Recursive)처럼 기계적으로 분할하지 않고, 유사성 기반으로 분할합니다.



Embeddings

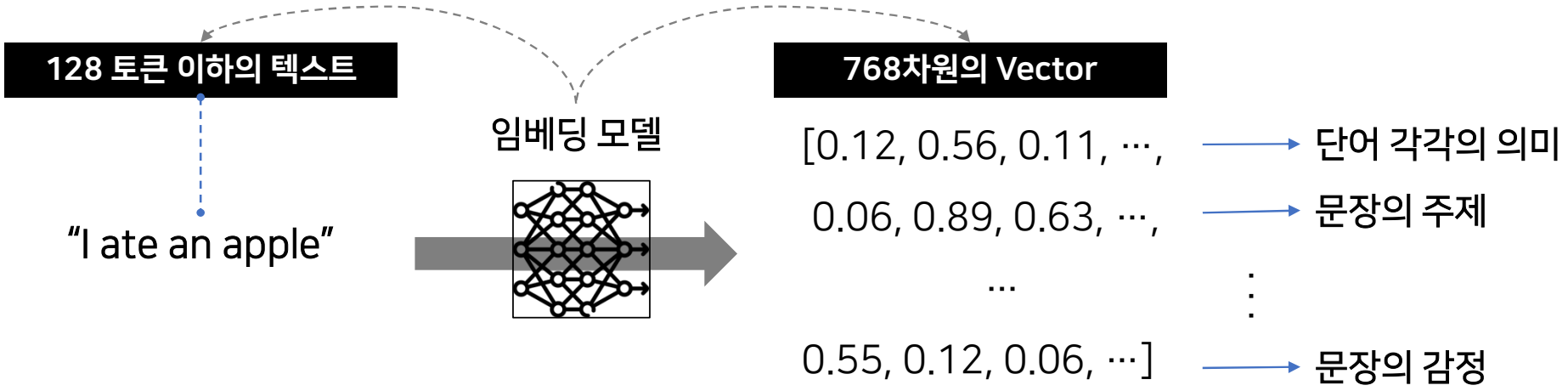
Embedding의 개념

▶ **Embedding은 텍스트 데이터를 수치 데이터로 변환하는 역할을 합니다. 이는 RAG에서 핵심적인 역할을 합니다.**



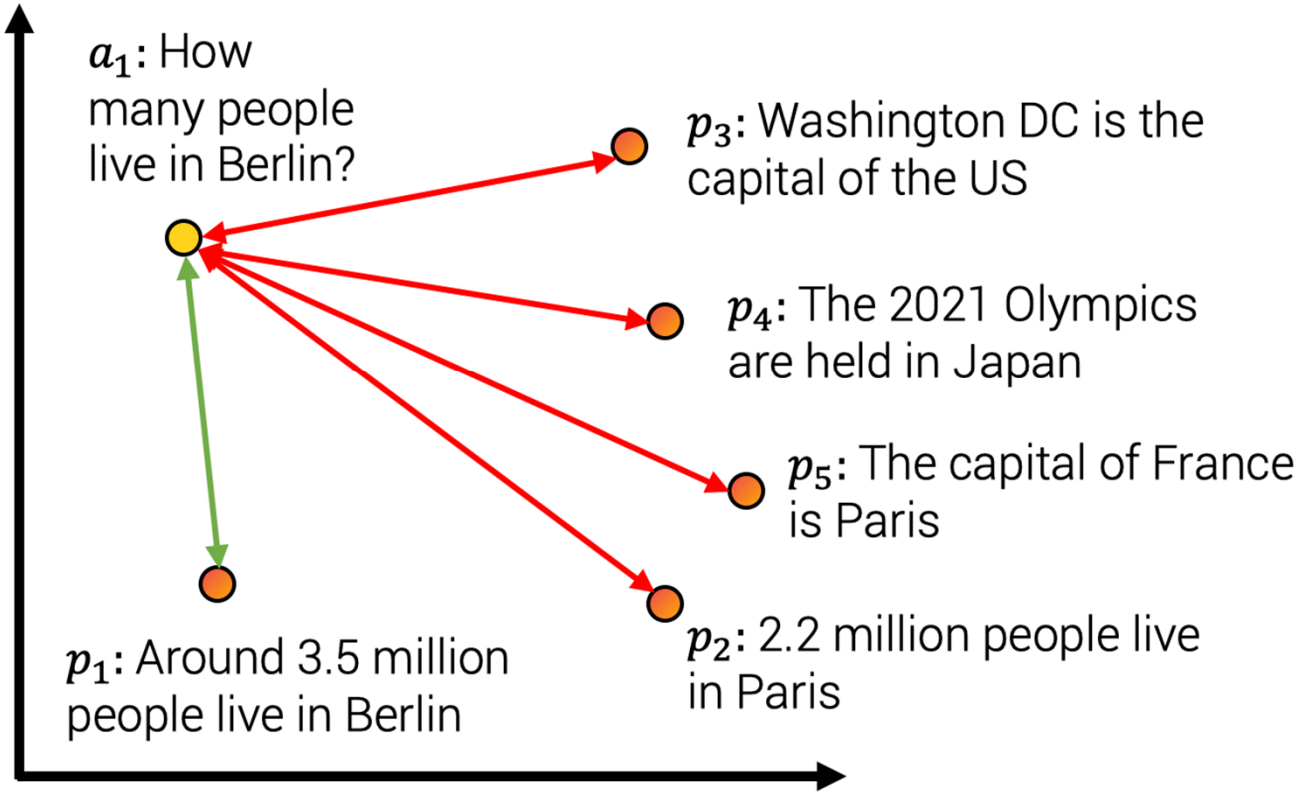
Embedding의 개념

▶ Embedding 과정에서 문장의 의미, 주제, 감정 등 다양한 정보를 담기 위해 고차원의 Vector로 변환합니다.



Embedding 원리

▶ 대부분의 경우 대용량의 말뭉치를 통해 사전학습된 모델을 통해 쉽게 임베딩합니다.



대용량의 말뭉치로 임베딩 모델 훈련

⇩

Pre-trained Embedding Model

⇩

학습 없이 Embedding 수행 가능

Embedding 모델의 종류

▶ 사전학습 임베딩 모델에는 대표적으로 OpenAI에서 제공하는 ada 모델과, HuggingFace의 모델들이 있습니다.
사용 목적과 요구사항에 따라 적절한 임베딩을 고르는 것은 RAG의 가장 중요한 부분입니다.

구분	기업명	모델명	장단점
유료 임베딩 모델	OpenAI Cohere Amazon . : :	text-embedding-ada-002 embed-multilingual-v2.0 titan-embed-text-v1 . : :	- 사용하기 편리하지만 비용 발생 - API 통신 이용하므로 보안 우려 - 한국어 포함 많은 언어 임베딩 지원 - GPU 없이도 빠른 임베딩
로컬 임베딩 모델	HuggingFace	bge-large-en-v1.5 multilingual-e5-large instructor-xl ko-sbert-nli KoSimCSE-roberta-multitask	- 무료지만 다소 어려운 사용 - 오픈소스 모델 사용하므로 보안 우수 - 모델마다 지원 언어가 다름 - GPU 없을 시, 느린 임베딩

Vectorstores

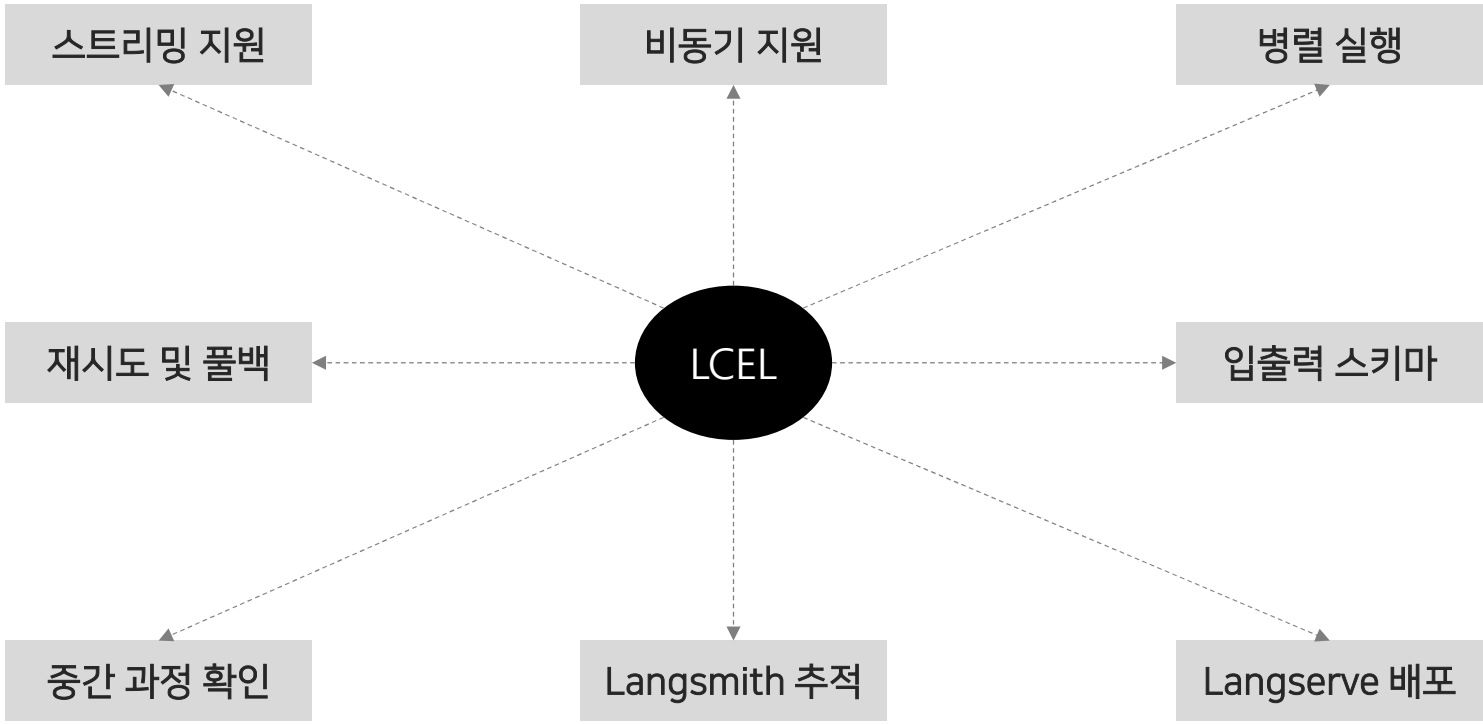
VectorStore의 개념

▶ Vector DB(Store)는 고차원의 벡터 임베딩을 인덱싱(저장)하고 검색하는 것에 특화된 데이터베이스입니다.

LCEL 소개 및 기본 구조

LCEL이란?

▶ LCEL(Langchain Expression Language)은 Langchin 프레임워크에서 사용되는 언어로, 다양한 장점이 있습니다.



LCEL의 형태

▶ LCEL은 파이프 오퍼레이터("|")를 통해 짧은 코드만으로 Chain을 구성할 수 있습니다.

[LCEL로 농담 생성 Chain 구축 예시 코드]

```
from langchain_openai import ChatOpenAI
from langchain_core.output_parsers import StrOutputParser
from langchain_core.prompts import ChatPromptTemplate

model = ChatOpenAI(model="gpt-4o-mini") --- 모델 선언

prompt = ChatPromptTemplate.from_template("tell me a short joke about {topic}") --- 프롬프트 선언
output_parser = StrOutputParser() --- 출력 형태 지정

chain = prompt | model | output_parser --- Chain 구성

chain.invoke({ " topic " :  " ice cream " }) --- Chain 호출
```

LCEL을 호출하는 방법

▶ **LCEL은 가독성 높은 코드를 바탕으로, invoke 함수를 통해 LLM의 답변을 받아볼 수 있습니다.**

[Langchain 없이 Chain 구축 및 실행]

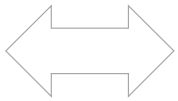
```
from typing import List
import openai

prompt_template = "Tell me a short joke about
{topic}"
client = openai.OpenAI()

def call_chat_model(messages: List[dict]) -> str:
response = client.chat.completions.create(
model="gpt-3.5-turbo",
messages=messages,
)
return response.choices[0].message.content

def invoke_chain(topic: str) -> str:
prompt_value =
prompt_template.format(topic=topic)
messages = [{"role": "user", "content":
prompt_value}]
return call_chat_model(messages)

invoke_chain("ice cream")
```



[LCEL로 Chain 구축 및 실행]

```
from langchain_openai import ChatOpenAI
from langchain_core.prompts import
ChatPromptTemplate
from langchain_core.output_parsers import
StrOutputParser
from langchain_core.runnables import
RunnablePassthrough


prompt = ChatPromptTemplate.from_template(
"Tell me a short joke about {topic}"
)
output_parser = StrOutputParser()
model = ChatOpenAI(model="gpt-3.5-turbo")
chain = (
{"topic": RunnablePassthrough()}
| prompt
| model
| output_parser
)

chain.invoke("ice cream")
```

LCEL로 답변 스트리밍 하기

▶ **LCEL을 사용하면, Chain의 내장 함수인 Stream을 통해 두줄만으로 LLM 답변 스트리밍이 가능합니다.**

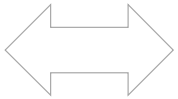
[Langchain 없이 LLM 답변 스트리밍]

```
from typing import Iterator

def stream_chat_model(messages: List[dict]) ->
Iterator[str]:
    stream = client.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=messages,
        stream=True,
    )
    for response in stream:
        content = response.choices[0].delta.content
        if content is not None:
            yield content

def stream_chain(topic: str) -> Iterator[str]:
    prompt_value = prompt.format(topic=topic)
    return stream_chat_model([{"role": "user",
"content": prompt_value}])

for chunk in stream_chain("ice cream"):
    print(chunk, end="", flush=True)
```



[LCEL로 LLM 답변 스트리밍]

```
for chunk in chain.stream("ice cream"):
    print(chunk, end="", flush=True)
```

LCEL로 여러 답변 Batch 처리

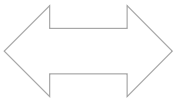
▶ LCEL을 사용하면, 여러 프롬프트를 한꺼번에 실행하는 Batch 처리가 용이합니다.

[Langchain 없이 배치 처리]

```python
from concurrent.futures import ThreadPoolExecutor


def batch_chain(topics: list) -> list:
    with ThreadPoolExecutor(max_workers=5) as executor:
        return list(executor.map(invoke_chain, topics))

batch_chain(["ice cream", "spaghetti", "dumplings"])
```



[LCEL로 배치 처리]

```python
chain.batch(["ice cream", "spaghetti",
"dumplings"])
```

LCEL로 비동기 실행

▶ LCEL을 사용하면, 동시에 API 호출을 여러 개 실행하는 비동기 처리가 용이합니다.

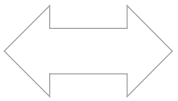
[Langchain 없이 비동기 처리]

```
async_client = openai.AsyncOpenAI()

async def acall_chat_model(messages: List[dict]) -> str:
    response = await
async_client.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=messages,
    )
    return response.choices[0].message.content

async def ainvoke_chain(topic: str) -> str:
    prompt_value = prompt_template.format(topic=topic)
    messages = [{"role": "user", "content":
prompt_value}]
    return await acall_chat_model(messages)


await ainvoke_chain("ice cream")
```



[LCEL로 비동기 처리]

```
await chain.ainvoke("ice cream")
```

Runnable 인터페이스와 Chaining

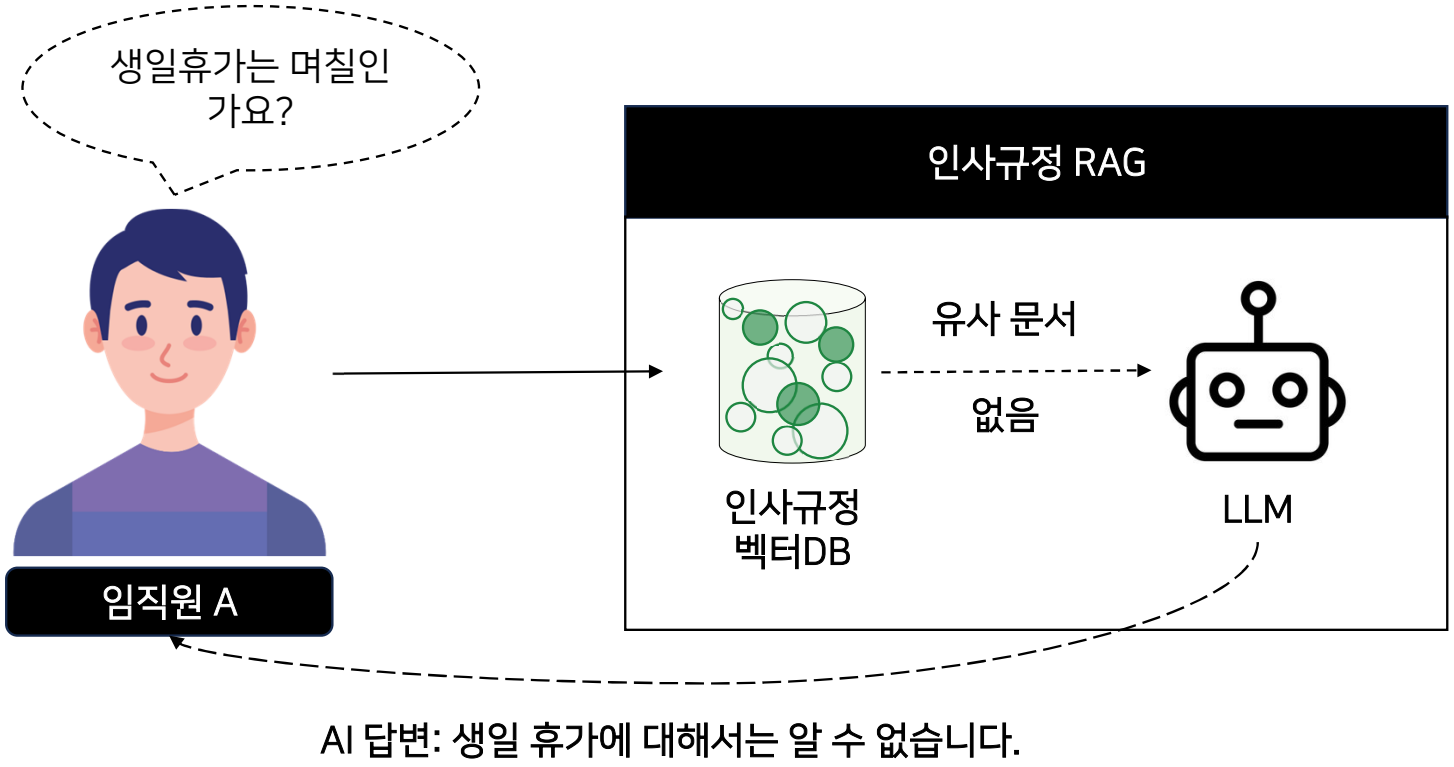
RunnablePassthrough
RunnableLambda
RunnableParallel

LCEL을 사용한 RAG 파이프라인 구성

다중 쿼리 생성 기법 (Multi-query Retriever)

기존 RAG의 문제점

▶ 기존의 RAG는 사용자의 질문이 모호할 경우, 벡터 DB 내 문장들과 매칭되지 않는 경우가 있습니다.

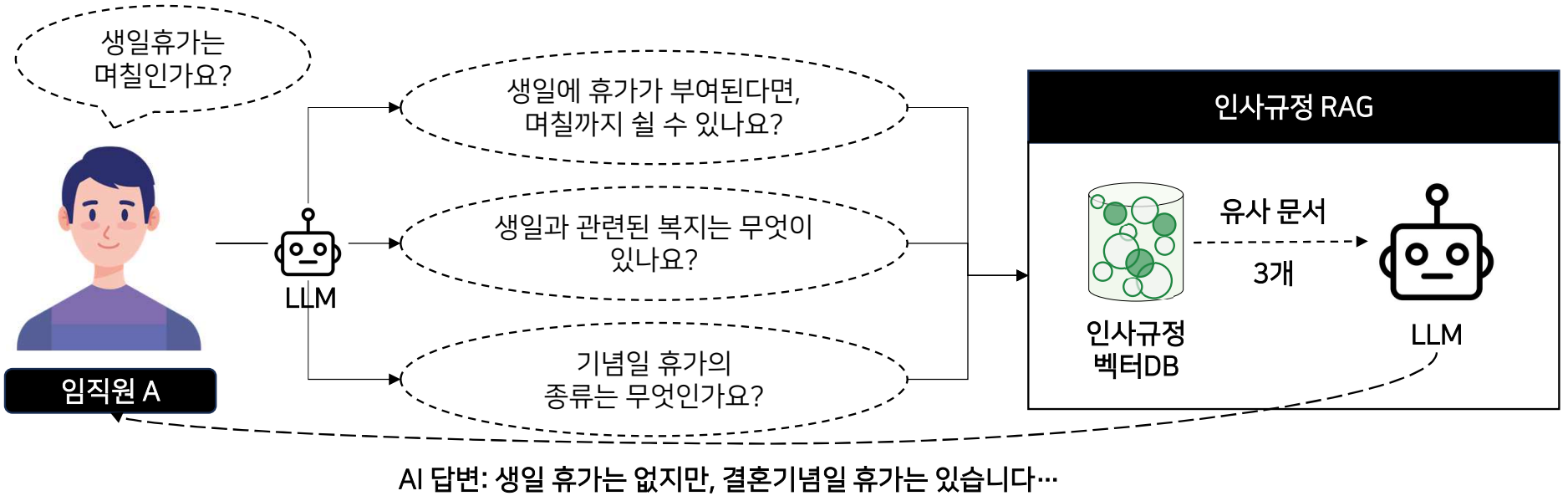


AI 답변: 생일 휴가에 대해서는 알 수 없습니다.

MultiQueryRetriever 원리

▶ **MultiQueryRetriever는 사용자 질문의 의도를 LLM이 이해하여 이를 여러 질문으로 재생성합니다.**

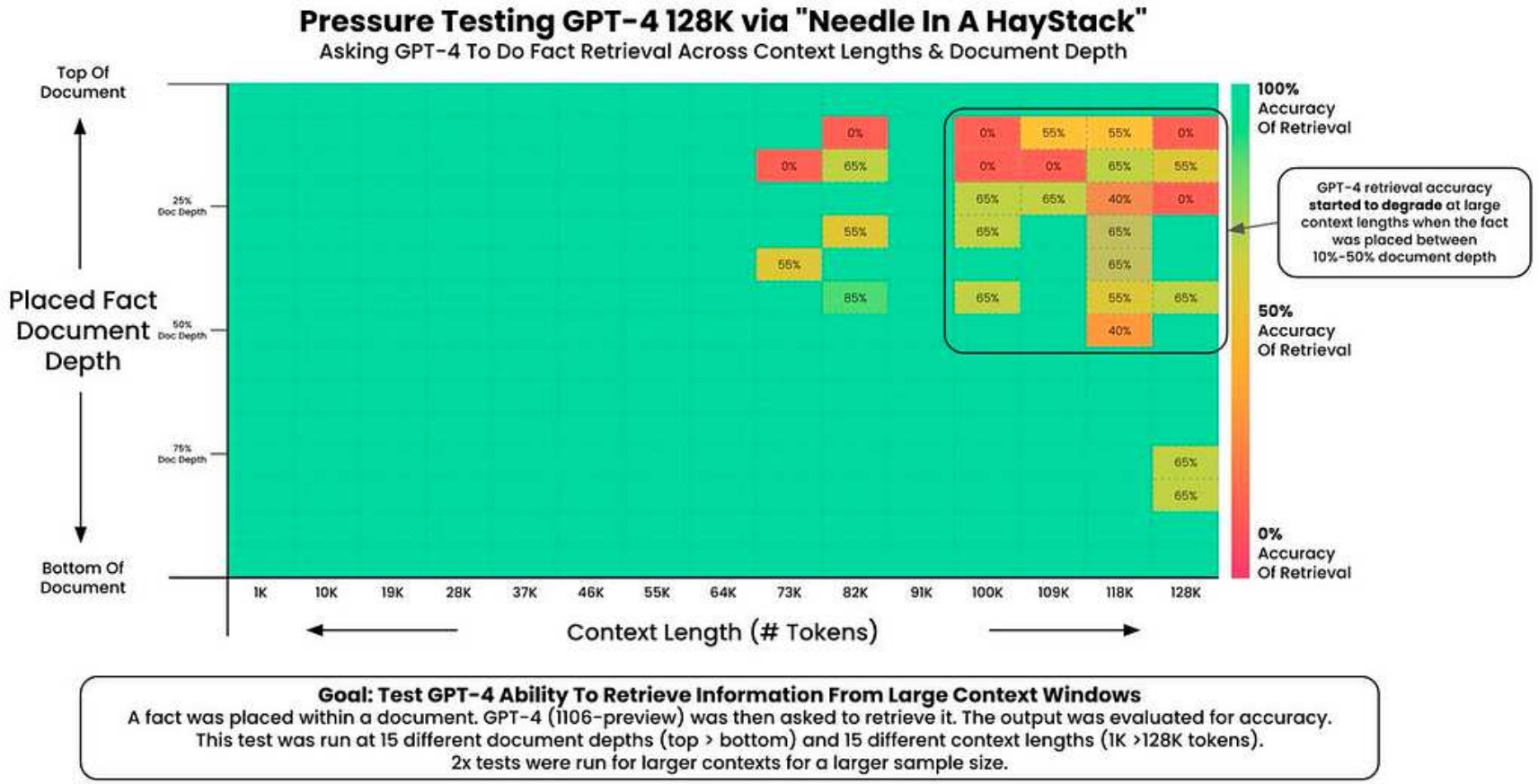
의도에 알맞은 여러 질문을 생성하면, 벡터 DB 내 유사 문서를 찾을 가능성이 이전 대비 크게 높아집니다.



재순위화 기법 (Rerank)

Needle in a Haystack

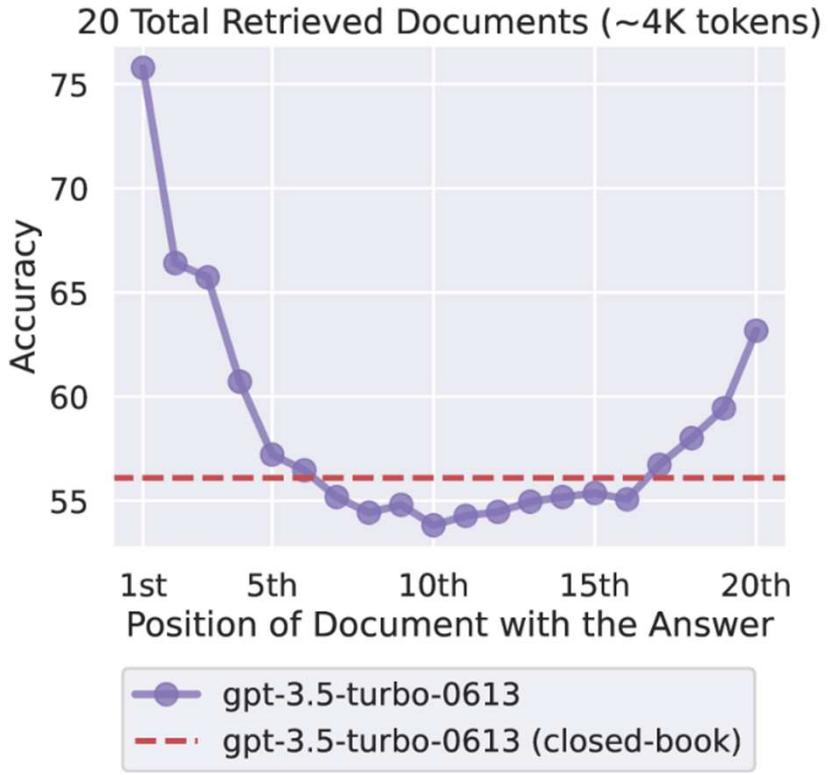
▶ LLM의 컨텍스트 사이즈가 늘어나며, 긴 입력이 가능해졌지만 중간 문서는 잘 찾지 못하는 문제가 있습니다.



MultiQueryRetriever 원리

▶ 실제 연구 결과 또한, LLM의 중간 문서 탐색 문제를 증명하고 있습니다.

▶ Langchain의 Long context reorder로 RAG에서 검색된 문서를 Reorder할 수 있습니다.

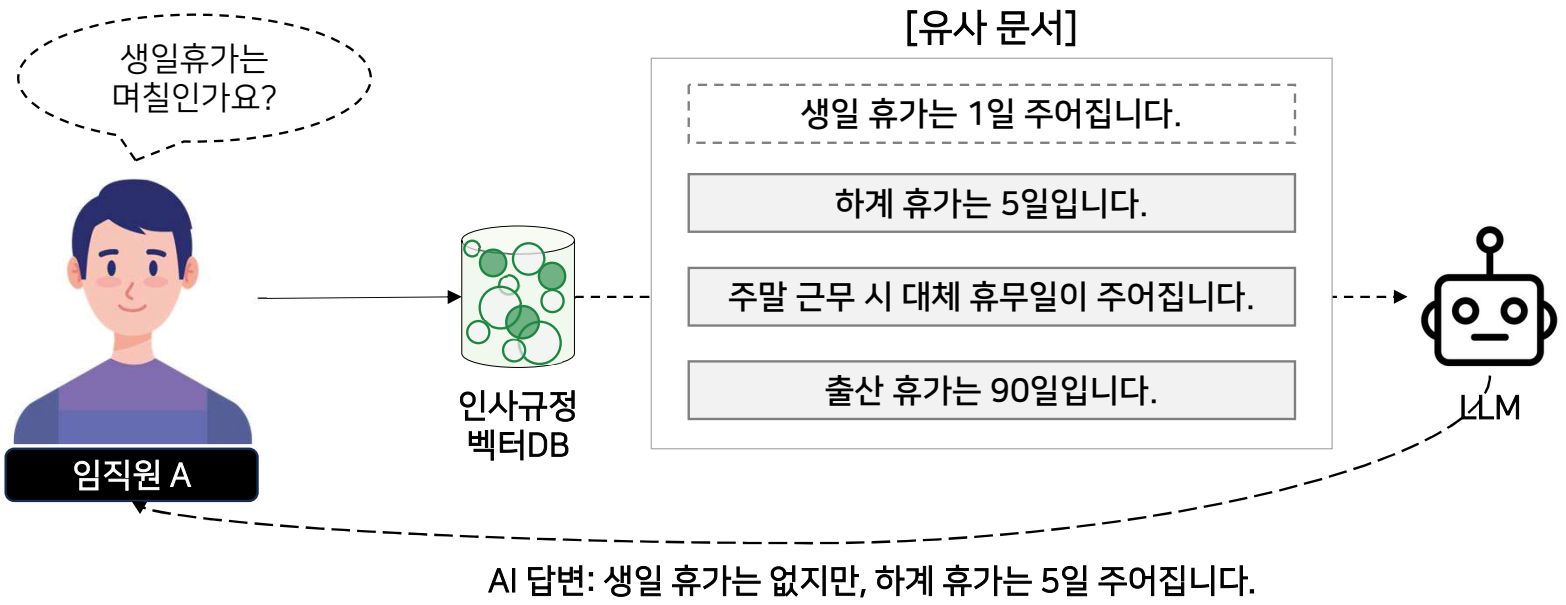


연관성이 높은 문서는 맨앞과 맨뒤로 Reorder!

맥락 압축 기법 (Context Compression)

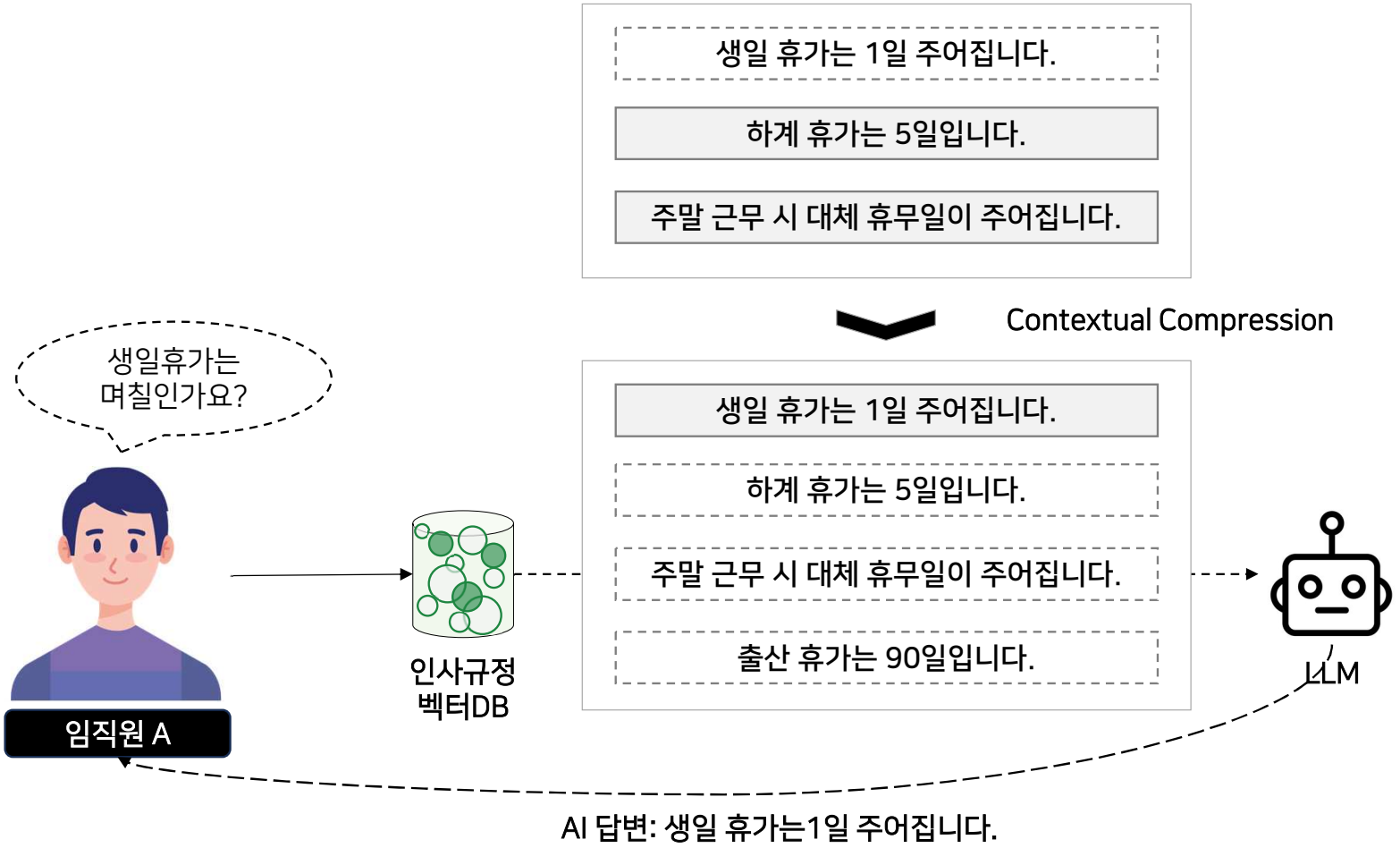
맥락압축의 필요성

▶ RAG가 제대로 작동하려면, 알맞은 유사 문서만 검색되어야만 하지만 그렇지 않은 경우가 있습니다.



Contextual compression

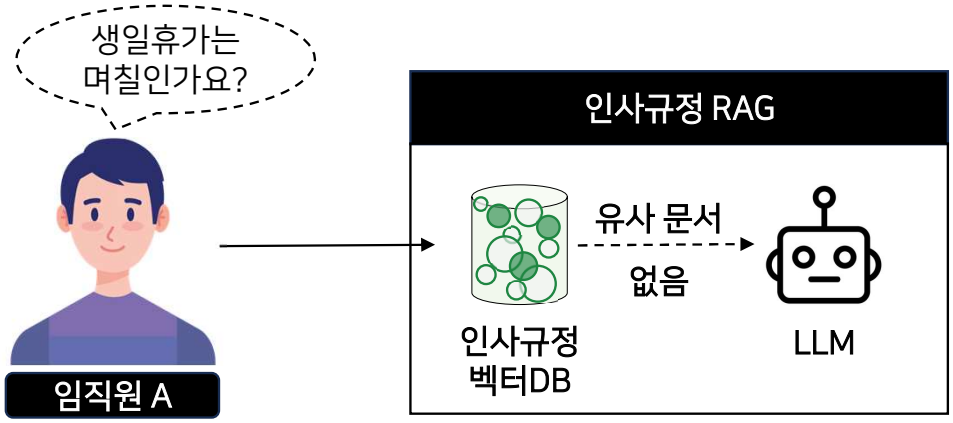
▶ Contextual compression은, 주어진 사용자 질문의 문맥을 사용하여 관련 정보만 반환합니다.



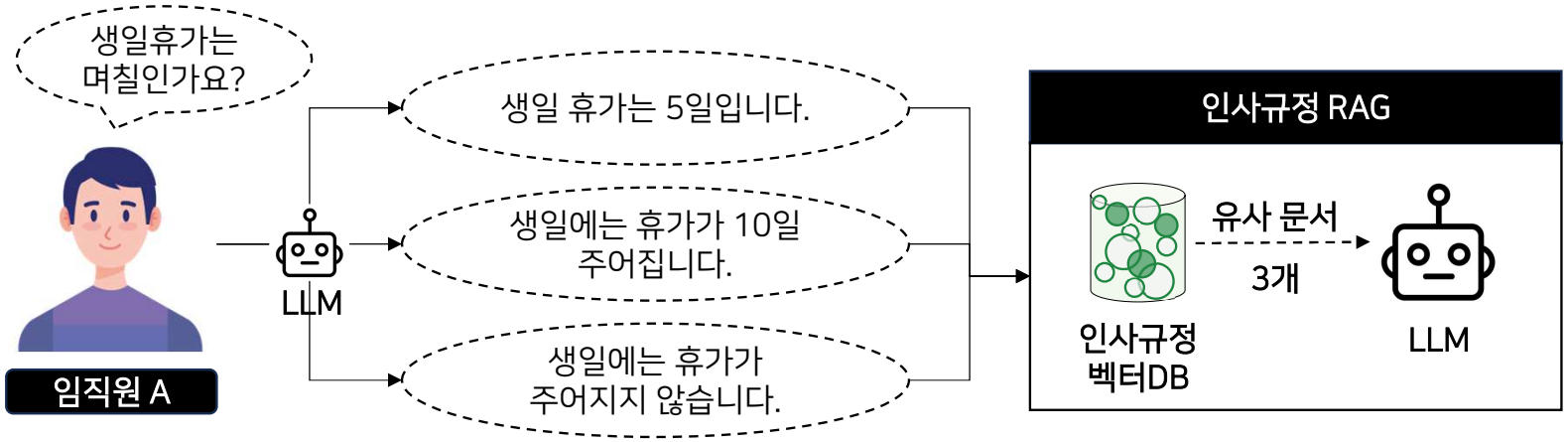
HyDE
(Hypothetical Document Embeddings)

기존 RAG의 문제와 HyDE

▶ 사용자의 질문과 벡터 DB 내 문장들의 유사성 비교는 종종 올바른 문서 검색이 어렵게 만듭니다.



▶ HyDE에서는 사용자 질문에 대한 가상의 답변을 작성하여 벡터 DB 속 유사 문장을 더 잘 검색할 수 있게 만듭니다.



THANK YOU

마소캠퍼스 www.masocampus.com
이메일 문의 biz@masocampus.com
전화 문의 02-6080-2022

Actionable Content
MASO CAMPUS