



# RAG:

# 구라같은데...



# 목차 페이지

01

챗지피티도 모르는게 있다.

02

RAG란?

03

RAG 체험



# 01

## 챗지피티도 모르는게 있다.

---

이걸 모르네...

# 챗지피티, LLM?



## LLM 이란?

대형 언어 모델(영어: large language model, LLM)은 수많은 파라미터(보통 수십억 웨이트 이상)를 보유한 인공 신경망으로 구성되는 언어 모델이다. 자기 지도 학습이나 반자기지도 학습을 사용하여 레이블링되지 않은 상당한 양의 텍스트로 훈련된다. LLM은 2018년 즈음에 모습을 드러냈으며 다양한 작업을 위해 수행된다. 이전의 특정 작업의 특수한 지도 학습 모델의 훈련 패러다임에서 벗어나 자연어 처리 연구로 초점이 옮겨졌다.



# LLM의 한계



01

학습된 데이터를 기반으로 적절하다고 생각되는 문장 출력  
→환각 현상(hallucination)이 발생한다

02

사내 정보, 최신 정보는 반영하기 힘들다.

03

명확한 근거를 가지고 오지 않는다.

# LLM의 한계



하나지방의 챔피언은 누구야?

하나지방(ポケモン 스カーレット・바이올렛의 무대가 되는 지역)의 챔피언은 **오모리 카이리(네모, Nemona)** 같은 "라이벌 챔피언"이 아니라, 스토리상 정식으로는 **플레이어(주인공 자신)** 이 챔피언이 돼요.

즉, 기존 지방처럼 특정 인물이 챔피언 자리를 지키고 있는 게 아니라, 플레이어가 챔피언로드(체육관 챌린지 → 챔피언 테스트 → 리포리움 사천왕과 챔피언 전투)를 클리어하면서 챔피언 칭호를 얻게 되는 구조예요.

- **네모(Nemona)** 는 이미 챔피언급 트레이너라서 마지막에 플레이어와 "챔피언 라이벌전"을 펼치는 역할을 하고,
- 그 외의 인물들(예: 오르티가, 카이덴, 아이노 등)도 체육관 관장이나 스토리 캐릭터일 뿐 챔피언은 아님.

정리하면:

👉 하나지방의 챔피언 = **플레이어 본인**

혹시 너가 묻는 게 "스토리 진행상 최종 보스로 싸우는 인물"을 말하는 거야, 아니면 "설정상 그 지방을 대표하는 챔피언"을 말하는 거야?





# 02

## RAG : 구라같은데...

---

???: 내가 가르쳐줄게

# RAG

## Retrieval



검색

외부에서 관련 자료를 찾아옴

## Augmented



보강

검색된 자료를 LLM 프롬프트에 추가

## Generation

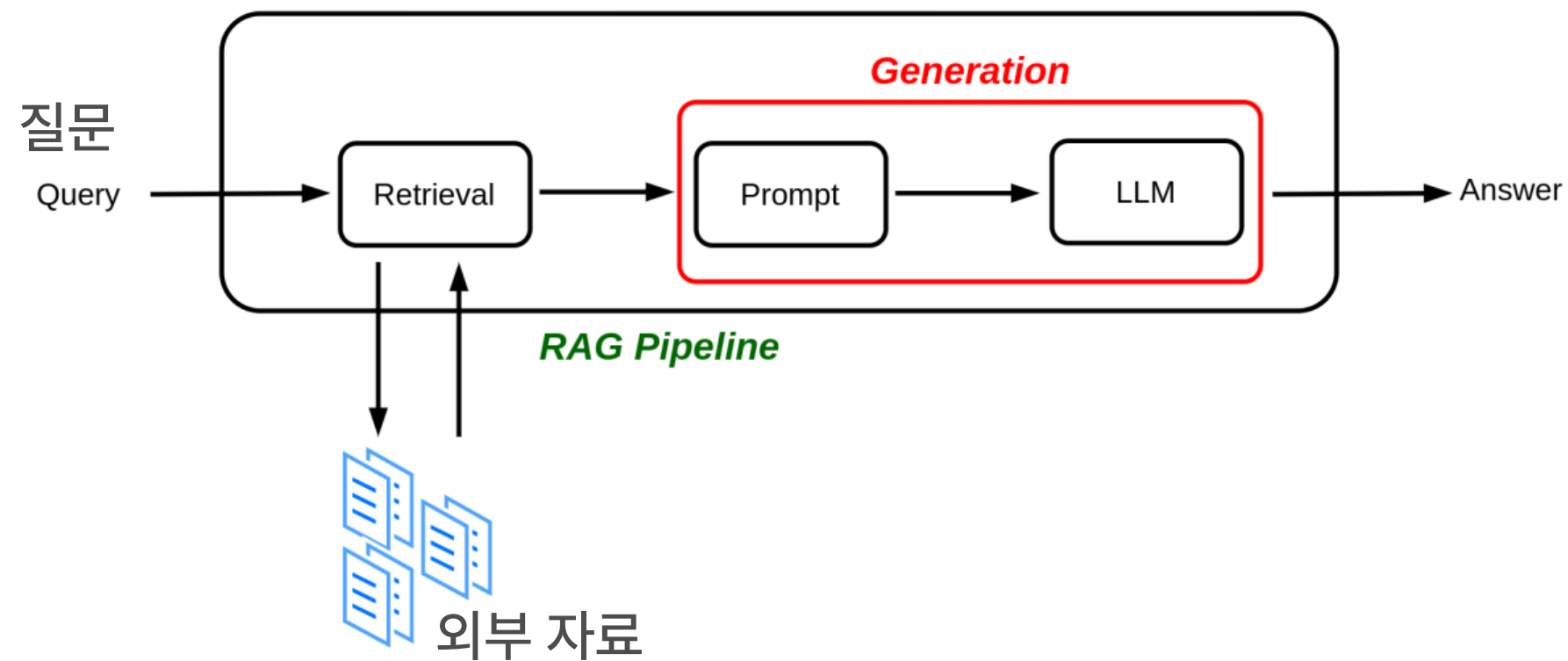


생성

LLM이 자료를 참고하여 답변 생성



# RAG

**[System/지침]**

- 너의 임무: 아래 '근거'만을 사용해 질문에 답하라.
- 근거에 없으면 "근거 없음"이라고 답하라.
- 가능한 한 간단히, 불확실한 부분은 추측하지 말라.
- 답의 마지막에 사용한 근거들의 식별자(ID)를 나열하라.

**[Context/근거]**

- (doc\_12) ...청크 내용...
- (doc\_27) ...청크 내용...
- (doc\_33) ...청크 내용...

**[User Question/질문]**

- RAG가 왜 필요한가?

**[Output Format/출력형식]**

- 3줄 요약 + 한 줄 결론
- 사용한 근거: (doc\_12, doc\_27)



# 03

## 데모

---

RAG를 체험해보자

```
1  from sklearn.feature_extraction.text import TfidfVectorizer # 문장을 숫자 벡터(TF-IDF)로 변환해줄
2  from sklearn.metrics.pairwise import cosine_similarity # 두 벡터가 얼마나 비슷한지 (코사인 유사도) 계산
3
4  # 1. 문서 준비 (지식 베이스)
5  docs = [
6      "Git은 분산 버전 관리 시스템이다.",
7      "Huffman 코딩은 압축 알고리즘의 한 종류다.",
8      "RAG는 Retrieval과 Generation을 합친 구조다.",
9      "CPU 캐시는 L1, L2, L3 단계로 나뉘며 속도를 높인다."
10 ]
11
12 # 2. 벡터화 (문서들을 숫자 벡터로 변환)
13 vectorizer = TfidfVectorizer()
14 X = vectorizer.fit_transform(docs)
15
16 # 3. 여러 질문 테스트
17 queries = [
18     "RAG가 뭐야?",
19     "압축 알고리즘에는 뭐가 있어?",
20     "Git은 어떤 시스템이야?",
21     "CPU 캐시가 뭐야?"
22 ]
```

```
24 for query in queries:
25     # 4. 질문 벡터화
26     q_vec = vectorizer.transform([query])
27
28     # 5. 유사도 계산 & 가장 비슷한 문서 찾기
29     similarities = cosine_similarity(q_vec, X)
30     best_idx = similarities.argmax()
31     best_doc = docs[best_idx]
32
33     # 6. 최종 답변 생성 (검색 + 생성 흉내내기)
34     print("="*50)
35     print("질문:", query)
36     print("검색된 문서:", best_doc)
37     print("최종 답변:", f"{best_doc} (이 문서를 참고했어요)")
```

=====

질문: RAG가 뭐야?

검색된 문서: Git은 분산 버전 관리 시스템이다.

최종 답변: Git은 분산 버전 관리 시스템이다. (이 문서를 참고했어요)

=====

질문: 압축 알고리즘에는 뭐가 있어?

검색된 문서: Huffman 코딩은 압축 알고리즘의 한 종류다.

최종 답변: Huffman 코딩은 압축 알고리즘의 한 종류다. (이 문서를 참고했어요)

=====

질문: Git은 어떤 시스템이야?

검색된 문서: Git은 분산 버전 관리 시스템이다.

최종 답변: Git은 분산 버전 관리 시스템이다. (이 문서를 참고했어요)

=====

질문: CPU 캐시가 뭐야?

검색된 문서: CPU 캐시는 L1, L2, L3 단계로 나뉘며 속도를 높인다.

최종 답변: CPU 캐시는 L1, L2, L3 단계로 나뉘며 속도를 높인다. (이 문서를 참고했어요)

PS C:\Users\CAU\Downloads\seo> □





QnA