

산출물

II. 테스트 계획 및 결과 보고서

1. 테스트 목적

본 테스트는 RAG 기반 검색 품질을 극대화하기 위한 일련의 실험 과정으로 구성되었다. 초기 단계에서는 크롤링된 논문 초록에 대해 NLTK 기반 전처리 후 TF-IDF 및 KeyBERT를 활용하여 주요 태그를 추출하고, 사용자 질문에 포함된 태그와 매칭하는 방식으로 태그 기반 필터링 전략의 효과를 검증하고자 하였다.

그러나 태그 기반 접근법은 문서 간 주제 다양성, 태그 중복, 과소·과다 추출 등 구조적 한계로 인해 검색 의존도가 낮고 품질 편차가 발생하였다. 이를 보완하기 위해 문서 임베딩을 기반으로 주제적 구조를 자동 분석하는 Auto-Clustering 기법을 적용하여 문서를 총 17개 의미 클러스터로 그룹화함으로써, 검색 시 노이즈를 줄이고 주제별 문서 집중도를 향상시키고자 하였다.

또한 검색 정확도 향상 및 RAG 응답 품질 개선을 위해 다양한 Chunking 조건(크기·overlap)과 Embedding 모델을 비교 평가하여 최적의 임베딩 기반 검색 구조를 도출하는 실험을 수행하였다.

최종적으로 본 테스트는 Chunking, Embedding, Clustering, Prompt 설계, LangChain 기반 Retrieval 파이프라인, LangGraph 의사결정(edge routing) 구조가 RAG 시스템의 검색 품질, 응답 정확성, 문맥 관련성에 미치는 영향을 종합적으로 검증하여, RAG 시스템의 최적 구성을 확립하는 것을 주요 목적으로 한다.

2. 태그 기반 검색 개선 및 Auto-Clustering 적용

2.1 태그 기반 검색을 위한 과정

- 본 단계에서는 사용자의 질문에 포함된 주요 토픽을 기반으로 문서를 선별하기 위해, 논문 초록을 crawling 하는 과정에서 텍스트 전처리 및 태그 추출 실험을 진행.
- step1: 전처리 NLTK 기반 기본 정규화
 - 소문자 변환, 불용어 제거, 토큰화 적용
 - 불용어 리스트 및 토큰 기준은 NLTK 기본 사전 기반 및 논문 도메인 특화 불용어 약 100개 추가
- step2-1: TF-IDF를 사용하여 keyword 추출 (태그 추출)
 - 전 문서 corpus 단위에서 TF-IDF 계산
 - 각 문서당 TF-IDF 상위 3개 를 태그로 선정
 - 특징: 빈도 기반으로 단어의 중요도를 산출하는 방식
- step2-2: KeyBERT 기반 의미적 keyword 추출 (태그 추출)
 - 문서 임베딩 기반 의미 유사도를 활용하여 핵심 단어·구 추출
 - TF-IDF 대비 더 “의미적으로 대표성 있는” 키워드를 얻기 위한 목적
 - 각 문서당 상위 3개의 semantic keyword 를 태그로 저장

2.2 태그 기반 검색을 적용한 결과

- 실험 결과, 태그 기반 접근법은 다음과 같은 이유로 검색 품질 향상에 실패함
 - 문서 주제가 다양하고 복합적이라 상위 태그 몇 개로 대표성이 부족함
 - TF-IDF는 단순 빈도 기반이라 중요 단어가 과소추출됨
 - 태그가 질문과 태그가 매칭되지 않아 검색 결과가 제대로 수행되지 않는 문제가 다수 발생

ToolOrchestra: Elevating Intelligence via Efficient Model and Tool Orchestration 이 논문 찾아줘



❌ 관련 없음 죄송합니다. "ToolOrchestra: Elevating Intelligence via Efficient Model and Tool Orchestration 이 논문 찾아줘"는 AI/ML 연구 논문과 관련이 없는 질문입니다.

이 시스템은 다음과 같은 질문에 답변할 수 있습니다:

- AI/ML 모델 및 아키텍처 (GPT, BERT, SAM, Diffusion 등)
- AI/ML 개념 및 기술 (Transformer, Attention, Fine-tuning 등)
- AI/ML 도구 및 라이브러리 (LangChain, Transformers, PyTorch 등)
- 최근 AI/ML 연구 동향 및 논문

AI/ML 관련 질문으로 다시 시도해주세요! 🙏

- 과도한 태그 다양성으로 인해 관련 문서를 제대로 찾지 못하는 문제 해결
- 문서의 주제 기반 의미적 그룹화 → 검색 효율 향상
- RAG 시스템에서의 semantic drift 감소
- 이러한 실험 결과를 통해 태그 기반 검색 전략의 한계를 확인하였고, 이를 보완하기 위해 문서 임베딩을 활용한 Clustering 을 적용
(세부 태그 분포 및 분석 결과는 1. 수집된 데이터 및 데이터 전처리 결과서를 참조)

- 본 시스템은 사용자의 질문이 입력되면 우선적으로 retriever의 검색 결과를 기반으로 문서 적합성(Relevance)을 평가하고, 그 결과가 *MEDIUM*으로 분류되는 경우 추가적인 Cluster Similarity Check 단계를 수행
- Auto-Clustering을 통해 생성된 cluster_id 및 클러스터 중심 정보가 활용됨.
- *MEDIUM* 등급 문서는 개별 문서 수준에서의 매칭 신뢰도는 다소 낮지만, 동일 클러스터 내 인접 문서의 집단적 의미 정보를 반영하여 재평가가 가능해짐
- 동일 클러스터 내에서 추가 검색을 수행하여 관련도가 높은 문서가 보완적으로 반영되고, 최종적으로 사용자 질의에 대해 더 높은 품질의 문맥(Context)을 구성할 수 있게 되었음



📌 Beyond Fact Retrieval: Episodic Memory for RAG with Generative Semantic Workspaces
Shreyas Rajesh · 2025 · [HuggingFace](#) —

3. Chunking 및 Embedding 모델 평가

3.1 Chunking 및 Embedding 평가 조건

- chunking 조건
 - chunking_size : 100, 200, 300
 - chunking_overlap : chunk_size의 0%, 15% 비율
 - 하기 사이트를 참조하여 일반적인 overlap 기준(10~25%)의 약 중간값과 화학논문에서 overlap 0% 일때 가장 좋았다는 논문 결과가 있어 선택하였음
 - 참조 사이트: [\[Tech Series\] kt cloud AI 검색 증강 생성\(RAG\) #3 : 청킹\(Chunking\) 전략과 최적화](#)
- Embedding 모델
 - HuggingFace 모델:
 - ① all-MiniLM-L6-v2 (384 dim / 경량 고속 범용 모델)
 - ② all-mpnet-base-v2 (768 dim / 고품질)
 - ③ msmarco-MiniLM-L-6-v3 (384 dim / 검색 최적화)
 - ④ allenai-specter (768 dim / 과학 논문 특화)
 - ⑤ bge-m3 (1024 dim / 다국어 지원, 높은 성능)
 - ⑥ paraphrase-multilingual-mpnet-base-v2 (768 dim/ 다국어 의역 검색 최적화)
 - OpenAI 모델:
 - ⑦ text-embedding-3-small (1536 dim / 고성능·대규모 의미 임베딩)

3.2 Chunking 및 Embedding 성능 평가 결과

- 평가 요소
 - Hit Rate@K : 상위 K개 결과 중 최소 1개의 관련 문서 포함 비율 (0~1)
 - MRR(Mean Reciprocal Rank) : 첫 번째 관련 문서의 역순위 평균
 - NDCG@K (Normalized Discounted Cumulative Gain) : 순위를 고려한 검색 품질 (0~1)
 - Avg Time : 쿼리당 평균 검색 시간 (초) → 평가에 배제. (GPU, CPU에 따라 성능차이 有)
- 평가 결과 (순위별) :
 - ① paraphrase-multilingual-mpnet-base-v2 / Chunk=200, overlap=30
 - ② text-embedding-3-small / Chunk=300, overlap=45
 - ③ paraphrase-multilingual-mpnet-base-v2 / Chunk=300, overlap=0
- 성능 평가 결과

청크 전략: 100_0

모델	HR@10	NDCG@10	MRR
Paraphrase-Multi	1.000	0.738	0.847
MsMarco	1.000	0.721	0.911
MiniLM-L6	0.929	0.703	0.893
MPNet	1.000	0.701	0.771
OpenAI-small	1.000	0.685	0.849
SPECTER	1.000	0.641	0.738
BGE-M3	0.929	0.633	0.788

청크 전략: 100_15

모델	HR@10	NDCG@10	MRR
Paraphrase-Multi	1.000	0.740	0.883
MiniLM-L6	0.929	0.737	0.857
MPNet	1.000	0.720	0.774
MsMarco	1.000	0.702	0.907
OpenAI-small	1.000	0.687	0.854
BGE-M3	1.000	0.678	0.746
SPECTER	1.000	0.646	0.719

청크 전략: 200_0

모델	HR@10	NDCG@10	MRR
Paraphrase-Multi	1.000	0.754	0.800
MiniLM-L6	1.000	0.729	0.807
OpenAI-small	1.000	0.729	0.839
MPNet	1.000	0.699	0.768
SPECTER	0.929	0.657	0.750
MsMarco	1.000	0.634	0.747
BGE-M3	1.000	0.603	0.801

청크 전략: 200_30

모델	HR@10	NDCG@10	MRR
Paraphrase-Multi	1.000	0.780	0.857
MPNet	1.000	0.750	0.857
OpenAI-small	1.000	0.730	0.839
MiniLM-L6	1.000	0.725	0.794
SPECTER	0.929	0.661	0.764
BGE-M3	1.000	0.660	0.821
MsMarco	0.929	0.658	0.792

청크 전략: 300_0

모델	HR@10	NDCG@10	MRR
Paraphrase-Multi	1.000	0.765	0.810
OpenAI-small	1.000	0.762	0.821
MPNet	1.000	0.721	0.798
MiniLM-L6	1.000	0.712	0.804
SPECTER	0.929	0.683	0.768
MsMarco	1.000	0.674	0.740
BGE-M3	1.000	0.642	0.792

청크 전략: 300_45

모델	HR@10	NDCG@10	MRR
OpenAI-small	1.000	0.778	0.857
Paraphrase-Multi	1.000	0.761	0.845
MiniLM-L6	1.000	0.745	0.881
MPNet	1.000	0.739	0.833
SPECTER	1.000	0.713	0.890
BGE-M3	1.000	0.691	0.815
MsMarco	1.000	0.670	0.780

전체 평가 결과

모델	청크 전략	HR@5	HR@10	MRR	NDCG@5	NDCG@10	Avg Time(s)
Paraphrase-Multi	200_30	1.000	1.000	0.857	0.783	0.780	0.08
OpenAI-small	300_45	1.000	1.000	0.857	0.813	0.778	0.54
Paraphrase-Multi	300_0	1.000	1.000	0.810	0.754	0.765	0.04
OpenAI-small	300_0	1.000	1.000	0.821	0.771	0.762	0.41
Paraphrase-Multi	300_45	1.000	1.000	0.845	0.754	0.761	0.04
Paraphrase-Multi	200_0	1.000	1.000	0.800	0.724	0.754	0.07
MPNet	200_30	1.000	1.000	0.857	0.715	0.750	0.08
MiniLM-L6	300_45	1.000	1.000	0.881	0.754	0.745	0.02
Paraphrase-Multi	100_15	0.929	1.000	0.883	0.762	0.740	0.10
MPNet	300_45	1.000	1.000	0.833	0.763	0.739	0.04

4. Prompt 설계

4.1 Langchain에서 Prompt

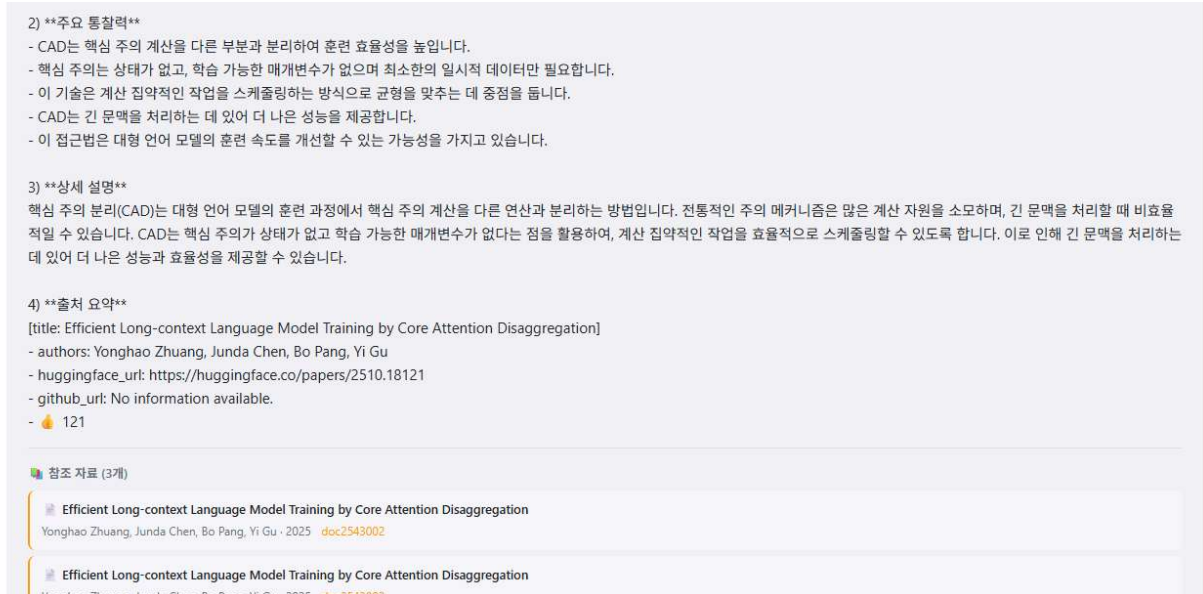
1) Prompt 설계

- 역할(Role)
 - 모델은 AI Tech Trend Navigator로서 최신 AI 논문 요약, 여러 논문 비교·종합, 실제 서비스/제품에 적용 가능성 설명을 담당하도록 정의됨.
- 입력규칙(Input Handling)
 - 모델은 user_question, retrieved context, 일반적인 AI/ML/DL/LLM 지식 요소만 사용하도록 제한됨
 - 문맥에 없는 논문/저자/수치 창작 및 웹검색 정보로 논문을 대체하는 행위를 금지함
- 문맥처리(context Handling)
 - 문맥에 논문이 존재할 때는 1~3개의 가장 관련성이 높은 논문만 선택하고 논문 기반으로 답변을 구성
 - “NO_RELEVANT_PAPERS”일 때는 논문 언급을 금지하고, DuckDuckGo 검색 결과의 URL만 사용
- 답변 출력 형식(Output Format)
 - 논문 기반일 때:
 - ① One-line summary, ② Key insights, ③ Related papers(1~3개), ④ Detailed explanation, ⑤ Sources summary (metadata 기반으로 정리)
 - 웹검색 기반일 때:
 - ① One-line summary, ② Detailed explanation, ③ Source: DuckDuckGo (URL 2개)
- 금지 규칙
 - 논문·저자·수치·결과 절대 창작 금지, 웹결과를 논문처럼 취급 금지, 무관한 논문 억지로 연결 금지

2) Langchain에서 Prompt 설계의 한계

- 역할 과부하로 인한 재현성(Robustness) 저하
 - 너무 많은 기능(질문 의도 파악, 문서 선택, 요약, 비교, 형식 구성)을 하나의 프롬프트가 수행하도록 구성됨.
 - 동일한 입력이라도 출력이 일정하지 않고 출력 변동성(variance)이 크게 나타남.
- 자연어 조건 기반 제어의 한계
 - 프롬프트에서 “관련 논문 상위 1~3개 사용”, “메타데이터 기반으로만 작성”, “context 없으면 웹 검색” 등의 절차적 조건을 자연어로 명시했지만, LLM은 이를 100% 확실하게 준수하지 못함.
 - 실제 출력 예시:
 - “상위 3개의 관련 논문을 활용하라” → 실제로는 2개만 사용하거나 4개가 섞임.
 - “메타데이터 없으면 'No Information'이라고 표기” → 가끔 누락하거나 임의로 생성.
 - 즉, 프롬프트만으로 논리 흐름을 강제하는 것은 신뢰성이 낮음.

- role의 분리가 되지 않아 유지보수가 어려움
 - 하나의 프롬프트가 모든 단계(질문 이해 → 검색 판단 → 요약 → 출력 형식)를 담당하니, 일부 기능만 수정하고 싶어도 전체 프롬프트를 뜯어고쳐야 하는 상황 발생.
 - 프롬프트의 길이가 길어질수록 오류 발견 및 수정이 매우 어려워짐.
- 출력형식과 참조자료가 중복 출력됨



4.2 LangGraph에서 Prompt 1 (초기)

1) prompt 설계

- 번역 prompt
 - 역할 : AI/ML/DL/LLM 전문가로써 번역
 - 입력규칙 : AI/ML/DL/LLM 에 전문적인 용어를 유지
 - 답변출력 : 영어로 출력
- 최종 답변 prompt
 [사용자의 질문이 관련 주제인 경우]
 - 역할 : AI/ML/DL/LLM 연구논문 전문가
 - 입력규칙 : 사용자 질문에 한글로 답변
 - 답변출력 :
 [문서가 있는 경우] 핵심요약, 주요 인사이트, 상세설명
 [문서가 없는 경우] 웹검색결과의 경우 출처를 웹으로 명시
 [사용자의 질문이 관련 주제가 아닌 경우]
 - 답변출력:
 질문에 대한 적절한 문서를 찾지 못했다는 출력과 함께 AI/ML/DL/LLM 주제로 질문 요청
- 출처:
 - 출처는 prompt 에서 답변 출력이 아닌 (langchain prompt), 카드형식(코드로 설계)으로 출력되게 하였음

2) LangGraph에서 Prompt 1 설계의 한계

- 번역 오류 존재
 - AI/ML/DL/LLM 전문가 역할을 지정한 결과,
예를 들어, “해리포터 줄거리를 설명해줘” 라고 사용자가 입력하였을 때,
“please explain the AI/ML/DL/LLM” 으로 임의로 번역을 변경함으로써 정확한
답변이 출력되지 않음
- 최종 답변에서 역할 오류
 - “관련성이 낮으면” 이라고 기재한 문구에 대해서 정확한 기준을 제공하지 않아서
답변출력의 재현성이 낮음
 - 소문자로 사용자가 질문하거나 한글로 기술용어를 질문하는 경우, 최종 답변으로
관련없는 질문으로 출력됨
예를 들어, “랭그래프에 대해서 설명해줘” 또는 “rag에 대해서 설명해줘”에 대한
답변이 출력되지 않음

4.3 Langgraph에서 Prompt 2 (최종 확정버전)

1) prompt 설계

가) AI 관련성 판별 prompt

- 역할: 사용자의 질문이 AI 분야와 관련되어 있는지 판단하는 이진 분류기
- 입력규칙(Input Handling)
 - [YES를 반환하는 기준]
 - 실제 존재하는 AI 관련 모델(GPT, BERT, ...)이나 프레임워크와
도구(PyTorch, TensorFlow, ...) 및 AI 관련 플랫폼, 개념, 아키텍처에 대한
내용이 질문에 들어있을 때
 - [NO를 반환하는 기준]
 - AI를 응용하지 않는 수학, 통계, 과학, 비즈니스, 금융 법률을 포함한 AI 분야와
관련 없는 질문을 받았을 때
 - [애매할 경우 판단 기준]
 - AI를 위한 데이터 분석이나 신경망을 위한 수학은 YES를 답변
 - 일반 데이터 분석이나 일반 미적분학은 NO를 답변
- 답변 판단에 있어 중요한 부분
 - 질문에 AI 관련 용어나 도구가 직접적으로 언급되면 YES로 판단
- 답변 출력 형식(Output Format): “YES” 또는 “NO”만 출력

나) 번역 prompt

- 역할: 사용자의 질의를 번역
- 입력규칙
 - AI, ML, 데이터, 모델명, 프레임워크, 학술 영어 등 정확한 기술 용어로 정규화
 - 기술적 맥락, 전문용어를 그대로 유지하며 불필요한 의역을 하지않음
 - 문장이 너무 길 경우 핵심 의미만 유지한 compact English query로 요약

- 알수없는 약어나 기술용어도 맥락상 AI/ML/DL/LLM 관련이면 그대로 유지
- 출력은 오직 영어로만 수행
- 금지규칙
 - 주제가 무엇이든 상관없이 “번역”만 수행
 - 내용의 적정성을 판단해서 임의의 번역을 수행하지 말것

다) 최종답변 prompt

- 상기 prompt 1과 동일하게 진행

3) LangGraph에서 Prompt 2 설계 결과

- 번역 오류 해결
 - 사용자의 질문과 정확하게 번역 수행
- 최종 답변 오류 해결
 - 관련성 판단에 대해서 상기 분류 prompt를 사전에 수행하고, langgraph 코드로 관련성을 판단하는 것을 추가 기재함으로써 답변의 정확도를 높임
 - 소문자 및 한글로 기술용어 질문하는 경우에도 정확한 답변을 출력함

오디오 기반 애니메이션의 캐릭터 정체성을 유지하는 모델이 있나요?

1.1

🔍 해커브러드 검색

1) 오디오 기반 애니메이션에서 캐릭터 정체성을 유지하는 'Lookahead Anchoring' 모델이 개발되었습니다.

2) 주요 통찰:

- 미래의 키프레임을 동적 가이드로 사용하여 애니메이션 품질을 향상시킴.
- 입술 동기화와 정체성 보존에서 우수한 성능을 보여줌.
- 다양한 아키텍처에서 시간적 조건 개선을 달성함.

3) 'Lookahead Anchoring'은 오디오 기반 인간 애니메이션의 품질을 높이기 위해 미래의 키프레임을 활용하는 방법입니다. 이 모델은 캐릭터의 입술 동기화와 정체성을 잘 유지하면서 시각적 품질을 향상시킵니다. 특히, 여러 애니메이션 모델에 적용했을 때, 정체성 유지와 시각적 품질에서 뛰어난 성과를 보였습니다. 이 기술은 게임이나 영화 제작에서 캐릭터의 일관된 표현을 필요로 하는 다양한 분야에 활용될 수 있습니다.

📄 참조 자료 (3개)

Lookahead Anchoring: Preserving Character Identity in Audio-Driven Human Animation

2025 HuggingFace →

Game-TARS: Pretrained Foundation Models for Scalable Generalist Multimodal Game Agents

Zhao Wang, Haoming Wang, Jiazhan Feng, Zhongkai Zhao - 2025 HuggingFace →

SteadyDancer: Harmonized and Coherent Human Image Animation with First-Frame Preservation

Jiaming Zhang - 2025 HuggingFace → GitHub →

해리포터 줄거리 알려주세요

1.2

❌ 관련 없음 죄송합니다. "해리포터 줄거리 알려주세요"는 AI/ML 연구 논문과 관련이 없는 질문입니다.

이 시스템은 다음과 같은 질문에 답변할 수 있습니다:

- AI/ML 모델 및 아키텍처 (GPT, BERT, SAM, Diffusion 등)
- AI/ML 개념 및 기술 (Transformer, Attention, Fine-tuning 등)
- AI/ML 도구 및 라이브러리 (LangChain, Transformers, PyTorch 등)
- 최근 AI/ML 연구 동향 및 논문

AI/ML 관련 질문으로 다시 시도해주세요! 📄

해리포터 언어 모델에 대해서 설명해줘

1.3

🔍 해커브러드 검색

1) 해리포터 언어 모델에 대한 구체적인 정보는 제공되지 않았습니다.

2) 주요 통찰:

- 언어 모델은 자연어 생성 능력이 뛰어나지만, 사실과 다른 정보를 생성하는 경향이 있습니다.
- ORION이라는 프레임워크가 언어 모델이 효율적으로 추론하도록 가르치는 방법을 제안합니다.
- 언어 모델은 언어를 모델링하는 데 한계가 있으며, 더 나은 성능을 위해 다른 접근 방식이 필요합니다.

3) 제공된 문서는 언어 모델의 한계와 개선 방법에 대해 논의하고 있습니다. 언어 모델은 자연어를 잘 생성하지만, 때때로 사실과 다른 정보를 만들어내는 '환각' 현상이 발생합니다. 이를 해결하기 위해 ORION이라는 새로운 프레임워크가 제안되었으며, 이는 언어 모델이 더 효율적으로 사고할 수 있도록 돕는 방법을 제공합니다. 또한, 언어 모델이 단순한 언어를 모델링하는 데 그치지 않고, 더 깊은 이해와 추론 능력을 갖추도록 하는 것이 중요하다는 점도 강조됩니다.

📄 참조 자료 (3개)

Stemming Hallucination in Language Models Using a Licensing Oracle

Simeon Emaniulov - 2025 HuggingFace →

ORION: Teaching Language Models to Reason Efficiently in the Language of Thought

Krisi Aggarwal - 2025 HuggingFace →

Language Models Model Language

Lutzack Buchmann - 2025 HuggingFace →

[LangChain에 대해서 설명해주세요](#)

LangChain

1) LangChain은 대형 언어 모델(LLM)을 활용한 애플리케이션 개발을 위한 오픈소스 프레임워크입니다.

2) 주요 특징:

- LangChain은 LLM과 외부 데이터 소스를 연결하여 자연어 처리(NLP) 애플리케이션을 쉽게 개발할 수 있도록 돕습니다.
- 개발자는 프롬프트 템플릿을 사용하여 LLM과의 상호작용을 구조화하고 더 정확한 응답을 얻을 수 있습니다.
- 챗봇, 가상 비서, 질문-응답 시스템 등 다양한 생성 AI 애플리케이션을 구축할 수 있는 기반을 제공합니다.

3) LangChain은 개발자들이 대형 언어 모델을 활용하여 복잡한 애플리케이션을 쉽게 만들 수 있도록 돕는 프레임워크입니다. 이 프레임워크는 OpenAI의 GPT-3.5 및 GPT-4와 같은 강력한 LLM을 외부 데이터 소스와 연결하여 자연어 처리 애플리케이션을 개발할 수 있게 해줍니다. LangChain은 프롬프트 템플릿 모듈을 제공하여 개발자가 LLM에 대한 구조화된 프롬프트를 만들 수 있도록 지원하며, 이를 통해 더 빠르고 정확한 상호작용과 정확한 응답을 이끌어낼 수 있습니다.

또한, LangChain은 복잡한 워크플로우를 구축하고 다양한 구성 요소를 통합할 수 있는 도구를 제공하여, 챗봇, RAG(검색-보장 생성) 기반의 어시스턴트, 디퓨즈드 에이전트 시스템 등 다양한 AI 프로젝트를 쉽게 시작할 수 있는 기반을 마련합니다.

참조 자료 (3개)

- What Is LangChain: Components, Benefits & How to Get Started
[링크 보기](#)
- What Is LangChain and How to Use It: A Guide - TechTarget
[링크 보기](#)
- LangChain Explained: The Ultimate Framework for Building LLM ...
[링크 보기](#)

5. LangChain 설계

5.1 LangChain 설계

- 상기 prompt 1 (초기 설계)를 적용하여 기본적인 prompt | llm | StrOutputParser()로 설계하였음
- 내부 문서가 아닌 경우에는 DuckDuckGo 웹서치를 통해 답변을 출력하도록 함

5.2 LangChain 설계의 한계

- 사용자의 질문에 따라 내부 문서를 서치하는 과정에서 코드로 분류를 하지 않고 prompt에 많은 역할을 부여하여 진행함으로써 정확도 및 재현성이 현저히 낮아짐
- 내부 문서를 통한 답변이 아닌 웹서치를 통해 답변하는 경우가 대다수로 확인됨
- 이러한 한계점으로 LangGraph를 설계하여 보완하고자 하였음

simple-rag-system-cc95c4cf
SKN20-...0d251
Compact Full Diff Default
Group by
Display
+ Compare

	Inputs	Reference Outputs	Outputs	Latency	Completed
1.	ToolOrchestra: Elevating I... #359d →	title : ToolOrchestra: Elevating Intellige...	1) 간단한 요약 ToolOrchestra는 다양한 지능형 도구를 효율적으로 조정하여 복잡한 작업을 해결하는 방법을 제시합니다. ...	11.66s	COMPLETED 1
2.	LLM에서 캐시와 관련된 논문... #72c6 →	title: Cache-to-Cache: Direct Semantic...	1) LLM에서 캐시와 관련된 논문은 없습니다. 2) 현재 제공된 논문들 중에서 LLM의 캐시와 직접적으로 관련된 내용을 다른 ...	2.90s	COMPLETED 1
3.	LLM에서 긴 문맥의 추론을 향... #8112 →	title: Beyond Fact Retrieval: Episodic M...	1) GSW(Generative Semantic Workspace)는 LLM의 긴 문맥 추론을 향상시키는 새로운 접근법입니다. 2) - GSW는 ...	6.07s	COMPLETED 1
4.	LLM에서 환각탐지를 할 수 있... #88f6 →	title: When Models Lie, We Learn: Multi...	1) LLM에서 환각 탐지를 위한 데이터셋에 대한 정보입니다. 2) 최근 연구에서는 LLM의 환각 탐지를 위한 다양한 방법과 ...	6.23s	COMPLETED 1
5.	core attention disaggrega... #8dcb →	title : Efficient Long-context Language ...	1) 핵심 주의 분산(core attention disaggregation, CAD)은 긴 문맥의 대형 언어 모델 훈련을 개선하는 기술입니다. 2) ...	6.52s	COMPLETED 1
6.	Adrian Kosowski 저자의 최... #9214 →	title: The Dragon Hatchling: The Missin...	NO_RELEVANT_PAPERS [WebResult 1] title: Adrian Kosowski - Google Scholar url: https://scholar.google...	6.17s	COMPLETED 1
7.	GUI-360: A Comprehensi... #9f83 →	title: GUI-360: A Comprehensive Datas...	1) GUI-360은 컴퓨터 사용 에이전트를 위한 대규모 데이터셋 및 벤치마크입니다. 2) - GUI-360은 실제 작업에서의 데이...	6.16s	COMPLETED 1
8.	RFT를 LVLMS (large video... #a7ad →	title: VIDEO2R: Video Understanding f...	1) RFT를 대형 비디오 언어 모델(LVLMs)로 확장하는 것은 도전적입니다. 2) - RFT를 LVLMs에 적용하는 것은 여러 기술...	12.45s	COMPLETED 1
9.	해리포터 줄거리 알려주세요 #cde5e →	해당없다	NO_RELEVANT_PAPERS [WebResult 1] title: 해리포터 줄거리 url: https://example.com/harry-potter-summar...	5.70s	COMPLETED 1
10.	최근 공개된 논문에서 좋아요... #eafa →		NO_RELEVANT_PAPERS [WebResult 1] title: Recent Advances in AI and Machine Learning url: https://exa...	3.13s	COMPLETED 1
11.	오디오 기반 애니메이션의 정... #fd3c →	title: https://huggingface.co/papers/25...	1) 오디오 기반 애니메이션에서 캐릭터 정체성을 유지하는 방법에 대한 연구입니다. 2) - Lookahead Anchoring 기법을 ...	8.02s	COMPLETED 1

6. LangGraph 설계

6.1 초기 LangGraph 설계

- prompt : 상기 LangGraph prompt 1 적용
- state:
 - question: 검색에 사용될 질문 (영어 번역된 질문 또는 원본)
 - original_question: 사용자가 입력한 원본 질문
 - translated_question: 영어로 번역된 질문 (한글인 경우)
(사용자가 질문을 영어로 했다면 사용하지 않음)
 - is_korean: 원본 질문이 한글인지 여부
 - documents: 검색된 문서 리스트
 - doc_scores: 각 문서의 유사도 점수 (L2 distance)
 - cluster_id: 검색된 문서의 클러스터 ID
(질문이 클러스터와 관련 없을 경우 사용하지 않음)
 - cluster_similarity_score: 클러스터 내 평균 유사도 점수
(질문이 클러스터와 관련 없을 경우 사용하지 않음)
 - search_type: 검색 유형 ("internal", "cluster", "web", "rejected")
 - relevance_level: 문서 관련성 수준 ("high", "medium", "low")
 - answer: LLM이 생성한 최종 답변
 - sources: 참조 문서 정보 리스트
- Node:
 - translate: 사용자가 한글 질문을 한 경우 영어로 번역
 - retrieve: VectorStore에서 질문과 유사한 문서를 검색 상위 5개의 문서와 유사도 점수 및 최상위 문서의 클러스터 정보를 반환
 - evaluate: 검색된 문서의 유사도 점수를 기반으로 관련성을 HIGH, MEDIUM, LOW로 분류
 - cluster_check: 검색된 최상위 문서와 같은 클러스터에 속한 추가 문서들을 검색하여 클러스터 평균 유사도를 HIGH 또는 LOW로 반환
 - web_search: DuckDuckGo를 사용한 웹 검색
 - generate: 최종 답변 생성
 - reject: 답변 거부
- Edge 구성:


```

graph TD
    START --> translate
    translate --> retrieve
    retrieve --> evaluate
    evaluate -- HIGH --> generate
    evaluate -- MEDIUM --> cluster_check
    cluster_check --> generate
    generate --> web_search
    web_search --> generate
    generate --> END
    evaluate -- LOW --> reject
    reject --> END
      
```

6.2 초기 LangGraph 설계의 한계

- 단순 벡터 기반 `similarity_search` 를 진행하였으나, 문장이메딩만으로는 정확한 의미 유사도 판단이 어려움
 - 의미가 유사해도 표현 방식이 다른 경우 검색에 실패함
 - 질문이 짧거나, 선언형 형태일 경우 벡터 모델이 문맥을 제대로 잡지 못함
- 상기 LangGraph의 prompt 1 의 한계 참조

6.3 최종 LangGraph 설계

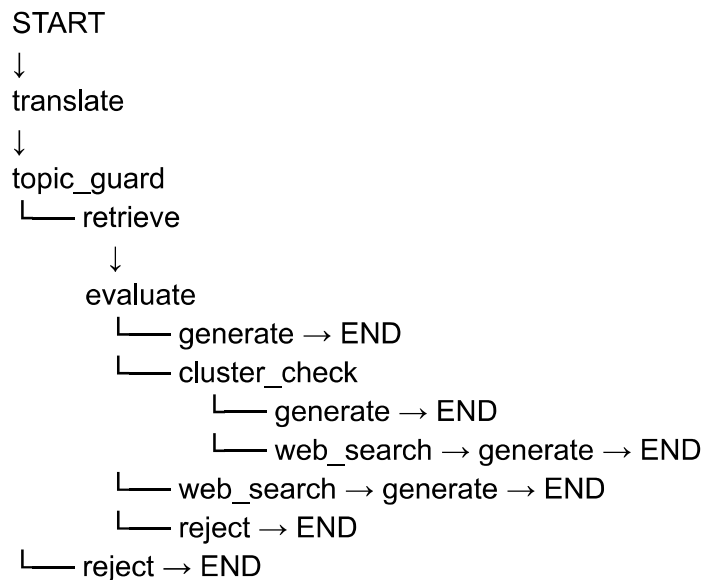
- 상기 한계를 극복하기 위해 하기 사항을 수행
 - 가) 하이브리드 검색 추가
 - `retrieve_node`에서는 Vector 검색과 BM25 검색 결과를 RRF 방식으로 합산하고 vector와 BM25의 가중치를 조절 (벡터: 1.5 가중, BM25: 0.5 가중)
 - `retrieve_node`에서는 (1) BM25 미사용 시에는 벡터 거리(score)를 부스팅 반영하여 조정하고, (2) 하이브리드(RRF) 사용 시에는 fusion score에 부스팅을 가산하여 최종 랭킹을 보정

나) 메타데이터 부스팅(Boosting) 로직 도입

- 코드에서 `calculate_metadata_boost()`는 질의 키워드(`extract_keywords`)를 기준으로 다음 항목을 검사해 점수를 가산함.
 - Title 매칭: +0.05, keywords(태그/키워드) 매칭: +0.02, doc_id 매칭: +0.01
- state:
 - question: 검색에 사용될 질문 (영어 번역된 질문 또는 원본)
 - original_question: 사용자가 입력한 원본 질문
 - translated_question*: 영어로 번역된 질문 (한글인 경우)
(사용자가 질문을 영어로 했다면 사용하지 않음)
 - is_korean: 원본 질문이 한글인지 여부
 - documents: 검색된 문서 리스트
 - doc_scores: 각 문서의 유사도 점수 (L2 distance)
 - cluster_id: 검색된 문서의 클러스터 ID
(질문이 클러스터와 관련 없을 경우 사용하지 않음)
 - cluster_similarity_score: 클러스터 내 평균 유사도 점수
(질문이 클러스터와 관련 없을 경우 사용하지 않음)
 - search_type: 검색 유형 ("internal", "cluster", "web", "rejected")
 - relevance_level: 문서 관련성 수준 ("high", "medium", "low")
 - answer: LLM이 생성한 최종 답변
 - sources: 참조 문서 정보 리스트
 - is_ai_ml_related: 질문이 AI 주제와 관련성이 있는가
- Node:
 - translate: 사용자가 한글 질문을 한 경우 영어로 번역

- topic_guard: AI와 관련된 키워드가 질문에 들어있는지 확인
- retrieve: VectorStore 유사도 검색 점수와 BM25 점수를 3:1 비율로 합산하고 AI와 관련된 내용이 title, keywords, doc_id에 있으면 부분점수를 부여
- evaluate: 검색된 문서를 평가. 메타데이터와 매칭되는 문서가 있을 경우 관련성 HIGH 반환하고 없으면 점수를 기반으로 high, medium, low를 반환
- cluster_check: medium으로 평가되었을 때 질문의 키워드가 클러스터 안에 있으면 관련 문서를 더 찾아 관련 문서 추가
- web_search: medium으로 평가되었을 때 질문 키워드가 클러스터에 없으면 웹 서치로 이동
- generate: 최종 답변 생성
- reject: 답변 거부

- Edge 구성:

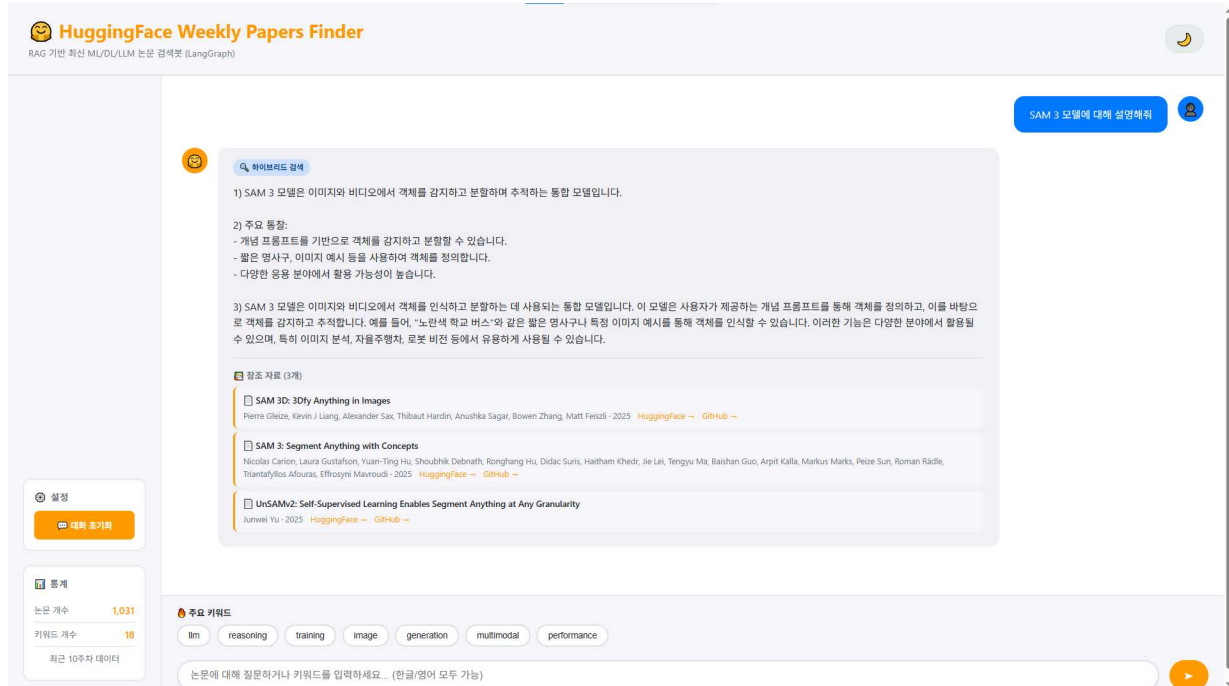


6.4 최종 LangGraph 설계 결과

- BM25 도입 및 가중치 부여
 - BM25 도입 및 vectorstore 유사도 1:3으로 설정하여 정확한 의미 + 키워드의 균형잡힌 검색 가능
 - 상기 설정을 통해서 기존에 medium이 아닌 low로 분류되어(score가 낮음) 웹서치로 분류되었던 사용자의 질문이 medium score를 만족하여 cluster_check 로 수행되면서 내부검색의 정확도가 증가됨
- AI 관련 키워드 추가
 - AI 관련 키워드가 나오면 평가 부분에서 추가 점수를 부여(score - boost * 2.0)하여 평가 노드에서 문서에 없는 AI 관련 키워드는 낮은 점수를 받아 답변을 거부하는 경우를 방지함
 - 테스트에서 특정 고유명사/모델명 질의(예: “SAM3”, “LLaMA” 등)에 대해 제목·키워드·doc_id 매칭을 통한 부스팅이 Top-K 문서의 순위를 안정화하여 응답 품질이 향상되는 것을 확인.

7. UI 구현

- html 로 Langgraph를 구현



8. 최종 LangGraph 테스트

8.1 LangGraph 테스트 목적

- 실제 챗봇 사용 환경에서 LangGraph의 라우팅 조건이 RAG의 응답 품질에 어떤 영향을 주는지 직관적으로 검증하고자 함
- 확인 사항
 - 관련 문서가 존재하는 경우 → 올바른 라우팅을 통해 정확한 답변을 반환하는가?
 - 관련 문서가 전혀 없거나 모호한 질문인 경우 → Web Search 또는 Reject로 잘 분기하는가?
 - 라우팅 구조가 전체 RAG의 안정성과 품질 향상에 기여하는가?

8.2 LangGraph 테스트 범위

- 내부 검색(Vector / Cluster) 라우팅 동작 상태 확인
- Web 검색 fallback 조건 검증
- Reject 노드 동작 검증
- 라우팅 결과에 따른 응답 자연스러움·관련성·오류 여부 평가

- 사용자 입력 종류에 따른 라우팅 적합성 판단

8.3 LangGraph 테스트 시나리오

- 약 11개 질문을 직접 수행하여 예상하는 답변이 나오는지 확인
 - CASE1. 관련 문서가 분명히 있는 입력
ex) “LLM에서 캐시와 관련된 논문이 있나요”
→ Vector/Cluster에서 정확한 논문을 선정하여 답변하는지
 - CASE2. 관련 문서는 있지만 모호하거나 길게 표현된 입력
ex) “LLM에서 환각 탐지를 할 수 있는 모델을 알려주세요”
→ Cluster or Hybrid 라우팅 후 답변하는지
 - CASE3. 내부 DB에 전혀 없는 정보
ex) “langgraph란 무엇인가요?”
→ Web Search 작동 여부 확인
 - CASE4. 무의미한 입력 / 답변 불가능한 입력
ex) “해리포터 모델이 뭔가요?”
→ Reject 노드 동작 여부 확인

8.4 LangGraph 평가 기준

1. 관련성(Relevance)

- 답변이 질문의 의도에 부합하는가?
- 실제 논문 내용과 크게 벗어나지 않는가?

2. 라우팅 적절성(Routing Validity)

- 관련 문서가 있을 때 내부 검색으로 잘 들어갔는가?
- 없을 때 Web으로 잘 넘어갔는가?
- 불가능한 경우 Reject 처리가 되었는가?

3. 응답 신뢰성(Answer Quality)

- 환각(Hallucination) 발생 여부
- 사실과 크게 벗어난 답변 여부
- 문장 구조·논리의 자연스러움

8.5 LangGraph 테스트 절차

- 각 질문을 챗봇에 입력하여 라우팅 경로 확인 : Hybrid / Cluster / Web / Reject
- 각 질문에 대해 다음을 기록
 - 라우팅이 적절했는가?

- 답변이 관련성이 있었는가?
- 환각 또는 오답이 없는가?
- 라우팅 유형별 품질 비교

8.6 LangGraph 테스트 결과 테이블

CASE	예상 라우팅	실제 라우팅	라우팅 적절성	답변 관련성	비고
1	Hybrid	Hybrid	적절	높음	정상 동작
2	Cluster	Cluster	적절	높음	정상 동작
3	Web	Web	적절	높음	정상 동작
4	Reject	Reject	적절	-	정상 동작

● Case 1

LLM에서 캐시와 관련된 논문이 있나요?

🔍 하이라이트 검색

1) LLM에서 캐시를 활용한 두 가지 논문이 있습니다: Elastic-Cache와 Cache-to-Cache입니다.

2) 주요 통찰:

- Elastic-Cache는 확산 언어 모델을 위한 적응형 KV 캐시 전략을 제안하여 속도를 크게 향상시킵니다.
- Cache-to-Cache는 LLM 간의 직접적인 의미적 통신을 가능하게 하여 정확성을 높이고 지연 시간을 줄입니다.
- 두 연구 모두 캐시를 효과적인 모델 간 통신 수단으로 활용합니다.

3) 첫 번째 논문인 "Attention is All You Need for KV Cache in Diffusion LLMs"에서는 Elastic-Cache라는 새로운 KV 캐시 전략을 소개합니다. 이 전략은 모델의 각 레이어에 맞춰 조정되며, 생성 품질을 유지하면서도 속도를 크게 향상시킬 수 있습니다. 이는 확산 언어 모델의 실용성을 높이는 데 기여합니다.

두 번째 논문인 "Cache-to-Cache: Direct Semantic Communication Between Large Language Models"에서는 LLM 간의 텍스트 기반 통신을 넘어서는 새로운 통신 패러다임인 Cache-to-Cache를 탐구합니다. 이 방법은 신경망 프로젝션을 사용하여 LLM 간의 직접적인 의미적 통신을 가능하게 하며, 이를 통해 정확성을 개선하고 지연 시간을 줄일 수 있습니다. 이 연구는 다양한 LLM 시스템이 서로의 강점을 활용할 수 있는 방법을 제시합니다.

📚 참조 자료 (3개)

- Attention Is All You Need for KV Cache in Diffusion LLMs
Quen Nguyen-Tr, Mukul Rangan - 2025 [HuggingFace](#) →
- Cache-to-Cache: Direct Semantic Communication Between Large Language Models
Tianyu Fu, Zhan Min, Hanling Zhang, Ichao Yan - 2025 [HuggingFace](#) → [GitHub](#) →
- From Code Foundation Models to Agents and Applications: A Practical Guide to Code Intelligence
Jian Yang, Jiajun Wu, Wei Zhang, Terry Yue Zhuo - 2025 [HuggingFace](#) →

● Case 2

LLM에서 긴 문맥의 추론을 향상시키는 GSW (Generative Semantic Workspace)에 대한 논문이 있다면 소개시켜주세요.

🔍 클러스터 검색

1) GSW(Generative Semantic Workspace)는 LLM의 긴 문맥 추론 능력을 향상시키는 신경 영감을 받은 메모리 프레임워크입니다.

2) 주요 통찰:

- GSW는 구조화된 해석 가능한 상향 표현을 생성하여 LLM의 추론 능력을 개선합니다.
- 기존 방법보다 에피소드 메모리 벤치마크에서 더 나은 성능을 보입니다.
- 추론 시간을 단축시켜 효율성을 높입니다.

3) GSW는 LLM이 긴 문맥을 이해하고 추론하는 데 필요한 구조화된 메모리 시스템을 제공합니다. 이는 LLM이 변화하는 역할, 행동 및 시간적 맥락을 고려하여 더 나은 결정을 내릴 수 있도록 돕습니다. 기존의 LLM은 긴 문서에서 정보를 처리하는 데 한계가 있었으나, GSW는 이러한 문제를 해결하여 더 복잡한 상황에서도 효과적으로 작동할 수 있게 합니다. 이 기술은 고객 서비스, 교육, 콘텐츠 생성 등 다양한 분야에서 활용될 수 있습니다.

📚 참조 자료 (1개)

- Beyond Fact Retrieval: Episodic Memory for RAG with Generative Semantic Workspaces
Shreyas Rajesh - 2025 [HuggingFace](#) →

● Case 3



● Case 4



8.7 LangGraph 테스트 최종 결론

- 본 테스트는 실제 사용자 관점에서의 RAG 라우팅 품질 검증을 목표로 수행되었음
- 결과 요약:
 - 내부 검색(Vector/Cluster)은 관련 문서 존재 시 안정적으로 동작함
 - 최신 정보에 대해서는 Web Search fallback이 효과적으로 작동함
 - 무의미 입력에 대한 Reject 처리가 적절히 수행됨
 - 전체적으로 LangGraph 라우팅 구조는 RAG 시스템의 정확성과 안전성 향상에 기여함

9. 결론

9.1 HuggingFace Weekly Papers Finder 챗봇 개발 결론

RAG 기반 AI 논문 요약·검색 시스템의 검색 품질 향상과 라우팅 안정성 확보를 위해 수행된 일련의 실험과 개선 과정을 종합적으로 평가하였다.

초기 단계에서는 TF-IDF 및 KeyBERT 기반 태그 추출, 단일 벡터 검색 기반 문서 매칭, 단일 Prompt 의존 방식 등 비교적 단순한 전략을 채택하였다.

그러나 이러한 방식은 **검색 성능 편차, 태그 대표성 부족, 벡터 기반 유사도 한계, 프롬프트 재현성 저하** 등의 구조적 한계를 드러냈다.

이를 보완하기 위해 본 프로젝트는 다음과 같은 주요 개선을 단계적으로 적용하였다.

1. Auto-Clustering 기반 의미 구조 도입

문서 임베딩을 기반으로 한 자동 클러스터링을 적용하여 문서 주제 구조를 정교하게 모델링함으로써, 검색 시 노이즈 감소와 주제 집중도 향상을 달성하였다.

2. Chunking 및 Embedding 최적화 실험 수행

다양한 Chunk size, overlap, Embedding 모델 조합을 평가하여 최적의 검색 품질을 제공하는 조합(예: Chunk 300 / overlap 45 / text-embedding-3-small)을 도출하였다.

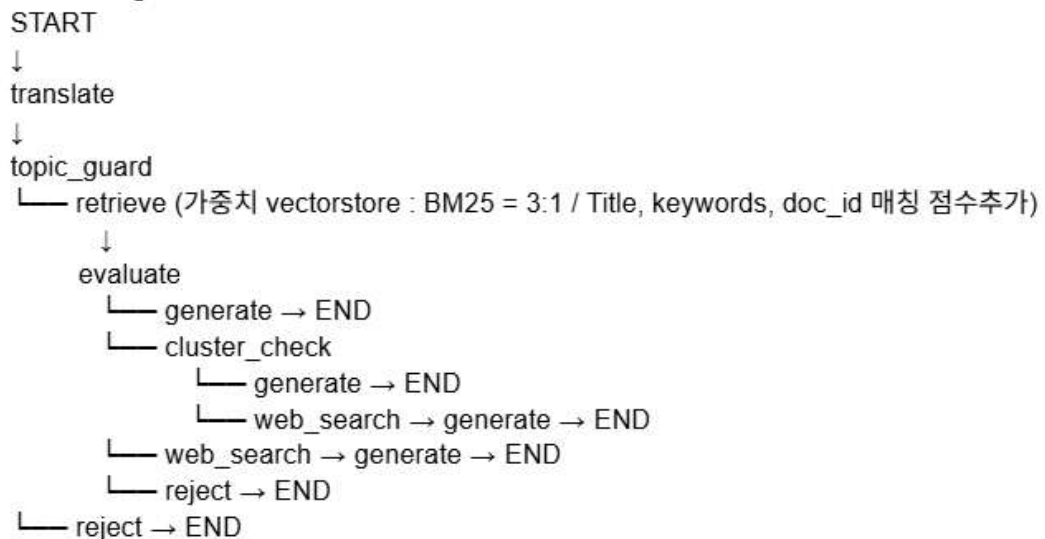
3. Prompt 구조의 단계적 재설계

프롬프트가 수행하던 과도한 역할을 분리하여 “AI주제 여부 판단, 번역, 최종답변 생성”으로 나눔으로써 **출력의 재현성, 정확도, 유지보수성**을 크게 향상시켰다.

4. LangChain 단일 파이프라인의 한계를 극복한 LangGraph 라우팅 구조 도입

단일 프롬프트 및 단일 체인의 한계를 해결하기 위해 LangGraph 기반 의사결정 라우팅을 적용하였다.

이에 따라 검색 흐름이 하기와 같이 단계적 구조로 분기되며, 검색 품질이 크게 향상되었다.



최종 테스트 결과는 다음과 같다.

- 문서 존재 시 → 내부 검색(Hybrid/Cluster)이 안정적으로 작동
- 문서 미존재 시 → Web Search Fallback이 정확히 수행
- AI 비관련·무의미 입력에 대해 → Reject가 올바르게 작동
- 전체적으로 Hallucination이 감소하고 응답 논리·품질이 크게 향상됨

본 테스트를 통해, RAG 시스템의 성능은 단순한 벡터 검색이나 단일 프롬프트 개선만으로는 충분히 향상되기 어렵다는 사실이 확인되었다.

다양한 검색 신호(Vector + BM25 + Metadata + Cluster), 프롬프트 역할 분리, LangGraph 기반 라우팅 구조의 결합이 필수적임이 검증되었다.

최종적으로 본 프로젝트는 하기와 같은 주요 성과를 달성하였다.

- 검색 정확도와 안정성 향상
- 모호한 질의 처리 능력 강화
- 웹 최신 정보 활용 구조 확립
- 비관련 입력에 대한 안전성 강화
- 프롬프트 재현성과 유지보수성 대폭 개선

따라서 본 보고서에서 확립된 **LangGraph + Hybrid Retrieval + Structured Prompting 기반 RAG 아키텍처**는 향후 확장성과 실서비스 적용 가능성이 매우 높으며, 추후 데이터 규모 확장·도메인 확장·모델 고도화 등 다양한 개선 작업에도 견고한 기반을 제공할 것으로 판단된다.

9.2 개선 사항

① 문서 최신성(Recency) 유지의 어려움

현재 벡터DB는 초기 구축 이후 정적(Static)으로 유지되고 있으며, 새로운 논문 또는 HuggingFace DailyPapers 데이터가 업데이트되더라도 자동 반영되지 않음

- 문제점
 - 문서가 업데이트될 때마다 수동으로 VectorStore를 재구축해야 함.
 - 최신 AI 논문 트렌드를 반영하지 못해 검색 정확도 및 활용성이 제한됨.
- 개선 방향

- 데이터 소스(논문 API, RSS, WeeklyPapers 등)가 갱신될 때마다 주기적 동기화 파이프라인을 구성하여 벡터DB를 자동 업데이트.
- 증분 업데이트(incremental update) 방식 도입하여 전체 Embedding 재생성 비용 절감.

② 논문 제목·저자 기반 검색 기능의 구조적 미흡

현재 시스템은 질문을 벡터 기반 검색으로만 처리하므로, 사용자가 “저자 이름으로 검색” 과 같은 명확한 정형 검색(intention-based retrieval) 을 요청할 때 최적화된 결과를 제공하지 못함

● 문제점

- 제목 그대로 입력해도 의미 벡터화 과정에서 표현이 왜곡됨.
- 특정 저자명을 검색할 때 정확한 저자 Matching 기능 부재.

● 개선 방향

- LangGraph 단계에 정형검색 Intent Detection 노드 추가
(author 검색 / 일반 질의 검색 구분)
- 저자 기준 검색 시:
저자가 작성한 논문을 **최신순(top-3)** 으로 정렬하여 제공하는 로직 추가

③ Auto-Clustering의 품질 및 알고리즘 개선 필요

현재 문서 클러스터링은 선행적으로 계산된 **tech_terms**를 하드코딩하여 사용하고 있음.
이는 Auto-Clustering의 “의미 기반 군집화 성능”을 충분히 반영하지 못함.

● 문제점

- cluster_id가 고정되어 있어 신규 문서 추가 시 군집 재학습이 불가.
- 문서 의미 변화(주제 변화)를 반영하지 못해 cluster-check 정확도가 제한됨.

● 개선 방향

- KMeans, HDBSCAN, Spectral Clustering 등 다양한 클러스터링 실험을 진행하여 최적 군집 수 및 알고리즘 검증.
- 클러스터 중심(centroid) 업데이트 자동화.
- 향후 신규 문서를 포함하여 Online Clustering 또는 Incremental Clustering 방식으로 확장.