

**SK네트웍스 Family AI과정 14기  
 모델링 및 평가 테스트 계획 및 결과 보고서**



**□ 개요**

* 산출물 단계 : 모델링 및 평가
* 평가 산출물 : 테스트 계획 및 결과 보고서
* 제출 일자 : 2025.09.29
* 깃허브 경로 : https://github.com/sungjaecho98/SKN14-Final-3Team
* 작성 팀원 : 조성재, 이수미, 김의령

1. **sLLM Fine-tuning 평가**

| **개요** | * 프로젝트 목표**:** KB국민은행 내부 데이터(예: 상품 매뉴얼, 규정, FAQ 등)와 금융데이터를 기반으로, 행원의 업무 질문에 대해 정확하고 신속하게 답변을 제공하는 소형 언어 모델(sLLM) 개발. 이를 통해 행원 업무 효율성 증대 및 교육 시간 단축을 목표로 함. * **모델 목표: 기반 모델의 범용 한국어 능력을 유지하면서 금융 도메인 전문성을 극대화하는 것** * 개발 모델**:** Alibaba Cloud의 Qwen2.5-7B-instruct 모델을 Full Fine-tuning하여 KB국민은행 특화 모델 개발 * 핵심 결과:   + Fine-tuning 하기 전 base 모델 보다 금융 벤치마크 및 한국어 벤치마크 우수.   + ArenaHard 벤치마크를 통해 Human preference 성능 향상 측정 * 결론 **:** 본 프로젝트를 통해 개발된 KB국민은행 특화 소형 언어모델(sLLM)은 금융 특화 벤치마크 및 한국어 벤치마크에서 기존 base 모델 대비 우수한 성능을 보임. 또한 정성 평가 결과 실제 사용 시나리오에서 90% 이상의 만족도를 확보하여, 행원 업무 지원에 실질적인 효과를 제공할 수 있음을 입증하였음..   따라서 KB국민은행 내부 서비스에 적용할 경우 **업무 효율성 증대 및 교육 시간 단축**에 기여할 수 있는 높은 실용성을 가진 것으로 평가됨.. |
| --- | --- |
| **모델링 개발 계획** | * 정량적 목표:   + Fine-tuning 하기 전 base 모델보다 금융 벤치마크 성적 상승   + Base 모델보다 한국어 벤치마크 성적 상승 혹은 유지 * 정성적 목표:   + 별도의 추가 검색 없이 답변을 신뢰하고 업무에 활용할 수 있는 수준.   + 복잡하고 여러조건이 포함된 질문에 대해서도 논리적인 답변 생성. * 개발 환경   + H/W: GPU 서버 (Runpod, NVIDIA H100 SXM x 1)   + S/W**:** Python 3.11.11, PyTorch 2.1, Hugging Face Transformers, TRL 등   + 모델: Qwen2.5-7B-instruct     - 선정 사유: 한국어 성능이 우수하고, 7B 수준의 모델중 상위권의 추론(inference) 능력을 보유. 상업적으로도 이용가능한 라이선스. |
| **학습 데이터** | * **Supervised Fine-Tuning (SFT) 데이터:** * 데이터 소스:   자체 생성 데이터 (KB국민은행 상품 설명서, 내부 업무 규정, 행원 교육 자료, FAQ 등)  허깅페이스 오픈소스 데이터(aiqwe/FinShibainu)   * 데이터 전처리:   aiqwe/FinShibainu QA 데이터셋: 금융 지식 관련 QA 데이터.   * + - 각 질의응답 쌍을 user와 assistant 역할의 대화 형식으로 변환     - 이후 모델 학습에 적합한 최종 대화 문자열 형식으로 만듦.     - 결과:   aiqwe/FinShibainu mcqa 데이터셋: 금융 지식 관련 MCQA 데이터 셋   * + - 객관식 질문에 답만 대답하는 형식으로 변환..     - 결과:   KB국민은행 상품 설명서, 내부 업무 규정, 행원 교육 자료등 데이터:   * + - PDF 데이터셋을 Plumber parser로 파싱후 Batch API를 통해 QA 데이터셋을 생성     - 결과: * 최종적으로 총 QA 금융데이터 15,000건, MCQA 금융데이터 15,000건 분할 하여 총 30,000 건의 질의응답 쌍 데이터셋 생성하고 자체 생성한 KB 특화 데이터 1,157건을 합치고 섞음 * 데이터셋을 train 29,599건, test 1,558건으로 분할 하여 hugging face에 업로드   최종 데이터셋 구조:   * + - 데이터셋 리포지토리: rucipheryn/combined-dataset-30K-final-v4 * **DPO 데이터:**   + 데이터 소스:     - 허깅페이스 오픈소스 데이터(aiqwe/FinShibainu) 금융 DPO셋     - 허깅페이스 오픈소스 데이터(javirandor/hh-rlhf-safety-v3-dpo) safety DPO셋   + 데이터 특성:     - 질문, 선호하는 형식의 답변 , 선호하지 않는 형식의 답변     - 예시인 aiqwe/FinShibainu의 데이터의 anwer\_A를 거절, answer\_B를 선호   + 데이터 전처리:     - Batch api를 통해 GPT-4o-mini로 영어 safety DPO 데이터셋인 javirandor/hh-rlhf-safety-v3-dpo을 번역     - 결과:      * + - aiqwe/FinShibainu데이터의 prompt, chosen, rejected칼럼으로 매핑하고 추출하여 DPO 데이터셋 생성     - 이후 위의 번역된 safety 데이터와 8:2 비율로 10,000개 합성 후 허깅페이스에 업로드     - 데이터 셋 리포지토리: sssssungjae/dpo\_shiba\_safety1     - 결과: |
| **모델 파인튜닝(SFT, DPO)** | * **Supervised Fine-Tuning (SFT):**  본 프로젝트에서는 Hugging Face의 trl 라이브러리를 활용하여 Supervised Fine-Tuning (SFT)을 수행하였음.   또한, 메모리 사용량 최적화 및 학습 속도 향상을 위해 unsloth 라이브러리를 적용하였음.  특히, Full Fine-tuning 방식을 채택하여, 모델의 모든 파라미터를 업데이트함으로써 데이터에 모델을 깊게 특화 시킴.  주요 하이퍼파라미터 (Hyperparameters):  *# 학습 하이퍼파라미터*  *TRAINING\_EPOCHS = 2*  *BATCH\_SIZE = 8*  *GRADIENT\_ACCUMULATION 4*  *LEARNING\_RATE = 1e-5*  *OPTIMIZER = "adamw\_torch\_fused"*  *max\_seq\_length=4096*  *warmup\_ratio=0.1*  *weight\_decay=0.01,*  *lr\_scheduler\_type="cosine"*  **결과 및 주요 지표 그래프:**   * **Direct Preference Optimization(DPO):** Supervised Fine-Tuning(SFT)이 완료된 모델을 기반으로, 자체 생성한 Direct Preference Optimization(DPO) 데이터셋을 활용하여 DPO 학습을 수행하였음.   DPO의 경우에는 SFT보다 많은 메모리가 소모 되므로 하나의 GPU에서 학습을 진행하기 위해 PEFT 기법을 사용함.  학습 과정에서는 메모리 절약 및 학습 효율성 향상을 위해 unsloth 라이브러리를 적용하였으며, Hugging Face의 DPOTrainer를 이용하여 학습을 진행함.  이를 통해 모델이 단순히 정답을 맞추는 수준을 넘어, 사용자가 선호하는 말투와 응답 형식을 학습하여 대화 품질을 향상시키고자함.  주요 하이퍼파라미터 (Hyperparameters):  *# 학습 하이퍼파라미터*  *TRAINING\_EPOCHS = 2*  *BATCH\_SIZE = 8*  *GRADIENT\_ACCUMULATION 4*  *LEARNING\_RATE = 1e-6*  *OPTIMIZER = "adamw\_torch\_fused"*  *max\_seq\_length=4096*  *warmup\_ratio=0.1*  *weight\_decay=0.01*  *lr\_scheduler\_type="cosine"*  *beta=0.1*  *lora\_alpha=16*  *lora\_dropout=16*  **결과 및 주요 지표 그래프:** |
| **평가 테스트** | * 정성 평가: 일상대화와 금융관련 질문을 통해 정성평가   + LM Studio를 이용한 일상적인 답변 평가 * 정량 평가: SFT가 완료된 모델을 한국어 관련 벤치마크와 금융 벤치마크로 모델 정량적 평가   **SFT (Supervised Fine-Tuning) 완료 후 성능**   |  | ArenaHard | HAERAE | Kobest | HRM8k\_GSM8k | HRM8k\_MATH | | --- | --- | --- | --- | --- | --- | | Qwen/Qwen2.5-7B-instruct (베이스 모델) | 72 | 0.5 | 0.6 | 0.81 | 0.62 | | \*sssssungjae/qwen2\_5-7b-instruct-finance-full-final-15\_15 (SFT 완료 모델) | 69.8 | 0.57 | 0.61 | 0.79 | 0.61 |   **SFT 모델에 DPO (Direct Preference Optimization) 완료 후 성능**   |  | ArenaHard | HAERAE | Kobest | HRM8k\_GSM8k | HRM8k\_MATH | | --- | --- | --- | --- | --- | --- | | sssssungjae/qwen2\_5-7b-instruct-finance-full-final-15\_15 (SFT 완료 모델) | 69.8 | 0.57 | 0.61 | 0.79 | 0.61 | | \*sssssungjae/qwen2.5-dpo-shi2  (DPO 완료 모델) | 71.8 | 0.57 | 0.6 | 0.79 | 0.62 |  * **금융 성능 평가 지표**   + - **금융 벤치마크로 allganize/financial-mmlu-ko를 lm-eval-harness에 추가하여 평가함.**  1. Qwen/Qwen2.5-7B-instruct (베이스 모델):0.7099 2. sssssungjae/qwen2\_5-7b-instruct-finance-full-final-15\_15(SFT 완료 모델): 0.7451 3. \*sssssungjae/qwen2.5-dpo-shi2 (DPO 완료 모델): 0.7495 |
| **결론 및 계획** | 결론: 본 프로젝트에서는 **Qwen2.5-7B-Instruct** 모델을 기반으로 KB국민은행 내부 데이터와 금융 특화 데이터를 활용해 **Full Fine-tuning**을 수행하였음.  **본 프로젝트의 모델 목표는 "기반 모델의 범용 한국어 능력을 유지하면서 금융 도메인 전문성을 극대화하는 것"이었음. 이를 검증하기 위해 SFT와 DPO 각 단계별로 금융 특화, 한국어 능력, 범용 추론 능력 세 가지 측면에서 성능을 측정했으며, 결과는 아래와 같음.**   | **평가지표** | **베이스 모델 (Qwen2.5-7B-instruct)** | **SFT 완료 모델** | **DPO 완료 모델 (최종)** | | --- | --- | --- | --- | | 금융 (financial-mmlu-ko) | 0.7099 | 0.7451 | **0.7495** | | 한국어 (HAERAE) | 0.50 | 0.57 | **0.57** | | 한국어 (Kobest) | 0.60 | 0.61 | 0.60 | | 인간선호 (ArenaHard) | **72.0** | 69.8 | **71.8** |  * **성능 분석**  1. **SFT 단계: 금융 전문 지식의 성공적 주입**   **SFT 튜닝 결과, 금융 벤치마크(financial-mmlu-ko) 점수가 0.7099에서 0.7451로 대폭 상승했습니다. 이는 모델이 금융 관련 데이터셋을 성공적으로 학습하여 해당 도메인에 대한 깊은 지식을 갖게 되었음을 의미함.**  **동시에 HAERAE, Kobest 등 한국어 능력 지표도 함께 상승하여, 학습 과정에서 한국어 이해 및 생성 능력이 저하되지 않고 오히려 강화되었음을 확인함.**  **다만, 전문화 과정에서 범용 추론 능력(ArenaHard)이 소폭 하락하는 일반적인 트레이드오프가 관찰됨.**   1. **DPO 단계: 범용 성능 회복 및 균형 달성**   **DPO 튜닝을 통해 SFT 단계에서 하락했던 범용 추론 능력(ArenaHard)이 71.8점으로 베이스 모델 수준까지 회복됨.**  **이는 모델의 답변 스타일이 인간의 선호도에 맞게 교정되면서, 전문 지식뿐만 아니라 일반적인 질문에도 더 적절하고 유용한 답변을 생성하게 되었음을 시사함.**  **가장 중요한 점은, 이러한 범용성 회복 과정에서 금융 지식(0.7495)과 한국어 능력(0.57)은 손실 없이 안정적으로 유지되었다는 것임.**   * **결론**   + **최종 개발된 sssssungjae/qwen2.5-dpo-shi2 모델은 기반 모델의 한국어 능력을 저하시키지 않으면서, 목표했던 금융 도메인 지식을 성공적으로 학습했음을 객관적인 지표를 통해 입증하였음.**   + **SFT를 통해 금융 전문가로 만들고, DPO를 통해 다시 똑똑한 제너럴리스트로 균형을 맞추는 2단계 파인튜닝 전략은 매우 효과적이었음.**   + **본 모델은 금융 분야의 실무적인 질의응답 태스크에서 높은 활용 가치를 가질 것으로 기대됨.** * **향후 계획 및 개선 방향** * **이번 DPO 학습을 통해 모델의 응답 품질이 크게 개선되었지만, ArenaHard 점수가 베이스 모델의 72점에는 소폭 미치지 못하는 71.8점을 기록한 점은 향후 개선의 여지를 남깁니다. 이는 현재 DPO 데이터셋이 모델의 잠재력을 최대한 끌어내기에는 아직 부족한 부분이 있음을 시사함.** * **따라서 다음과 같은 방향으로 추후에 DPO 데이터를 고도화하여 모델 성능을 향상시킬 계획.** * **데이터 품질 고도화: 현재의 DPO 데이터셋을 분석하여, 'chosen'과 'rejected' 답변 간의 차이가 미묘하고 까다로운 예시들을 추가 예정. 이를 통해 모델이 단순히 맞고 틀림을 넘어, 더 정교하고 세련된 답변 스타일을 학습하도록 유도.** * **금융 도메인 특화: 복잡한 금융 시나리오나 여러 상품을 비교 분석해야 하는 고난이도 질문에 대한 선호도 쌍(Preference Pair)을 집중적으로 구축할 예정. 이를 통해 일반적인 유용성을 넘어, 금융 전문가 수준의 깊이 있는 답변을 생성하는 능력을 강화.** |

1. **Langgraph 평가**

| **개요** | * 프로젝트 목표: KB국민은행 내부 데이터(예: 상품 매뉴얼, 규정, FAQ 등)를 실시간으로 검색·활용할 수 있는 RAG 기반 LangGraph 시스템을 구축하여, 행원의 업무 질문에 대해 **정확하고 신속한 답변**을 제공하는 것을 목표로 함. 이를 통해 행원의 **업무 효율성 증대**와 **신규 행원의 교육** **시간 단축**을 도모함. * 개발 시스템   + LangGraph 기반 RAG 파이프라인 설계   + Pinecone 벡터DB에 내부 문서 임베딩 저장 후, 쿼리 의도(Intent) 분류 및 문서 검색을 통한 하위 플로우(상품, 규정, 법률) 라우팅 구현   + openai 모델을 실험용 LLM으로 활용하여 실시간 답변 생성 + 컨텍스트 근거 제공 * 핵심 결과   + 내부 데이터를 활용한 RAG 검색을 통해, 단순 LLM 호출 대비 금융 상품 문의 응답 정확도와 구체성 향상   + 정성 평가 결과, 내부 문서를 통해 구축된 시나리오에서 90% 이상의 ‘만족’ 응답 획득   + 근거 문서와 함께 답변 제공 → 답변 신뢰도 및 내부 준법 감수성 강화 * 결론 : 본 프로젝트를 통해 개발된 RAG 기반 LangGraph 시스템은 금융 업무 특화 질문에 대해 높은 성능과 신뢰성을 보였으며, 이를 바탕으로 추후 더 정교한 Re-ranking 기법 및 멀티에이전트 구조로 확장하여 서비스 적용을 계획함. |
| --- | --- |
| **모델링 개발 계획** | 정량적 목표질문과 관련된 정확한 PDF 문서(상품 설명서, 약관 등)를 참조하여 검색되는 비율을 96% 이상 달성참조된 문서와 비교했을 때, 생성된 답변 내 핵심 정보 유지 정성적 목표답변 내용이 실제 PDF 문서 근거와 일치하여 행원이 추가 검색 없이 업무에 활용 가능한 수준 달성여러 조건이 포함된 질문에서도 관련 문서를 정확히 찾아내고, 논리적으로 일관된 답변 생성 개발 환경  * **H/W**: CPU 서버 (11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz, RAM 16GB) * **S/W**: Python 3.11.11, LangChain, LangGraph, OpenAI API SDK, Pinecone * **모델**: OpenAI GPT-4o-mini 모델 * **선정 사유**:    + **모델 크기 측면 :** GPT-4o-mini는 수십억(7B~8B급)에 해당하는 sLLM 크기와 비슷한 성능 레벨을 갖고 있어,  Qwen2.5-7B를 실제로 구동하지 않고도 **비슷한 파라미터 규모의 환경**을 실험 가능.   + **API 기반 사용성 :** CPU 환경에서는 대형 LLM 로컬 추론이 어려운데, GPT-4o-mini는 **API 호출만으로 빠르게 테스트 가능** → LangGraph 파이프라인 구조 검증에 최적.   + **추론 속도와 비용 :** 작은 규모 모델이라 **API 응답 속도가 빠르고 비용도 낮아** 프로토타입용으로 적합.   + **보완할 점 :** GPT-4o-mini는 상업 서비스용 오픈소스 모델(sLLM)과는 다른 계열(OpenAI 독점 모델)이므로,  본 프로젝트에서는 “실험 환경에서 구조 검증용”으로 활용하였음을 명확히 함.  결과  * LangGraph 기반 RAG 파이프라인에서 **문서 참조 + 답변 생성** 과정 정상 작동 확인 * 특정 질문 시, 관련 PDF가 Pinecone에서 검색되고, 그 내용을 기반으로 LLM이 일관된 답변을 생성 * CPU 환경에서도 RAG 구조를 검증할 수 있었으며, **추후 GPU 환경 및 sLLM 파인튜닝으로 확장 가능성** 확인 |
| **활용 데이터** | * 데이터 소스:   **상품 정보**: 주택담보대출, 전세자금대출, 신용대출, 예·적금 상품 설명서 등  **KB 내부 규정집 및 지침**: 윤리강령, 반부패 정책, 개인정보보호 정책 등  **여신 고객 응대 매뉴얼**: 신규 행원 교육 자료, 고객 상담 시나리오   * 데이터 전처리 : 상품/규정/매뉴얼 데이터셋 * PDF 문서를 **pdfplumber**로 텍스트 추출 * **1000 토큰 단위**로 문서 chunking 수행하며, overlap은 100 토큰 단위로 허용함 * 임베딩 생성: openai embedding(text-embedding-3-small, 1536차원) * **Pinecone 인덱스**에 id + chunk + metadata 형태로 저장 * Metadata 정보는 아래와 같음    + Business\_unit : 부서 정보   + Contains\_application\_info   + Contains\_interest\_rate : 상품 정보 포함 여부   + Contains\_policy : 윤리 정보 포함 여부   + Content\_length : text 길이   + Document\_category: 상품/ 규정/ 법률file\_name   + File\_path   + Keywords : text에서 키워드 추출   + Main\_category   + Page\_number : 원본 문서 페이지 번호   + Product\_type : 상품 대분류   + Sub\_category : 대출 유형(Kor)   + Subcategory : 대출 유형(Eng)   + Target\_customer : 고객 유형(개인/기업)   + Text   + upload\_date    테스트 데이터셋  * 시스템 검증을 위해 별도의 테스트셋 설계 * 질문–정답–참조문서 구조로 120개 구성 |
| **Langgraph 구조** | **1) 개요**  LangGraph는 질문(query)을 입력받아 지능형 라우팅을 통해 적절한 처리 경로를 선택한다.   1. FAQ 경로: 단순한 상식/절차 문의는 빠르게 응답 2. RAG 경로: 상품·규정·법률 등 문서 근거가 필요한 경우, 상품명 추출 → 문서 검색 → 관련성 필터링 → PDF 근거 기반 답변 생성 3. 맥락 경로: 이전 대화 맥락을 활용한 연속 질문 처리    2) 노드 정의 및 역할  * **A. SESSION\_INT (세션 초기화)**   + 입력: query, session\_id   + 출력: session\_context, conversation\_history, is\_first\_turn   + 설명: 대화 세션 생성/조회, Django 연동으로 대화 히스토리 로드   + 특징: 첫 대화 여부 판단으로 후속 라우팅 결정 * **B. SUPERVISOR (중앙 관리자)**   + 입력: query, conversation\_history, session\_context   + 출력: selected\_tool, reasoning\_trace   + 설명: LLM 기반 지능형 라우팅으로 적절한 도구 선택   + 도구 선택 로직:     - 상품명 명시 → product\_extraction     - 이전 맥락 활용 가능 → context\_answer     - 일반 FAQ → answer     - 문서 검색 필요 → rag\_search      * **C-1. PRODUCT\_EXTRACTION (상품명 추출)**   + 입력: query   + 출력: extracted\_product\_name, product\_category, keywords   + 설명: 정규표현식 기반 상품명 추출 및 카테고리 분류   + 예시: "햇살론 대출" → "햇살론" 추출 * **C-2. PRODUCT\_SEARCH (상품별 검색)**   + 입력: extracted\_product\_name, query   + 출력: relevant\_documents, search\_metadata   + 설명: 상품별 정밀 검색 (파일명 매칭 → 키워드 검색 → 일반 검색)   + 최적화: 검색 결과 캐싱으로 성능 향상 * **C-3. RAG\_SEARCH (RAG 문서 검색)**   + 입력: query   + 출력: retrieved\_documents, context\_text, sources   + 설명: Pinecone 기반 벡터 유사도 검색   + 특징: 메타데이터 필터링으로 정확한 검색 * **C-4. CONTEXT\_ANSWER (맥락 기반 답변)**   + 입력: query, conversation\_history   + 출력: contextual\_response, reasoning   + 설명: 이전 대화 맥락을 활용한 연속 질문 처리   + 사용 시나리오: "나이 제한이 있나요?", "조건은 어떻게 되나요?" 등 * **D. SESSION\_SUMMARY (세션 제목 생성)**   + 입력: first\_turn\_query   + 출력: session\_title, session\_metadata   + 설명: 첫 대화 시 세션 제목 자동 생성   + 실행 조건: 첫 대화인 경우에만 실행 * **GUARDRAIL\_CHECK (가드레일 검사)**   + 입력: response, context   + 출력: validated\_response, quality\_score, compliance\_flags   + 설명: 응답 품질 검증 및 금융 규제 준수 확인   + 검증 항목: 완전성, 정확성, 용어 표준화 * **F. ANSWER (최종 답변)**   + 입력: query, context, sources   + 출력: final\_response, citations, confidence\_score   + 설명: 검색된 컨텍스트 기반 최종 답변 생성   + 특징: 출처 정보 포함, 응답 포맷팅   **3) 엣지(전이) 규칙 기본 흐름**  START → SESSION\_INIT → first\_turn\_router  ├── 첫 대화: SESSION\_SUMMARY → SUPERVISOR  └── 멀티턴: SUPERVISOR  **SUPERVISOR 라우팅**  SUPERVISOR → supervisor\_router  ├── product\_extraction → PRODUCT\_SEARCH → ANSWER  ├── rag\_search → ANSWER  ├── context\_answer → 조건부 라우팅  └── answer (직접) → ANSWER  **조건부 라우팅**  CONTEXT\_ANSWER → 조건 확인  ├── 맥락으로 충분: ANSWER  └── 추가 검색 필요: RAG\_SEARCH → ANSWER  **최종 처리**  ANSWER → GUARDRAIL\_CHECK → 최종 응답 |
| **평가 테스트** | 1) 정량 평가(Quantitative)  * Precision@3 = 0.133  상위 3개 중 관련 문서 비율이 13.3% → Top-3에 비관련 문서가 많음. * Recall@3 = 0.400  정답 문서의 40%가 Top-3 안에 회수됨 → 일부 정답 누락. * F1@3 = 0.200  P@3와 R@3의 조화 평균(낮음). * MRR = 0.400  첫 관련 문서가 평균적으로 약 2~3위에 위치(1/2.5 ≈ 0.4 가량의 체감). * NDCG@3 = 0.626  상위 3개 결과의 순서 품질을 보는 지표.   + DCG = Σ (relᵢ / log₂(i+1)), NDCG = DCG / IDCG로 0~1 범위.   + 0.626은 중간 이상 수준으로, Top-3 안에서 상대적으로 중요한 문서가 앞쪽에 어느 정도 정렬되었음을 의미.    2) 정성 평가(Qualitative)  * 방식: OpenAI 모델로 LLM-as-a-judge 프롬프트를 호출해, 생성 답변의 사실성·충분성·명확성 등을 주관적으로 채점/코멘트함. |
| **결론 및 계획** | * 결론:   본 실험으로 **모델 계열의 적용 가능성**과 **LangGraph RAG 아키텍처의 유효성**을 확인함.  **STOCK 기반 Merge → 데이터/모델 확장 파인튜닝 → DPO**의 단계적 로드맵과 RAG 검색/생성 양면 개선을 병행하면, 실무 투입 품질(정확한 PDF 인용 + 일관된 답변)에 도달할 수 있을 것으로 예상됨.   * 앞으로의 계획:   **sLLM 결합(최우선) :** Qwen2.5 7B sLLM과 **LangGraph RAG** 통합   * + - 방식: vLLM(OpenAI 호환) 또는 로컬 엔드포인트로 교체     - 체크: 프롬프트/토큰 한도/배치·지연시간, 인용 템플릿 적용   **최종 성능평가(결합 후)**   * **정량**: P@k, R@k, F1@k, **MRR**, **NDCG@k**, *(MAP 재산출: 0~1 스케일 확인)* * **정성**: LLM-judge + 소규모 휴먼 리뷰 병행 * **데이터셋**: 기존 벤치마크(SKN14-Final-3Team-Data2) + 사내 시나리오 확장 |