

SK네트웍스 Family AI과정 14기

모델링 및 평가 수집된 데이터 및 전처리 문서

산출물 단계	모델링 및 평가
평가 산출물	수집된 데이터 및 전처리 문서
제출 일자	2025.10.01
깃허브 경로	SKN14-Final-1Team
작성 팀원	안윤지, 정민영

1. 개요

1.1 데이터 설명

본 문서는 LLM을 활용하여 내부 고객의 업무 효율성을 높이기 위한 API 전문 개발자 지원 AI 기반 문서 검색 시스템(현재는 Google API만 지원)의 sLLM 파인튜닝 및 RAG 전처리용 데이터의 출처·수집·전처리·저장 방식을 정리한다.

데이터는 구글 개발자 공식 문서(11개 구글API)와 사내용 합성 문서로 구성된다. 구글 API 문서는 원문과 QA 데이터셋을 분리하여 각각 임베딩하고, 검색 과정에서 두 데이터를 함께 활용함으로써 문서-근거 포함 생성(LLM) 흐름에서 검색 정확도와 응답의 사실성을 높인다.

사내 문서는 원문만 임베딩하여 RAG 검색과 sLLM 파인튜닝 학습 데이터 생성에 활용되며, 사내용 챗봇 제공과 모델 성능 향상에 기여한다.

1.2 데이터 수집 목적

[구글 api 문서]

- Google 제품/서비스 문서(API/Reference/가이드)의 전문화 용어와 절차/파라미터/제약을 정확히 반영한다.
- 내부 사용자가 API 관련 이슈를 빠르게 해결하도록 문맥 검색 + 키워드 검색을 결합한 답변을 제공한다.
- 원문을 그대로 벡터화할 때 생길 수 있는 코드·문맥 단절 문제를 줄이고자 QA 변환 셋을 추가로 구축한다.

[기업 내부 문서]

- 사내 정책, 규정, 회의록 등을 축적하여 RAG 검색 환경 구축
- 직급&부서별 권한 기반 검색을 적용해 보안성과 효율성을 확보
- 사내 문서를 기반으로 sLLM 파인튜닝 학습 데이터를 생성하여, 사내용 챗봇 구축 및 응답 품질 개선

1.3 수집 대상 및 범위

- 전체 수집 데이터 건수:
 - 구글 API 문서: 약 2,000개 문서 (txt 기준)
 - 회사 내부 문서: 60개의 txt 문서 (각 부서별 15개씩)
- 전처리 데이터 건수:
 - 구글 API 문서 QA: 약 14,800개의 QA 데이터셋 (jsonl)

1.4 법적·윤리적 고려 사항

[구글 api 문서]

- 개인정보 포함 여부: 아니오
- 민감 정보 여부 및 조치 사항: 해당 없음
- 자동 크롤링 시 robots.txt 확인 여부: O
- 수집 데이터의 활용범위 명시 여부: O
- 출처 및 사용권
 - Google API 공식 문서 활용
 - QA 데이터셋으로 가공하여 원문과 QA 모두 내부 연구/교육 목적으로 사용

[기업 내부 문서]

- 개인정보 포함 여부: 아니오
 - 프롬프트 차원에서 개인정보 포함을 원천적으로 차단
- 출처 및 사용권
 - OpenAI API 활용 합성 문서, 외부 저작권 문제 없음, 회사 내부 자산으로만 활용
- 공개 여부
 - 내부 전용, 부서 및 직급별 접근 권한에 따라 제한

2. 데이터 수집

2.1 데이터 출처

- 구글 **API** 공식 문서([Google for developers](#))
 - [OAuth 2.0 / Google Identity](#)
 - [People API](#)
 - [Google Drive API](#)
 - [Google Sheets API](#)
 - [Gmail API](#)
 - [Calendar API](#)
 - [YouTube Data API](#)
 - [Google Maps](#)
 - [Firestore API](#)
 - [Firebase Authentication](#)
 - [BigQuery API](#)
- 기업 내부 문서: OpenAI 합성 데이터셋 (자체 생성)

2.2 데이터 유형

[구글 api 문서]

- 구글 API 원문: txt 파일
- 구글 API QA : jsonl

[기업 내부 문서]

- 기업 내부 문서 : txt 파일
- 기업 내부 문서 기반 sllm 파인튜닝 데이터 : json 파일

2.3 데이터 수집 방법 및 도구

[구글 api 문서]

- 수집 방식
 - Google api 문서 : 웹 크롤링 → 텍스트 추출 → 텍스트 파일 저장
 - 사내문서 : OpenAI API 프롬프트 합성 → 텍스트 파일 저장
- 수집 도구 또는 스크립트 설명:
 - 사용한 언어/라이브러리
 - 구글 API 문서 크롤링 : python, Selenium, lxml, beautifulsoup 등
 - 사내 내부 문서 : python, openai 등
 - 자동화 여부 및 주기: 2~3주일마다 수집 예정
- 자동화 흐름
 - 1. 원문 크롤링 : 지정된 링크에서 원문 데이터를 크롤링하여 확보
 - 2. 원문 벡터 데이터베이스 삭제 : 재크롤링한 원문이 중복되지 않도록 기존 벡터DB에 저장된 원문 중 특정 태그를 가진 문서를 삭제 (메타 필터링)
 - 3. 원문 벡터 DB에 저장 : 원문 벡터DB를 로드하여 새로 크롤링한 원문을 모두 ChromaDB에 저장

- 4. QA 데이터 생성 및 파일 저장 : 생성된 QA 데이터는 JSON 파일로 저장
- 5. QA 벡터 데이터베이스 삭제 : 새로 생성된 QA 데이터가 중복되지 않도록 기존 벡터DB에 저장된 QA 중 특정 태그를 가진 문서를 삭제 (메타 필터링)
- 6. QA 벡터 DB에 저장 : 새로 생성된 QA 데이터를 QA 벡터DB에 저장

【 기업 내부 문서 】

- 사내 내부 문서 OPENAI API 생성 흐름도
 - 문서 사양 정의 → 팀별 (사내 개발팀: 프론트엔드팀, 백엔드팀, DATA&AI팀, CTO)로 15개씩 카테고리·제목을 리스트에 정리
 - 프롬프트 생성 → 각 문서 사양에 맞춰 실행 지침과 섹션 구조를 포함한 프롬프트 작성 (마크다운으로 만들어달라고 요청하여, 정규화 처리 미리 정의)
 - **OpenAI API** 호출 → 프롬프트를 넣어 문서 본문을 생성
 - 파일 저장 → 생성된 문서를 직급/제목 규칙에 맞게 .txt 파일로 저장
 - 반복 구조 → 팀별(CTO 포함 4개) × 문서별(15개) 루프를 돌며 전체 60개 문서를 자동 생성 (txt파일)

3. 데이터 전처리

[구글 api 문서]

3.1 구글 api 문서 QA 셋

- 저장 포맷: JSONL(QA), VectorDB(임베딩)
- 파일 및 필드 예시

파일명	구분	필드명	데이터 타입	설명	예시
Google API Docs QA	page_content	question	string	질문 텍스트	Firestore 감사 로그에서 요청 호출자를 식별하기 위해 어떤 필드를 참조해야 하나요?
		answer	string	답변 텍스트	요청 호출자를 식별하기 위해 `AuditLog` 객체 내의 `AuthenticationInfo` 필드를 참조해야 하며, 여기에는 사용자의 `principalEmail`이 포함될 수 있습니다.
	metadata	source	array of string	출처 URL 리스트	["https://cloud.google.com/firestore/docs/audit-logging?hl=ko"]
		tags	string	태그(주제/분류)	firestore
		last_verified	string	최종 검증일	2025-08-19
		source_file	string	원문 파일명	cloud.google.com_firestore_docs_audit-logging_hl=ko.txt

- QA셋 생성 방법

1. 청크 분할 + 오버랩 적용

- 청크 크기: 900 토큰 / 오버랩: 150 토큰
- 문맥 연결 유지 + 중요 정보 누락 최소화

2. 페어 단위 처리

- (청크1+청크2), (청크2+청크3) 방식 Q&A 생성
- 각 내용이 두번씩으로 검토되어 정보 손실 방지

3. 질문-답변 생성 규칙 강화

- 문서 범위 밖 정보는 생성 금지
- 유효한 Q&A만 저장 (불필요/잘못된 Q&A 배제)
- 각 언어별 코드를 Q&A에 담을 수 있도록 프롬프트 강화
- 이전 페어들에서 생성된 질문을 프롬프트에 금지질문으로 전달하여, 중복질문 생성 방지

[기업 내부 문서 기반 파인튜닝 데이터 생성]

1. 사내 내부 문서 생성

파이프라인 흐름:

단계	설명
단계 1	사내 개발팀별 문서 사양 정리 (프론트엔드팀, 백엔드팀, DATA&AI팀 , CTO)
단계 2	정의된 개발팀 문서를 만들기 위한 OpenAI 호출 프롬프트 작성
단계 3	마크다운으로 생성되도록 요청하며, 정규화 규칙을 프롬프트에 사전 정의
단계 4	각 팀별 사내 문서 생성 후 TXT 파일로 저장

2. 초기 질문 생성

2-1. 문서 검색 질문 생성

목적: 각 팀별 문서에서 자연스러운 질문을 자동 생성

방법: **GPT-4o-mini**를 활용한 문서 기반 질문 생성

질문 유형:

- 구체·상세 질문: "기술적 기여도 평가에서 **A/B** 테스트 도입이 긍정적인 결과를 가져온 사례를 자세히 설명해 주세요."

- 간결한 질문: "백엔드 팀 업무 알려줘"

처리 과정:

단계	내용
1	각 팀별 문서 폴더 순회
2	TXT 파일을 읽어 문서 내용 추출
3	GPT-4o-mini 로 질문 5~25 개 생성
4	정규식으로 질문만 추출
5	JSONL 형식으로 저장

2-2. 일상 질문 생성

목적: 사내 직원이 챗봇에게 할 수 있는 일상적인 질문 생성

특징: 문서/보고서 요청 및 기능 설명 제외, 개인 경험 요구 제외, 추천/감상/잡담 위주 구어체

처리 과정:

단계	내용
1	GPT-4o-mini 로 25 개 캐주얼 질문 생성 (6 회 반복)
2	정규식으로 질문 추출
3	총 150 개 일상 질문 확보
4	모든 팀별 초기 질문 JSONL 파일에 동일 일상 질문 추가(자체 검증 후 추가)

3. sLLM 파인튜닝 학습 데이터 생성

RAG 기반 대화 데이터 생성 과정 (TOOL CALL 멀티턴):

단계	설명
단계 1: 벡터 데이터베이스 설정	BGE-M3 임베딩으로 생성한 사내 문서 Chroma 벡터 DB 로드
단계 2: 질문 체인 설정	초기 시스템 메시지(팀/직급별 TOOL PROMPT 차등 적용, 공손한 말투), 기본 질문 생성, 후속 질문(문서/일상) 생성
단계 3: TOOL CALL/RESPONSE 처리	업무 관련 질문: 문서 검색 필요 → TOOL CALL 구조 / 일상 질문: 캐주얼 대화 → TOOL CALL 없이 바로 답변
단계 4: 친구말투 변환	GPT-4o-mini로 공손말투 답변을 친구말투로 변환
단계 5: 데이터 통합	8개 파일 데이터를 순차 로드하여 qwen3_company_train_dataset_combined.json 저장

4. 생성된 학습 데이터 예시(cto용 공손말투 데이터)

- 문서 검색 질문은 tool_call(assistant) -> tool_response(user) -> 답변(assistant)
- 일상 질문은 바로 답변(assistant)

```

{
  "messages": [
    {
      "role": "system",
      "content": "\n당신은 사내 지식을 활용하여 사용자의 질문에 정확하고 유용한 답변을 제공하는 코드노바의 사내 문서 AI 챗봇입니다. \n\n사용자는 cto로서, 모든 팀의 문서를 열람할 수 있는 개발팀 최고 관리자입니다. \n당신은 <tools> 안해 있는 to"
    },
    {
      "role": "user",
      "content": "프론트엔드팀의 임원급 및 커뮤니케이션 부분에서 발생한 의견 충돌에 대한 구체적인 사례는 무엇인가요?"
    },
    {
      "role": "assistant",
      "content": "<tool_call>{\n  \"name\": \"cto_search\", \n  \"arguments\": {\n    \"keyword\": \"프론트엔드팀의 임원급 및 커뮤니케이션 의견 충돌 사례\"\n  }\n}</tool_call>"
    },
    {
      "role": "user",
      "content": "<tool_response>검색 결과: \n-----\n\n임 성과 / 내부 평가 | 내부 평가 피드백 문서_프론트엔드팀 근무 태도 및 기술 기여도 평가\n\n작성일: 2025-08-29\n회사: CodeNova | 대상: CTO\n\n---\n\n내부 평가 피드백 문서_프론트엔드팀 근무"
    },
    {
      "role": "assistant",
      "content": "프론트엔드팀의 임원급과 커뮤니케이션에서 생긴 구체적인 의견 충돌 사례로는 일부 팀원끼리 소문이 부족해서 업무가 늦어진 일이 보고됐어. 이 때문에 일괄 관리할 리소스 배분이 비효율적이었고, 팀 내 갈등 해결을 위해 커뮤니케이션 교육이나 근정"
    },
    {
      "role": "user",
      "content": "오늘 날씨가 초리네, 무슨 형질지 판단해줘."
    },
    {
      "role": "assistant",
      "content": "오늘 날씨가 초리면 배 울 가들성 있으니까 우산 챙기는 게 좋아, 안전하게 준비하는 게 좋겠어."
    }
  ]
},

```

[공손말투 데이터셋 시스템 프롬프트] - cto용 예시

당신은 사내 지식을 활용하여 사용자의 질문에 정확하고 유용한 답변을 제공하는 코드노바의 사내 문서 AI 챗봇입니다.

사용자는 cto로서, 모든 팀의 문서를 열람할 수 있는 개발팀 최고 관리자입니다.

당신은 <tools></tools> 안에 있는 tool을 호출하여 문서를 검색할 수 있습니다.

일상적인 질문(ex: 안녕, 안녕하세요, 반가워 등)의 경우, tool 호출 없이 바로 답변하세요.

Tools

You may call one or more functions to assist with the user query.

You are provided with function signatures within <tools></tools> XML tags:

<tools>

```
{"type": "function", "function": {"name": "cto_search", "description": "사내 문서 검색을 위한 도구입니다. 대화 내역을 바탕으로 사용자가 원하는 문서를 찾고, 관련된 문서를 반환합니다.", "parameters": {"type": "object", "properties": {"keyword": {"type": "string", "description": "검색할 문서 키워드 (예: '코드노바 API 서버 설정')"}, "required": ["keyword"], "additionalProperties": false}}
```

</tools>

For each function call, return a json object with function name and arguments within

<tool_call></tool_call> XML tags:

<tool_call>

```
{"name": "<function-name>", "arguments": "<args-json-object>"}
```

</tool_call>

그리고 다음 지침을 반드시 따르세요:

1. 기존의 말투는 잊고 정중하고 사무적인 어조로 답변해야 하세요.
2. 대화 내역의 말투도 참고하지 말고 무조건 정중하고 사무적인 어조로 답변하세요
3. 사실에 기반한 정보를 사용하세요.
4. 사용자의 질문에 대한 답변을 문서에서 찾을 수 없을 경우, "잘 모르겠습니다"라고 솔직하게 말하세요.
5. 사용자가 문서에 대한 질문이 아닌, "안녕"과 같은 일상적인 질문을 한다면 해당 내용에 대해서 적절히 답변해주세요.
6. 답변이 너무 길지 않게 하세요.
7. 사용자의 말투와 상관 없이, 반드시 정중하고 사무적인 어조로 답변해야 합니다.

[친구말투 데이터셋 시스템 프롬프트] - cto용 예시

당신은 사내 지식을 활용하여 사용자의 질문에 정확하고 유용한 답변을 제공하는 코드노바의 사내 문서 AI 챗봇입니다.

사용자는 cto로서, 모든 팀의 문서를 열람할 수 있는 개발팀 최고 관리자입니다.

당신은 <tools></tools> 안에 있는 tool을 호출하여 문서를 검색할 수 있습니다.

일상적인 질문(ex: 안녕, 안녕하세요, 반가워 등)의 경우, tool 호출 없이 바로 답변하세요.

Tools

You may call one or more functions to assist with the user query.

You are provided with function signatures within <tools></tools> XML tags:

<tools>

```
{"type": "function", "function": {"name": "cto_search", "description": "사내 문서 검색을 위한 도구입니다. 대화 내역을 바탕으로 사용자가 원하는 문서를 찾고, 관련된 문서를 반환합니다.", "parameters": {"type": "object", "properties": {"keyword": {"type": "string", "description": "검색할 문서 키워드 (예: '코드노바 API 서버 설정')"}, "required": ["keyword"], "additionalProperties": false}}
```

</tools>

For each function call, return a json object with function name and arguments within

<tool_call></tool_call> XML tags:

<tool_call>

```
{"name": "<function-name>", "arguments": "<args-json-object>"}
```

</tool_call>

그리고 다음 지침을 반드시 따르세요:

1. 기존의 말투는 잊고 가볍고 친근한 반말로 답변해야 하세요.
2. 대화 내역의 말투도 참고하지 말고 무조건 가볍고 친근한 반말로 답변하세요
3. 사실에 기반한 정보를 사용하세요.
4. 사용자의 질문에 대한 답변을 문서에서 찾을 수 없을 경우, "잘 모르겠어"라고 솔직하게 말하세요.
5. 사용자가 문서에 대한 질문이 아닌, "안녕"과 같은 일상적인 질문을 한다면 해당 내용에 대해서 적절히 답변해주세요.
6. 답변이 너무 길지 않게 하세요.
7. 사용자의 말투와 상관 없이, 반드시 가볍고 친근한 반말로 답변해야 합니다.

5. 최종 데이터셋

- 허깅페이스 데이터 링크: [SKN14-Final-1Team/qwen-finetuning-data-ko-250919](https://huggingface.co/SKN14-Final-1Team/qwen-finetuning-data-ko-250919) · Datasets at [Hugging Face](https://huggingface.co)

항목	내용
데이터 규모	총 2,368 개 멀티턴 대화 데이터
팀별 분포	Backend, Frontend, Data_AI, CTO (거의 동일 비중)
말투 변형	공손말투와 친구말투 각각 생성 (동일 비중)
질문 유형 분포	업무 관련 질문 : 일상 질문 = 2 : 1 비율
핵심 특징	다양한 질문 유형, 자연스러운 대화 흐름, TOOL CALL/RESPONSE 구조, 멀티턴 맥락, 반말·공손말투 모두 학습

6. 기술 스택

항목	내용
언어모델	GPT-4o-mini (질문 생성/반말 변환), BGE-M3 (문서 임베딩)
라이브러리	LangChain, Chroma, OpenAI API, HuggingFace
데이터 형식	JSONL (초기 질문), JSON (학습 대화), TXT (원본 문서)

7. 향후 사용 계획

- 본 데이터셋은 Qwen2.5-7B-Instruct 및 Qwen3-8B 모델의 LoRA 파인튜닝에 활용할 예정이다.
- Hugging Face Dataset Hub에 저장된 데이터셋을 로드하여 학습 파이프라인에 투입하고, 데이터셋 품질 유지와 최신성 반영을 위해 주기적인 데이터셋 업데이트 및 버전 관리를 수행할 계획이다.
- 이를 통해 최신 트렌드를 반영하고, 사내 내부 문서 검색 챗봇에서의 성능을 효과적으로 향상시키는 것을 목표로 한다.