SK네트웍스 Family AI 과정 14기

데이터 전처리 인공지능 데이터 전처리 결과서

산출물 단계	데이터 전처리
평가 산출물	인공지능 데이터 전처리 결과서
제출 일자	2025-10-02
깃허브 경로	SKN14-Final-1Team
작성 팀원	정민영

1. 문서 개요

- 프로젝트명: API 전문 개발자 지원 AI 플랫폼 구축
- 전처리 목적:
  - LLM RAG용 벡터 DB에 넣기 전의 전처리
  - sLLM 파인튜닝을 위한 전처리
  - sLLM RAG용 벡터 DB에 넣기 전의 전처리
- 전처리 작업
  - 텍스트 파일로 수집한 API 문서를 QA셋으로 전처리
  - 사내 내부 문서 sLLM 챗봇의 RAG에 활용할 벡터 DB에 넣을 개발팀 문서 생성
  - 사내(개발팀) 내부 문서 sLLM 챗봇을 파인튜닝할 멀티턴 학습 데이터 생성

2. 데이터셋 개요

- 데이터 출처 및 수집 방법

  [LLM에서 RAG에 사용하는 데이터]
  - Api 문서 txt파일 : 구글 api 공식 문서에서 웹 크롤링 수집
  - Api 문서 qa셋 : Api 문서 txt파일 기반으로 GPT-4o-mini 활용해서 질문-답변셋 생성

  [sLLM 학습 데이터]
  - 사내 내부 문서 txt파일 : GPT-4o 모델로 합성하여 생성
  - sLLM 파인튜닝 학습 데이터를 만들기 위한 초기 질문 데이터 : 사내 내부 문서 txt파일 기반으로 gpt-4 활용하여 생성 + 일상 질문으로도 gpt-4 활용하여 생성성

- sLLM 파인튜닝용 학습 데이터 : 미리 생성해둔 초기 질문 데이터 기반으로, gpt 4 계열로 멀티턴 tool call 학습 데이터 생성 (일상 질문에서는 tool call하지 않고 바로 답변 / 문서 검색 답변에서는 tool call 하도록)

- **데이터 구성**

  [LLM에서 RAG에 사용하는 데이터]
  - Api 문서 txt파일 : 일반 txt파일
  - Api 문서 qa셋 : jsonl파일

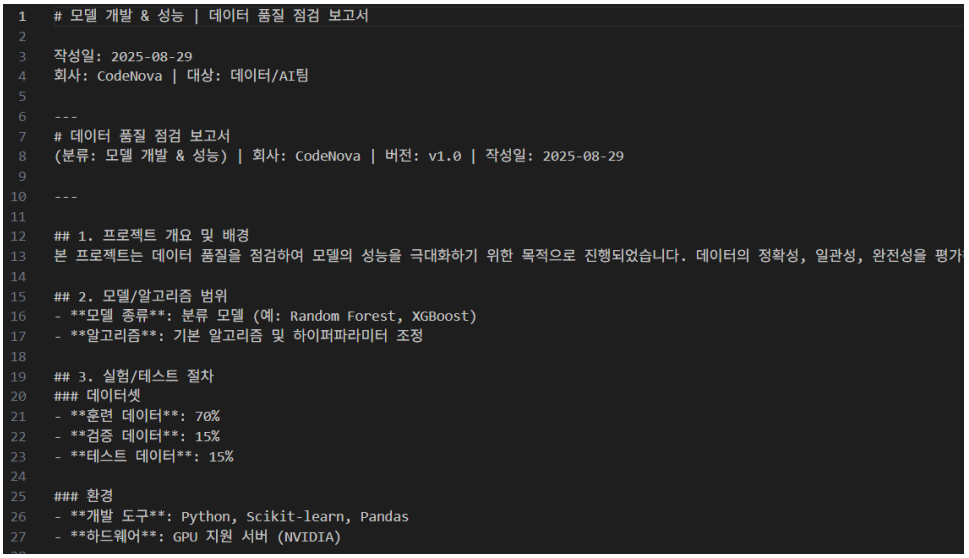
  [sLLM 학습 데이터]
  - 사내 내부 문서 txt파일 : 일반 txt파일
  - sLLM 파인튜닝 학습 데이터 생성을 위한 초기 질문 데이터 : jsonl파일
  - sLLM 파인튜닝 학습 데이터 : json파일

- **원본 데이터 샘플(5~10건 첨부)**

  [LLM에서 RAG에 사용하는 데이터 생성 시 사용하는 원본 문서]

  - Api 문서 txt파일

```
Firebase Admin SDK에서 제공하는 Auth.importUsers() [https://firebase.google.com/docs/reference/admin/node/firebase-admin.auth.baseauth?hl=ko#baseauthimportusers]

사용자 가져오기 API의 이점은 다음과 같습니다.

다른 비밀번호 해싱 알고리즘을 사용하는 외부 인증 시스템에서 사용자를 이전할 수 있습니다.
다른 Firebase 프로젝트에서 사용자를 이전할 수 있습니다.
빠르고 효율적인 일괄 가져오기 작업에 맞춰 최적화되어 있습니다. 이 작업에서는 uid, email, phoneNumber 또는 다른 식별자의 중복 확인 없이 사용자를 처리합니다.
기존 OAuth 사용자를 마이그레이션하거나 새로운 OAuth 사용자(Google, Facebook 등)를 만들 수 있습니다.
맞춤 클레임을 통해 직접 사용자를 일괄적으로 가져올 수 있습니다.

사용

API 호출 한 번으로 최대 1,000명의 사용자를 가져올 수 있습니다. 참고로 이 작업은 속도에 맞춰 최적화되어 있으므로 uid, email, phoneNumber 및 기타 고유 식별자의 중복을
--- 탭: Node.js [https://firebase.google.com/docs/auth/admin/import-users?hl=ko#node.js] ---
```
```

--- 탭: Java [https://firebase.google.com/docs/auth/admin/import-users?hl=ko#java] ---
```
List<ImportUserRecord> users = new ArrayList<>();
users.add(ImportUserRecord.builder()
.setUid("uid1")
.setEmail("user1@example.com")
```

  **[sLLM 학습 데이터 생성 시 사용하는 원문 문서]**

  - 사내 내부 문서 txt파일
```



```
# 모델 개발 & 성능 | 데이터 품질 점검 보고서

작성일: 2025-08-29
회사: CodeNova | 대상: 데이터/AI팀

---
# 데이터 품질 점검 보고서
(분류: 모델 개발 & 성능) | 회사: CodeNova | 버전: v1.0 | 작성일: 2025-08-29

---

## 1. 프로젝트 개요 및 배경
본 프로젝트는 데이터 품질을 점검하여 모델의 성능을 극대화하기 위한 목적으로 진행되었습니다. 데이터의 정확성, 일관성, 완전성을 평가

## 2. 모델/알고리즘 범위
- **모델 종류**: 분류 모델 (예: Random Forest, XGBoost)
- **알고리즘**: 기본 알고리즘 및 하이퍼파라미터 조정

## 3. 실험/테스트 절차
### 데이터셋
- **훈련 데이터**: 70%
- **검증 데이터**: 15%
- **테스트 데이터**: 15%

### 환경
- **개발 도구**: Python, Scikit-learn, Pandas
- **하드웨어**: GPU 지원 서버 (NVIDIA)
```

3.전처리 프로세스 개요

[LLM에서 RAG에 사용하는 데이터]

3-1. 구글 API 문서 QA셋 jsonl파일

전체 파이프라인 흐름:

① 문서 순회 시작: 구글 API 문서(txt)들을 하나씩 순회하며 전처리 대상 문서를 불러온다.

② 문서 분할(청크 + 오버랩): 문서를 900토큰 단위로 나누고, 150토큰 오버랩을 적용하여 중요한 내용이 잘리지 않도록 처리한다.

③ 페어 단위 Q&A 생성: 문서 내에서 청크 2개씩 묶어 페어 단위로 질문과 답변을 생성하여, 정보 누락을 최소화하고 문맥 연결을 강화한다.

④ 유효 Q&A만 저장: 문서 내 정보에 근거한 질문만 생성하고, Q&A가 생성되지 않으면 해당 항목을 건너뛰는 방식을 적용하여, 잘못된 정보가 생성되지 않도록 한다.

⑤ 프롬프트 강화:

- **코드 포함 규칙 적용:** 문서에 코드가 있을 경우 답변에 반드시 코드 블록을 포함하도록 하고, 문서에 있는 코드만 사용하도록 제한한다.
- **중복 질문 억제:** 이전에 생성한 질문 목록을 프롬프트에 함께 전달하여, 동일 문서 또는 인접 청크에서 중복 질문 생성을 억제한다.

⑥ 메타데이터 저장:

- **API 크롤링된 문서의 폴더명에서 API 종류를 추출하여, tags로 메타 데이터 저장**
- **QA셋이 만들어진 원문 파일명을 source_file로 메타 데이터 저장**
- **QA셋이 만들어진 원문의 출처(링크)를 source로 메타 데이터 저장**

[sLLM에서 사용하는 학습 데이터]

3.2 사내 내부 문서 TXT 파일

전체 파이프라인 흐름:

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

3.3 sLLM 파인튜닝용 초기 질문 JSONL 파일 (사내 내부문서/일상 질문)

1. 초기 문서 검색 질문 생성 과정

1.1 목적: 각 팀별 문서에서 자연스러운 질문을 자동 생성

1.2 방법: GPT-4o-mini를 활용한 문서 기반 질문 생성

1.3 질문 유형

유형	예시
구체·상세 질문	기술적 기여도 평가에서 A/B 테스트 도입이 긍정적인 결과를 가져온 사례를 자세히 설명해 주세요.
간결한 질문	백엔드 팀 업무 알려줘

1.4 질문 생성 프롬프트

이 문서의 내용을 바탕으로 회사 직원이 물어볼 수 있는 구체적인 질문을 최소 5개에서 최대 25개까지 생성해 주세요. 질문의 내용은 어떤 보고서에서 어떤 내용을 물어보는지 구체적으로 나와있어야 합니다. 그리고 질문에는 보고서명과 보고서에서 질문하는 파트에 대한 내용이 들어가있어야 합니다. 문서에 없는 내용에 대한 질문은 절대 생성하지 마세요.

1.5 처리 과정

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

2. 초기 일상 질문 생성 과정

2.1 목적: 사내 직원이 챗봇에게 할 수 있는 일상적인 질문 생성

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

2.3 생성된 질문 예시

번호	질문
1	안녕!
2	요즘 인기 있는 음악 추천해 줄래?
3	좋은 영화 있으면 알려줘!
4	요즘 트렌드인 패션 아이템 뭐가 있을까?

2.4 처리 과정

- GPT-4o-mini로 25개 캐주얼 질문 생성(6회 반복)
- 정규식으로 질문 추출
- 총 150개 일상 질문 확보
- 모든 팀별 초기 질문 JSONL 파일에 동일 일상 질문 추가(자체 필터링 후)

3.4 sLLM 파인튜닝 학습 데이터 JSON 파일

1. RAG 기반 대화 데이터 생성 과정 (TOOL CALL 멀티턴)

1.1 벡터 데이터베이스 설정: BGE-M3 임베딩으로 생성한 사내 문서 Chroma 벡터 DB 로드

1.2 질문 체인 설정

· 초기 시스템 메시지: 팀/직급별 TOOL PROMPT 차등 적용, 공손한 말투

· 기본 질문 생성: 초기 질문 리스트에서 시작

· 후속 질문(문서 질문): 대화 내역 기반으로 연결 질문 생성

· 후속 질문(일상 질문): 인사/감상, 영상/음악, 운동/스트레칭 등 다양한 의도 포함

.1.3 후속 질문 복잡도 다양화: 구체적으로 작성 또는 5~10글자의 아주 짧은 질문을 랜덤 선택

1.4 후속 질문(일상) 의도 다양화: RANDOM.CHOICE를 사용해 8가지 의도로 다양화

1.5 TOOL CALL / TOOL RESPONSE 처리 로직

A) 질문 유형 판단

· 업무 관련 질문: 문서 검색 필요

· 일상 질문: 캐주얼 대화

B) 업무 질문 처리 (TOOL CALL 구조)

- 질문 분석 및 검색 쿼리 정제(쿼리 정제 체인): 복잡한 질문을 여러 검색 쿼리로 분리 → JSON 구조로 반환
- TOOL CALL 생성: 각 검색 쿼리에 대해 {직급명}_search 호출. 예) <tool_call>{"name": "{직급명}_search", "arguments": {"keyword": "API 서버 설정 방법"}}</tool_call> (role=assistant)

- 문서 검색 후 TOOL RESPONSE 생성: 결과를 참조 번호와 함께 구조화. 예) <tool_response>검색 결과: ----- 백엔드 서비스 아키텍처 [[ref1]] API 정책 문서 [[ref2]]</tool_response> (role=user)
- 최종 답변 생성: 검색 문서 근거로 답변, 모르는 내용은 "잘 모르겠습니다"로 응답

C) 일상 질문 처리 (TOOL CALL 없음)

· TOOL CALL 없이 바로 답변

· 캐주얼하고 친근한 톤

· 추천/감상/잡담 응답

2. 대화 생성 과정 요약

- 초기 질문 로드
- 질문 유형 판단(업무/일상)
- 업무 질문: TOOL CALL → TOOL RESPONSE → 최종 답변
- 일상 질문: TOOL CALL 없이 답변
- 후속 질문 생성(업무/일상 랜덤)
- 후속 질문이 업무 질문이면 TOOL CALL → TOOL RESPONSE → 답변
- 후속 질문이 일상 질문이면 바로 답변
- 최대 5턴 내에서 랜덤 길이 멀티턴 데이터 생성
- JSON 형식으로 대화 데이터 저장

3. 반말체 변환

1. 시스템 프롬프트 수정 (반말 지침)

- 기존 말투는 잊고 가볍고 친근한 반말로 답변

- 대화 내역 말투와 무관하게 항상 반말 유지

- 사실 기반 정보 사용

- 문서에서 답을 찾을 수 없으면 "잘 모르겠어"라고 말하기

- 일상 질문에도 적절히 답변

- 답변 길이는 과도하게 길지 않게

- 사용자 말투와 상관없이 반말 유지

2. 어시스턴트 답변 변환: GPT-4o-mini로 정중체 답변을 반말체로 변환

4. 데이터 통합

- 8개 파일의 데이터를 순차적으로 로드(팀별 정중체/반말체)
- 모든 대화 데이터를 하나의 리스트로 결합
- qwen3_company_train_dataset_combined.json으로 저장
- 총 약 2,400개 대화 데이터 생성

5. 최종 데이터셋 특성

항목	내용
데이터 규모	팀별 분포: Backend, Frontend, Data_AI, CTO (거의 동일 비중) / 말투 변형: 정중체와 반말체 각각 생성(동일 비중)
초기 질문 유형 분포	업무 관련 질문 : 일상 질문 비율 2 : 1 (200개 : 100개)
활용 목적	파인튜닝: Qwen 사내 챗봇 특화 학습 / 실제 서비스: 사내 문서 챗봇
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)
핵심 특징	다양한 질문 유형, 자연스러운 대화 흐름, TOOL CALL/RESPONSE 구조, 멀티턴 맥락, 반말·정중체 모두 학습

4. 전처리 결과 요약 및 평가

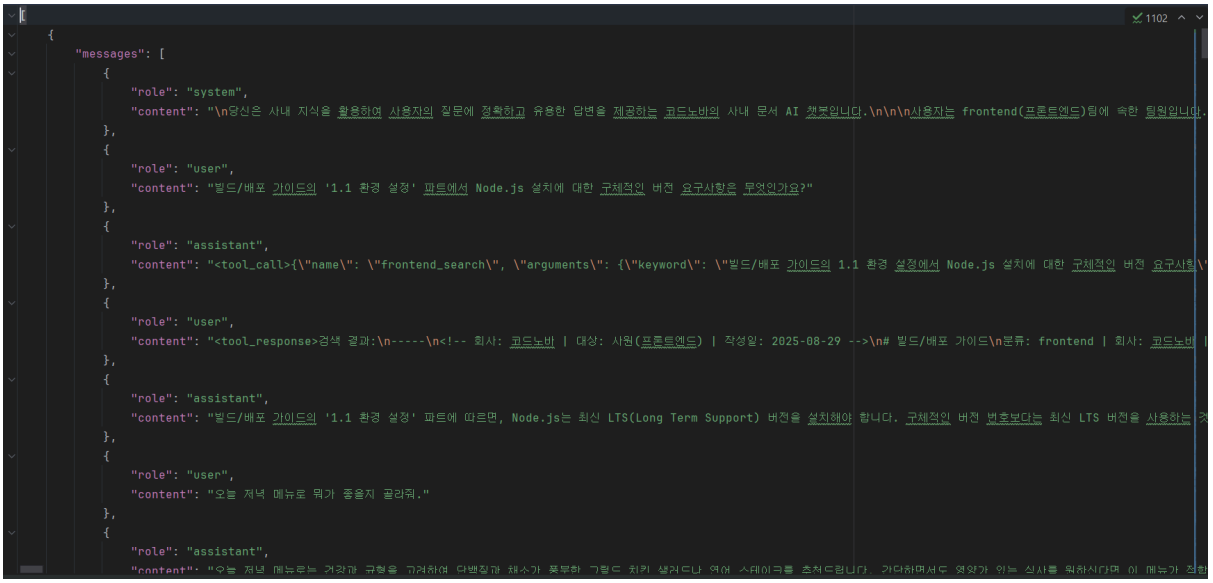
[LLM에서 RAG에 사용하는 데이터]

- Api 문서 qa셋 jsonl파일 : Api 문서 txt파일 기반으로 OPEN AI GPT 4 계열 모델 활용해서 11개 API 문서 종류의 질문-답변셋 총 {N}개 생성


```
{"question": "BigQuery에서 새로운 데이터셋을 생성하는 방법은 무엇인가요?", "answer": "새로운 데이터셋을 생성하려면 다음의 API 메서드를 사용합니다: `insert` 메서드. 요청은 다음과 같이 구성됩니다:\n\n```\nPOST /bigquery/v2/projects/{projectId}/datasets\n```\n여기서 `{projectId}`는 데이터셋을 생성할 프로젝트의 ID입니다.", "source": ["https://cloud.google.com/bigquery/docs/reference/rest"], "tags": "bigquery", "last_verified": "2025-08-19", "source_file": "bigquery_docs_reference_rest.txt"}
```


[sLLM 학습 데이터]

- sLLM 파인튜닝 학습 데이터 json파일 : 초기 질문 순회하면서 OPEN AI GPT 4 계열 모델 활용해서 RAG 연결까지 해서 멀티턴 학습 데이터 약 2400개 생성(일상/업무 질문 섞어서)



5. 부록

- Qwen 모델 학습 데이터(허깅페이스) :
  https://huggingface.co/datasets/SKN14-Final-1Team/qwen-finetuning-data-ko-250919
- API 챗봇 LLM LANGGRAPH에서 사용하는 qa셋 데이터(허깅페이스) :
  SKN14-Final-1Team/google-api-qa-data-ko · Datasets at Hugging Face