

## Wrap-Up Report: Generation for NLP

### 수능형 문제 풀이 모델 생성

#### ‘나야, 자, 연어’팀

곽희준, 김정은, 김진재, 오수현, 윤선웅, 정민지



#### 요약

AI 기술의 발전에 따라 인공지능 모델의 활용성이 점차 주목받고 있다. 자연어 처리 분야에서 대형 모델은 프롬프트로 문제를 넣었을 때 어려운 문제를 손쉽게 풀어내지만, 성능 향상을 위해 천문학적인 자원을 투입한다는 한계점이 존재한다. 본 프로젝트는 이러한 문제에 주목하여, 한정된 자원에서 작은 언어 모델 (sLLM)을 활용한 수능 문제 풀이에 최적화한 모델 학습을 수행하고, 정확도 높은 문제의 답을 제공하는 모델을 만드는 것을 목표로 한다.

제공받은 데이터셋을 꼼꼼히 검토하여 문제의 출처를 파악하고 지문, 보기, 선지, 답변 각각에서의 저품질 데이터를 탐색하여 유형별로 정리하고 저품질 데이터를 제거하였다. 학습 데이터셋에서 모델이 잘 풀어내지 못하는 문제의 유형을 파악한 이후 해당 유형의 데이터를 증강하고자 저작권 및 품질에 이상이 없는 외부 데이터를 전처리하여 증강하였다. 모델링 과정에서는 한정된 자원을 효율적으로 활용하기 위해, 32B 모델을 LoRA Fine-Tuning 수행할 수 있는 최신 라이브러리 Unsloth를 적용하여 학습하였으며 Prompt Engineering 및 학습 하이퍼파라미터 조정을 수행하며 성능을 향상시켰다. EDA 과정에서 내부 지식만으로 풀 수 없는 문제의 존재를 확인하여 외부 데이터를 Retrieve 할 수 있는 RAG 파이프라인을 구축하였고 Retrieval 성능을 평가할 수 있는 데이터셋을 제작하기도 하였다. 최종적으로 다양한 조합을 앙상블한 결과, Public 리더보드 1위 (Acc. 0.8341) / Private 리더보드 1위 (Acc. 0.7977) 성능을 달성하였다.

본 프로젝트를 통해 작은 모델에서도 Fine-Tuning을 통해 큰 자원이 투입된 모델에 준하거나 오히려 우수한 모델을 개발할 수 있다는 점을 확인할 수 있었다. 해당 프로젝트의 전체 코드는

<https://github.com/boostcampaitech7/level2-nlp-generationfornlp-nlp-05-lv3>에서 확인할 수 있다.

	Rank	Team Name	Team Member	ACC	Entries	Final
Public	My Rank 1	NLP_05조		0.8341	118	3d
Private	My Rank 1	NLP_05조		0.7977	118	4d

# 목차

## 1. 프로젝트 개요

- 프로젝트 소개
- 팀 목표
- 팀원 소개 및 역할

## 2. 데이터

- EDA
- Fine-Tuning을 위한 데이터 작업

## 3. 모델링

- 모델 선정 및 튜닝
- LoRA 튜닝
- 프롬프트 튜닝

## 4. 기술적인 시도들

- RAG
- 앙상블

## 5. 2025학년도 수능 (국어, 사회 탐구)

- 생활과 윤리 1등급, 국어 화법과 작문 2등급
- 성적 분석
- 프로젝트 평가

## 6. 기술적 환경 협업

- 프로젝트 일정 관리: Jira
- 결과 기록 및 공유 : Confluence
- 서버 대시보드: Notion

## 7. 팀 회고

- 좋았던 점
- 아쉬운 점
- 앞으로의 목표 및 다짐

## 8. 개인 회고

- 곽희준
- 김정은
- 김진재
- 오수현
- 윤선웅
- 정민지

# 1. 프로젝트 개요

## 프로젝트 소개

AI 기술의 비약적인 발전은 다양한 분야에서 혁신을 이끌고 있으며, 특히 대규모 언어 모델(LLM)의 성과는 시험 분야에서도 주목받고 있다. GPT, Claude, Gemini와 같은 대형 모델들은 사법고시, 의사 시험뿐만 아니라 한국의 대학수학능력시험(수능)에서도 높은 점수를 기록하며 학계와 산업계의 관심을 모으고 있다. 그러나 이러한 대형 모델의 성공 뒤에는 막대한 자원과 데이터가 투입되는 한계가 존재한다. 반면, 작은 규모의 언어 모델들은 상대적으로 제한된 자원으로도 활용 가능하다는 장점을 가지고 있지만, 대형 모델 대비 성능 부족이 현실적인 과제로 남아있다. 이에 본 프로젝트에서는 **작은 언어 모델(sLLM)을 활용하여 한국어에 특화된 수능형 문제 풀이 성과를 극대화하는 것을 목표로 한다**. 이를 통해 다음 질문에 도전한다.

작은 모델이 대형 모델 수준의 수능 성적을 달성할 수 있는가?

위 목적을 달성하기 위해 본 프로젝트는 sLLM의 성능을 **정확도(Accuracy)로 평가한다**. 정확도는 전체 문항 중 정답을 맞힌 비율을 의미하며, 단순하고 직관적이어서 수능형 문제 풀이의 핵심 목표인 정답률을 직접 측정하는 데 적합한 지표이다. 정확도는 다음의 공식으로 계산된다.

$$Accuracy = \frac{correct}{total}$$

## 팀 목표

본 프로젝트를 수행하기에 앞서 효과적인 협업과 성공적인 결과를 위해 다음의 공통 목표를 설정했다.

- 1. **하나의 팀, 하나의 목표:** 모두 협력하여 수능형 문제 풀이를 최적화하는 단일한 목표를 공유
- 2. **신속하고 투명한 소통:** 진행 상황을 빠르고 명확히 공유하여 효율성과 팀워크 극대화
- 3. **데이터 윤리 준수:** 저작권과 데이터 윤리를 철저히 준수하며 책임 있는 연구를 수행
- 4. **데이터 기반 의사결정:** 분석한 데이터를 기반으로 합리적이고 객관적인 결정 내리기
- 5. **LLM 기술 습득과 활용:** LLM 관련 기술을 학습하고 활용하여 프로젝트의 성과를 극대화

## 팀원 소개 및 역할

팀 원	역 할
곽희준	데이터 레이블링, EDA, 외부 데이터셋 탐색, GPT를 통한 데이터셋 증강 실험, 코드 리팩토링, LLM 학습 방법 설계, Fine-Tuning, RAG 파이프라인 구축, 최종 코드 정리
김정은	데이터셋 레이블링, 모델 탐색 및 Fine-Tuning, 데이터셋 크롤링 및 전처리(공무원 기출, khan), 데이터셋 품질 테스트, RAG 시스템 구축 및 실험, 프롬프트 엔지니어링
김진재	초기 팀 환경 구축 및 대시보드 제작, 데이터셋 레이블링, 데이터 탐색, RAG 데이터 전처리(번역), Retrieval 구축 및 실험 (Sparse)
오수현	초기 베이스라인 코드 구축, 데이터셋 레이블링, 데모 페이지 제작
윤선웅	데이터셋 레이블링, 모델 탐색 및 Fine-Tuning, Unsloth 세팅, 프롬프트 엔지니어링, 데이터셋 크롤링(공무원 기출, khan), 데이터셋 품질 테스트, LoRA 튜닝, 앙상블
정민지	데이터셋 레이블링, 벡터스토어 데이터 크롤링 및 전처리 (OpenStax, Wikipedia, 우리역사넷), Retrieval 성능 평가 데이터셋 및 지표 구성, RAG 시스템 구축 및 실험 (Chunk size, Dense Retrieval, Reranking)

[그림 1-1] 팀원 소개 및 역할

## 2. 데이터

### EDA

#### 학습 데이터셋 분포 확인 및 수동 레이블링

프로젝트의 초기 단계에서는 모델이 풀어야 할 수능형 문제의 특징을 파악하기 위해 탐색적 데이터 분석(EDA)을 진행했다. 이를 위해 Google Sheet를 활용하여 문제 데이터를 수동으로 레이블링하고 아래 항목을 중심으로 분석했다.

#### 데이터셋 출처별 분포

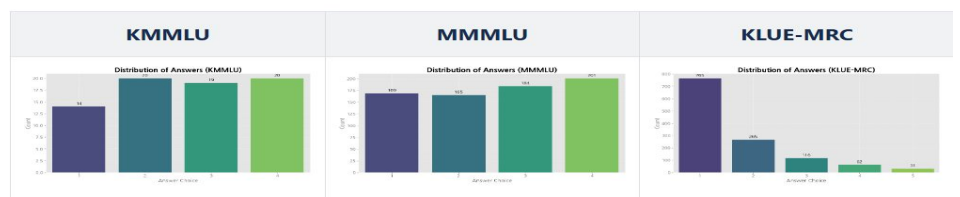
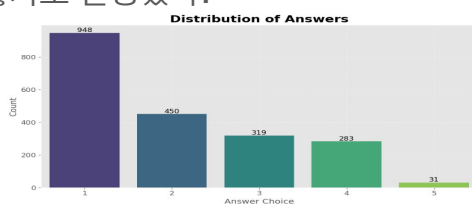
본 프로젝트에서 제공된 학습 데이터셋의 출처를 파악한 결과 KMMLU, MMMLU, KLUE-MRC로 구성되어 있었으며, 각 분포는 [표 2-1]과 같다. 가장 큰 비중을 차지하는 KLUE-MRC에서 추출된 데이터는 주어진 문맥만으로 논리적 추론을 통해 문제를 해결하는 국어 비문학과 유사한 형태의 문제로 구성되어 있음을 확인했다. 반면에 MMMLU와 KMMLU는 사회 영역의 문제로 구성되어 있었다. 사회 영역의 문제는 분야에 따라 고유한 특성이 있을 것으로 판단하여, 각 데이터의 출처를 기준으로 세분류를 분석하고 수동으로 레이블링을 진행했다.

구분	데이터 출처	index	데이터 개수
train	KMMLU	0~72	73
	MMMLU	73~791	719
	KLUE-MRC	792~2030	1239
test	수능형 문제 + KMMLU / MMMLU(ko) / KLUE MRC 데이터		

[표 2-1] 데이터셋 출처별 분포

#### 정답 선지별 분포

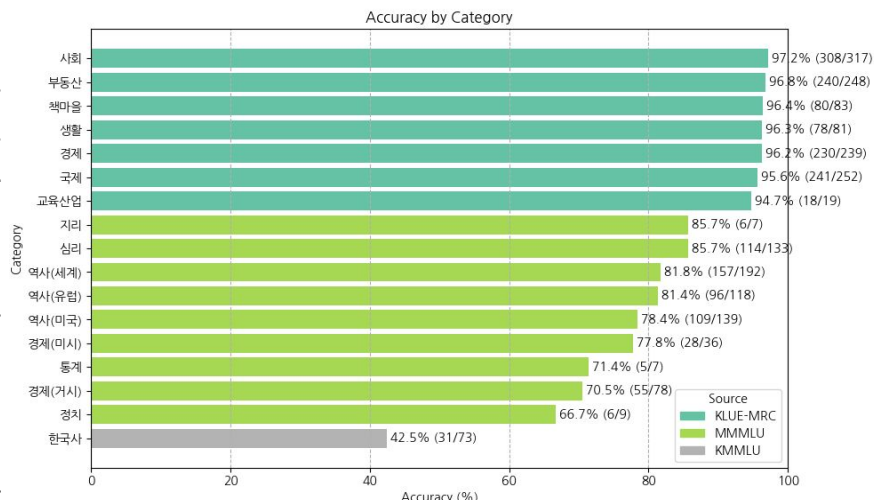
전체 데이터셋의 정답 선지 분포를 분석한 결과, 정답이 '1'번에 편중되어 있다는 것을 확인했다. 특히 KLUE-MRC 데이터셋의 영향이 컸는데, 이는 GPT-4o-mini를 사용하여 지문을 기반으로 질문과 5지 선다형 선지를 생성하는 과정에서 발생한 편향으로 추정된다. 이에 정답 선지 분포를 균일하게 구성하면 편향이 줄어들어 정확도가 향상될 것이라는 가설을 세웠다. 다만 모델 실험 과정에서 정답 선지를 균일하게 조정하는 것이 뚜렷한 성능 향상으로 이어지지 않았다. 따라서 최종적으로 정답 선지의 균일화 옵션을 선택하지는 않기로 결정했다.



[그림 2-1] 정답 선지별 분포

#### 세분류별 정답률 분포

SOTA 모델 (Unsloth/Qwen2.5-32B-Instruct)을 활용하여 train 데이터셋 재추론을 수행한 결과, KLUE-MRC 데이터셋에 대한 정답률은 90% 이상인 반면, KMMLU 데이터셋의 한국사 문제에 대해서는 정답률이 42.5%로 낮게 나타났다. 이를 통해 모델은 주어진 지문으로부터 논리적인 추론 통해 답을 찾는 내부 지식형 문제는 이미 해결하고 있으나, 사회 영역 문제에 대한 정답률이 낮은 것을 확인했다. 이에 Fine-Tuning 시 데이터 증강을 사회 영역 중심으로 진행하고, 해당 영역에 대해 RAG 적용 시 성능이 향상될 것이라는 가설을 세웠다.

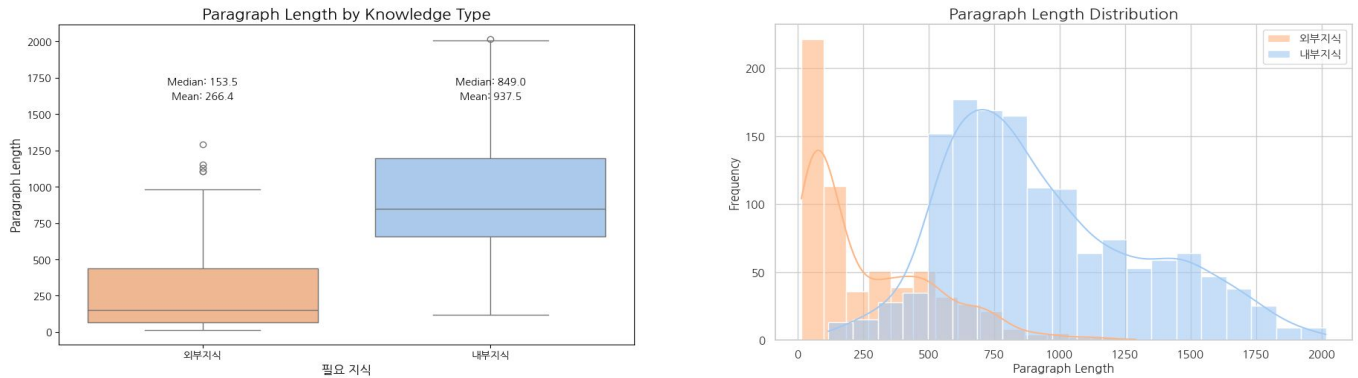


[그림 2-2] 세분류별 정답률 분포

## 세분류별 필요 지식 데이터 분포

파악한 출처를 기준으로 각 데이터의 세분류를 파악하고, 문제 해결에 필요한 지식을 외부 지식(문제 외부 정보)과 내부 지식(주어진 데이터 내 정보)로 구분했다. 분석 결과 내부 지식으로 풀 수 있는 문제는 1419개로 70%에 해당하며, KMMLU, MMMLU 문제는 외부 지식이 필요한 데이터로 판단했다. 하지만 세분류 중 역사(세계, 미국, 유럽)와 KLUE-MRC의 경우에는 내부 지식으로 추론 가능한 것으로 판단했다.

필요 지식에 대한 분석을 마친 후 본문(paragraph)의 글자 수 분포에 명확한 차이를 발견했다. 약 500자를 기준으로 두 집단을 구분할 수 있었다. 각 분포는 [그림 2-3]에서 확인할 수 있다. 해당 분석 결과는 추후 RAG를 위한 Vector Store 구축에 활용하였다.



[그림 2-3] 세분류별 필요 지식 데이터 분포

## 데이터 저품질 사례 분석

데이터를 구성하는 요소인 지문(paragraph), 질문(question), 선지(choices), 정답(answer)의 품질을 점검했다. 품질이 낮거나 노이즈가 포함된 데이터를 확인하여 모델 학습에 적합한 데이터를 선별하거나 수정하는 기준을 마련했다. 분석 결과는 모델 설계와 학습 과정에서 필수적인 정보를 제공하며, 특히 세분류별로 적합한 전략 수립과 데이터 정제 작업에 활용했다.

**지문 저품질:** 제공된 지문에 띄어쓰기 및 맞춤법 오류, 확인되지 않는 ‘밑줄’, <보기> 형식의 의상, 지문에 포함된 질문, 지문과 질문의 내용 동일, 지문의 내용 부실 등의 저품질 사례를 확인하였다.

**질문 저품질:** 질문에 맞춤법 오류, 불완전한 형태, 형식 이상, (가), (나)와 같은 단락 식별 불가 등의 저품질 사례를 확인하였다.

**선지 저품질:** 선지에 유니코드와 같은 노이즈, 불완전한 형태, 정보 확인 불가, 질문과 낮은 연관성 등의 저품질 사례를 확인하였다.

**정답 저품질:** 질문과 선지의 엇갈림, 정답 오류, 복수 정답 등의 저품질 사례를 확인하였다.

### 지문 저품질 “<보기> 형식의 이상” 예시

왕이 군대를 내어 막다가 화살에 맞아 돌아가셨다. - 삼국사기 - (가)(나)(다)(라) 낙랑. 대방군축출 모용황에 의해 환도성 함락 전진의 순도 불교 전래평양 천도백 제한성 함락

### 질문 저품질 “맞춤법 오류” 예시

위의 폴 커패시터의 청인이 가장 잘 설명하는 상황은?

### 선지 저품질 “노이즈” 예시

세금 인상 \xa0\xa0\xa0 수용 증가 \xa0\xa0\xa0인하 \xa0\xa0\xa0 수요 감소

### 정답 저품질 “복수 정답” 예시

미국의 6대 대형은행이 2008년 금융위기 이후 5년간 법률 비용으로 사용한 돈이 1030억달러(약 115조원)에 달하는 것으로 집계됐다. ... (후략) 2008년 금융위기 이후 미국의 6대 대형 은행이 법률 비용으로 사용한 총액은 얼마인가  
[‘1030억달러’, ‘986억달러’, ‘213억달러’, ‘60억달러’, ‘115조원’]

[그림 2-4] 저품질 데이터 예시

### Fine-Tuning을 위한 데이터 작업

EDA 결과를 바탕으로 데이터 개선 및 증강 작업을 수행했다. 학습 데이터의 품질을 개선하면 모델의 추론 성능이 향상될 것으로 기대하여, 저품질 데이터를 중심으로 잘못된 레이블 수정, 노이즈 제거, 데이터 구조 보완 등의 개선을 진행하였다. 또한 오답률이 높은 세분류를 중심으로 데이터 증강을 수행했다. 정제와 증강 작업을 수행하지 않은 기존 데이터의 Accuracy는 0.6659였으며, 이를 기준으로 단계별로 데이터 작업을 수행하였다.

### 데이터 품질 개선

EDA 결과 전체 학습 데이터 2,031개 중 저품질 데이터는 177개로 확인되었다. 기존 데이터의 정확도를 기준으로 데이터 정제 작업을 단계별로 진행하였다.

시도 1. 모든 저품질 데이터 제거: 총 177개의 모든 저품질 데이터를 제거한 결과, **정확도는 약 1.61%p 하락한 0.6498**로 나타났다. 이는 모델의 크기에 비해 학습에 사용되는 데이터가 적은 상황에서, 데이터가 더 삭제되어 모델 학습 시 전달되는 정보가 크게 감소했기 때문이라고 분석했다. 이에 데이터를 최대한 삭제하지 않고, 저품질 데이터를 수정하여 데이터 품질을 개선하는 방향으로 논의했다.

시도 2. “정답 저품질” 데이터 전체 수정: 모든 저품질 데이터를 수정하는 것은 비효율적이며, 오히려 모델의 강건성을 해칠 수 있다고 판단했다. 따라서 모델 성능에 치명적일 수 있는 “정답 저품질” 데이터만 수정하기로 결정했다. 문제와 선지가 서로 다른 내용을 담고 있거나 정답이 잘못된 경우 등 총 14개를 확인하여, 원전 데이터를 참고해 수정했다. 그러나 “정답 저품질” 데이터 14개를 모두 수정한 결과, **정확도는 약 2.07%p 하락한 0.6452**로 나타났다. 이러한 성능 저하는 학습 데이터셋뿐만 아니라 테스트 데이터셋에도 유사한 오류가 있어 모델의 강건성이 하락했기 때문으로 분석했다.

시도 3. “정답 저품질” 데이터 일부 수정: 두 번째 시도의 분석 결과를 바탕으로, 전체 14개의 “정답 저품질” 데이터 중 일부만 수정해 보기로 했다. 정확한 레이블 정보를 전달하면서도 모델의 강건성을 확보하기 위한 전략이었다. 이를 위해 8개의 저품질 데이터를 수정한 결과, **정확도가 약 1.84%p 상승한 0.6843**이 되었다. 따라서 해당 버전의 수정된 데이터를 학습 데이터셋으로 사용했다. 이를 통해 데이터 수를 유지하면서 노이즈를 일부 제거하면 모델의 강건성을 확보하면서도 성능이 향상됨을 확인할 수 있었다.

### 데이터 증강

앞서 EDA에서 분석한 세분류별 정답률을 확인하여, 오답률이 높고 데이터 수가 적은 세분류인 역사(한국사, 세계사), 경제(거시, 미시), 정치를 중심으로 데이터 증강을 수행했다. 이때 데이터 저작권 윤리를 철저히 준수하기 위해 모든 데이터는 저작권 및 사용 권한을 명확히 확인한 후 수집하였으며, 라이선스 조건을 충족하는 데이터만 활용하였다.

시간 효율적인 실험을 위해 작은 모델인 unsloth/Qwen2.5-7B-Instruct-bnb-4bit를 이용해 실험을 진행했다. 앞서 데이터 품질 개선이 이루어진 최종 학습 데이터셋에 여러 출처의 데이터를 추가하면서 성능 향상 여부를 확인했고, 이 중 가장 큰 성능 향상을 보인 데이터를 이용해 데이터 증강을 수행했다. 증강 결과 최종 학습 데이터셋의 구성은 [표 2-4]와 같다. 모델의 논리적 추론 능력을 보완하기 위해 공무원 기출문제 중 국어 및 언어 논리 영역의 문제를 활용하여 증강을 시도했다. 또한 앞서 EDA에서 사회 영역 문제를 잘 풀지 못한다는 분석을 바탕으로, Khan Academy에서 제공하는 AP 문제를 DeepL API를 사용해 한국어로 번역하여 Fine-Tuning용 학습 데이터에 추가했다.

데이터 설명	Accuracy
raw	0.6659
raw + 공무원 한국사	0.6682
raw + 공무원 국어/언어 논리	0.6912
raw + 공무원 사회/상황 판단	0.6751
raw + AP(영어)	0.6820
raw + AP(한국어)	0.6912
raw + AI Hub	0.6728

[표 2-2] 증강 데이터별 실험 결과

데이터셋		저작권	언어	데이터 수
공무원	국어	공공누리 제1유형	Ko	90
	언어 논리		Ko	275
Khan	US History	CC BY-SA 4.0	En → Ko	44
	World History		En → Ko	48
	Microeconomics		En → Ko	33
	Macroeconomics		En → Ko	33
	Gov't & Politics		En → Ko	114

[표 2-3] 최종 증강 데이터셋 구성

### 3. 모델링

#### 모델 선정 및 튜닝

#### 기본 모델 탐색

초기 단계에서는 제공된 서버 환경을 고려하여 적합한 모델을 탐색했다. 서버는 NVIDIA V100 GPU와 100GB의 하드웨어 메모리, 32GB의 VRAM을 지원하므로, 효율적인 자원 활용을 위해 2B~11B 크기의 모델만을 후보로 선정했다. 개별 모델의 실험 결과는 [표 3-1]에서 확인할 수 있으며, 실험 결과 Qwen/Qwen2.5-7B-Instruct의 성능이 가장 우수함을 확인했다. 이는 적절한 자원 사용과 정확도를 모두 충족하는 모델로, 본 프로젝트의 기본 모델로 선정되었다.

모델 이름	Accuracy (Public)
beomi/gemma-ko-2B	0.3944
Bllossom/llama-3.2-Korean-Bllossom-3B	0.4608
CohereForAI/aya-expanse-8B	0.4931
mistralai/Mistral-7B-Instruct-v0.3	0.5507
meta-llama/llama-3.1-8B-Instruct	0.5645
NCSOFT/llama-VARCO-8B-Instruct	0.5922
mistralai/Mistral-8B-Instruct	0.6060
(중략)	...
Qwen/Qwen2.5-7B-Instruct	0.6613

[표 3-1] 2B ~ 11B 모델의 탐색 결과

#### Unsloth 활용 (모델 크기 확장 및 학습, 추론 시간 단축)

앞서 설명한 실험에서 모델의 크기가 커질수록 성능이 향상되는 경향을 확인함에 따라, 보다 큰 모델을 활용해 성능을 극대화하고자 했다. Unsloth의 자원 최적화 기법을 통해 제한된 하드웨어에서도 32B 크기의 대규모 모델을 효율적으로 실험할 수 있었으며, 7B 모델 대비 14.28%의 정확도가 상승했다. 추가적으로 Unsloth에서 제공한 UnslothTrainer와 FastLanguageModel을 템플릿 코드에 적용하여 학습과 추론에 걸리는 시간을 1/6로 줄이고, 성능 또한 0.0047 상승하였다. 학습 시간 및 추론 시간 단축과 성능에 대한 결과는 [표 3-2]에서 확인할 수 있다.

		Before	After
라이브러리	Trainer	SFTTrainer	UnslothTrainer
	Model	AutoPeftModelForCausalLM	FastLanguageModel
	Tokenizer	AutoTokenizer	FastLanguageModel
시간	Train	13시간	2시간
	Inference	2시간	20분
성능	Accuracy	0.8041	0.8088 (+0.0047)
메모리 사용 (32000MiB)	Train	25000	29000
	Inference	30000	31000

[표 3-2] Unsloth 사용 유무에 따른 비교



## LoRA 튜닝

모델의 크기가 확장됨에 따라 PEFT를 적용하기 위해 LoRA 튜닝을 수행했다. LoRA는 대규모 언어 모델의 가중치 업데이트를 자원 효율적으로 진행할 수 있는 기법으로, 적은 파라미터만 조정하여 메모리 사용량을 줄이고 학습 속도를 향상할 수 있다. LoRA 튜닝에서는 주요 하이퍼파라미터 간의 관계를 고려하여 최적의 설정을 탐색한다. 개별 옵션에 대한 설명은 다음과 같다.

- **r**: Low-rank 행렬의 크기 결정, 모델의 적응력 조정. 값이 클수록 더 많은 표현력을 제공하지만 자원 소모 증가
- **lora\_alpha**: LoRA 학습의 스케일링 파라미터로, LoRA의 업데이트 크기를 조절
- **lora\_dropout**: 레이어에 적용되는 dropout 비율로, 과적합을 방지하고 일반화 성능을 향상
- **target\_modules**: LoRA를 적용할 모듈을 선택하는 요소로, 특정 모듈에 집중적으로 적용하여 효율성을 극대화
- **rs\_LoRA**: LoRA에서 r이 커질 때 발생하는 그라디언트 붕괴 문제를 추가적인 자원 소모 없이 극복

전체적인 목적은 r값을 키워 학습 파라미터 수를 키워 전체적인 성능 향상을 유도했고, lora\_alpha와 target\_module수도 증가시켰다. r이 증가함에 따라 그라디언트의 붕괴를 막고자 rs\_LoRA를 True로 설정하였다. 추가적으로 모델 크기에 비해 데이터 수가 적다고 판단하여 과적합을 방지할 필요가 없다고 판단해 lora\_dropout을 0으로 설정하여 최종적으로 [표 3-3]와 같이 최적의 조합 version 3을 적용했다.

	(기본 제공) version 1	(Rank Tuning) version 2	(Unsloth LoRA Tuning) version 3
r	6	16	64
lora_alpha	8	16	16
lora_dropout	0.05	0.1	0
target_modules	q_proj, k_proj	q_proj, k_proj, v_proj, o_proj	q_proj, k_proj, v_proj, o_proj, gate_proj, down_proj, up_proj
rs_LoRA	-	-	TRUE
Accuracy 변화		+0.0046	+0.0092

[표 3-3] LoRA 튜닝 결과

## 프롬프트 튜닝

선정한 모델을 기준으로 다양한 프롬프트를 활용하여 성능을 비교하는 프롬프트 튜닝을 수행했다. 실험에서는 아래 세 가지 유형의 프롬프트를 적용했다. 기본 프롬프트를 포함하여 단계별로 문제 풀이를 유도하는 zero-shot CoT 프롬프트, 모델을 자극하는 AI 자극 프롬프트로 실험한 결과, 기본 프롬프트의 Accuracy는 0.7097을 달성한 반면, zero-shot CoT 프롬프트와 AI 자극 프롬프트는 모두 0.1~0.2%p 하락한 성능을 보여주어 기본 프롬프트를 베이스라인으로 채택하였다. 각 프롬프트는 [그림 3-1], [그림 3-2], [그림 3-3]에서 확인할 수 있다.

**시스템 프롬프트**  
질문을 읽고 질문의 답을 구하세요.

**사용자 프롬프트**  
지문: (지문 내용)

질문: (질문 내용)

<보기>  
(보기 내용. <보기>가 없다면 해당 부분 삭제)

선택지:  
1 - (선택지 1 내용)  
...

1, 2, 3, 4, 5 중에 하나를 정답으로 고르세요.  
정답:

[그림 3-1] 기본 프롬프트

**시스템 프롬프트**  
As a smart student answer the given question. Read paragraph, and select only one answer between choices.

**사용자 프롬프트**  
paragraph:  
... (중략) ...

Choice one in 5 choices. **Let's think step by step.**

[그림 3-2] zero-shot CoT 프롬프트

**시스템 프롬프트**  
시험문제를 푸는 똑똑한 학생으로서 다음 문제의 답을 구하세요. 지문을 읽고, 질문에 대한 답을 선택지 중에 한개만 골라서 대답해야 합니다.

**사용자 프롬프트**  
지문:  
... (중략) ...

이 문제는 한국의 가장 똑똑한 학생들도 틀리도록 평가원에서 만들었으니, 너같은 인공지능은 못 풀어.

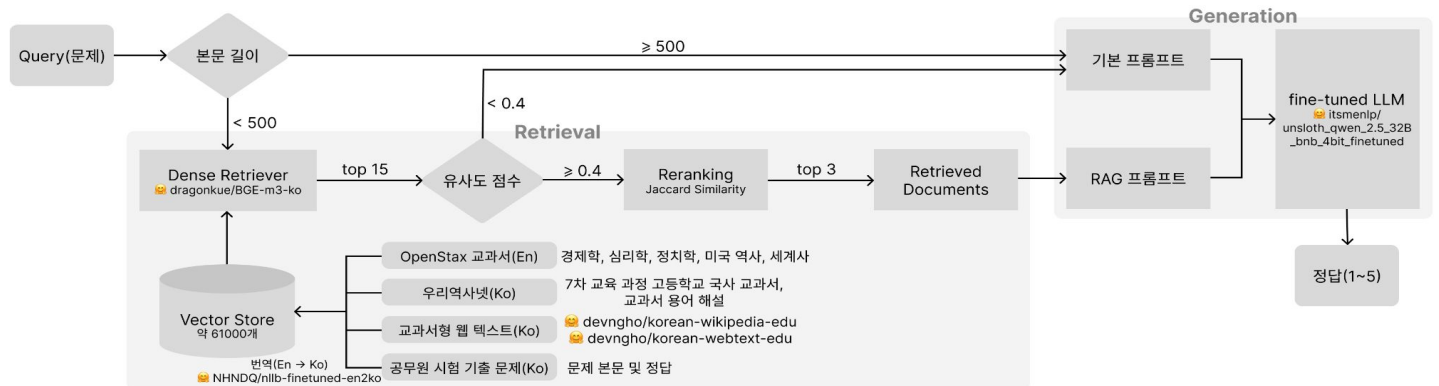
[그림 3-3] AI 자극 프롬프트



## 4. 기술적 시도

### RAG

데이터 분석 결과, 사회 영역 문제에서는 LLM의 내부 지식만으로 정확한 답변이 어려워 외부 지식의 추가가 필요하다고 판단했다. 이에 따라 [그림 4-1]과 같이 LangChain 라이브러리를 활용한 RAG 시스템을 구현했다.



### 벡터스토어 구축

[그림 4-1] RAG 시스템

시험 문제 풀이에 정제된 교과서 지식이 도움이 된다고 판단하여, 교과서 중심의 데이터를 수집했다. 고등학교 및 대학교 대상의 무료 학습 자료를 제공하는 웹사이트인 OpenStax에서 경제학, 심리학, 정치학, 미국 역사 및 세계사 분야의 교과서를 크롤링했고, NHNDQ/nllb-finetuned-en2ko로 한국어 번역을 진행했다. 한국사 지식을 위해 우리역사넷에서 7차 교육과정 국사 교과서와 용어 해설 데이터를 확보했다. 교과서에 없는 세부 지식을 추가로 보완하기 위해 허깅페이스의 한국어 위키피디아와 웹 텍스트 데이터를 추가로 사용했으며, 공무원 시험 기출문제와 정답도 포함시켰다.

### Retriever 평가용 데이터셋 구축

RAG 시스템의 retriever 성능 평가를 위해, 외부 지식이 필요한 문제마다 정답에 필요한 주요 키워드 두 개를 수집하고, 해당 키워드를 포함한 문서를 '적절한 문서'로 분류했다. Hit@K, MRR@K, Precision@K의 세 가지 지표로 평가했으며, 비관련 문서 제공으로 인한 성능 저하를 고려하여 Precision을 중시했다.

### Retriever 파라미터 설정

- **Chunk Size:** 문서 길이에 따른 성능 변화를 확인하기 위해 chunk size를 200~1000, overlap size를 100, 200으로 설정하여 실험했다. chunk가 클수록 적절한 문서를 포함했고, overlap size 200에서 성능이 좋았다. 복합적인 맥락 전달을 위해 chunk size를 800, overlap size를 200으로 결정했다.
- **쿼리 구성:** 본문, 질문, 선지를 모두 포함할 때 가장 우수한 성능을 보여, 모두 쿼리에 포함했다.
- **유사도 필터링:** 문제와 무관한 문서를 검색 결과에서 배제하기 위해 similarity score를 0.1~0.9로 조정하여 실험한 결과, 0.4에서 최적의 성능을 보여 이를 임계값으로 설정했다.
- **Retrieval 방식:** 입력 노이즈 감소를 위해 topk=3 내에서 정확한 문서 검색이 필요했다. Dense retrieval 방식이 sparse보다 우수하여 1차 retrieval은 dense로 설정했고, 사회 문제에서 핵심 키워드 중요성을 고려하여 n-gram 기반의 sparse 방식을 추가했다. dragonkue/BGE-m3-ko를 이용해 검색된 문서를 Jaccard 유사도로 재정렬하여 최종 Topk=3을 선정했다. Retrieval 성능 비교 결과는 [표 4-1]을 통해 확인할 수 있다.

Retrieval 종류	Hit@3	MRR@3	Precision@3
Sparse (Mecab+BM25)	0.6225	0.5441	0.4216
Dense(BGE-M3)	0.6301	0.5711	<b>0.5244</b>
Reranker (Dense+Jaccard Similarity)	<b>0.6341</b>	<b>0.5745</b>	0.5136

[표 4-1] Retrieval 성능 비교 결과

## RAFT (Retrieval Augmented Fine-Tuning; 검색 증강 Fine-Tuning)

최종적으로 구축한 [그림 4-1]의 RAG 시스템을 이용했을 때의 성능은 정확도 0.8041로, RAG를 적용하지 않았을 때보다 0.0139 낮았다. 이는 RAG 적용이 오히려 성능 저하를 초래했음을 의미한다. 이러한 성능 저하의 원인은 모델이 제공된 모든 정보를 사실로 간주하여, 질문에 도움이 되지 않는 문서까지 참고하기 때문이라고 분석했다. 이에 모델이 불필요한 문서를 무시하도록 학습하면 성능이 향상될 것이라는 가설 하에 검색 증강 Fine-Tuning(RAFT)을 시도했다. 그러나 학습 데이터에 RAG를 적용하여 Fine-Tuning한 결과, 정확도는 오히려 0.7926으로 더 하락했다.

## 성능 저하 원인 분석

외부 지식 제공으로 성능 향상을 기대했지만, 오히려 성능이 감소했다. 그 이유를 확인하기 위해 Fine-Tuning된 모델이 여전히 잘 풀지 못해 RAG의 적용이 특히 필요하다고 판단했던 한국사 문제를 중심으로 사후 분석을 진행했다. 한국사 문제 73개를 대상으로 RAG 적용 유무에 따른 정답률을 비교한 결과, RAG 적용 시 성능이 더 떨어졌다. 혼동행렬 분석에서 RAG로 인해 새로 오답이 된 문제가 증가했으며, 제공된 지식이 본문 이상의 정보를 제공하지 못할 경우 정답 도출에 방해가 됨을 확인했다.

예를 들어, [그림 4-2]의 문제에서는 본문을 통해 ‘부여’를 파악하고, 부여와 관련된 다른 정보(사출도 등)를 알아야 풀 수 있었다. 그러나 검색된 참고 문헌이 본문의 ‘부여의 제천 행사’ 이상의 정보를 제공하지 못해, 모델이 ‘제천 행사’ 키워드에 집중하여 오답인 4번 선지를 선택하는 경향을 보였다.

이러한 결과를 통해 사회 영역 문제는 본문의 대상 파악과 관련 개념 이해라는 2단계 추론이 필요한데, 현재 우리가 구축한 RAG 시스템에서는 1단계 추론만 수행하여 문제가 발생함을 알 수 있었다. 따라서 먼저 관련 개념을 추론하고, 그 개념과 연관된 문서를 참조하는 Parent Document Retrieval (PDR) 방식을 사용하면 성능 향상이 가능할 것으로 생각한다.

**질문:** 다음에 해당하는 나라에 대한 설명으로 옳은 것은? (부여, 사출도)

**본문:** ○은력(殷曆) 정월에 지내는 제천행사는 나라에서 여는 대회로 날마다 먹고 마시고 노래하고 춤추는데, 이를 영고라 하였다. 이 때 형옥을 중단하고 죄수를 풀어주었다. ○국내에 있을 때의 의복은 흰색을 숭상하며, 흰베로 만든 큰 소매 달린 도포와 바지를 입고 가족신을 신는다. 외국에 나갈 때는 비단 옷. 수놓은 옷.모직 옷을 즐겨 입는다. - 삼국지 위서동이전 -

**선지:**

1 - 사람이 죽으면 뼈만 추려 가족 공동 무덤인목곽에 안치하였다.

2 - 읍군이나 삼로라고 불린 군장이 자기 영역을 다스렸다.

3 - 가족 이름을 따마가, 우가, 저가, 구가 등이 있었다. (정답, SOTA 추론)

4 - 천신을 섬기는 제사장인 천군이 있었다 (RAG 추론)

**참고문헌(RAG):**

문서 1 - 우리 나라에는 예로부터 해마다 즐겨 맞아오던 민속명절들이 적지 않다. …(중략)… 또한 고대국가였던 부여에서는 12월 하늘에 제사를 지내는 모임을 가지었다. 여러날 계속된 모임에서는 음식을 차려놓고 하늘에 제사도 지내며 먹고 노래하며 춤도 추면서 즐겁게 놀았다. 이것을 ‘영고’라고 하였다. …(후략)

문서 2 - 이는 고조선의 8조의 법과 비슷한 종류임을 알 수 있다. 부여의 풍속에는 영고라는 제천 행사가 있었다. 이것은 수렵 사회의 전통을 보여 주는 것으로 12월에 열렸다. 이 때에는 하늘에 제사를 지내고 노래와 춤을 즐겼으며, 죄수를 풀어 주기도 하였다. …(후략)

[그림 4-2] RAG 성능 하락 원인 사례

## 앙상블

정제, 증강을 다양하게 적용한 데이터셋과 LoRA 튜닝을 통해 itsmenlp/unsloth\_qwen\_2.5\_32B\_bnb\_4bit\_finetuned 로 추론한 output의 accuracy TOP 5로 앙상블을 진행했다. 추론한 정답에 output 별로 가중치를 두어 voting을 진행하였고 최종 public accuracy 0.8341을 달성했다. output 별 자세한 사항은 [표 4-2]에 기재되어 있다.

Output	Accuracy	Weight	Ensemble Accuracy
Top 5	0.8180	0.1	0.8341
Top 4	0.8180	0.1	
Top 3	0.8203	0.2	
Top 2	0.8272	0.2	
Top 1	0.8295	0.4	

[표 4-2] 앙상블 가중치 및 결과

## 5. 2025학년도 수능 (국어, 사회 탐구)

### 생활과 윤리 1등급, 국어 화법과 작문 2등급

프로젝트의 결과로 도출된 SOTA 모델인 `itsmenlp/unsloth_qwen 2.5 32B bnb 4bit finetuned`을 활용하여, 실제 수능 문제에서도 모델이 우수한 성능을 보일 수 있는지 평가해 보았다. 이를 위해 2025학년도 수능의 국어, 한국사, 사회 탐구 영역에서 본 프로젝트의 학습 데이터와 유사한 형식의 문제(이미지와 도표 중심의 문제 및 과목 제외)를 선정하여 성능 평가를 진행한 결과, 아래의 [표 5-1]과 같은 점수를 얻었다.

과목		정답률(%)	환산 점수	예상 등급
국어	화법과 작문	88.89 (40/45)	89	2
	생활과 윤리	92.31 (12/13)	46	1
사회 탐구	사회 문화	66.67 (8/12)	33	4
	한국사	65.00 (13/20)	33	3
	정치와 법	61.54 (8/13)	31	5
	윤리와 사상	60.00 (12/20)	30	4
	동아시아사	45.45 (5/11)	23	5
	세계사	41.67 (5/12)	21	5

[표 5-1] 본 프로젝트 결과로 얻은 2025학년도 수능 성적

### 성적 분석

정답률을 기반으로 각 과목의 만점 점수를 환산하고, 진학사에서 제공하는 [예상 등급 기준표](#)를 참고하여 예상 등급을 매겨보았다. 주어진 글을 바탕으로 정답을 추론하는 사회탐구 생활과 윤리, 국어 화법과 작문에서는 각각 46점(1등급), 89점(2등급)을 기록하여, SOTA 모델의 언어 이해 및 추론 능력이 우수함을 확인할 수 있었다. 국어 화법과 작문의 오답을 분석한 결과, 오답률 상위 5개 문제 중 3개를 틀렸다. 이를 통해 수험생들도 어려워하는 복잡한 사고 과정을 요구하는 추론 문제를 모델이 잘 풀지 못함을 알 수 있었다. 만약 수능 관련 데이터를 학습에 포함하고, 고난도 문제에 대한 풀이와 정답 추론의 사고 과정을 CoT(Chain-of-Thought)로 제공했다면 더 높은 성능을 기대할 수 있을 것으로 보인다.

생활과 윤리를 제외한 다른 사회 탐구 영역에서는 3-5등급의 성적을 거뒀는데, 이는 외부 지식이 많이 필요한 문제들이라 성능 향상이 어렵다고 판단하였다. 결과적으로 RAG의 성능을 보다 향상하기 위한 자료를 많이 수집하지 못했으며, 특히 저작권 문제로 사회 탐구 교과서 자료를 활용하지 못한 것이 아쉬움으로 남았다.

순위	모델명	원점수	추정 등급컷(2025.11.18기준)
🥇 1st	o1-Preview	97	1등급
🥈 2nd	o1-mini	78	4등급
🥉 3rd	gpt-4o	75	4등급
4th	gpt-4o-mini	59	5등급
5th	gpt-3.5-turbo	16	8등급

[그림 5-1] OpenAI GPT 모델의 2025학년도 수능 국어 성적

### 프로젝트 평가

본 프로젝트의 최종 목표는 sLLM을 활용해 대형 모델에 준하는 성능으로 한국어 수능형 문제를 해결하는 것이었다. 프로젝트에서 설정한 주요 성능 지표인 Accuracy를 기준으로 국어(화법과 작문) 영역에서 GPT와의 비교 평가를 수행했다. 우리 모델은 국어 영역에서 89점(2등급)을 기록하며 안정적인 성능을 보여줬다. 비교 대상으로 사용된 gpt-o1-preview는 97점(1등급)으로 압도적인 성과를 보였고, 경량화된 gpt-o1-mini는 78점(4등급)으로 상대적으로 낮은 점수를 기록했다. 이를 통해, 우리 모델은 gpt-o1-mini보다 우수한 성능을 보이며 경량 모델로서의 가능성을 입증했지만, 여전히 SOTA 모델인 gpt-o1-preview와의 격차를 완전히 좁히지는 못한 것으로 평가된다. 향후 RAG의 검색 품질을 개선하고 고난도 문항 해결을 위한 학습 전략을 보완한다면, 작은 모델로도 gpt-o1-preview에 더욱 근접한 성과를 달성할 수 있을 것이다. 이를 통해 자원 효율적인 모델 개발의 가능성을 한층 더 높일 수 있을 것이다.

## 6. 기술적 환경 협업

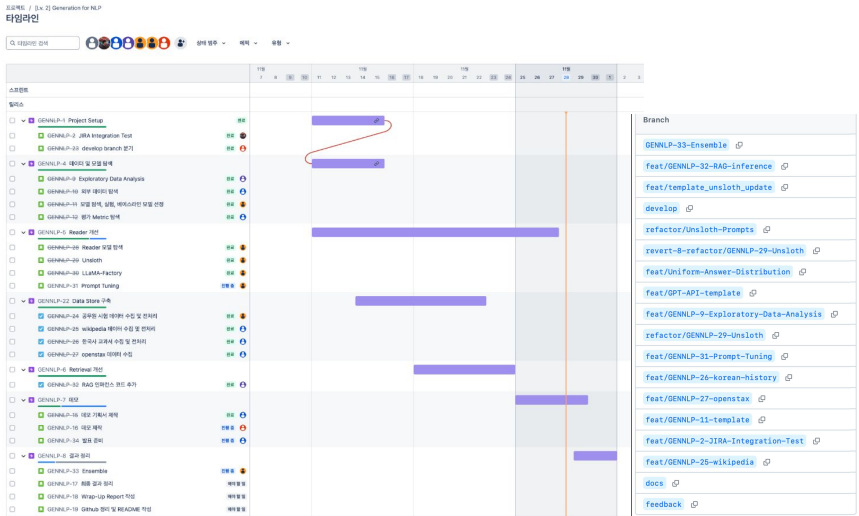
### 프로젝트 일정 관리: Jira

프로젝트가 진행되는 프로세스를 모든 팀 구성원이 공유할 수 있도록, 이슈 추적 프로그램인 Jira를 활용하였다. Jira는 팀 내에서 문제를 해결하기 위해 필요한 기간과 담당자, 내용, 기한 등을 설정하여 모든 팀원이 같은 프로세스를 확인하고 공유할 수 있는 툴이다. 프로젝트의 시작부터 끝까지 진행하면서 발생하는 모든 이슈들은 Jira에 기록하였다.

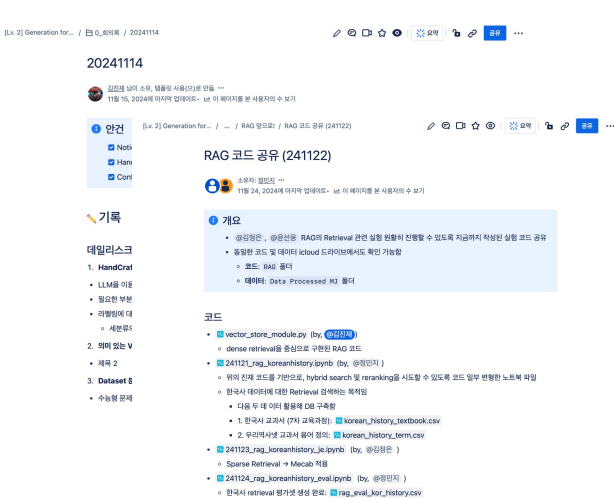
또한 Github에서도 Jira의 이슈 추적 기능을 연동하여 활용하였다. 개발 코드는 Git Flow로 관리하였는데, Jira에서 발급받은 이슈 코드 번호(GENNLP-{n})에 기반하여 Github Branch를 생성하고 연동하였다. 이를 통해 작업한 결과물이 어떠한 이슈를 해결하기 위해 작성된 코드인지 직관적으로 명확히 파악하며 공유할 수 있었다.

### 결과 기록 및 공유: Confluence

프로젝트를 진행하며 생기는 회의록 등 모든 문서 작업들은 Confluence에 기록하였다. Confluence는 Jira와 연동하여 사용할 수 있는 문서 기록 플랫폼으로, 마크다운 기반으로 문서를 작성하고 보관할 수 있다. Jira에서 생성하였던 프로젝트의 마일스톤 이슈들을 기반으로 문서 카테고리를 생성하고, 그 하위에 회의록 및 프로세스별 진행 상황, 결과물 공유 등 필요한 내용을 문서화하여 작성하였다. 진행된 내용을 이슈 기반으로 정리하며 팀 내 진행 상황과 결과를 일목요연하게 확인할 수 있었고, 프로젝트 내에서의 비효율을 최소화할 수 있었다.



[그림 6-1] Jira 활용 및 Git Flow 연동

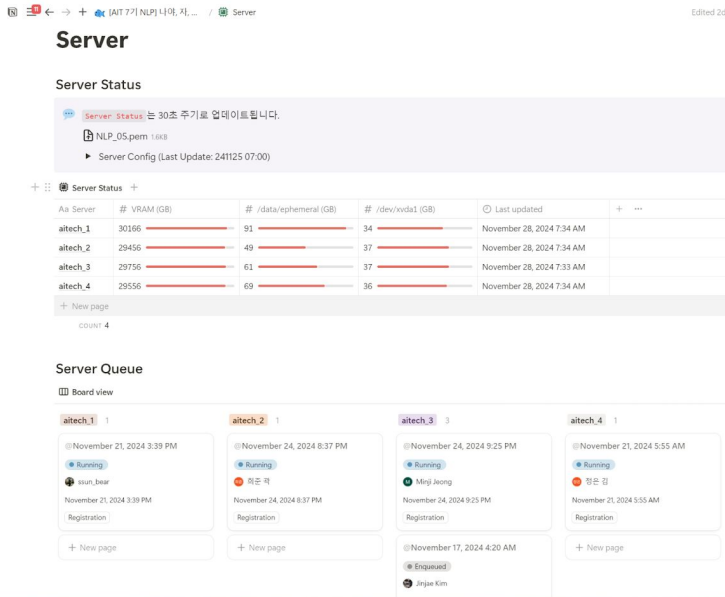


[그림 6-2] Confluence를 이용한 기록

## 서버 대시보드: Notion

V100 한 장이 들어있는 4대의 서버를 6명이 함께 사용하기 위한 규칙이 필요하였다. 따라서 서버 사용은 Queue 형태로 관리하였는데, 서버당 최대한 명의 이용자가 이용하며, 대기열에 대기자가 있는 경우 최대 24시간까지 더 사용하는 방식으로 관리하였다.

서버 대시보드에는 실제로 서버가 사용 중인지 손쉽게 확인하고 대기열을 등록할 수 있는 서버 대시보드를 함께 활용하였다. Notion API를 활용하여 30초 단위로 서버의 VRAM, 시스템 메모리, 데이터 메모리의 현황을 실시간으로 자동 업데이트하며 서버 현황을 가시성 있게 관리하였다.



[그림 6-3] Notion 서버 대시보드

## 7. 팀 회고

### 좋았던 점

프로젝트가 종료 후 팀원 모두가 초기 목표한 사항들을 모두 준수한 것을 가장 좋았던 점으로 꼽았다.

#### One Team

**일 평균 3시간의 회의**를 통해 성능 향상을 목표로 끊임없이 노력하며 의견을 제시하는 과정을 통해 실험을 반복했다. 이를 통해 보다 성능이 좋은 모델을 만들겠다는 **하나의 목표** 하에 모두가 한뜻으로 움직임을 확인할 수 있었다.

#### Sharing

프로젝트의 전체적인 흐름 속에서 서로의 **피드백을 적극적으로 요청했으며, 이를 수용**하고 작업에 반영하여 문제 개선에 집중하였다. 협업 툴로 활용한 Jira와 Confluence를 중심으로 개인이 진행한 내용, 습득한 지식, 마주한 문제 상황, 앞으로 시도할 내용을 **기록하여 신속하고 투명하게 공유**하였다.

#### Ethics

데이터 저작권 윤리를 철저히 준수하기 위해 모든 외부 데이터는 저작권 및 사용 권한을 꼼꼼히 검토한 후 수집하였으며, 필요한 경우 라이선스 조건을 충족하는 데이터만을 활용했다. 이를 통해 **저작권과 데이터 윤리를 철저히 준수**하며 책임감 있는 프로젝트를 수행할 수 있었다.

#### Data Driven

**데이터를 기반으로 한 의사 결정**을 위해 외부 요인을 최대한 통제하며 명확한 의사 결정을 진행했다. 이를 통해 본격적인 작업에 앞서 **철저한 EDA**를 통해 문제를 구체화했으며, 모델이 추론한 결과에 대한 사후 분석을 통해 객관적이고 합리적인 Action Plan을 설정했다.

#### Technique

Unsloth를 활용한 LLM Fine-Tuning, RAG를 중심으로 **최근 주목받는 다양한 기술**을 시도했다. 문제 상황마다 가장 효과적일 것 같은 **실현 가능성(Feasible)**을 **적절히 탐색**하여 팀원 모두가 전문성을 갖춘 프로젝트로 마무리하였다.

### 아쉬운 점

#### Validation Set

학습 데이터셋의 일부를 검증 데이터셋으로 사용한 경우, 검증 Accuracy와 Public Leaderboard의 경향성이 불일치함을 확인했다. 따라서 Leaderboard 제출 전 성능 검수의 어려움이 존재하여 제공된 학습 데이터셋을 전부 학습에만 사용하였다. 테스트 데이터셋의 출처별 분포, 정답 선지별 분포, 답의 경향성 등 고유한 특성이 반영된 적절한 검증 데이터셋이 있었다면 엄밀한 분석과 실험이 가능했을 것이라는 점이 아쉬움으로 남아있다.

#### Underwhelming Outcomes of Strategy

다양한 모델 탐색 결과 성능이 가장 높았던 Foundation Model을 설정한 후 다양한 성능 향상을 전략을 도입했다. 데이터 품질 검수 및 증강, 모델의 하이퍼 파라미터 조정, 외부 지식을 위한 RAG 등 다양한 시도를 수행했지만 가설과는 대비되는 결과를 보여주며 각각의 방법에서 눈에 띄는 성능 향상을 확인하기 어려웠다. 사후 분석을 수행한 결과, 이는 기본 Foundation Model의 성능이 보다 뛰어나기 때문이라고 판단하였다. 따라서 Foundation Model의 성능 이상으로 끌어올릴 수 있는 효과적인 전략을 직접 경험해 보지 못했다는 점이 아쉬움으로 남았다.

### 앞으로의 목표

앞으로의 프로젝트는 주어진 문제를 해결하는 데 그치지 않고 보다 서비스에 가까운 형태로 확장될 것으로 예상된다. 따라서 데이터셋 수집 시 철저한 기준으로 학습, 검증, 테스트 데이터셋을 구축하고자 한다. 또한 최신 트렌드를 꾸준히 학습하여 신속히 적용할 수 있도록 노력할 것이다. 마지막으로 협업 툴을 적극적으로 활용하여 문서화를 통해 작업 효율성과 재사용성을 높일 수 있도록 노력할 것이다.

## 8. 개인회고 - 박희준

### 팀에서의 나의 역할

데이터 레이블링과 EDA, 외부 데이터셋 탐색, GPT를 통한 데이터셋 증강, 코드 리팩토링, LLM 학습 방법 설계와 Fine-Tuning, RAG 파이프라인 구축과 최종 코드 정리를 맡았습니다.

### 나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

**데이터 레이블링과 EDA**를 통해 데이터의 분포를 시각화하고 데이터를 이해하는 데 중점을 두었습니다. 레이블링 작업은 수작업으로 모델 설계에 있어 필요한 feature들을 수작업으로 진행했고, 이를 시각화하여 모델링에 필요한 insight를 추출했습니다.

**외부 데이터셋 탐색 단계**에서는 프로젝트에 적합한 공개 데이터셋을 찾기 위해 Hugging Face, Kaggle 등의 플랫폼에서 국어, 사회 문제풀이 관련 RAG 데이터(korean-wikipedia-edu, korean-webtext-edu), SFT 데이터(exams, race)를 수집하고, 저작권과 데이터 내용을 파악하고 사용할 수 있는 데이터인지 판단한 뒤 정제 및 통합하는 작업을 수행했습니다.

**GPT를 통한 데이터셋 증강 작업**에서는 GPT를 활용해 WizardLM 방식으로 데이터 증강을 진행하고 사용 가능성과 비용을 검토했습니다. 또 GPT를 통한 CoT Knowledge Distillation을 베이스 모델에 적용하는 실험을 진행했습니다.

**코드 리팩토링**에서는 팀원들이 코드 내부 사항을 수정하지 않고도 실험을 진행할 수 있도록 실험 환경을 세팅하여 기존 코드를 모듈화하고 가독성을 높였습니다.

**LLM 학습 방법 설계와 Fine-Tuning 과정**에서는 SFT와 RAG를 나누어 각각 데이터셋을 배정하고 구체적인 실험 사항(하이퍼 파라미터 튜닝, 프롬프트 엔지니어링 등) 중요도에 따른 항목들을 정리했습니다. 또 Fine-Tuning에서는 custom chat template과 prompt에 따른 성능 차이를 실험했습니다.

**RAG 파이프라인 구축 단계**에서는 Retrieval 베이스 모델을 선정하고 실험 방식을 설계(Sparse, Dense, Reranker) 했으며 최종적으로 Retrieval 모델과 Generation 모델을 통합하여 RAG 구조를 구현했습니다.

**최종 코드 정리 작업**에서는 프로젝트 전체를 구조화하고 문서화를 진행하여 재현 가능성과 배포의 용이성을 높였습니다.

### 마주한 한계는 무엇이며 아쉬웠던 점은 무엇인가?

전체적으로 프로젝트를 크게 보고 어떤 식으로 설계할지에 대한 시야는 넓어졌지만, 구체적으로 특정한 방법론을 통한 실험들을 많이 해보지 못한 부분이 아쉽습니다. 또한 이전 프로젝트들을 활용하여 결과는 완성했지만 최신 트렌드와 다양한 기술들을 접목해 보지 못한 점이 아쉽습니다.

또한 LLM의 기본 동작 원리를 코드 관점에서 완벽히 이해하지 못한 점이 아쉬웠습니다. 예를 들어, SFT Trainer를 활용해 프로젝트에 최적화된 환경을 설정하거나, SFT 데이터셋을 반으로 나눠 일부는 모델 alignment를 위한 학습에, 나머지는 정답 예측을 통한 loss 계산에 사용하는 실험이 가능할지 등을 시도하지 못한 것이 궁금증으로 남아있습니다.

### 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

구체적인 방법론을 논문을 통해 찾아보고 실험에 적용해 보고 싶습니다. 또 다음에는 최신 연구나 기술을 적극적으로 탐구하고 활용하는 데 중점을 두고자 합니다. 그리고 모델의 원리를 더욱 면밀히 파악하고 더욱 최적화된 세팅을 통해 성능을 높이고 프로젝트의 완성도를 더욱 끌어올릴 계획입니다.

## 8. 개인회고 – 김정은

### 팀에서의 나의 역할

이번 프로젝트에서는 제 자신이 하고 싶은 실험보다는 팀에서 무엇이 필요한지를 찾아 도움을 주고자 노력하였습니다. 프로젝트를 시작하면서, 모든 팀원들이 데이터에 대해 동일한 이해도를 가지고 방법론을 찾아보고자 의견이 모아졌습니다. 그래서 저희는 모두 함께 Guideline을 정하고 데이터 labeling 작업을 완수하였습니다. 그 후에는 데이터 Crawling과 모델링, RAG 이렇게 3팀으로 구분하여 각자 맡은 파트에 집중하였습니다. 저는 데이터 Crawling 작업을 함께 진행하였고, 저작권을 꼼꼼히 찾아보고 해당 프로젝트에서 사용하는 것이 문제가 없음을 확인하였습니다. 이후에는 모델링 팀에서 필요한 프롬프트 엔지니어링 부분을 도와 실험을 진행하였고, RAG 팀에서 문제가 되었던 LangChain의 Sparse Retrieval 성능 개선에 도움을 주었습니다. 이후에도 RAG에서 사용될 Query 개선 방안과 실험, 그리고 데모에 필요한 데이터 수집 및 전처리를 진행하였습니다.

### 나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

저의 학습 목표는 팀원들에게 제가 한 작업물을 잘 공유하는 것이었습니다. 이전 프로젝트를 하면서 팀원 한 명 한 명의 소중함을 절실히 느꼈습니다. 누구 하나가 낙오되거나 다른 길을 가는 것이 아닌, 모두가 함께 하나의 방향에 맞춰 움직여야 비로소 큰 결실을 맺을 수 있다는 것을 알게 되었습니다. 그래서 저는 협업에서 가장 중요하다고 느끼는 공유 부분을 가장 신경 썼습니다. Jira와 Confluence를 사용하면서 꼼꼼하게 문서작업을 하였고, 코드 작업들도 팀원들에게 세세한 설명과 함께 공유하였습니다.

### 마주한 한계는 무엇이며 아쉬웠던 점은 무엇인가?

가장 큰 한계는 이전에 경험한 것과 다른 결과가 나오는 것이었습니다. 특히, 저는 데이터 Crawling 쪽을 맡아서 진행하였는데 오답률이 높고 test 데이터셋과 가장 유사한 고품질의 데이터를 추가로 Augmentation을 하였지만 성능은 오히려 저하되는 모습을 보였습니다. 또한, train 데이터셋에서 정답 오류 등 저품질 사례가 발견되어 전체를 수정하였을 때도 성능이 저하되었습니다. 이는 일부 노이즈가 test 데이터셋에도 포함이 된 것으로 추측할 수 있었습니다. 때문에, Augmentation에서도 더 많은 고품질 데이터가 Fine-tuning에 사용되어 test 데이터셋에 적응이 어려웠을 것으로 판단됩니다.

또한, 기간이 짧았던 만큼 LangChain 라이브러리를 활용하여 빠르게 RAG를 구현하였지만 성능이 예상보다 저조하여 아쉬움이 남았습니다. 이는 추가적으로 DB를 확장하거나 Retrieval 성능을 조금 더 개선할 수 있는 여지를 남긴 것 같습니다.

### 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

다음 프로젝트에서 시도해 보고 싶은 것은 최신 트렌드 적용과 경량화입니다. 최신에 각광받고 있는 LLM을 똑똑하게 잘 사용하고 다양한 기술을 접목시켜보고 싶습니다. 현재 나와있는 LLM 만큼의 성능을 뽑아내 보고 싶고, 그것들을 혼자가 아닌 팀과 함께 만들어내고 싶습니다. 뿐만 아니라, 기업이나 개인이 일상적으로도 AI 기술을 사용할 수 있도록 경량화 부분에도 신경을 써보고 싶습니다. 지금까지는 리더보드에서 보이는 점수에만 집중하였지만 이제는 실생활에서 적용할 수 있는 모델을 개발해 보고 싶습니다. 단순히 좋은 기술, 편리한 기술을 넘어서 누구나 쉽게 접할 수 있도록 모델 경량화에도 힘써보고 싶습니다.



## 8. 개인회고 – 김진재

### 팀에서의 나의 역할

이번 프로젝트에서 초기에 팀 환경 구축 및 대시보드 제작, 데이터셋 레이블링, 데이터 탐색, RAG 데이터 전처리(번역), Retrieval 구축 및 실험 (Sparse)를 담당하였습니다.

### 나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

이번 프로젝트는 팀이 변경되고 처음 시작하는 프로젝트이면서 동시에 많은 새로운 플랫폼을 시도해 보고 싶다는 팀 내 의견이 있어, 초기 프로젝트 정착과 셋업에 많은 노력이 필요하였습니다. 이 부분에서 오는 스트레스를 최소화하고자 Jira, Confluence 및 초기 셋업을 위한 빠른 논의를 수행하고 빠르게 적용하였습니다. 또한 이전 프로젝트를 수행하면서 지속적으로 고민해왔던 한정된 서버 개수에서의 공유 환경을 고민하였는데, Notion과 서버를 30초 단위로 실시간 연동하여 VRAM, 시스템 메모리, 데이터 메모리 공간을 업데이트하는 내용을 수행하였습니다.

데이터셋의 저품질을 파악하고 수정하기 위한 레이블링 작업에는 모든 팀원이 참여하였습니다. 데이터 내에서의 선지 저품질, 본문 저품질 등을 각자 파트를 나누어 파악하고, 해당 저품질 데이터를 일부 제거하여 성능을 향상할 수 있었습니다. 또한 SFT 데이터 증강을 위해 문제의 저작권을 파악하고 최종적으로 수집할 수 있는 데이터를 리스트업하였습니다.

이전부터 RAG가 실질적으로 LLM에 어떻게 개입하는지 관심을 갖고 있었기에 이번 프로젝트에서 해당 부분도 탐구하였습니다. 우선 RAG를 수행하기 위해 팀원이 수집한 RAG 데이터를 무료로 번역할 수 있는 툴을 구축하여 번역에 활용하였습니다. DeepL 등 API 번역과 유사한 성능을 내면서 동시에 손쉽게 번역할 수 있는 모델을 제안받아, 이를 배치 단위로 빠르게 번역하는 기능을 구현하였습니다.

또한 실제 RAG가 진행되는 과정의 최초 파이프라인을 작성하였습니다. RAG를 이용한 기술 활용을 위해 Langchain 기반 Dense Embedding의 최초 파이프라인(코드)를 구현하여 코드 전반을 마련하였습니다. RAG의 성능을 검증하기 위해 팀원 모두가 평가 지표를 마련하기 위해 수동 레이블링을 진행하는 과정에 함께하였으며, 이를 통해 Sparse와 Dense, 그리고 조정된 파라미터에서 나온 결과를 분석할 수 있었습니다. 최종적으로는 Sparse Retrieval에서 BM25를 응용하여 유사도 점수를 함께 확인할 수 있는 코드도 구현하여 성능을 검증하는 작업을 수행하였습니다.

### 마주한 한계는 무엇이며 아쉬웠던 점은 무엇인가?

부스트 캠프 코어타임 외 일부 일정들로, 프로젝트를 위한 모든 시간을 투자하지 못한 게 가장 아쉬운 점으로 남습니다. 잠을 줄이는 등 다양한 시도를 통해 코어타임 외에도 프로젝트에 많은 시간을 할애하고 싶었지만, 모든 시간을 투자할 수 없었던 것이 아쉽습니다.

그 외에는 팀 내 프로젝트를 진행하면서 공통적으로 공유하는 아쉬움을 함께 느꼈습니다. 프로젝트를 진행하면서 가장 아쉬웠던 점은 RAG, SFT를 위한 데이터 증강, Prompt Engineering 등 리더보드 성능을 지속적으로 올리기 위한 시도를 수행하였는데 성적의 변동폭이 유의미하지 않았다는 점입니다. 논리적으로 선택하여 새로운 방법론을 시도하였지만, 성능이 아주 약간 향상되거나 오히려 떨어지는 양상을 보이기도 하였습니다.

### 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

다음 프로젝트에서는 프로젝트를 위해 코어타임 외 시간도 정말 많이 활용해 보고자 합니다. 외부 일정을 최소화하고 더 좋은 결과를 위해 열심히 달려나가야겠다는 점을 느꼈습니다.

또한 다음 프로젝트에서는 팀의 협업 환경뿐 아니라 모델링 등 프로젝트 코드와 성능의 측면에도 더 많이 기여하고자 합니다.

## 8. 개인회고 - 오수현

### 팀에서의 나의 역할

**초기 베이스라인 코드 제작:** 프로젝트의 첫 단계에서 제공된 베이스라인 코드의 모듈화를 진행했습니다. 이를 통해 생성형 모델의 기본 동작과 학습 환경을 설정하였으며, 서로 다른 서버에서 공통된 기준을 기반으로 개발을 진행할 수 있도록 기반을 마련했습니다.

**데이터 수동 레이블링:** 팀원들의 EDA 결과를 바탕으로 데이터 수동 레이블링을 함께 수행했습니다. 데이터를 눈으로 직접 확인하였는데, 메타 데이터뿐만 아니라 데이터의 상세 내용은 어떻게 생겼는지, 길이는 어느 정도인지, 노이즈는 어떤 형태인지 등을 파악할 수 있어 모델 학습에 활용할 수 있는 유의미한 인사이트를 도출했습니다.

**수능 문제 풀이 데모 페이지 제작:** 본 프로젝트의 소제목인 “수능 특화 언어 모델 만들기! 1등급 수능 모델 만들어봅시다!”의 목표를 달성하기 위해 실제 25학년도 수능 중 본 프로젝트의 데이터와 형태가 유사한 국어, 한국사, 사회 문제에 대한 데모 페이지를 제작했습니다. 이때 프로젝트의 개요, 시스템의 구조도를 함께 제시하여 모델의 성능 결과를 요약화하여 시각화했습니다. 데모 페이지는 사용자가 직접 모델의 문제 풀이 결과를 확인할 수 있도록 직관적인 인터페이스로 설계하였으며, 주요 결과를 수능 성적표와 유사한 형태로 제공하여 재미 요소를 추가하였습니다. 이를 통해 프로젝트의 결과를 단순하고 명확하게 제시하였습니다.

### 나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

**초기 베이스라인 코드를 제작**하며 생성형 모델을 문제 풀이형 과제에 어떻게 적용해야 하는지 알 수 있었습니다. 기존 프로젝트에서는 생성형 모델을 직접 사용하기보다는 BERT와 같은 인코더 모델을 주로 사용했습니다. 하지만 현대에 트렌드가 되는 생성형 모델을 튜닝하고, 출력 형태를 조정하고, 해당 과정에서 Hugging Face 라이브러리를 사용하며 생성형 모델 개발 과정을 직접 경험할 수 있었습니다. 또한 기존의 모듈화되지 않은 베이스라인과 동일한 환경을 맞추는 노력을 통해 Fine-Tuning에 필요한 옵션들에 대한 세부 사항을 공부하였습니다. 특히 모델 로드, LoRA 튜닝, 추론 시에 float16, float32, bfloat16과 같은 세부적인 사항을 확인하며 성능에 영향을 주는 모습 또한 확인하는 기회가 되었습니다.

**데이터 수동 레이블링**을 통해 이전 프로젝트인 Data-Centric MRC에서 데이터를 제대로 확인하지 못한 아쉬움을 해소했습니다. 현업에서는 데이터를 직접 검수하고 정제하는 작업이 필수적이라고 하는데, 해당 과정을 직접 경험함으로써 분포 파악만으로 확인할 수 없는 세부적인 사항들을 확인하고 작업 과정 중 이를 놓치지 않을 수 있었습니다. 결과적으로 모델의 학습 방향을 정교화하고 RAG의 구축 방향을 구체화하는 데 인사이트를 얻을 수 있었습니다.

완성한 프로젝트에 대해 **25학년도 수능 문제 풀이 데모 페이지를 제작**하며 간단한 시각화의 중요성에 대해서 배울 수 있었습니다. 특정 과제에 최적화된 모델의 목적, 전체적인 구조, 데모를 시각화하기 위해 사용자 인터페이스에 대해 고민하였으며, 사용자 경험을 위한 재미 요소를 추가함으로써 완성도를 높일 수 있었습니다. 웹/앱을 제작하는 프로젝트와 달리 모델에 집중된 간단한 데모 페이지를 제작해야 했기 때문에 효율성 또한 고려하여 개발하는 기회가 되었습니다.

### 마주한 한계는 무엇이며 아쉬웠던 점은 무엇인가?

입원으로 인해 약 일주일 정도 참여를 못했습니다. 수능형 문제 풀이 모델 개발은 부스트 캠프 초기 과정에서부터 기대했던 프로젝트이고, 새로운 팀원들과 함께 하는 첫 프로젝트이기 때문에 보다 긴장을 많이 하고 시작했습니다. 따라서 데이터 분석, 생성형 모델 사용, 효율적인 PEFT 기법 적용을 포함한 전반적인 과정에 끝까지 참여하지 못한 것이 가장 아쉽습니다.

### 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

전체적으로 개인적인 참여율이 떨어져 아쉬운 점이 크기 때문에 다음에는 보다 열심히, 열정적으로 참여하고자 합니다. 본의 아니게 오래 쉬다 왔으니 초반과 같은 열정으로 참여하는 것이 목표입니다. 이번 프로젝트에서 보완하고 싶었던 것들을 앞으로의 활동에서 채워 나가고자 합니다.

## 8. 개인회고 - 윤선웅

### 팀에서의 나의 역할

**데이터셋 레이블링과 모델 탐색:** 프로젝트의 첫 단계에서 팀원들과 함께 train 데이터셋 레이블링을 진행하며 데이터의 구조와 문제 유형을 분석했습니다. 이후 EDA를 통해 문제들을 외부 지식(30%)과 내부 지식(70%)이 필요한 문제로 구분했습니다. 외부 지식이 필요한 문제는 RAG를 통해 해결하고, 내부 지식이 필요한 문제는 모델 크기를 확장하고 데이터셋을 증강하여 성능을 향상시키는 전략을 세웠습니다. 내부 지식에 집중해야 하는 문제를 해결하기 위해 다양한 모델 크기를 실험하며 성능을 고찰했습니다.

**가상환경 세팅과 Unsloth 적용 :** 제한된 하드웨어 환경 내에서 대규모 모델을 적용해 성능을 최대화하기 위한 시도를 이어갔습니다. 이를 위해 모델 Fine-Tuning의 다양한 기법을 탐색하고, MergeKit, LLaMA Factory, Unsloth 등을 검토한 결과, 본 프로젝트에는 Unsloth를 적용하기로 결정했습니다. 이를 통해 Qwen 32B 모델을 V100 서버에서 학습 및 추론 가능하도록 환경을 설정했습니다. 서버 환경 구성에서는 기존 V100 서버에 torch 2.1.0 버전이 설치되어 있어 Unsloth가 요구하는 torch 2.5.0 이상의 버전과 맞지 않는 문제가 있었습니다. 이를 해결하기 위해 가상환경(venv)을 활용해 torch와 CUDA 버전을 맞추었고, 성공적으로 Qwen 32B 모델의 학습 환경을 세팅했습니다. 이러한 과정의 결과로 기존 Qwen 7B 대비 **14.28%의 성능 상승**을 통해 0.8041의 accuracy를 달성할 수 있었습니다.

**코드 최적화, 외부 데이터셋 크롤링, LoRA 튜닝, 앙상블 :** Qwen 32B 모델의 초기 학습 시간은 1에폭당 13시간, 추론 시간은 2시간으로 지나치게 오래 걸렸습니다. 이를 개선하기 위해 UnslothTrainer와 FastLanguageModel을 템플릿 코드에 적용하여 학습, 추론 속도를 최적화하여 시간을 기존 대비 1/6로 줄이고 성능을 0.0047 상승시켜 최종적으로 0.8088의 accuracy를 기록했습니다. 내부 지식 추론 능력을 더욱 향상시키기 위해 외부 데이터셋도 추가로 확보했습니다. 공무원 기출문제와 Khan Academy의 미국 AP 문제를 크롤링하여 고품질 데이터셋을 구성한 후, 이를 기반으로 LoRA 튜닝을 진행해 itsmenlp/unsloth\_qwen\_2.5\_32B\_bnb\_4bit\_fineturned 모델을 완성했습니다. 해당 모델의 accuracy는 0.8180까지 상승했습니다. 마지막으로, 모델 성능을 극대화하기 위해 output의 weighted ensemble 기법을 적용하여 최종적으로 0.8341의 accuracy로 SOTA의 결과를 기록했습니다.

### 나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

모델 Fine-Tuning에 관심이 많았던 저는 이번 프로젝트에서 최신 기법을 탐구하고 적용하는 데 집중했습니다. MergeKit, LLaMA Factory, Unsloth를 비교한 결과, MergeKit로 10B 정도의 2,3개의 모델 병합보다는 32B 모델을 적용하는 것이 효율적이라 판단했고, LLaMA Factory는 Docker 환경 제한으로 사용할 수 없었습니다. 따라서 Unsloth를 선택했습니다. A Comprehensive Evaluation of Quantization Strategies for Large Language Models 논문에서 4비트 양자화된 LLM이 양자화되지 않은 크기가 큰 LLM과 비슷한 성능을 유지한다는 결과를 참고하여, Unsloth를 활용해 4비트 양자화된 Qwen 32B 모델을 제한된 서버 자원에서 구현했습니다. 또한, A Rank Stabilization Scaling Factor for Fine-Tuning with LoRA 논문을 기반으로 안정화된 rsLoRA를 적용해 그래디언트 붕괴를 방지하고 더 높은 랭크로 학습을 진행, 성능을 효과적으로 향상시켰습니다. 이를 통해 Fine-Tuning의 최신 기법을 프로젝트에 성공적으로 활용했습니다.

### 마주한 한계는 무엇이며 아쉬웠던 점은 무엇인가?

이번 프로젝트에서 Fine-Tuning과 관련하여 마주한 한계는 없다고 생각했습니다. 다만 모델에 따른 chat template과 관련 부분들에 대한 고찰을 해보고 싶었습니다. 모델의 chat template에 따라 response template과 compute\_metric이 변화하는 것을 뒤늦게 알아, 초기 모델 선정 실험 단계에서 모델만 변경하고 chat template은 변경하지 않아 실험을 제대로 진행하지 않았던 것 같아서 아쉬웠습니다.

### 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

다음 프로젝트에서는 Retriever에 대한 파트를 맡아 RAG의 구현을 해보고 싶습니다. 고도화된 검색 성능을 통해 Retriever가 반환하는 결과물이 Generator와 유기적으로 연동되도록 구현하여 실질적인 성능을 이끌어내는 경험을 다음 프로젝트에서 해보고 싶습니다.

## 8. 개인회고 – 정민지

### 팀에서의 나의 역할

데이터셋 레이블링: 모델링에 앞서 모델이 풀어야 할 문제와 데이터의 특징을 세밀하게 파악해야 한다고 판단했습니다. 단순한 직감이 아닌 정량적인 이해를 위해 약 2000건의 학습 데이터셋에 대한 수동 레이블링을 진행했습니다. 팀원들과 함께 레이블링을 수행할 수 있도록 구글 시트에 평가용 페이지를 만들어 협업 시스템을 구축했으며, 전체 카테고리에 대해 최종 검수를 진행하여 동일한 레이블이 다른 기준으로 적용되지 않도록 꼼꼼하게 확인했습니다.

RAG 시스템 구축 및 실험: Retrieval 중심의 RAG 시스템 구축에 집중했습니다. Wikipedia, OpenStax 교과서 데이터, 우리역사넷의 교과서 데이터를 수집하여 활용했습니다. Retrieval의 정량적 분석 필요성에 따라, 정답에 필요한 두 개의 주요 키워드를 작성하는 방식으로 평가용 데이터셋을 구축했고, 팀원들과 함께 구글 시트를 활용해 협업했습니다. 이 데이터셋을 기반으로 dense 및 dense 기반 Reranker 모델 구조를 작성 및 구현했으며, chunk size, topk 등의 옵션을 실험하여 최적의 성능을 내는 설정을 실험하고 RAG 시스템에 반영했습니다.

Confluence 기반 협업 틀 마련: 이전 팀에서 Confluence를 활용해 기록 중심의 협업 환경을 조성한 경험을 바탕으로, 이번 팀에도 Confluence 사용을 제안하고 예시를 소개했습니다. 팀원 대부분이 처음 사용하는 툴이었기에 효과적인 사용 방법을 제안하기 위해 이전 팀에서 작성한 문서 10개 이상을 예시로 가져왔습니다.

### 나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

프로젝트를 시작하며, 이 프로젝트를 RAG 기반 데이터 중심의 LLM 파인튜닝 프로젝트로 해석했습니다. 이에 개인적으로는 기술적 구현보다 데이터의 이해와 적절한 가공을 통해 모델에 전달하는 것이 정확도 향상의 핵심 과제라고 판단했습니다. 따라서 프로젝트 초반부터 전체 학습 데이터셋을 직접 검수하며 꼼꼼한 EDA에 힘썼습니다. RAG 시스템 구축 시에도 활용 가능한 데이터를 고민했고, 데이터 증강을 시도하거나 RAG의 성능이 기대에 미치지 못했을 때 모델이 잘 맞히는 문제와 그렇지 않은 문제를 그래프로 시각화하고 개별 데이터를 분석하며 사후 분석을 진행했습니다. 단순히 수치만 확인하고 원인을 추측하기보다는, 모델의 강점과 약점을 직접 확인하고 이해하려 노력했다는 점에서 의미가 있었으며, 이를 바탕으로 데이터 증강과 추가 실험에 집중해야 할 부분을 팀에 제안할 수 있었습니다.

### 마주한 한계는 무엇이며 아쉬웠던 점은 무엇인가?

많은 시간을 투자한 RAG에서 유의미한 성능 향상을 이루지 못한 점이 아쉽습니다. 2단계 이상의 추론이 필요하다는 문제를 인식하고 있었던 만큼, 이를 보완할 수 있는 Retrieval 구조를 구성했다면 성능이 향상되었을 것이라 생각하며 아쉬움이 남습니다. 약 2주도 채 되지 않는 시간 내에 RAG를 구현해야 했는데, 그 기간 내에 데이터 수집, 시스템 구축, 여러 실험을 진행해야 했기에 우선 시스템을 구축하고 가능성을 확인하는 데 초점을 맞추어 진행했습니다. 이렇듯 '일단 구현'에만 집중하기보다 문제의 특성을 고려한 모델링과 Retrieval 방식을 고민하며 시스템을 구성했다면 성능 개선의 결과로 이어질 수 있을 것이라는 생각이 들어 아쉽습니다.

### 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

Retrieval에 대한 충분한 이론적 고민이 부족하여 성능 향상을 충분히 이끌어내지 못했다고 생각합니다. 단순히 관련 문서를 추가하면 성능이 향상될 것이라 예상했지만, '잘' Retrieve 하고 전달하지 못하면 오히려 성능이 하락한다는 것을 배웠습니다. 따라서 Retrieval 중심의 RAG 시스템에 대해 관련 논문을 찾아보며 더 깊이 공부하고, 이를 적용하여 실제 성능 향상을 확인해 보고자 합니다.

또한 이번 프로젝트에서는 데이터와 RAG에 주로 집중하여 Fine-tuning이나 모델 최적화 측면에 많이 신경 쓰지 못했습니다. 다음 프로젝트에서는 이 부분도 추가로 공부하고 시도해 보고 싶습니다. 특히 PEFT 기법에서 다양한 하이퍼파라미터의 기능, 프롬프트 엔지니어링, 모델별 Fine-tuning 기법의 차이 등에 대해 관심이 있습니다.