SK네트웍스 **Family AI** 과정 **13**기

데이터 전처리 학습된 인공지능 모델

산출물 단계	데이터 전처리
평가 산출물	학습된 인공지능 모델
제출 일자	2025.08.08
깃허브 경로	https://github.com/SKNETWORKS-FAMILY-AICAMP/SKN13-FINAL-3TEAM.git
작성 팀원	강지윤, 기원준, 우민규, 전진혁, 최호연

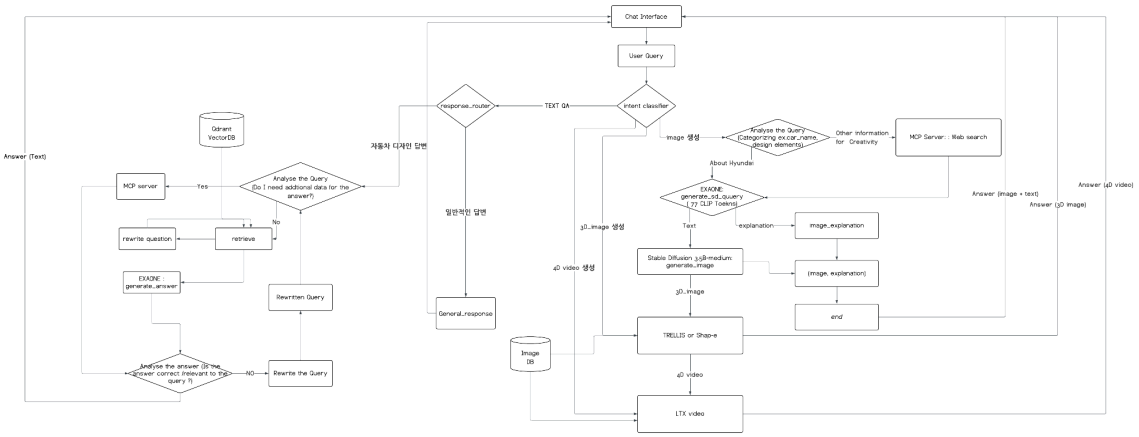
1. 모델 목적: 디자이너의 요구에 맞는 관련 지식 대화 및 이미지, 3D, 4D 로 이어지는 LangGraph 파이프라인의 메인 모델

2. 모델 아키텍처 설계

- 선정 모델: EXAONE 4.0 1.2B
- 아키텍처 개요:

계층	구성 요소	역할
Embedding	입력 문장을 벡터화	• BBPE tokenizer 사용 • Vocab size: 102,400 • 한국어, 영어, 소량의 다국어 토큰으로 구성
Encoder (Transformer block)	의미 표현 학습	• Global Attention • GQA(Grouped Query Attention) 적용 • QK-Reorder-LN
Classification Head (Final Layer)	토큰 예측	• 다음 토큰 확률 계산을 위한 Softmax 함수 사용

- 전체 아키텍처 시각화: intent classifier 역할



- 설계 근거
  1) global attention 을 기반으로 한 1.2B의 경량 모델
  2) 강화 학습의 새로운 패러다임인 AGAPO 가 적용되어 사용자 query 분석 능력이 뛰어남
  3) Agentic Tool Use 부문에서 동일 사이즈의 다른 오픈 모델보다 우수한 성능을 보임

3. 모델 학습 요약

- 학습 데이터 수: 266건
- 검증 데이터 수: 50건
- 평가 데이터 수: 40건
- 베이스 모델 성능 평가 결과:

지표	값
Precision	43.13%
Recall	37.32%
F1 Score	39.82%

- 파인튜닝 모델 성능 평가 결과:

지표	값
Precision	57.05%
Recall	58.93%
F1 Score	57.83%

- 일반화 성능 평가:
  - 미검증 데이터셋(Test set)에 대한 성능 평가: BERTScore 기반의 평가 결과는 미검증 데이터셋에 대한 성능을 측정
  - 과적합 방지: 파인튜닝 코드(finetuning01.ipynb)에서 별도의 Early Stopping이나 Dropout 설정은 확인되지 않음. 향후 모델 학습 시 과적합

방지를 위한 기법을 추가할 계획

- 과소적합 발생: 훈련 정확도와 테스트 정확도가 유사하다는 직접적인 증거는 없으나, BERTScore F1 점수가 60% 미만인 점을 고려할 때, 모델이 데이터의 패턴을 충분히 학습하지 못하는 과소적합(Underfitting) 현상이 발생했을 가능성이 있습니다.

4. 저장 및 배포

- 저장 형식:

항목	설명
저장 파일명	llm_finetuned_model
저장 형식	LoRA 어댑터 가중치 파일
저장 방법	model.save_pretrained(OUTPUT_DIR)
모델 불러오기 코드 예시	PeftModel.from_pretrained(base_model, adapter_path)

- 모델 사양 요구 사항:
  - 프레임워크: PyTorch
  - GPU/CPU 호환 여부: 학습 시 GPU를 사용하며, CPU에서도 추론이 가능
  - 환경 설정 파일: requirements.txt 포함 (transformers, torch, peft, accelerate, datasets,..)
- 모델 테스트:
  - 모델 적재 및 추론 테스트 완료: 파인튜닝된 LoRA 어댑터를 기본 모델에 병합한 뒤 정상적으로 추론이 가능함을 확인
  - Inference 예시:

    입력: “내일 날씨 알려줘”
    출력: “날씨” (예측 정확)

5. 종합 평가 및 활용 방안

- 모델 안정성: 모델을 저장하고 다시 불러와도 동일한 추론 결과를 재현 가능
- 일반화 가능성: 미사용 데이터셋에 대해 BERTScore가 상승한 것으로 보아, 어느 정도 일반화 성능을 확보

- 재사용성: 파인튜닝된 모델은 LoRA 어댑터 형태로 저장되어 용량이 매우 작고, 추론 속도가 빠른 편입니다. (용량 420MB, 추론 평균 시간 0.3초/건)
- 향후 활용
    - API 서버 탑재: 경량화된 모델을 REST API로 배포하여 다양한 서비스에서 활용할 수 있습니다.
    - 챗봇 응답 분류/유사 문장 검색: 모델이 질문의 의도를 파악하는 능력이 향상되어 챗봇의 핵심 기능에 적용할 수 있습니다.
    - COT(Chain of Thought) 적용: 더 복잡하고 논리적인 답변 생성을 위해 COT 기법을 도입할 계획입니다.

6. 추가 기재

- 모델 저장 링크
- 모델 불러오기/병합 코드

```
base_model_name = "./exaone_4.0_1.2b"
adapter_path = "./llm_finetuned_model"
base_model = AutoModelForCausalLM.from_pretrained(
        base_model_name,
        trust_remote_code=True,
        device_map="auto",
        torch_dtype=torch.bfloat16
    )
    tokenizer = AutoTokenizer.from_pretrained(base_model_name)

    # --- LoRA 어댑터 병합 ---
    print("LoRA 어댑터를 베이스 모델에 병합합니다...")
    finetuned_model = PeftModel.from_pretrained(base_model, adapter_path)
    finetuned_model = finetuned_model.merge_and_unload()
    print("✅ 모델 병합 완료")
```

- 학습 로그

```
Applying LoRA configuration...
trainable params: 7,618,560 || all params: 1,287,010,048 || trainable%: 0.5920
Starting LoRA fine-tuning...
/tmp/ipykernel_247/357885995.py:151: FutureWarning: `tokenizer` is deprecated
  trainer = Trainer(
```

[60/60 04:22, Epoch 5/5]

Step	Training Loss
10	3.223000
20	2.385000
30	2.206400
40	2.088800
50	2.028000
60	2.036800