[영어분석을 위한 기계학습] - 202403804 김진희

1. 프로젝트 개요 및 문제 정의

AI Hub 데이터를 활용하여 PDF 문서를 자동 번역하고, 그 결과를 JSON 형태로 반환하는 API 서비스 구현을 목표로 했다. 개발 동기는 전공 강의 자료의 대부분이 영어로 구성되어 있어, 매 학기 방대한 양의 자료를 수동으로 번역해야 했던 비효율성을 해소하는 데 중점을 두었다. 궁극적으로는 이 서비스를 통해 영어 자료의 장벽 없이 핵심 내용에 즉각적으로 집중할 수 있는 기계 번역 시스템을 구축하는 것을 최종 목표로 설정했다.

2. 진행 과정

2.1. 주제 선정 및 문제 정의

복잡한 문서 형식인 PDF에 담긴 영문 텍스트를 효율적으로 번역하는 서비스 개발로 주제를 선정했다. 이를 위한 핵심 과제는 다음과 같이 세 가지로 정의했다. 첫째, 대용량 모델을 서버 환경에 안정적이고 효율적으로 통합하는 방안. 둘째, 사용자가 파일 업로드 방식을 통해 번역 서비스를 이용할 수 있도록 구현하는 방안. 셋째, PDF 텍스트 추출 시 발생하는 비정형 노이즈를 처리하는 방안이다.

2.2. 데이터 수집 및 학습

데이터 수집 및 학습 단계에서는 AI Hub의 대규모 영어-한국어 병렬 코퍼스 데이터를 활용했다. 학습 데이터 20,000쌍과 검증 데이터 2,000쌍을 사용했으며, mBART 사전 학습 모델을 기반으로 3 Epoch 동안 훈련을 진행했다. 특히 중요한 전처리 과정은 토큰화와 레이블 마스킹이었다. 입력 문장과 정답 문장은 모델이 이해하는 토큰 ID 배열로 변환했고, 정답 배열의 패딩 자리에는 손실 계산 시 패딩 부분을 무시하도록 -100 값을 채워 넣는 레이블 마스킹 기법을 적용했다. 모델 훈련 손실은 훈련 스텝이 진행됨에 따라 0.6726까지 안정적으로 감소하여 모델이 데이터 패턴을 학습했음을 확인했다.

2.3. 서비스 구축 및 시스템 구조

학습 완료된 모델을 서비스로 전환하기 위해 API 구조를 채택했다. Flask 웹 서버에 모델을 통합했으며, Gunicorn을 서버로 사용하고 ngrok을 통해 공용 인터넷 주소를 할당하여 /translate_pdf엔드포인트를 확보했다. 시스템 최적화를 위해 서버 시작 시 Google Drive에 저장된 대용량 모델을 GPU(CUDA)를 통해 로드하여 추론 속도를 최적화했다. 클라이언트로부터 PDF 파일을 받으면 PyPDF2를 통해 텍스트를 추출하도록 구현했으며, 빔 크기 4를 설정한 빔 서치 전략을 사용하여 최적의 번역 결과 후보 경로를 탐색하도록 설계했다.

3. 실제 사용 결과

3.1. 5회 이상 사용 기록

서버 구동 후, 5개의 실제 PDF 강의 자료 파일을 활용하여 API 테스트 클라이언트를 통해 개별적인 POST 요청을 5회 이상 성공적으로 전송했다. 모든 요청이 서버에 도달하여 JSON 응답을 반환했으며, 서버 구동 및 추론 시스템이 정상적으로 작동함을 확인했다. 응답 시간은

서버 안정화 후 약 5초~20초 내외로 측정되었다.

3.2. 성능 및 한계점

모델의 Validation BLEU 스코어는 24.14점을 달성했으나, 실제 PDF 강의 자료에 대한 Test BLEU 스코어는 7.61점으로 크게 하락했다. 이러한 격차는 모델이 훈련 데이터와 성격이 다른 비정형 노이즈 입력에 대해 일반화 성능이 매우 취약하다는 점을 명확히 시사한다. 또한, 모델 학습의 한계로 인해 번역 결과가 짧게 잘리거나 "힐힐힐힐..."과 같은 반복 생성 문제가 나타났다.

4. 배운 점 및 개선 방향

4.1. 배운 점

프로젝트를 진행하며 트랜스포머 모델의 핵심 원리(마스킹, 빔 서치)를 코드로 깊이 있게 이해하는 귀중한 경험을 할 수 있었다. 그러나 대용량 모델 학습 및 배포 과정에서 GPU 메모리 부족으로 코랩 세션이 여러 번 재시작되어 훈련을 여러 차례 시도해야 하는 어려움이 있었다. 이 외에도 서비스화 단계에서 예상치 못한 종속성 문제, 서버 연결 실패, 추론 시간 초과와 같은 다양한 기술적 난이도에 직면했다. 가장 큰 교훈은 모델 성능(24.14점)뿐 아니라 실제 적용 환경에서의 텍스트 정제 및 전처리 품질(7.61점)이 성능에 핵심적인 영향을 미친다는 사실이었다.

4.2. 개선 방향

현재 모델의 출력 길이 제한을 우회하기 위해 PDF 전체 텍스트를 청크 단위로 분할하고 반복적으로 번역한 후 통합하는 기능을flask_app.py에 추가하는 것을 고려할 수 있다. 결론적으로, 실제 테스트 스코어가 낮은 문제를 해결하기 위해 노이즈 데이터에 강한 텍스트 정제 파이프라인을 구축하는 것이 앞으로의 핵심 개선 방향이다.