DU-도전학기 결과보고서

과제명	교통사고 과실비율 예측 및 RAG·파인튜닝 기반 질의응답 시스템 개발		
참여자	성명	소속	학번
		AI학과	
		AI학과	
		AI학과	
지도교수 의견	본 과제는 교통사고 과실비율 예측과 법률 기반 질의응답 시스템이라는 복합적인 주제를 실현한 도전적인 프로젝트입니다. 학생들은 역할을 분담해 데이터 정제, 모델 구현, UI/UX 개선, 질의응답 시스템 구축 등 각자 맡은 영역에서 성실히 수행하였고, 기술적 완성도와 실용성을 고루 갖춘 결과물을 만들어냈습니다. 특히 사용자 피드백을 반영한 개선 과정이 인상 깊었으며, 실제 활용 가능성을 고려한 설계가 돋보였습니다. 학습과 협업 과정 모두에서 성장 가능성을 보여주었으며, 향후 발전이 기대됩니다. (소속) AI학과		

1. 도전 과제 내용

본 프로젝트는 교통사고 영상 데이터를 기반으로 과실 비율을 예측하는 AI 모델과, 관련 법률·사례에 대한 질의응답 기능을 제공하는 시스템을 개발하는 것을 목표로 한다.
단순한 수치 예측을 넘어, 사용자가 결과를 이해하고 참고할 수 있도록 하기 위해 검색 증강 생성(Retrieval-Augmented Generation, RAG) 기술을 도입하여, 관련 문서 기반의 자연어 응답을 함께 제공한다.
이를 통해 사고 상황에 대한 정량적 분석과 정성적 정보 제공이 동시에 이루어지는 교통사고 분석 지원 시스템을 구축하고자 한다.

2. 도전 과제 수행 결과 및 성과

- 김00:

① 서버-클라이언트 연동 구조 설계:

FastAPI를 활용해 백엔드 서버를 구축하고, 클라이언트인 Svelte와의 연결을 통해 사용자 요청에 응답할 수 있는 기본 API 구조를 설계했다. 이때 버튼 클릭, 입력창 작성 등 사용자 인터페이스 상의 동작이 서버로 연결되도록 흐름을 구성하였고, 실제 응답을 받을 수 있도록 클라이언트와 서버 간 데이터 주고받기 기능을 구현하였다. 이러한 연동을 통해 향후 다양한 기능을 추가하거나 외부 모델을 연결하는 데 필요한 기반이 마련할 수 있었다.

② 기능별 코드 분리 및 확장성 고려한 설계:

처음부터 하나의 기능만 고려하지 않고, 예측 결과 시각화, 사고 정보 요약, 챗봇 질의응답 등 다양한 기능을 통합하기 위해 각 기능별로 FastAPI 라우터와 함수들을 분리하여 작성했다. 이는 기능 추가 시 전체 코드를 건드릴 필요 없이 독립적으로 유지 보수하거나 확장할 수 있

도록 하기 위한 구조적 고려하였다.

- 텍스트 크기 단계별 조정 (제목, 본문, 설명 구분)
- 불필요한 기능 제거
- 정보 영역 간 시각적 경계(라인, 그림자 등) 등 구현

③ 사용자 피드백 기반 UI 개선 및 UX 고려한 UI 구성:
주변 지인들과 팀원들에게 받은 피드백을 바탕으로, 정보가 너무 많아 한눈에 보기 어려운 문제를 해결하기 위해 시각적 흐름을 조정하였다. 텍스트 정렬이나 크기, 정보 간 여백을 조절해 혼잡함을 줄였고, 불필요 기능 등은 생략하였다. 이를 통해 한 화면에 여러 기능이 있더라도 각각의 기능이 명확히 구분되어 보일 수 있도록 개선하였다.

- 김00:

① 데이터 정제・편향 완화
프로젝트를 시작할 때 가져온 데이터셋 전체 1만 5 천여 건의 사고 영상 가운데 60프로 이상이 '직선도로-직진 충돌' 패턴에 집중돼 있었다. 이렇게 한쪽으로 기울어진 데이터를 그대로 쓰면 모델이 교차로나 회전 사고를 거의 경험하지 못해 실제 상황에서 오답을 낼 우려가 컸다. 이 편향을 줄이기 위해 직선도로 클립을 절반가량 줄이고, 교차로・회전・후방 추돌 영상 네 종류 장면 비율을 비교적 고르게 맞췄다.

1. 장면 비율 재편성 : 직선도로 영상을 절반 가까이 덜어내어 네 가지 장면이 고른 비율이 되도록 재구성했다.
2. 라벨 신뢰도 점검 : 잘못 표시된 바운딩 박스와 누락된 '사고 위치・차량 진행 방향' 정보를 전수 조사해 모두 잘못 적힌 부분을 다시 데이터셋 참고자료에 맞게 수정했다.
3. 깨끗한 학습 프레임 확보 : 각 영상에서 사고 전・후 5초(총 10초)만 15fps로 뽑아 150장의 정제 프레임을 만들고, 검은 화면・인트로 같은 무의미한 장면은 자동으로 걸러냈다.

결과적으로 데이터셋이 균형을 되찾아, 이후 모델이 다양한 사고 유형을 고르게 배우는 기반이 마련되었다.

② 학습 포맷 정비와 '두 개의 핵심 스크립트'
모델마다 달랐던 폴더 구조・라벨 이름・평가 방식이 실험 효율을 떨어뜨려, 이를 한 번에 해결할 PredictionWithConstraint.py와 FinalResult.py를 새로 정비 ・ 도입했다.

- PredictionWithConstraint.py
- 각 모델의 원시 예측에서 블랙 화면・인트로처럼 의미 없는 프레임을 자동으로 걸러낸다.
- 영문・약어・혼합으로 섞여 있던 라벨을 표준 한글 라벨로 다시 매핑해, 누구나 검증 가능한 깔끔한 JSON 결과로 변환한다.
- FinalResult.py
- 위에서 정제된 로그 여러 개를 한꺼번에 불러온다.
- Top-k 정확도, 혼동행렬, Precision・Recall・F1, PR Curve를 자동 계산하고, CSV・PNG・HTML 리포트로 저장해 팀원이 바로 비교・검토할 수 있게 한다.

③ 객체 탐지 전처리 · 학습 실험

영상 속 차량·사람·신호등을 찾아내는 단계에서는 CNN 계열(Cascade R-CNN)과 Transformer 계열(DETR)을 같은 조건에서 돌려 성능과 안정성을 비교했다. 결론은 CNN 계열인 Cascade R-CNN을 사용하는 것이 결과가 나오는데 조금 더 유리했다. 이 선택은 실제 교통량이 많은 장면에서도 꾸준히 물체를 잡아내는 데 큰 도움이 되었다.

④ 비디오 분류용 클립 구축 · 학습 · 평가

사고 영상을 15 프레임씩 잘라 "사고가 어디서 일어났는지, 교차로 특성은 무엇인지, 두 차량이 어떤 방향으로 움직였는지"를 동시에 분류하도록 설계했다.

세 가지 모델(SlowFast-R50, VideoMAE-Base, TimeSformer-Small)을 같은 데이터로 학습시켜 비교한 결과 SlowFast 구조가 네 항목 모두에서 가장 안정적인 성능을 보여 최종 선택했다.

⑤ 탐지·분류 결과 통합과 과실비율 룰 테이블 결합

이제 "물체가 어디에 있었는지"와 "사고가 어떤 상황이었는지" 정보를 합쳐야 실제 책임 비율을 계산할 수 있다.

두 결과를 묶어 사고 장소 × 두 차량의 진행 방향 × 부딪힌 위치 조합으로 요약한다.

준비해 둔 430여가지 사고 책임 과실 비율 테이블(예: '사거리에서 직진 vs 좌회전 → 70:30')에 맞춰 자동으로 과실비율을 산출한다.

만약 규칙표에 없는 새 조합이 발견되면 로그에 기록해서 나중에 없는 케이스들만 체크해서 추후 테이블을 수정할 수 있게 세팅해두었다.

⑥ 3일 베타 테스트와 결과 출력 개선

연구실 동료와 대학 동기가 실제 테스트를 위해 3일간 시스템을 사용해 봤다.

크게는 지적했던 문제점이 두가지 정도 존재하였다:

문제 ① "분석이 멈춘 줄 알았다" → 원형으로 돌아가고 있음을 그림으로 표현하며 실시간으로 작동중임을 표시했다.

문제 ② "결과가 한눈에 안 들어온다" → '사고 상황 카드'와 '과실비율 카드'를 분리하고, 큰 책임 비율 차이는 막대그래프로 강조했다.

테스트 후 설문에서 눈에 보이는 불편함이 확실히 줄었다는 응답을 얻었고, 평균 처리 속도도 15초 남짓으로 유지돼 성능 저하 없이 UX가 개선됐음을 확인했다.

- 박00

① 텍스트 질의 임베딩 구조 개발

사용자 사고 설명과 같은 자연어 질의를 그대로 ColPali 모델에 임베딩할 수 있도록 입력 파이프라인을 구성하였다. ColPaliProcessor를 명시적으로 선언하여 tokenizer와 image processor를 개별 로딩하고, 텍스트-only 입력도 별도의 처리 없이 임베딩할 수 있도록 구현하였다.

- 자연어 쿼리를 직접 ColPaliProcessor.process_queries()에 전달하여 벡터로 변환
- 출력된 질의 벡터는 FAISS 검색에 바로 활용 가능
- processor, model, index, vectors 등 전체 객체를 runtime_objs_colpali.pkl로 저장하여 빠르게 로딩 가능하게 구성

② 유사 문서 검색 및 GPT-4o 응답 연결 흐름 구현
쿼리 임베딩 후 FAISS 인덱스를 활용해 유사 문서 3건을 자동 선별하고, 해당 문서 이미지를 GPT-4o에 전달해 질의응답이 이뤄지도록 전체 흐름을 구성하였다.
- query_vecs를 기준으로 FAISS에서 top-k 문서 추출
- 문서 이미지를 base64 포맷으로 GPT-4o에 전송
- 질의와 이미지 조합을 기반으로 사고 상황 분석 및 판단 근거 중심 응답 생성

③ 심의사례 2페이지 병합 및 이미지 임베딩 재구축
하나의 사고 사례가 PDF 이미지 단위를 조정하고 임베딩을 다시 구축하였다.
- ColPaliProcessor를 통해 각 이미지에서 약 1천여 개 벡터 추출
- FAISS 인덱스를 새로 구축하고 page_ids, vectors, index 파일을 갱신

④ 전체 파이프라인 통합 및 자동화
텍스트 질의 → 임베딩 → 유사 문서 검색 → GPT-4o 질의응답까지의 흐름을 하나의 루프 내에서 자동으로 처리할 수 있도록 통합 코드를 구성하였다. 사용자 입장에서 사고 설명만 입력하면 전체 질의응답 결과를 확인할 수 있도록 하여 실용성과 응답 일관성을 높였다.

3. 자기 평가

- 김00: 도전학기를 시작할 때는 FastAPI나 Svelte 같은 도구들이 낯설어서 구조를 잡는 데 시간이 좀 걸렸지만, 매주 하나씩 기능을 직접 구현해보면서 익숙해졌다. 특히 프론트엔드랑 백엔드가 데이터를 어떻게 주고받는지, 전체 흐름을 직접 설계해보면서 많이 배울 수 있었다.
  처음에는 기능이 돌아가기만 하면 된다고 생각했는데, 시간이 지나면서 사용자 입장에서 화면을 어떻게 구성해야 할지도 고민하게 됐다. 그래서 피드백을 받아서 화면 배치나 글자 크기, 색상 등을 수정하는 과정도 직접 해봤고, 그게 실제로 보기 좋게 바뀌는 걸 보면서 보람도 느꼈다.
  한 번에 잘 되는 경우보다는 시행착오가 많았지만, 그럴 때마다 계속 테스트해보고 방법을 찾아보면서 해결하려고 했던 점이 스스로 가장 뿌듯하다. 이번 도전학기를 통해 처음보다 확실히 성장했다고 느끼고, 다음에는 더 복잡한 프로젝트도 자신 있게 도전해보고 싶다.

- 김00: 먼저, 직선도로 장면이 과도하게 많아 모델이 특정 패턴에만 치우치는 문제를 해결하고자 원천 데이터를 재구성하며 데이터 엔지니어링의 중요성을 체감했다. 클래스 불균형을 완화하기 위해 교차로·회전·추돌 클립을 증설하고 라벨 구조를 프레임·메타 2계층 JSON으로 통합하면서, 데이터 설계가 이후 모든 모델과 로직의 토대가 된다는 사실을 다시금 깨달았다.
  둘째, Detection·Classification·룰 엔진이 서로 다른 포맷을 사용해 생기던 병목을 해소하기 위해 학습 포맷을 범용화하고, PredictionWithConstraint.py · FinalResult.py를 개발해 실험 재현성과 분석 효율을 크게 높였다. 이 과정을 통해 표준화된 파이프라인 구축 능력을 키울 수 있었습니다.또한 사용자 테스트에서 드러난 “분석 진행 상태 미표시”와 “결과 가독성 부족” 같은 UX 이슈를 신속히 반영하여, 기술적 완성도뿐 아니라 사용자 신뢰도를

확보하는 접근이 얼마나 중요한지 깨달았다. 특히 실제 사용자의 피드백을 바탕으로 UI를 개선한 경험은 모델 성능만큼이나 인터랙션 설계가 서비스 성공에 핵심임을 느끼게 해 주었다.

- 박00: 도전학기 활동을 통해 AI모델에 대해서 이해하고 개선해나가는 과정을 경험할 수 있었다. 특히 처음에는 ColPali 모델의 입출력 구조나 처리 방식이 명확하지 않아 시행착오를 반복했지만, 그 과정 속에서 멀티모달 모델이 어떤 전처리를 필요로 하고, 텍스트와 이미지가 어떻게 결합되어야 의미 있는 출력이 나오는지를 구체적으로 이해할 수 있었다. 또한 기존 구조의 비효율성을 인식하고 이를 구조적으로 개선하면서 단순히 코드를 수정하는 것을 넘어서 '왜 이 구조가 문제였는지', '어떤 방향이 더 바람직한 설계인지'를 고민하는 습관이 생겼다. 무엇보다 이번 활동을 통해 '동작하는 코드'를 넘어서 '신뢰할 수 있고 유지보수 가능한 구조'를 고민하게 되었고, 이는 앞으로의 연구나 개발 과정에서도 중요한 기준점이 될 것이라 생각한다. 이번 활동을 통해 단순 구현에서 벗어나 시스템 설계의 관점까지 고민해볼 수 있었던 의미 있는 시간이었다.

4. 최종 결과물