

멀티모달 AI 를 이용한 노인복지시설 안전사고 실시간 모니터링 플랫폼 개발



저자 1 장인성

저자 2 송세연

저자 3 손현경

지도교수 김태운

목 차

1. 서론	1
1.1. 연구 배경	1
1.2. 기존 문제점	2
1.3. 연구 목표	3
2. 연구 배경	4
2.1. H/W	4
2.1.1. 라즈베리파이	4
2.1.2. 마이크	5
2.1.3. 열화상 카메라	5
2.1.4. Jetson AGX Xavier	6
2.2. 인공지능	7
2.2.1. 음향 기반 낙상 분류 데이터셋	7
2.2.2. 열화상 동영상 기반 낙상 분류 데이터셋	9
2.3. 플랫폼	10
2.3.1. 데이터베이스	10
2.3.2. 백엔드	11
2.3.3. 프론트엔드	12
2.3.4. 스트리밍	13
2.3.5. 클라우드	14
3. 연구 내용	16
3.1. H/W	16

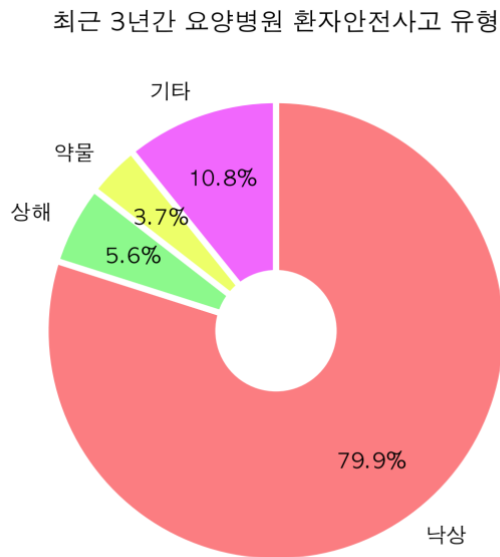
3.1.1.	라즈베리파이 및 센서 환경 구축.....	16
3.1.2.	Jetson AGX Xavier 환경 구축.....	16
3.1.3.	공인 IP 신청 및 포트포워딩.....	16
3.2.	인공지능.....	17
3.2.1.	음향 기반 낙상 분류 모델.....	17
3.2.2.	열화상 동영상 기반 낙상 분류 모델.....	19
3.2.3.	멀티모달 모델.....	20
3.2.4.	태스크 오프로딩.....	20
3.2.5.	딥러닝 모델 최적화.....	20
3.3.	플랫폼.....	22
3.3.1.	데이터베이스 테이블.....	22
3.3.2.	실시간 동영상 스트리밍.....	24
3.3.3.	데이터 흐름.....	25
3.3.4.	인프라 구조.....	27
3.3.5.	UI/UX.....	27
4.	연구 결과 분석 및 평가.....	37
4.1.	인공지능.....	37
4.1.1.	음향 기반 낙상 분류 모델 성능 평가.....	37
4.1.2.	열화상 동영상 기반 낙상 분류 모델 성능 평가.....	38
4.1.3.	딥러닝 모델 추론 속도 최적화 성능 평가.....	38
4.2.	플랫폼.....	39
4.2.1.	비회원 기능.....	39
4.2.2.	회원 기능.....	39
4.2.3.	핵심 기능 구현 상세.....	40

4.2.4.	플랫폼 배포	41
4.2.5.	API	41
4.3.	공통	43
4.3.1.	코드 버전 관리 및 CI/CD	43
4.3.2.	Notion을 활용한 문서화 및 협업	44
5.	결론 및 향후 연구 방향	45
6.	참고 문헌	46

1. 서론

1.1. 연구 배경

다수의 어르신이 생활하는 노인복지시설에서는 고령으로 인한 인지 및 운동 능력 저하로 안전사고가 종종 발생한다. <그림 1>에 따르면 최근 3년간 요양병원에서 발생한 환자 안전사고를 유형별로 분류한 결과, 낙상이 절반을 훌쩍 넘는 비율을 차지하고 있는 것으로 나타났다.

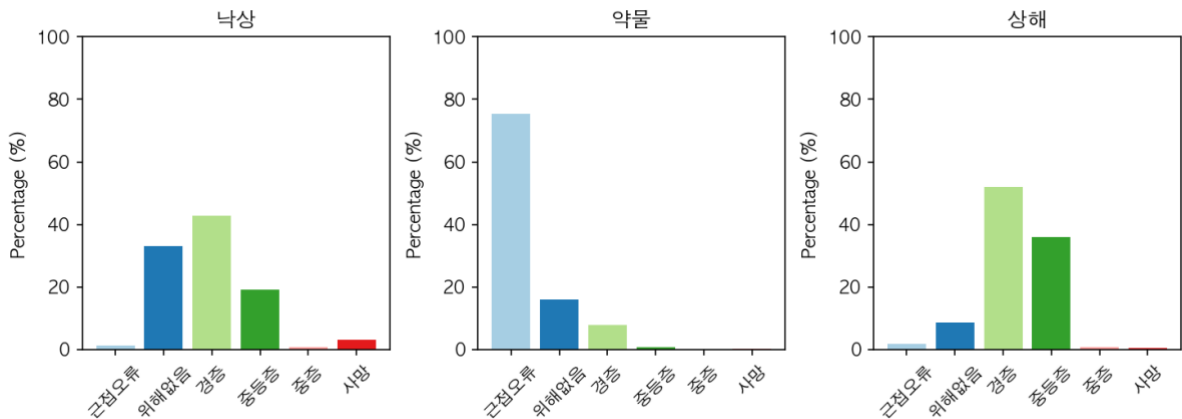


<그림 1> 2023년 의료기관평가인증원에 대한 국정감사 자료

또한, <그림 2>에 따르면 낙상, 약물, 상해와 같은 안전사고 중 중증 및 사망으로 이어지는 비율이 낙상에서 가장 높았다. 낙상과 같은 안전사고가 발생할 경우, 골든아워를 지키기 위해 신속하게 사고 발생 여부를 파악하고 조치를 취해야 한다. 그러나, 시설 내 CCTV를 모니터링하는 전담 인력의 피로도 증가와 환자의 사생활 침해 문제가 발생한다.

이러한 문제를 해결하기 위해 개인의 식별이 불가능한 정보를 인공지능을 이용해 실시간으로 안전사고 발생 여부를 판단하고 신고하는 시스템이 필요하다. 안전사고 발생 시 신속하게 대처하기 위해, 실시간으로 사고를 탐지하고 의료 관계자 및 보호자에게 자동으로 신고하는 시스템을 개발하고자 한다.

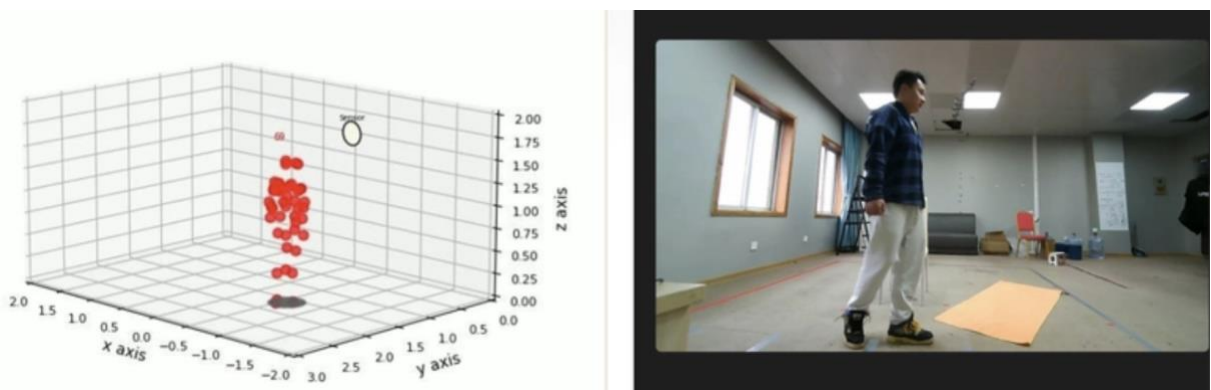
사고 종류에 따른 피해정도별 현황



<그림 2> 2022 환자안전 통계연보

1.2. 기존 문제점

과제 주제가 타당한지를 검토하기 위해 의료계 종사자로부터의 자문을 통해 병원에서는 낙상사고가 발생했을 때 사고 발생에 대한 책임을 논하는 과정에서 환자와 간호사 간의 분쟁이 종종 발생한다는 것을 알게 되었다. 제안하는 플랫폼이 단순히 안전사고에 대응하는 데에서 나아가 환자와 간호사 간의 분쟁을 해소할 수 있도록 사고 후 책임논쟁에 대한 증거자료로 채택될 상황을 고려하여 사고 발생 1분 내외의 영상을 녹화하기로 하였다. 사고를 기록하기 위해 RGB 카메라를 사용하는 경우, 야간에는 이미지를 식별하기 어렵고 프라이버시 침해 문제가 발생한다.



<그림 3> mmWave 레이더 센서를 이용하여 획득한 사람의 포인트 클라우드

현재 상용화된 시스템들은 프라이버시 보호 및 안전사고를 탐지하기 위해 mmWave 레이더 센서를 사용하여 안전사고를 탐지한다. 그리고 mmWave 레이더 센서 기반 시스템

의 경우 센서의 측정 범위가 약 4m에 불과하여 공간마다 장비를 구축해야 하므로 초기 비용이 많이 발생한다. 그리고 mmWave 레이더 센서를 통하여 얻은 포인트 클라우드로는 사고 발생 과정 및 대처 과정을 해석하기 어려워 환자와 간호사 간 분쟁 발생 시 책임 논쟁에 대한 증거자료로 채택되기 힘들다는 단점이 있다.

이러한 문제를 해결하기 위해 음향과 열화상 정보를 함께 이용하고자 한다. 열화상 카메라를 이용하여 사고 발생 및 대처 과정을 녹화한다. 마이크를 이용하여 음향 기반으로 넓은 지역을 모니터링하며, 카메라의 사각지대 문제를 해결한다. 또한 안전사고 발생 여부를 자동으로 파악하여 신고하기 위해 음향과 열화상 정보를 함께 이용하는 멀티모달 AI를 이용하여 안전사고를 탐지하고자 한다.

1.3. 연구 목표

음향과 열화상 정보를 인공지능을 활용해 실시간으로 안전사고 발생 여부를 파악하여 자동으로 보호자에게 실시간 영상을 제공하고, 자동 신고 기능을 통해 즉각적인 대처를 할 수 있도록 한다. 더불어 전국 요양병원의 안전사고 발생 현황을 모니터링하는 통계 시스템을 구축하여 사고 취약 지점을 파악하고, 안전사고 예방을 위한 조치를 취하는 데 도움을 주고자 한다. 사고에 대한 즉각적인 대처와 사후 영상자료 확보를 위해 사고 발생 시 보호자와 병원 관계자에게 사고 영상을 즉시 제공하고, 119 자동 신고 기능이 탑재된 플랫폼을 제작한다.

더 나아가 안전사고 발생에 대한 통계 정보를 공개하여 사후 조치를 개선하고 공공의 안전을 확보하고자 한다. 이를 위해 전국 요양병원으로부터 안전사고가 발생한 건에 대한 데이터를 수집하고, 안전사고가 발생한 빈도, 사고 경위, 사후 조치에 대한 기록 등 전국의 사고 발생 현황을 파악할 수 있는 통계 자료를 제공하고자 한다. 이러한 통계 자료를 통해 전국의 사고 발생/경위/조치 현황을 모니터링하고, 사고 취약 지점으로 판단되는 병원을 찾아 안전사고 예방을 위한 조치를 취할 수 있을 것으로 기대된다.

2. 연구 배경

연구 배경에서는 데이터 수집 및 처리를 진행할 센서와 컴퓨팅 H/W, 딥러닝 기반 안전 사고 분류를 위한 데이터셋, 스트리밍 및 통계를 저장할 플랫폼에 필요한 요소들을 설명한다.

2.1. H/W

과제에 진행할 음향 및 열화상 데이터를 수집하기 위해 마이크와 열화상 카메라가 필요하다. 원하는 기능이 H/W에서 동작하게 하기 위해 오픈소스 H/W를 사용하여 진행하였다.

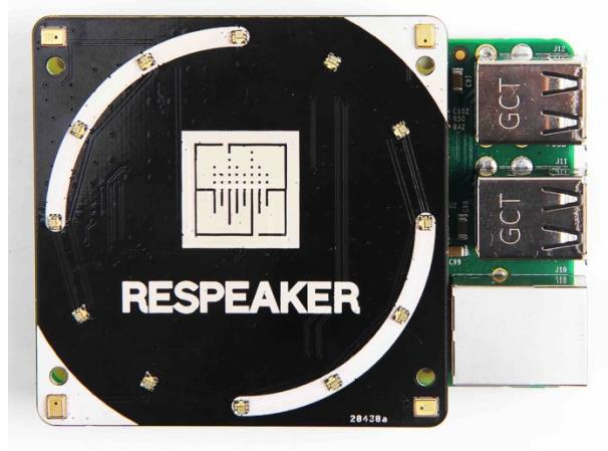
2.1.1. 라즈베리파이



<그림 4> 라즈베리파이 3B+

데이터 수집을 위해 라즈베리파이에 열화상 카메라와 마이크를 연결하여 사용한다. 수집한 데이터를 HTTP 통신을 통해 Jetson AGX Xavier로 전송한다.

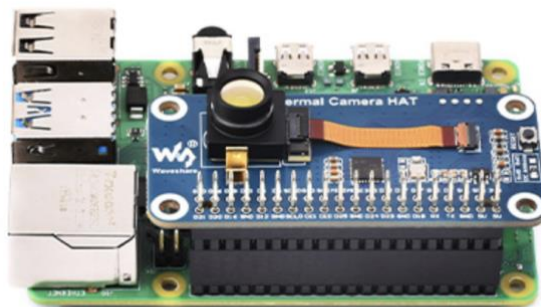
2.1.2. 마이크



<그림 5> 라즈베리파이와 연결한 마이크

음향 기반 안전사고 모니터링을 위한 마이크로 라즈베리파이에서 사용할 수 있는 seeed studio 사의 ReSpeaker 4-Mic Array for Raspberry Pi를 이용한다. 초기 기획 시 안전사고 발생 위치 추정을 위해 마이크의 DoA(Direction of Arrival) 기능을 이용하여 삼각 측량을 이용하여 음향이 발생한 위치를 추정하려고 하였으나, 열화상 카메라가 각 병상을 모니터링하므로 낙상 발생 지점을 알 수 있어 해당 기능을 제외하였다.

2.1.3. 열화상 카메라



<그림 6> 라즈베리파이와 연결한 열화상 카메라

열화상 카메라 기반 안전사고 모니터링을 위해 라즈베리파이에서 동작하는 waveshare

사의 Long-Wave IR Thermal Imaging Camera Module을 이용한다. RGB 카메라의 경우, 이미지에 포착된 인물을 식별할 수 있어 프라이버시 문제가 발생한다. 또한 빛이 없는 야간 시간에 획득한 이미지는 대부분이 검은 픽셀로만 구성되어 있어 모니터링 시스템에 사용할 수 없다. 이러한 문제를 해결하고자 RGB 카메라 대신 열화상 카메라를 사용한다.

2.1.4. Jetson AGX Xavier



<그림 7> Jetson AGX Xavier

일반적으로 센서를 제어하는 MCU 등의 연산장치에서는 컴퓨팅 성능의 한계로 인하여 복잡한 딥러닝 모델의 추론 연산을 진행할 수 없다. 본 과제에서는 센서 제어 장치로 라즈베리파이를 사용하고 있어 가벼운 모델의 추론이 가능할 것으로 보이나, 보다 일반적인 상황을 가정하기 위해 라즈베리파이는 수집한 열화상 동영상 및 음향 데이터를 전송하는 것으로 기능을 한정하였다. 수집한 데이터에서 안전사고를 탐지하는 딥러닝 연산을 클라우드/엣지 컴퓨팅으로의 태스크 오프로딩을 통해 처리할 수 있다. 그러나, 클라우드 컴퓨팅을 위해 센서에서 발생하는 많은 데이터를 외부 네트워크를 통해 전송하는 것은 혼잡 측면에서 바람직하지 않다. 내부 네트워크에 연결된 엣지 컴퓨팅으로의 태스크 오프로딩을 통해 위와 같은 문제를 해결할 수 있다.

본 과제에서는 딥러닝 연산을 위한 엣지 컴퓨팅 장치로 Nvidia의 Jetson AGX Xavier를 사용한다. Jetson AGX Xavier는 저전력으로 동작하는 딥러닝 추론을 위한 GPU를 탑재하고 있어 엣지 디바이스로 사용하기에 적합하다.

2.2. 인공지능

여기에 1의 내용 다시 쓰기

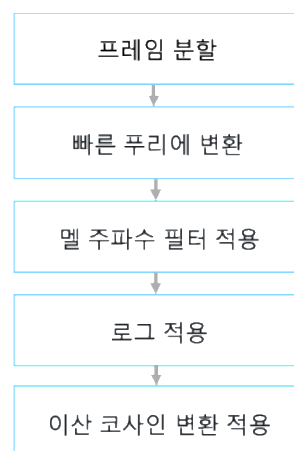
2.2.1. 음향 기반 낙상 분류 데이터셋

노인복지시설에서 발생하는 음향을 이용하여 안전사고 발생 여부를 탐지하는 딥러닝 모델을 학습하기 위해 AI Hub의 위급상황 음성/음향 데이터셋을 이용한다. 해당 데이터셋에는 다양한 클래스의 데이터가 포함되어 있으나 본 과제에서는 노인복지시설 안전사고에 초점을 맞출 수 있도록 낙상과 정상(실내) 클래스만 이용하여 학습을 진행한다. 데이터셋에 포함된 데이터 수는 <표 1>과 같다.

클래스	학습 데이터 수	테스트 데이터 수
낙상	18,322	1,020
정상(실내)	6,215	2,865

<표 1> 위급상황 음성/음향 데이터셋의 클래스 별 데이터 수

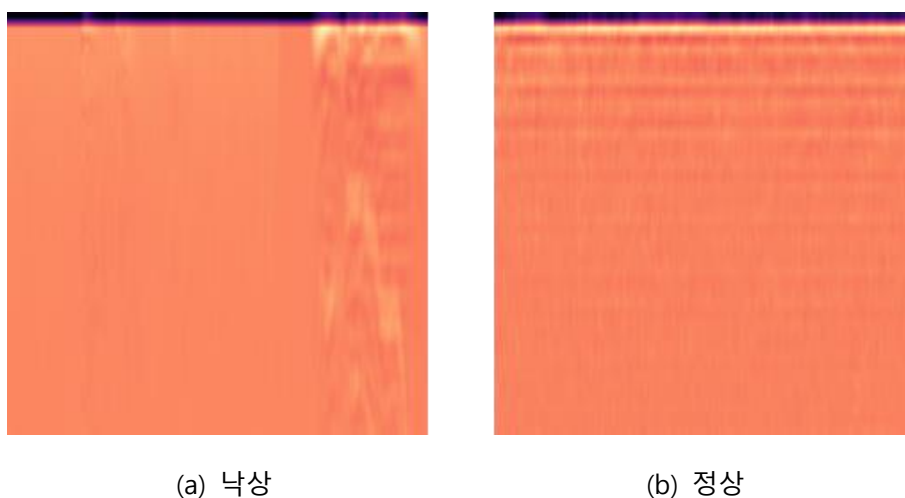
음향 데이터는 기본적으로 시간 도메인의 1차원 형태이다. 한 음향 내에 원하고자 하는 신호와 다른 신호가 섞일 경우 구분이 어려워지는 간섭 현상이 일어난다.[1] 이는 원하고자 하는 분류 모델의 수렴을 어렵게 만든다. 그러나 음향을 시간 도메인에서 2차원인 시간-주파수 도메인으로 변환하면, 주파수가 다른 여러 신호들을 동시에 식별할 수 있다. 본 과제에서는 음향을 시간 도메인에서 시간-주파수 도메인으로 변환하기 위해 MFCC(Mel Frequency Cepstral Coefficients)를 사용한다. MFCC는 인간의 청각 특성에 맞게 처리한 표현이다. 음향이 MFCC로 변환되는 과정은 <그림 9>와 같다.



<그림 8> 음향의 MFCC 변환 과정

프레임을 분할한 후, 각 프레임에 빠른 푸리에 변환을 적용하여 주파수 도메인으로 변환한다. 푸리에 변환이 적용된 프레임이 모이면 시간-주파수 도메인이 된다. 그 후 사람의 청각과 유사하게 인식하도록 멜 주파수 필터와 로그를 적용한다. 그 후 순차적으로 관련이 있는 데이터를 제거하기 위해 이산 코사인 변환을 적용한다.

획득한 MFCC는 시간-주파수 도메인의 2차원 형태로, 채널이 1인 이미지이다. 이를 분류하기 위해 딥러닝 기반 이미지 분류 모델을 사용할 수 있게 된다.

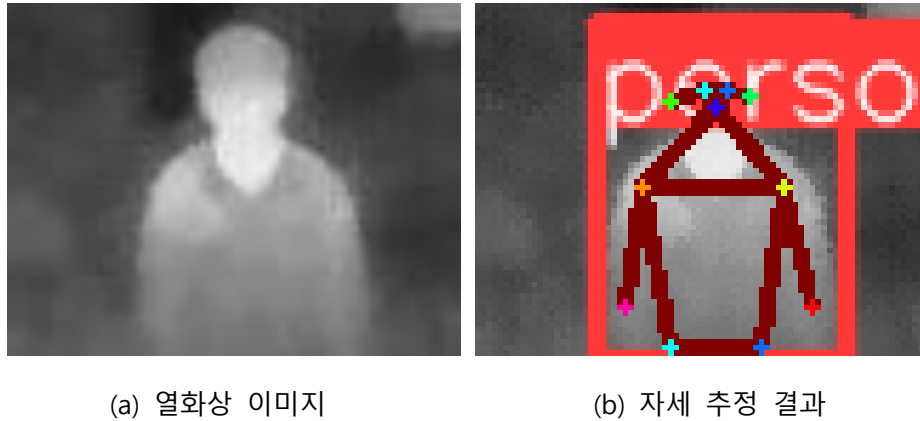


<그림 9> 클래스 별 음향의 MFCC 변환 결과

<그림 10>은 데이터셋의 음향을 MFCC로 변환한 결과이다. 사람이 보고 해석하기는 쉽지 않으나, 딥러닝 모델은 시간-주파수 도메인의 신호 차이를 학습하여 분류할 수 있다.

학습 검증을 위해 학습 데이터를 학습 데이터:검증 데이터=4:1로 분할하였다. 이때 StratifiedShuffleSplit을 이용하여 클래스 불균형이 존재하는 데이터셋의 분할 시 원래 데이터셋의 분포를 유지하여 클래스의 분포가 치우치지 않도록 하였다.

2.2.2. 열화상 동영상 기반 낙상 분류 데이터셋



<그림 10> ViTPose를 이용한 자세 추정

기존의 연구들은 RGB 카메라로 촬영한 동영상에서 자세 추정 모델을 이용하여 추출한 스켈레톤을 이용하여 낙상을 탐지하는 연구들[2]이 많았다. AI hub의 시니어 이상행동 영상 데이터셋과 High quality fall simulation data[3]는 각각 가정, 요양원 방 내에서 재현된 일상 및 낙상을 RGB 카메라로 촬영한 동영상 데이터셋이다. 원래는 위 데이터셋을 이용하여 ViTPose[4] 등 자세 추정을 모델 통해 사람의 스켈레톤을 추출한 후 낙상을 분류하는 모델을 학습하려 하였으나 열화상 이미지에서 스켈레톤을 추출할 경우 RGB 카메라를 이용하였을 때와 달리 신체 부위가 겹치는 부분이 명확히 추출되지 않았으며, 자세 추정 모델에서의 병목[5]으로 인하여 플랫폼의 실시간성 관점에서 원하는 처리율을 달성하기 어렵다고 판단하였다. 또한, 단편적인 이미지를 이용하여 낙상을 분류하는 경우, 정상적으로 누워있는 사람과 낙상으로 인해 쓰러진 사람을 구별하기 어렵다고 판단하였다.

이러한 근거를 바탕으로, 열화상 동영상을 기반으로 하는 낙상 탐지를 진행하고자 하였다. 또한, 열화상 동영상 기반 낙상 탐지 공개 데이터셋이 없어 열화상 카메라를 이용하여 직접 낙상과 정상을 재현한 모습을 촬영하여 데이터셋을 직접 구축하였다.

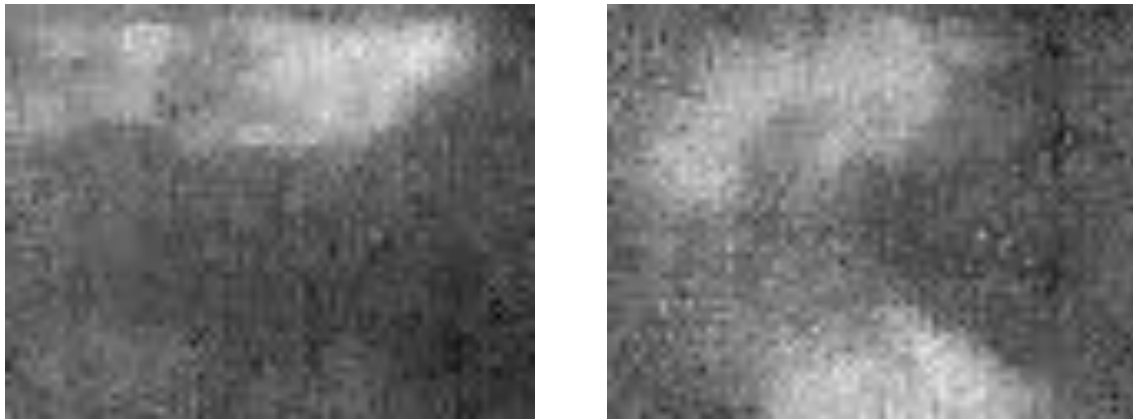
클래스	학습 데이터 수	테스트 데이터 수
낙상	21	5
정상	65	16

<표 2> 열화상 동영상 데이터셋의 클래스별 데이터 수

구축한 데이터셋은 열화상 카메라를 이용하여 10FPS(Frame per Speed)로 5초간 촬영한 50장의 이미지들로 이루어져 있다. 일련의 이미지 프레임들은 공간적 정보뿐만 아니라

시간적 정보도 포함하고 있는 3차원 정보이다. 시간적 정보를 추가로 처리하기 위해서는 이미지에서 정보를 추출하는 2D 컨볼루션 연산의 차원을 확장한 3D 컨볼루션 연산을 사용할 수 있다.

학습 검증을 위해 마찬가지로 StratifiedShuffleSplit를 이용하여 학습 데이터:검증 데이터 =4:1로 분할하였다. 부족한 데이터의 증강 및 카메라 흔들림에 강인한 모델을 획득하기 위해 학습 시 epoch 마다 프레임들을 -10~10도 사이에서 무작위로 회전하였다.



(a) 정상

(b) 낙상

<그림 11> 구축한 데이터셋의 클래스 별 예시 프레임

<그림 11>은 각각 정상 데이터와 낙상 데이터의 한 프레임이다. 정상 데이터의 경우, 침대 위에 한 명의 사람이 큰 움직임 없이 누워있는 데이터로, 열화상 카메라에서 하나의 열원으로 포착된다. 하지만 낙상이 발생한 경우, 사람이 침대에서 빠르게 이탈하므로, 열화상 카메라에는 바닥으로 떨어진 사람의 열원뿐만 아니라 침대에 사람 형태로 남은 잔열 두 가지가 모두 포착된다. 이러한 차이점을 바탕으로 동영상 기반 분류 모델이 낙상을 분류하도록 하였다.

2.3. 플랫폼

2.3.1. 데이터베이스

PostgreSQL

PostgreSQL은 관계형 데이터베이스 관리 시스템(RDBMS)으로, 데이터를 테이블 형식으로 저장한다. ACID를 준수하여 복잡한 트랜잭션이 필요한 애플리케이션에서 신뢰성을 보

장하고, 데이터 무결성을 유지할 수 있는 다양한 제약 조건을 지원한다. 또, 확장 가능한 저장소 구조로 되어 있어 많은 양의 데이터를 효율적으로 처리할 수 있어 본 과제에 사용하는 데이터베이스로 채택하였다.

2.3.2. 백엔드

주된 기능을 담당할 백엔드 서버 기술 스택으로는 Spring을 선택하였다. 다양한 라이브러리를 지원하는 Spring Boot를 사용하면 빠르고 편리하게 백엔드 서버를 구축하고 기능을 개발할 수 있다.

언어는 Kotlin을 사용하여 개발했다. Kotlin은 Java와 호환이 가능한 언어로, 간결한 문법과 Null 안정성을 보장해 주는 특징이 있어 높은 생산성이 보장된 언어이다.

Spring Security

Spring Security는 애플리케이션의 인증과 권한 부여를 처리하는 보안 프레임워크이다. OAuth 및 JWT를 지원하기 때문에 소셜 로그인, API 인증 등을 위한 표준 보안 프로토콜 지원이 가능하다는 특징이 있다.

본 과제에서는 병원의 관리자와 일반 회원의 권한을 구분하고, 토큰을 사용하여 회원을 식별하기 위해 사용하였다.

Spring Data JPA

Spring Data JPA는 Java Persistence API를 쉽게 사용할 수 있도록 도와주는 라이브러리이다. 추상화된 인터페이스를 제공하여 코드 작성을 최소화하면서도 데이터베이스 조회 및 저장을 쉽게 처리할 수 있다.

플랫폼 개발 과정에서 작성된 데이터베이스 테이블과 백엔드 코드 간의 격차를 줄이고, 특정 데이터베이스에 종속적이지 않은 코드를 작성하기 위해 사용하였다.

Spring MVC

Spring MVC는 HTTP 요청을 처리하고, 클라이언트에게 웹 응답을 제공하는 웹 프레임워크이다. Model, View, Controller로 구성된 MVC 패턴을 기반으로 동작하며, RESTful 웹 서비스 개발을 쉽게 도와준다.

플랫폼에서 제공하는 서비스 API는 Spring MVC 패턴에 따라 개발하였다.

JWT

JWT는 클라이언트와 서버 간의 인증 및 정보 전달에 사용되는 토큰 기반 인증 방식이다. 토큰 기반 인증 방식을 사용하면 클라이언트는 서버로부터 JWT 토큰을 발급받고 이후의 요청마다 HTTP 헤더에 토큰을 포함하여 인증을 수행한다. 서버는 클라이언트로부터 넘겨받은 토큰을 확인하여 요청자의 신원을 검증한다.

병원의 관리자와 일반 회원의 권한을 구분하고, 토큰을 사용하여 회원을 식별하기 위해 사용하였다.

RabbitMQ

RabbitMQ는 송신자와 수신자 사이에서 메시지를 안전하게 교환하는 표준 프로토콜인 AMQP를 구현한 오픈소스 메시지 브로커다. 여러 대의 서버를 구성하고, 서버 간의 상호작용이 필요할 경우 RabbitMQ를 활용하여 비동기식으로 메시지를 전달할 수 있다. 또 동일한 큐 내에서는 원소 간의 순서가 보장된다는 특징이 있다.

여러 서버 간에 사고 이미지를 공유하기 위해서 사용하였다. 병원의 카메라별로 각각 다른 큐에 연속적인 이미지를 저장하여 순서가 보장될 수 있도록 하였다.

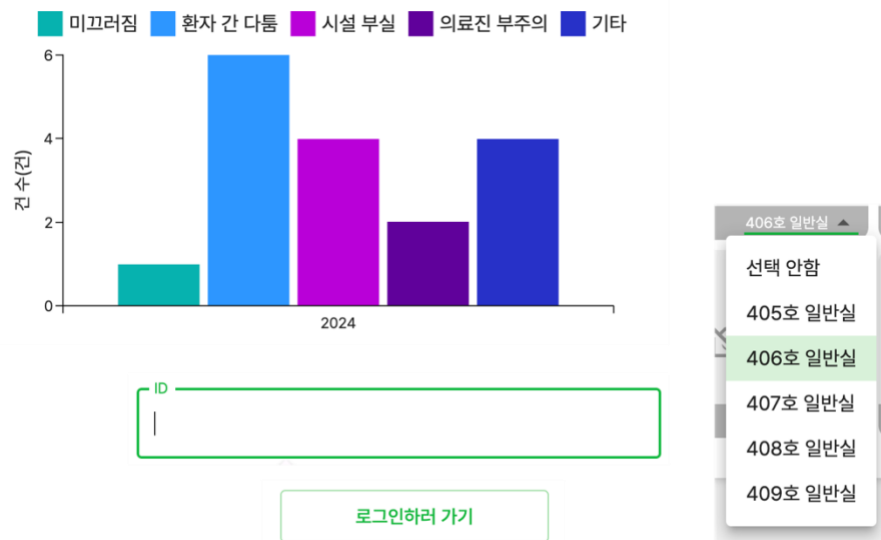
2.3.3. 프론트엔드

주된 기능을 담당할 프론트엔드 기술 스택으로는 프론트엔드 개발 담당 팀원의 학습곡선과 생산성을 고려하여 React 라이브러리를 선택하였다.

언어는 TypeScript를 사용하였다. TypeScript는 JavaScript의 확장 버전으로, 정적 타입 시스템을 제공한다. 정적 타입 시스템은 코드 작성 중에 발생할 수 있는 오류를 줄이고 유지 보수를 쉽게 만든다. TypeScript를 사용하면 코드의 가독성을 높일 수 있다. 코드의 의도를 더욱 명확하게 표현할 수 있기 때문에 코드의 구조를 변경하는 작업도 더욱 수월하고 빠르게 수행할 수 있다. 이에 개발 생산성과 코드 품질을 높이기 위해 기술 스택으로 채택하였다.

MUI

완성도 높은 디자인의 UI 컴포넌트를 제공하는 라이브러리로, 생산성을 높이기 위해 사용하였음. <그림 12>와 같이 클라이언트 서비스 내 막대그래프 및 표 UI, 사용자 입력을 받는 TextInput, Selector, Button 등의 입력 요소가 사용되었다.



<그림 12> MUI를 사용한 UI

Redux

애플리케이션의 데이터를 관리하고 동기화하는 데 사용되는 라이브러리이다. 로그인 정보와 STOMP 연결 동작을 수행하는 STOMP Client 객체 등 애플리케이션 내에서 자주 사용되는 데이터에 전역적으로 접근하기 위해 사용하였다.

TailwindCSS

웹 서비스에 적용될 CSS를 직관적이고 간단하게 적용할 수 있도록 지원하는 CSS 라이브러리로, UI 생산성을 높일 수 있다.

코드의 재사용성을 높이고 추후 기능 확장에 따른 유지보수를 용이하게 하기 위해 컴포넌트 중심의 설계를 하였으며, 플랫폼 전반에 사용되는 색상 및 규격 등을 상수화한 후 TailwindCSS 설정에 적용하여 디자인 유지보수성을 높였다.

ReactQuery

플랫폼의 데이터를 관리하고 동기화하는 데 사용되는 라이브러리이다. 서버로부터 받은 데이터를 쉽고 효율적으로 관리하기 위해 사용한다. 서버로부터 불러온 데이터를 캐싱하여 불필요한 API 호출을 줄일 수 있다.

2.3.4. 스트리밍

WebSocket

WebSocket은 클라이언트와 서버 간의 양방향(full-duplex) 통신을 가능하게 한다. 초기에 핸드 셰이크를 통해 연결을 시작한 후, 양방향 통신을 유지한다. 연결이 수립되면 클라이언트와 서버가 서로 데이터를 자유롭게 주고받을 수 있다. 클라이언트와 서버 간에 실시간으로 데이터를 주고받을 수 있어, 실시간 플랫폼에 적합하다. 연결이 수립된 후에는 재연결 없이도 계속해서 통신할 수 있어 실시간 데이터 전송에 용이한 통신 방법이다.

FastAPI와 Spring 서버 간의 이미지 통신과 Spring 서버와 React 클라이언트 간의 이미지 및 사고 발생 신호 통신을 위해 사용하였다.

STOMP 프로토콜

STOMP는 메시징 프로토콜로, WebSocket 위에서 메시지를 전송하기 위한 규약을 정의한다. WebSocket과는 달리, 메시지의 목적지(destination)를 지정하여 전송할 수 있는 구조로 되어 있으며, 발행-구독(pub-sub) 메시징 패턴을 쉽게 구현할 수 있다. 클라이언트는 미리 정의해둔 토픽을 구독하여 발행자가 발행한 데이터를 수신받을 수 있다.

React 클라이언트가 병원, 카메라별로 각각의 경로를 구별하여 구독할 수 있도록 사용하였다.

2.3.5. 클라우드

Docker

Docker는 애플리케이션과 종속성을 컨테이너라는 격리된 환경에 패키징하는 컨테이너 기반 가상화 기술이다. Docker를 사용하면 개발, 테스트, 운영 환경에서 동일한 컨테이너를 사용할 수 있어 환경의 일관성을 유지할 수 있다. 컨테이너를 이미지로 만들어 배포한다는 특징이 있다.

ECR

ECR은 AWS에서 제공하는 관리형 Docker 컨테이너 이미지 저장소 서비스다. 개발, 테스트, 운영 환경에 사용한 이미지를 관리하기 위해 사용하였다.

ECS (EC2 기반)

ECS는 Docker 컨테이너를 오케스트레이션하고, 클러스터 기반으로 컨테이너 배포 및 관리와 확장을 자동화하는 서비스다. ECR과 연계하여 Docker 이미지를 불러올 수 있으며 가상 서버 환경인 EC2 인스턴스에 컨테이너를 배포하고 실행할 수 있다. 태스크 단위로

컨테이너가 실행되며, 최소 태스크 수와 최대 태스크 수를 설정하여 오토 스케일링 환경을 구성할 수 있다.

추후 플랫폼 트래픽이 많아지게 될 경우를 대비해서 컨테이너 오케스트레이션 및 오토 스케일링 서비스를 제공하는 ECS를 활용했다.

ALB

ALB는 AWS의 로드 밸런싱 서비스로, 서버에 가해지는 HTTP 및 HTTPS 트래픽을 분산해서 처리하는 데 특화되어 있다. 요청 기반의 라우팅, WebSocket 지원, 리스너 규칙 등을 통해 트래픽을 다양한 대상 그룹으로 분산시킬 수 있다. 여러 대의 서버를 둘 경우를 대비하여 적절한 트래픽 분산을 위해 로드 밸런서를 사용하였다.

S3

S3는 AWS의 객체 스토리지 서비스로, 대용량 파일 데이터를 안전하게 저장하고 접근할 수 있는 스토리지다. 낙상 사고 영상과 프론트엔드 어플리케이션의 빌드 결과물을 저장하기 위해 사용하였다.

CloudFront

CloudFront는 AWS에서 제공하는 글로벌 콘텐츠 전송 네트워크(CDN) 서비스다. 전 세계에 분산된 엣지 로케이션을 통해 사용자에게 더 빠르게 콘텐츠를 제공할 수 있다. CloudFront와 S3를 연계하면 다양한 장점을 제공할 수 있는데, CloudFront의 엣지 로케이션을 통해 사용자가 S3에 저장된 콘텐츠를 요청할 때, 가까운 위치에서 캐싱된 콘텐츠를 전달하여 빠른 응답을 제공한다. CloudFront의 보안 기능(SSL/TLS)을 활용해, S3에서 제공하는 콘텐츠에 대한 보안 설정을 강화할 수 있다.

3. 연구 내용

3.1. H/W

3.1.1. 라즈베리파이 및 센서 환경 구축

라즈베리 3B+와 마이크를 결합한 후 seeed studio 공식 문서를 따라 드라이버를 설치하였다. 5초간 녹음한 음향을 HTTP Request를 통해 전송하도록 하였다.

라즈베리 4와 열화상 카메라를 결합한 후 waveshare 공식 문서가 제공하는 라즈베리파이 OS 이미지를 이용하였다. 마찬가지로 5초간 10FPS로 촬영한 50장의 열화상 이미지들을 HTTP Request를 통해 전송하도록 하였다.

3.1.2. Jetson AGX Xavier 환경 구축

Nvidia SDK Manager를 이용하여 Jetson AGX Xavier에 JetPack 5.1.3을 설치하였다. JetPack 설치 시 CUDA Toolkit이 함께 설치되어 추가적인 GPU 환경 구축은 필요하지 않았다. 이후 FastAPI 기반 웹 서버를 구축하여 라즈베리파이로부터 전달받은 정보를 이용하여 딥러닝 기반으로 안전사고 발생 여부를 탐지한 후 병원 관리 서버로 전달한다.

3.1.3. 공인 IP 신청 및 포트포워딩

장치/서비스	공인 IP:포트	사설 IP:포트
Jetson/Web	public:8000	192.168.0.118:8000
Jetson/ssh	public:8001	192.168.0.118:22
열화상 RPI/ssh	public:8002	192.168.0.43:22
마이크 RPI/ssh	public:8003	192.168.0.43:22

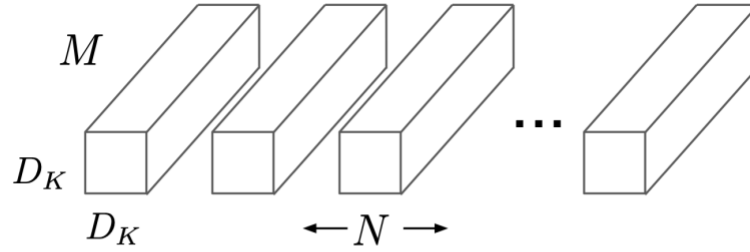
<표 3> 장치들의 서비스별 포트포워딩 목록

병원 관리 서버와 Web Socket 통신을 이용하여 스트리밍 및 사고 정보를 전송한다. 라즈베리파이와 Jetson AGX Xavier는 학교 내부망을 사용하는 WiFi 공유기와 연결되어 있어 외부에서 접속이 불가능하였다. 정보화본부에 공인 IP NAT 신청을 통해 WiFi 공유기에 공인 IP를 할당받은 후, 포트포워딩을 설정하여 서비스 별로 접속할 수 있게 설정하였다. 개발의 용이성을 위해 ssh 포트도 포트포워딩을 설정하여 진행하였다. 사용한 포트포워

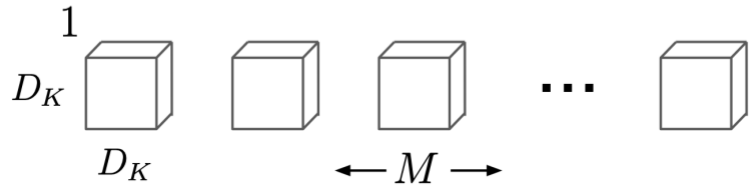
딩 목록을 <표 3>에 나타내었다.

3.2. 인공지능

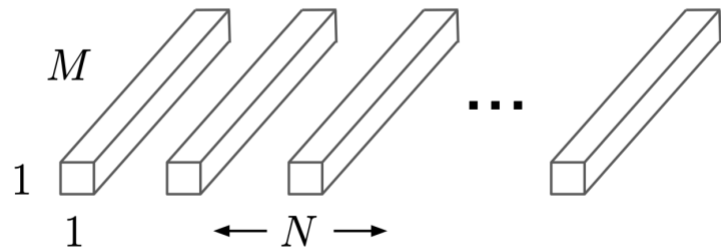
3.2.1. 음향 기반 낙상 분류 모델



(a) Standard Convolution Filters

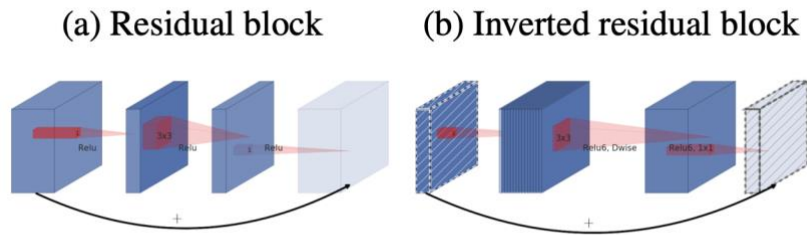


(b) Depthwise Convolutional Filters

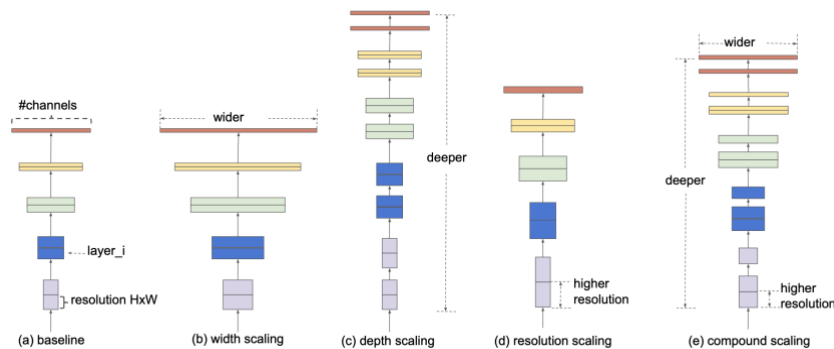


(c) 1×1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

<그림 13> MobileNetV1에 사용되는 Depthwise Separable Convolution



<그림 14> MobileNetV2에 사용되는 Inverted residual block



<그림 15> EfficientNet-B0에 사용되는 compound scaling

플랫폼의 실시간성을 고려하여 MobileNetV2[6]와 EfficientNet-B0[7]을 음향 분류 모델로 사용할 수 있다. MobileNetV2는 <그림 13>의 MobileNetV1[8]에서 제안한 Depthwise Separable Convolution을 이용하여 연산량과 모델 사이즈를 줄이며, <그림 14>와 같은 Inverted Residual 구조를 이용하여 작은 영역에 압축된 정보를 skip connection을 통해 모델의 깊은 층까지 전달해 주는 것이 특징이다. EfficientNet-B0은 <그림 15>에 설명된 모델의 넓이, 깊이, 해상도를 모두 고려한 scaling인 compound scaling을 통해 효율적인 파라미터 수를 갖는 모델이다.

본 과제에서는 채널 수가 3인 RGB 이미지를 처리하기 위해 설계된 MobileNetV2과 EfficientNet-B0이 채널 수가 1인 MFCC를 처리할 수 있도록 입력층의 채널 수를 3에서 1로 변경하였다. 또한 2개의 클래스를 분류할 수 있도록 출력 수를 2로 변경하였다.

또한 모델의 학습 시간을 단축하고, 높은 성능을 달성하기 위해서 모델을 처음부터 학습하는 것이 아닌, 사전 학습된 모델을 미세 조정하는 방법을 사용하였다. 본 과제에서는 ImageNet으로 사전 학습된 MobileNetV2과 EfficientNet-B0를 미세 조정하여 모델을 획득한 후 성능을 비교하였다.

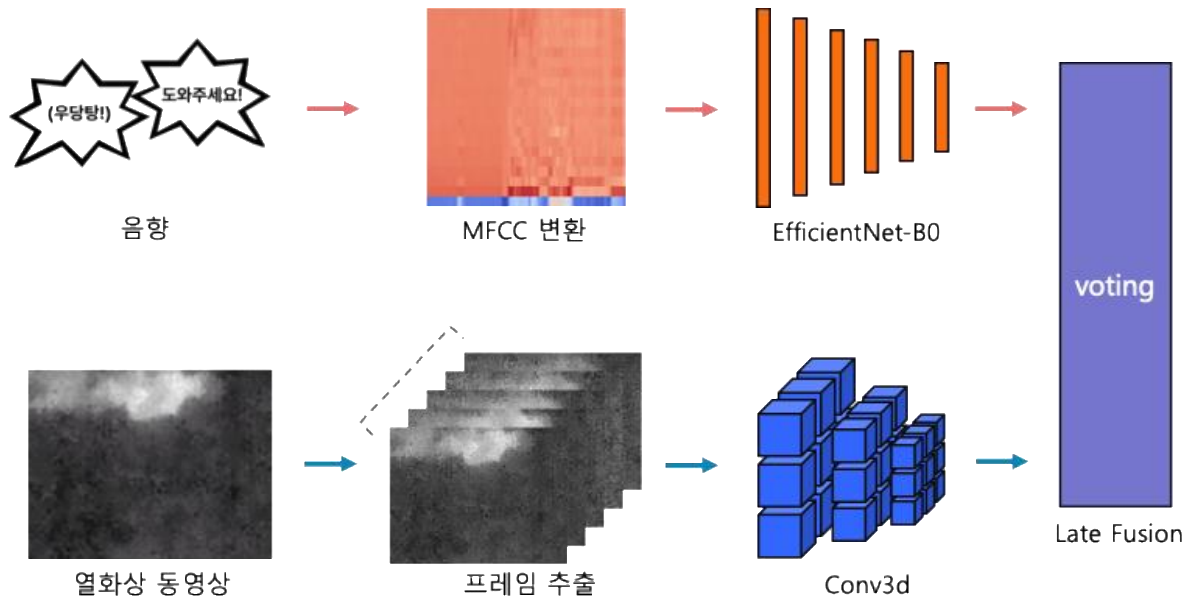
3.2.2. 열화상 동영상 기반 낙상 분류 모델

Layer (type:depth-idx)	Output Shape	Param #
BaseModel	[1, 2]	--
└Sequential: 1-1	[1, 256, 1, 1, 1]	--
└Conv3d: 2-1	[1, 8, 50, 62, 62]	80
└ReLU: 2-2	[1, 8, 50, 62, 62]	--
└BatchNorm3d: 2-3	[1, 8, 50, 62, 62]	16
└MaxPool3d: 2-4	[1, 8, 50, 31, 31]	--
└Conv3d: 2-5	[1, 32, 50, 29, 29]	2,336
└ReLU: 2-6	[1, 32, 50, 29, 29]	--
└BatchNorm3d: 2-7	[1, 32, 50, 29, 29]	64
└MaxPool3d: 2-8	[1, 32, 50, 14, 14]	--
└Conv3d: 2-9	[1, 64, 50, 12, 12]	18,496
└ReLU: 2-10	[1, 64, 50, 12, 12]	--
└BatchNorm3d: 2-11	[1, 64, 50, 12, 12]	128
└MaxPool3d: 2-12	[1, 64, 50, 6, 6]	--
└Conv3d: 2-13	[1, 128, 50, 4, 4]	73,856
└ReLU: 2-14	[1, 128, 50, 4, 4]	--
└BatchNorm3d: 2-15	[1, 128, 50, 4, 4]	256
└Conv3d: 2-16	[1, 256, 50, 2, 2]	295,168
└ReLU: 2-17	[1, 256, 50, 2, 2]	--
└BatchNorm3d: 2-18	[1, 256, 50, 2, 2]	512
└AdaptiveAvgPool3d: 2-19	[1, 256, 1, 1, 1]	--
└Linear: 1-2	[1, 2]	514
=====		
Total params: 391,426		
Trainable params: 391,426		
Non-trainable params: 0		
Total mult-adds (M): 364.90		
=====		
Input size (MB): 0.82		
Forward/backward pass size (MB): 55.96		
Params size (MB): 1.57		
Estimated Total Size (MB): 58.35		
=====		

<그림 16> Conv3D 모델의 파라미터 정보

직접 구축한 열화상 동영상 데이터셋에 적합한 분류 모델을 찾기 위해 여러 모델을 학습한 후 성능을 평가하였다. 성능의 베이스라인을 잡기 위해 일반적인 Conv3D 모델을 구현하였다. <그림 16>는 구현한 Conv3D의 파라미터에 대한 정보이다. 그리고 사람의 여러 가지 행동이 담긴 데이터셋인 Kinetics-400[8]으로 사전 학습한 S3D[9], R3D-18[10]을 미세 조정하였다. S3D는 기존의 계산 비용이 큰 3D 컨볼루션 연산을 공간을 위한 2D 컨볼루션 연산과, 시간을 위한 1D 컨볼루션 연산으로 분리한 3D 컨볼루션을 사용한다는 것이 특징이다. R3D-18은 3D 컨볼루션 연산과 ResNet[11]의 잔차 연결을 이용하는 모델이다. S3D와 R3D-18의 경우 채널 수가 3인 RGB 이미지를 대신 채널 수가 1인 열화상 이미지를 처리해야 하므로 입력층의 채널 수를 3에서 1로 변경하였다.

3.2.3. 멀티모달 모델



<그림 17> 각 모달의 추론 결과를 취합하는 멀티 모달 모델

멀티모달 모델은 <그림 17>과 같이 음향 기반 낙상 분류 모델의 추론 결과와 열화상 동영상 기반 낙상 분류 모델의 추론 결과를 voting하는 Late Fusion 방식을 통해 안전사고를 탐지한다.

3.2.4. 태스크 오프로딩

효과적인 딥러닝 추론을 위해 라즈베리파이에서 수집한 데이터를 Jetson AGX Xavier로 태스크 오프로딩한다. 태스크 오프로딩 구현을 위해 FastAPI를 사용하였다. FastAPI는 Python 기반 웹 프레임워크로, 비동기 처리에 특화되어 있어 비교적 긴 시간이 소요되는 딥러닝 추론을 제공하는 서버를 제작하기에 적합하다. 소켓 프로그래밍, SCP(Secure Copy Protocol)를 이용한 방법도 고려하였으나, 구현의 용이함을 위해 FastAPI를 선정하였다.

3.2.5. 딥러닝 모델 최적화

더욱 빠르고 효율적인 딥러닝 추론을 위해 TensorRT를 이용할 수 있다. TensorRT는 NVIDIA의 딥러닝 추론 속도 가속 라이브러리이다. 이때 모델의 기본 파라미터 표현 단위인 FP32에서 INT8 등으로 양자화를 적용할 수 있는데, 모델의 학습 후 양자화를 적용

하는 PTQ(Post Training Quantization)과 학습 중 양자화를 시뮬레이션하는 QAT(Quantization Aware Training)이 있다. 양자화 과정에서는 파라미터 표현 비트 수의 감소로 정확도의 하락이 필연적인데, QAT의 경우 PTQ에 비해 그 성능 하락의 폭이 작다고 알려져 있다. 양자화를 진행하기 위해 양자화 엔진이 필요한데, 진행해 본 결과 Jetson AGX Xavier의 아키텍처인 ARM64에서는 해당 엔진을 지원하지 않았다. Desktop에서 양자화가 완료된 모델을 Jetson AGX Xavier에서 실행하여도 양자화된 파라미터를 인식할 수 없었다.

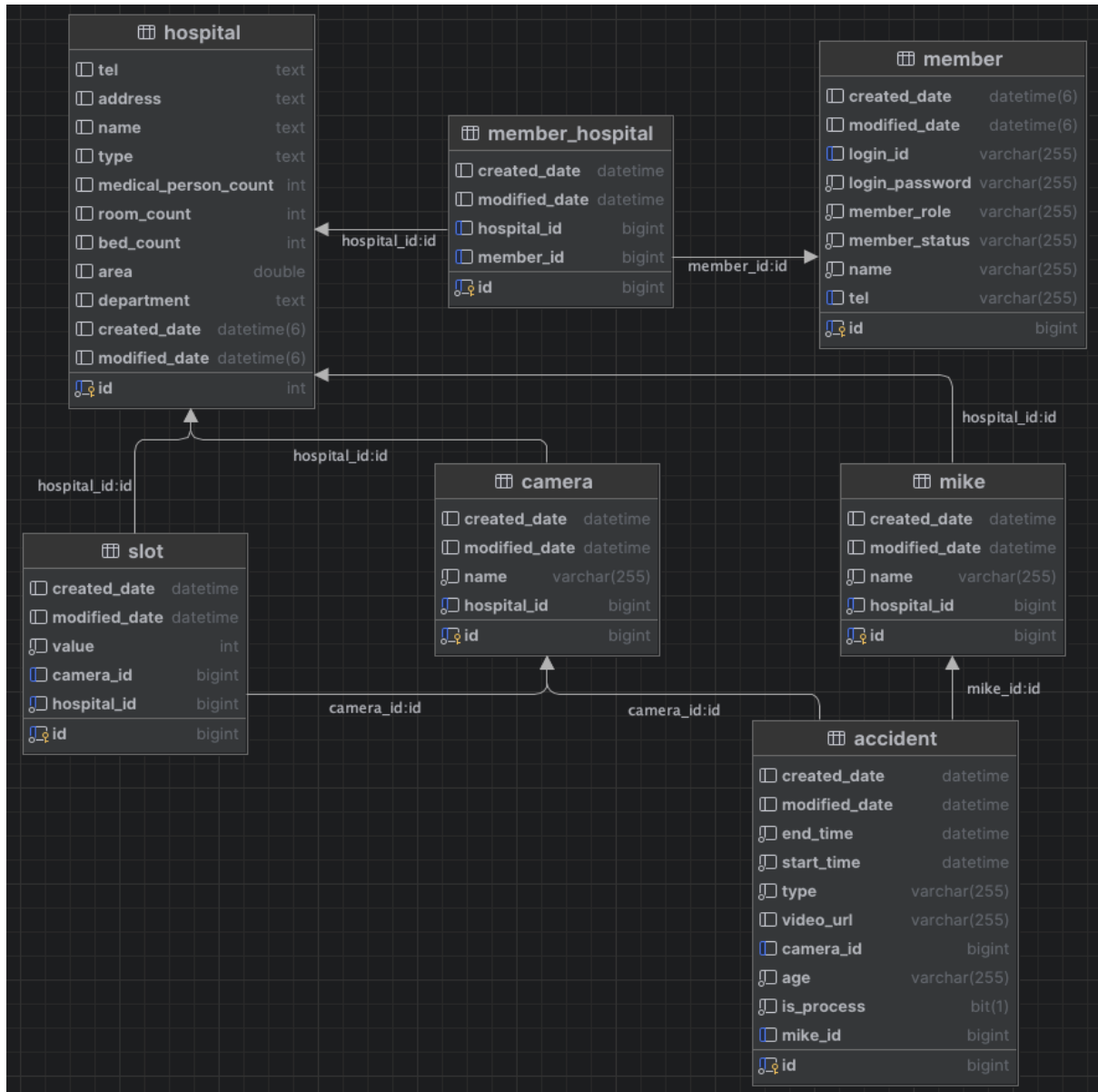
	Desktop	Jetson AGX Xavier
아키텍처	x86_64	ARM64
OS	Ubuntu 20.04 LTS	Ubuntu 20.04 LTS
CUDA	11.8	11.4(JetPack 5.1.3)
PyTorch	2.4.0+cu118	2.1.0a0+41361538.nv23.06
양자화 엔진	'qnnpack', 'none', 'onednn', 'x86', 'fbgemm'	'none'

<표 4> 과제에 사용한 컴퓨터의 정보

따라서 양자화는 진행하지 않고 torch2trt 라이브러리를 이용하여 PyTorch 모델을 CUDA에 최적화된 연산이 적용된 TensorRT 모델로 변환하여 추론 속도를 가속화할 수 있도록 하였다.

3.3. 플랫폼

3.3.1. 데이터베이스 테이블



<그림 18> 데이터베이스 다이어그램

member

서비스에 가입한 회원의 정보를 저장하기 위한 테이블이다.

hospital

병원 정보를 저장하기 위한 테이블이다. 정부에 등록된 병원만이 서비스에 가입할 수 있도록 하기 위해 실제 정부의 전국 병원 데이터를 다운받아 저장하였다. 이를 통해, 정부에 서비스에 가입하려는 병원의 신뢰성을 높일 수 있고 등록된 병원만을 대상으로 서비스를 제공함으로써 사용자에게 높은 품질의 서비스를 제공할 수 있다.

member_hospital

회원과 병원의 관계를 설정하기 위한 중간 테이블이다. 확장성을 고려하여 여러 명이 하나의 병원을 관리할 수 있고, 한 회원이 여러 개의 병원을 관리하는 것이 가능하도록 설계하였다.

camera,

병원에 등록된 카메라의 정보를 저장한 테이블이다.

mike

병원에 등록된 마이크의 정보를 저장한 테이블이다.

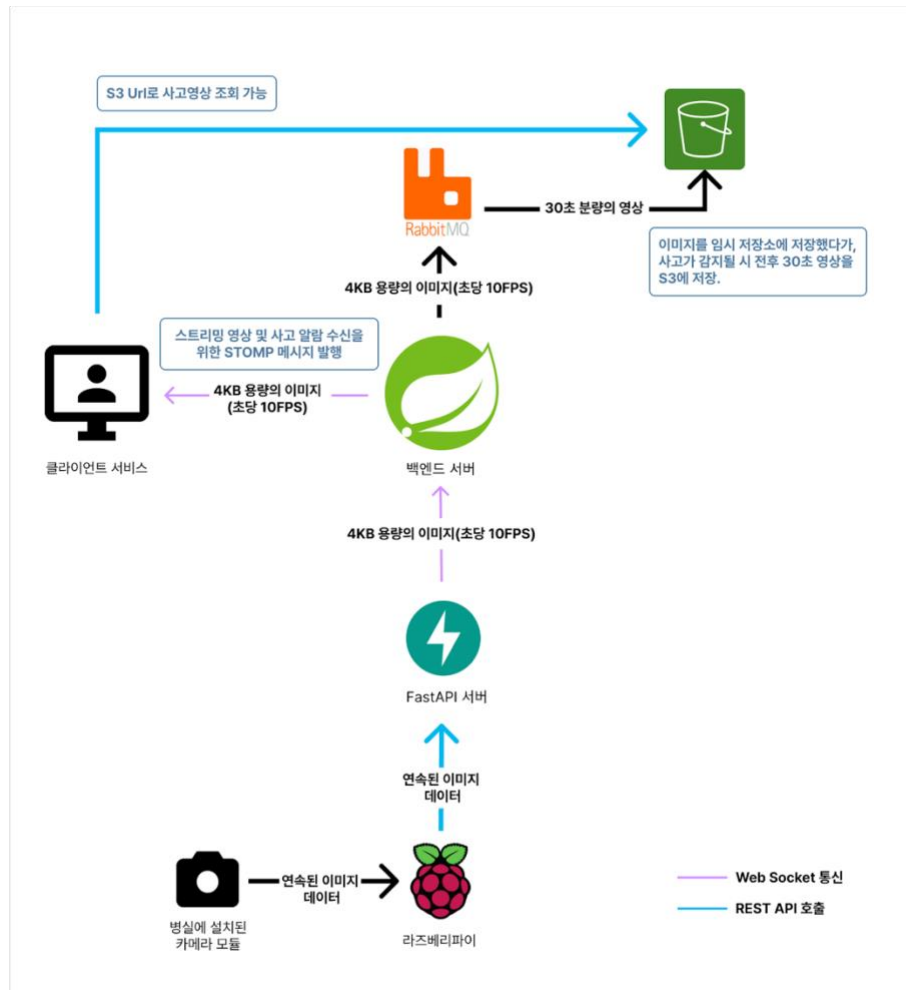
slot

병원에는 모니터링을 위한 슬롯이 있고, 슬롯에 카메라를 지정하여 모니터링할 카메라를 변경하고 실시간으로 영상을 조회할 수 있다.

accident

사고의 정보를 저장한 테이블이다. 사고 시간, 유형, 영상 S3 URL 등을 저장하고 있다.

3.3.2. 실시간 동영상 스트리밍



<그림 19> 실시간 동영상 스트리밍 설계

병원에 설치된 카메라 및 음성 모듈로부터 저장된 데이터를 라즈베리파이에서 Fast API 로 HTTP 통신을 통해 전송한다. 이미지를 전달받은 FastAPI 서버는 임포트된 AI 모델의 API를 호출하여 낙상사고 발생 여부를 판단한다. 이때 FastAPI 서버가 실행될 때 Spring 서버로 WebSocket 통신 연결을 미리 수립한 상태이며, 이미지와 사고 발생 여부를 Spring 서버로 WebSocket 통신을 통해 전송한다. 이때 이미지 파일의 경우, Base64로 인코딩한 값을 JSON 형식으로 전송했다.

WebSocket 통신으로 이미지 데이터를 전달받은 Spring 서버는 RabbitMQ에 30초의 TTL을 설정하여 데이터를 전송한다. 이때, 큐의 이름은 각각의 카메라에 대해 고유하게 설정하여 이미지 데이터가 섞이지 않도록 설정했다. 클라이언트에게는 사전에 정의된 "/ws/topic/image/hospitals/{hospitalId}/cameras/{cameraId}" 토픽으로 데이터를 발행한다.

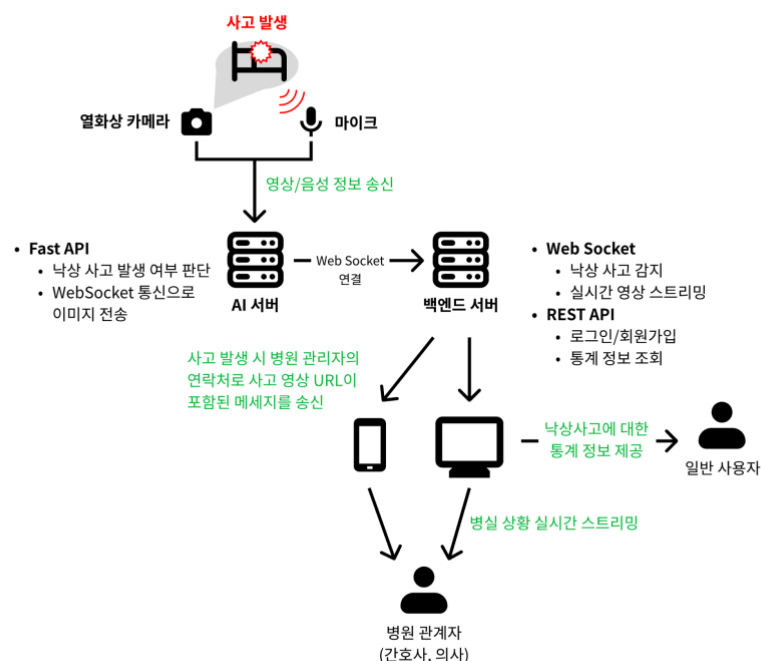
사고 발생 여부를 구독할 수 있는 토픽 경로는 `"/ws/topic/accident/hospitals/{hospitalId}"` 이다.

서버에 사고 발생 신호가 전달되었을 시 RabbitMQ에 저장되어 있던 이미지를 영상으로 변환하여 S3에 저장하여 추후 클라이언트가 사고 영상을 다시 조회할 수 있도록 하였다. 서버 포인트에서 이미지를 영상으로 변환하는 이유는, FastAPI 서버는 최대한 AI 모델만 구동하도록 다른 작업은 Spring 서버가 처리하는 방향으로 결정했기 때문이다.

또한, 카메라 모듈이 제공하는 스크립트는 촬영 시 바로 동영상으로 제공하지 않고 이미지 프레임들을 순차적으로 제공한다. 또한 Jetson AGX Xavier에서 열화상 동영상 기반 낙상 탐지 모델의 입력은 이미지 프레임들이다. 따라서 라즈베리파이와 Jetson AGX Xavier에서는 이미지 프레임들을 동영상으로 변환하지 않고, Spring 서버에서 동영상 변환 처리를 진행한다.

S3에 저장된 사고 영상은 평균적으로 40KB 정도의 크기를 갖고 있고, 추후에 열람이 가능하도록 서비스 운영 기간에는 영구적으로 S3 버킷에 저장할 예정이다. S3 버킷은 대용량 파일 저장에 최적화된 기술이기에 무리는 없다고 판단하였다.

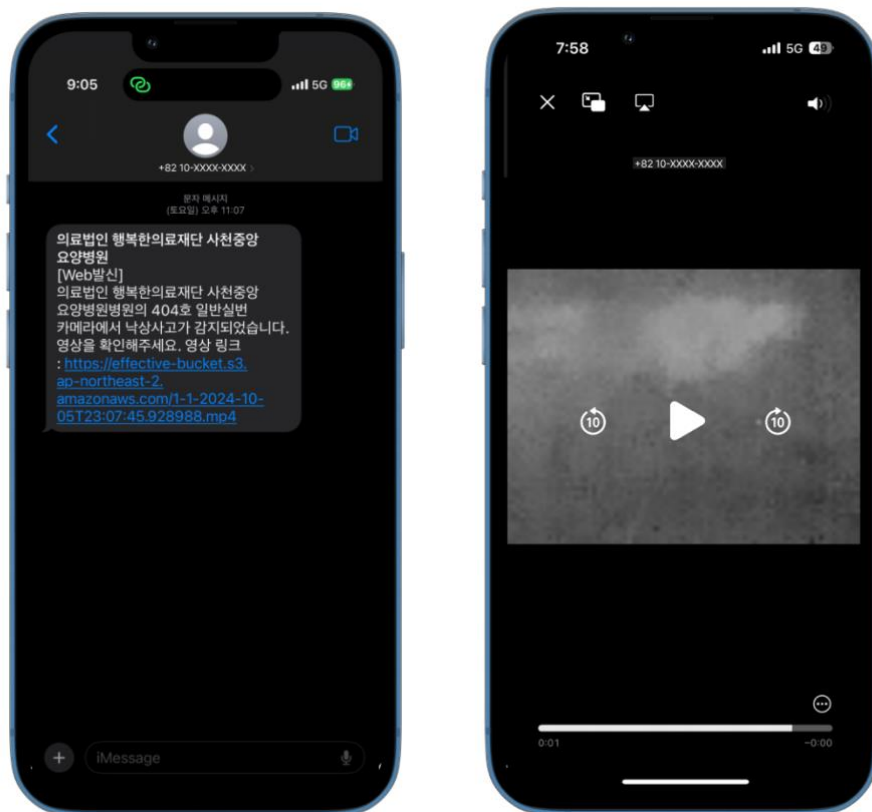
3.3.3. 데이터 흐름



<그림 20> 데이터 흐름도

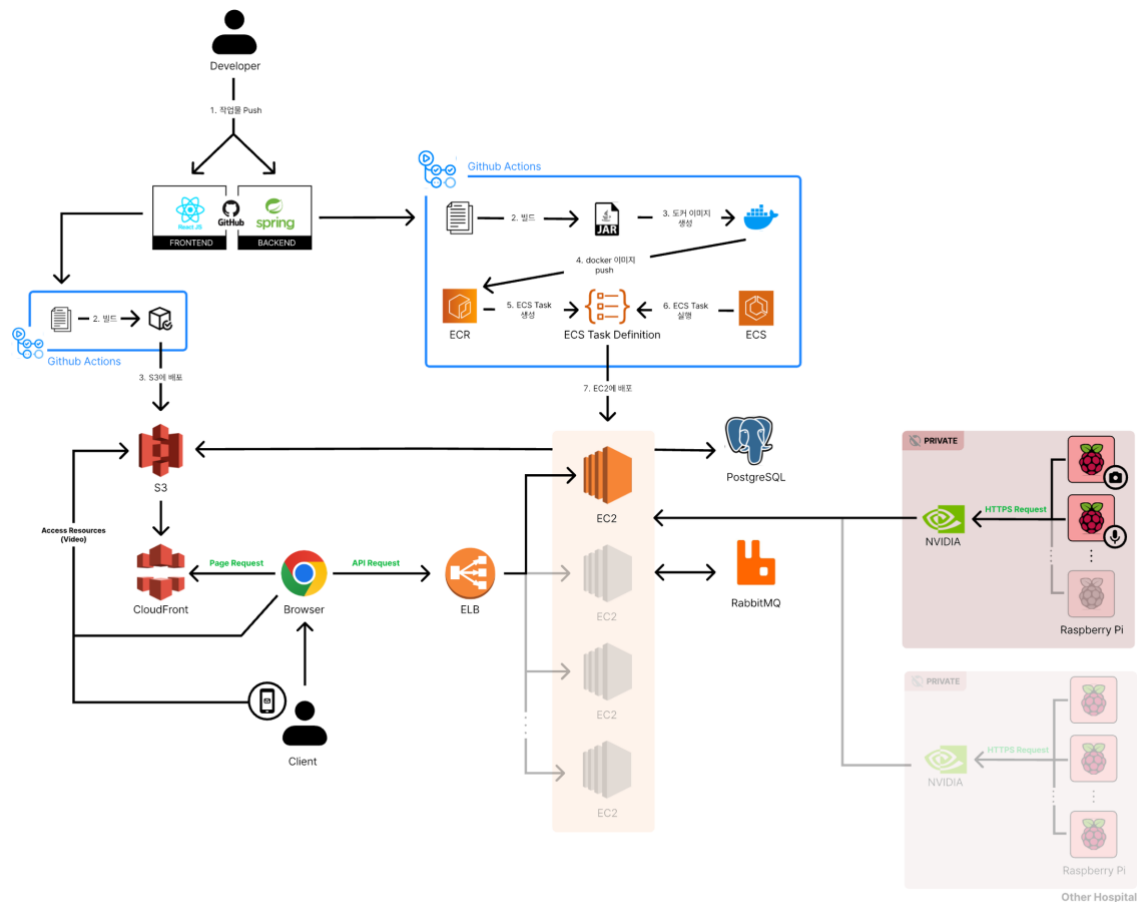
데이터 흐름도는 <그림 20>과 같다. 열화상 카메라 모듈과 마이크 모듈이 데이터를 수집하고 있으며, FastAPI 서버에서 낙상 사고 발생 여부를 판단한다. FastAPI 서버는 이미지 데이터와 낙상 사고 발생 시그널을 WebSocket 통신을 통해 Spring 백엔드 서버로 전달한다.

Spring 백엔드 서버의 경우 서비스 API들은 REST API로 제공하고 있으며, WebSocket 통신을 통해 낙상 사고 감지 시 클라이언트(React)가 구독한 STOMP 경로로 사고 발생 신호를 보낸다. 또한 이미지를 실시간으로 클라이언트에 보내 관리자가 병실의 상황을 실시간으로 스트리밍하는 것이 가능하게 한다. 백엔드 서버는 사고 영상 저장까지 모두 끝나고 나면, <그림 21>과 같이 SMS 문자 발송을 통해 병원 관리자에게 낙상 사고 발생 사실을 알리고 링크에 접속하여 낙상 사고 영상을 조회할 수 있다.



<그림 21> 낙상 사고 영상 URL이 포함된 SMS

3.3.4. 인프라 구조



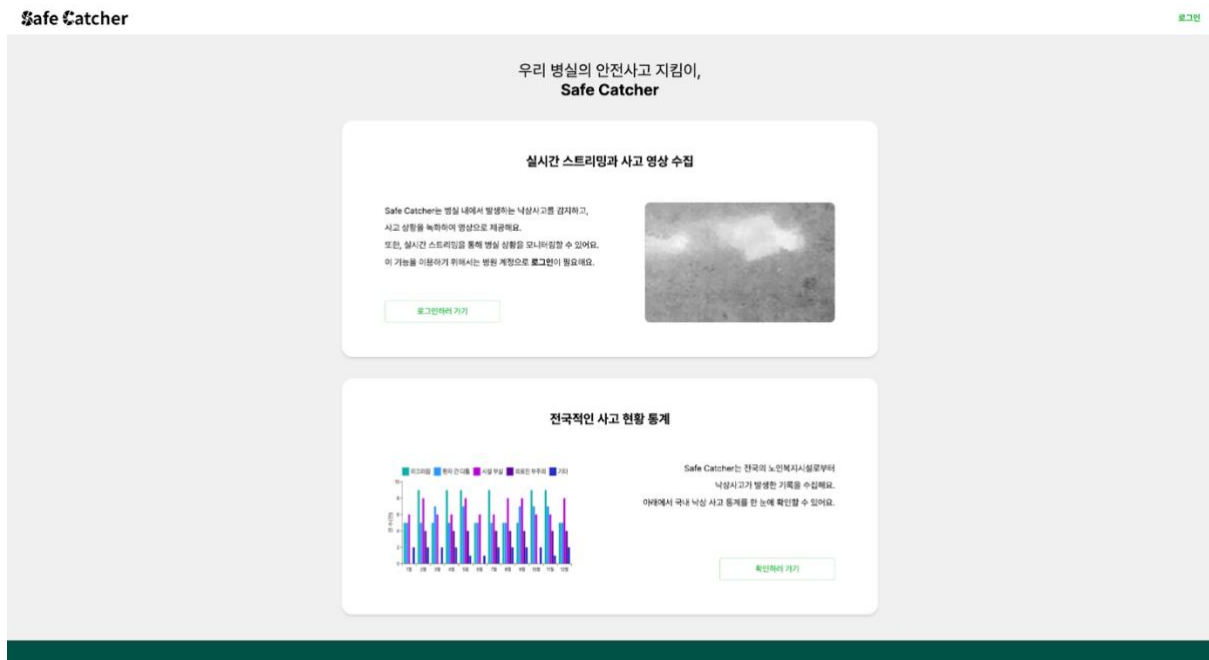
<그림 22> 인프라 구조도

인프라 구조는 <그림 22>과 같다. 병원 관리 서버는 Github Actions를 이용하여 개발이 완료될 시 자동으로 CI/CD를 통해 Amazon EC2 및 S3에 배포된다. 플랫폼 사용자는 CloudFront, ELB를 통해 Page Request와 API Request 시의 QoS가 보장된다. 병원 내에서 라즈베리파이를 통해 수집된 데이터는 FastAPI를 지나며 안전사고 탐지 결과와 함께 EC2로 전달된다. 해당 데이터는 안전사고 탐지 결과에 따라 스트리밍을 위한 RabbitMQ와 사고 발생 통계를 위해 PostgreSQL로 이동한다.

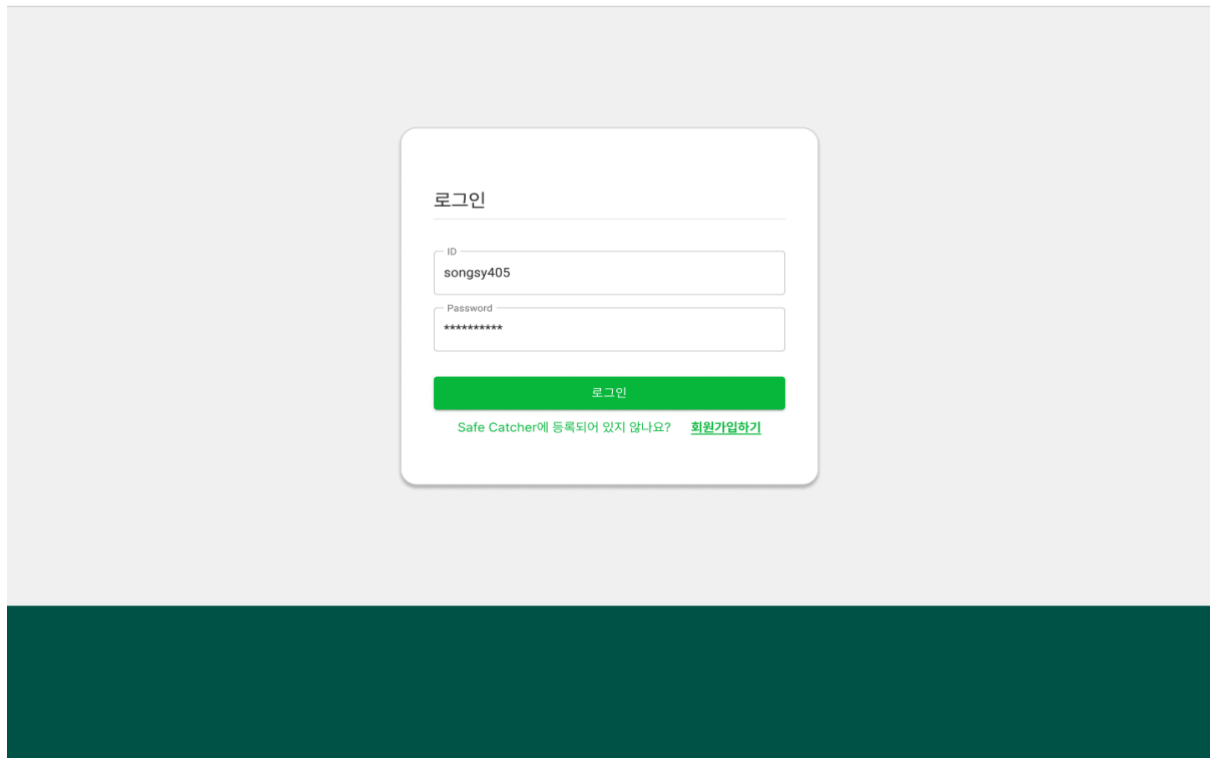
3.3.5. UI/UX

Figma 툴을 활용해 개발하고자 하는 서비스의 기획을 구체화하고 UI&UX를 디자인하였다.

포인트 컬러는 서비스의 주 이용 목적을 고려하여 신뢰도를 높일 수 있는 초록색으로 채택했고 서비스의 이용자 그룹이 정해져 있기 때문에, 핵심 정보를 직관적으로 보여줄 수 있게끔 UI를 구성하였다.



<그림 23> 비 로그인 상태의 메인 페이지

The image shows a web login page for 'Safe Catcher'. At the top left is the 'Safe Catcher' logo, and at the top right is a green '로그인' (Login) link. The main content area has a light gray background. In the center is a white rounded rectangle containing the login form. The form has a title '로그인' (Login) at the top. Below it are two input fields: 'ID' with the value 'songsy405' and 'Password' with masked characters '*****'. A green button labeled '로그인' (Login) is positioned below the password field. At the bottom of the form, there is a link that says 'Safe Catcher에 등록되어 있지 않나요? [회원가입하기](#)' (Not registered on Safe Catcher? [Join as a member](#)). The bottom of the page features a solid dark green horizontal bar.

<그림 24> 로그인 페이지

회원가입

계정 정보

아이디

songsy405

중복확인

비밀번호

비밀번호 확인

병원 정보

병원명

행복요양병원

🔍

병원 유형

요양원

▼

관리자 정보

관리자 이름

홍길동

관리자 전화번호

01012345678

회원가입

<그림 25> 회원가입 페이지

계정 정보 조회

내 병원 정보

행복 요양병원 **요양원**
부산광역시 금정구 62번길 2
Tel: 010-0000-0000

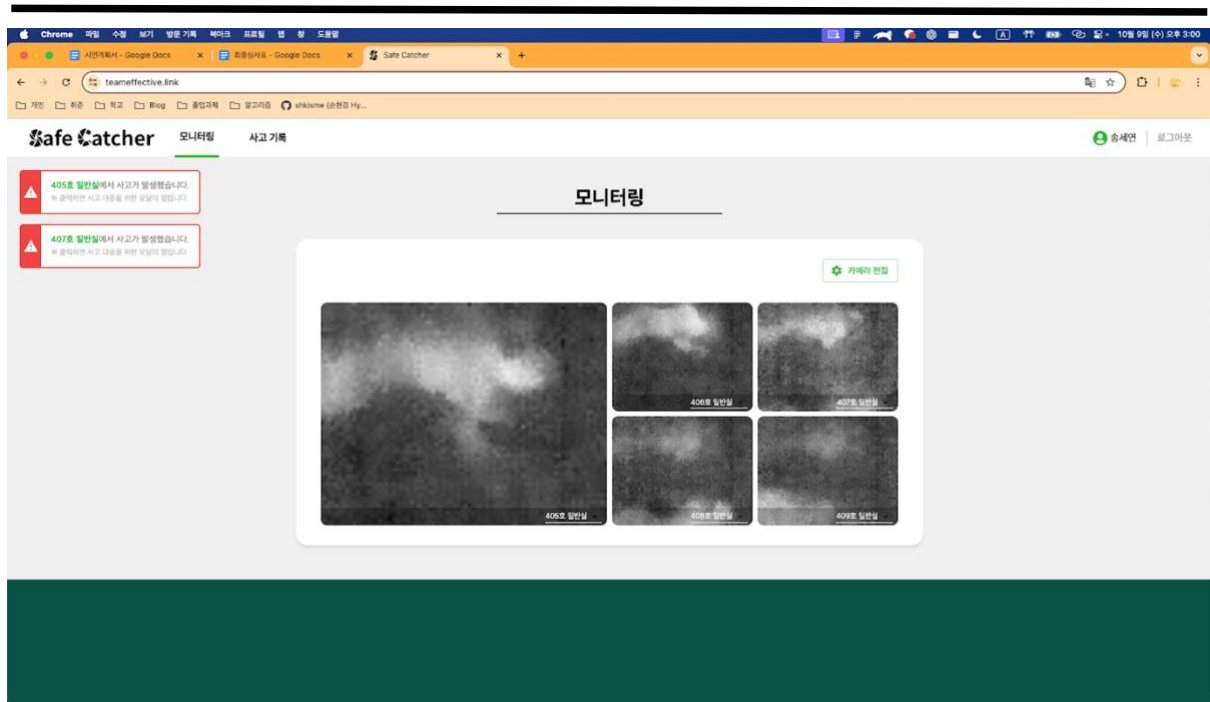
계정 관리자 정보

계정 ID songsy405
관리자 이름 송세연
전화번호 010-1234-5678

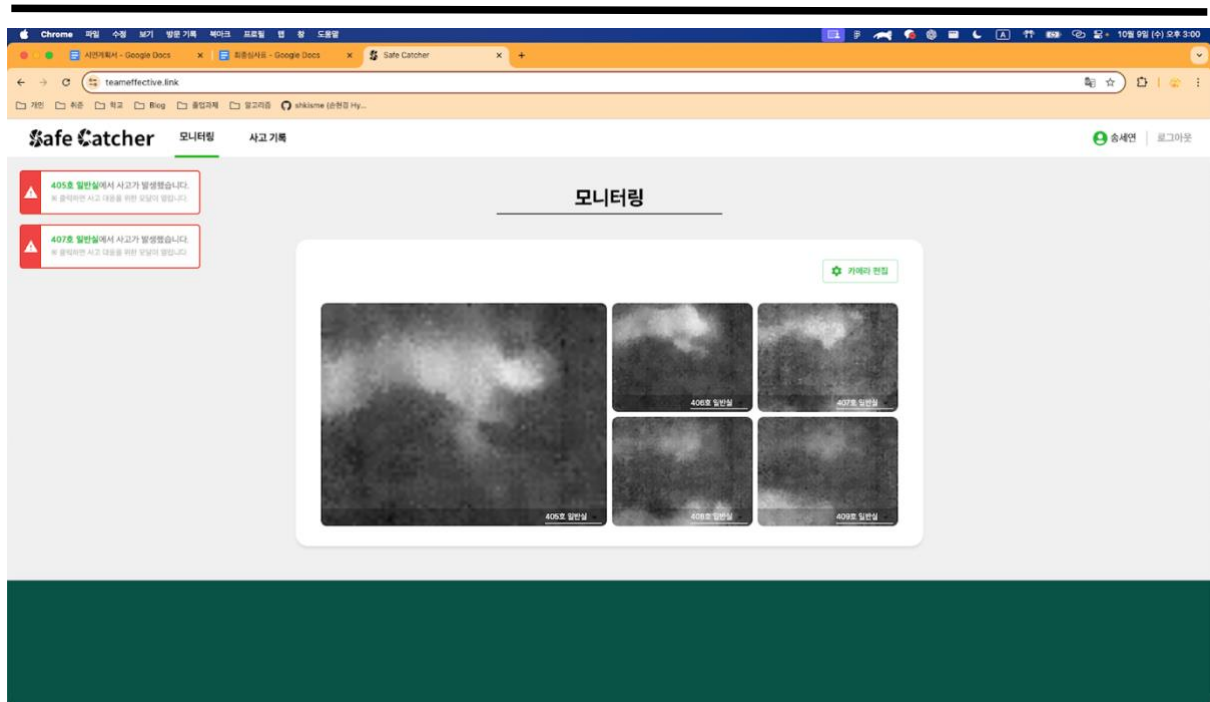
계정 관리

[로그아웃](#)[회원 탈퇴](#)

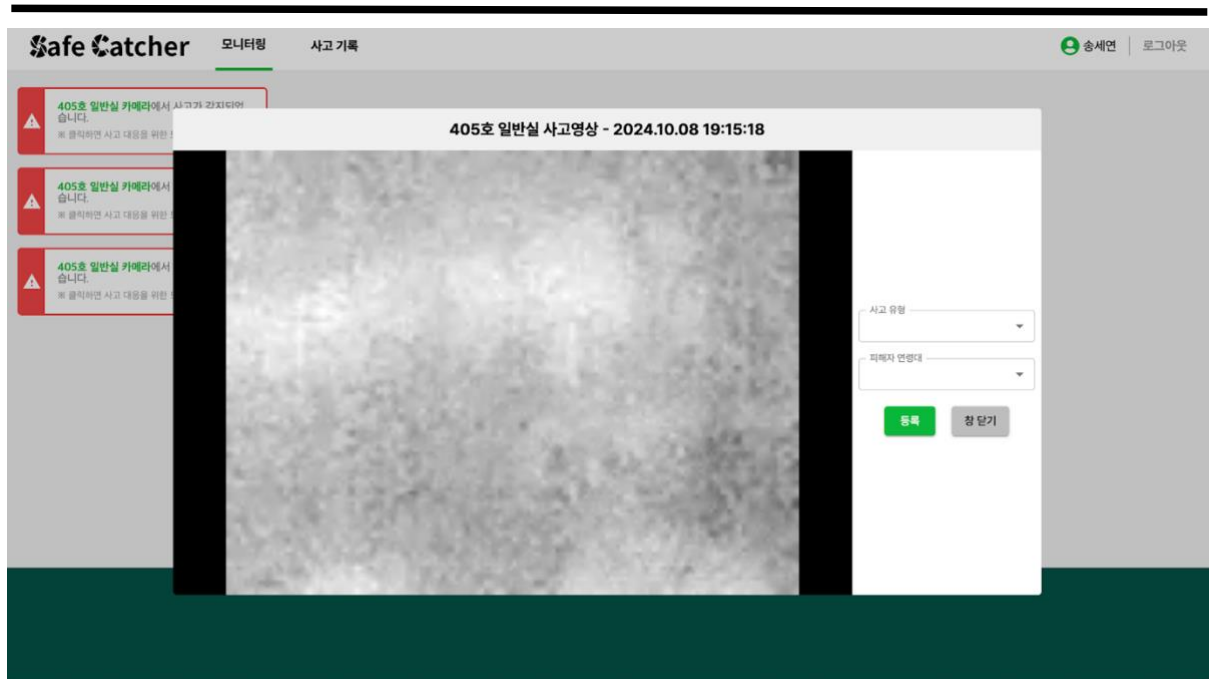
<그림 26> 계정 정보 조회 페이지



<그림 27> 로그인 상태의 메인 페이지(모니터링 페이지)



<그림 28> 사고 발생 시 수신되는 알림 및 사고 라벨링 인터페이스 모달



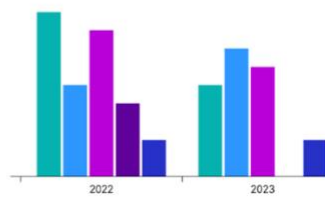
<그림 29> 사고 라벨링 인터페이스 모달 클릭 시

국내 낙상사고 통계

사고 통계

기간 연도
월간 2024
그룹화 기준
원인별

끄러짐 환자 간 다툼 시설 부실 의료진 부주의



2024 누적 사고 건수 82건

2023년 대비
30.5% 감소사고 감지 정확도
98.8%2024 주 사고원인
미끄러짐최다 사고 발생 월
8월최다 사고 발생 연도
2022

사고 이력

No	사고 발생 일자	분류	피해자 연령
1	2023.08.01	미끄러짐	10대
2	2023.09.23	오작동	20대
3	2024.01.01	환자 간 다툼	60~70대
4	2024.04.05	미끄러짐	40대
5	2024.09.07	시설 부주의	50대

Rows per page: 5 1-5 of 13

<그림 30> 국내 낙상사고 통계 페이지

사고 기록



사고 이력

No	사고 발생 일자	분류	카메라
1	2023.08.01	미끄러짐	404호 일반병실
2	2023.09.23	오작동	401호 3인실
3	2024.01.01	환자 간 다툼	401호 3인실
4	2024.04.05	미끄러짐	402호 2인실
5	2024.09.07	시설 부주의	404호 일반 병실

Rows per page: 5
1-5 of 13

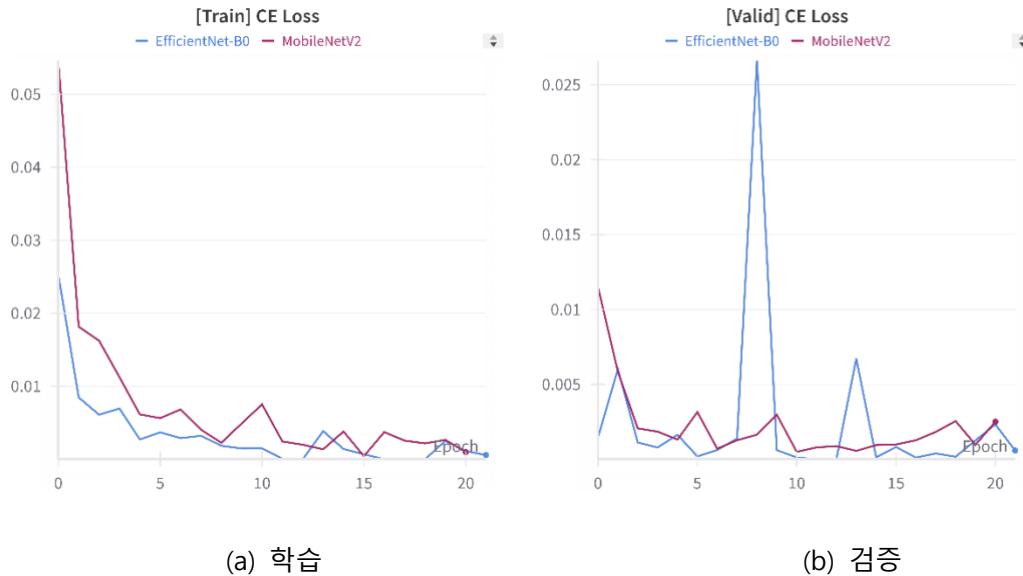
※사고 이력 클릭 시 사고 영상을 열람할 수 있습니다.

<그림 31> 병원 내 낙상사고 통계 페이지

4. 연구 결과 분석 및 평가

4.1. 인공지능

4.1.1. 음향 기반 낙상 분류 모델 성능 평가



<그림 32> 음향 기반 낙상 분류 모델의 손실 그래프

MFCC 연산 시 하이퍼파라미터를 $sr=44,100$, $n_mfcc=40$ 으로 설정하였다. 또한 음성이 5초보다 짧은 경우 0으로 패딩하였으며, 5초보다 긴 경우 뒤 음향을 잘라내었다. 딥러닝 모델에 고정된 크기로 전달하기 위해 (1, 224, 224) 형태로 크기를 조절하였다.

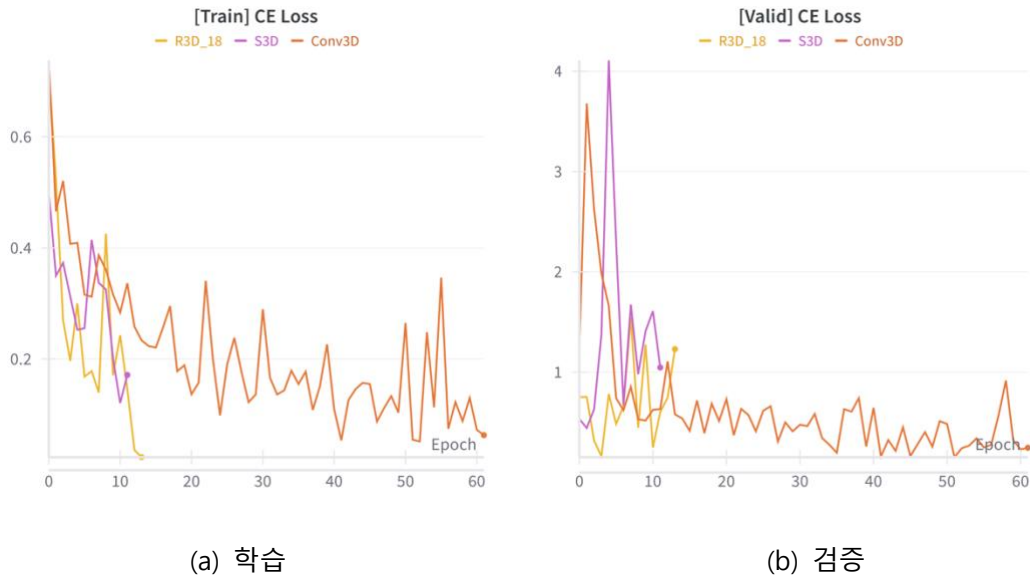
모델 학습 시 하이퍼파라미터는 학습률을 $3e-4$, 배치 사이즈를 8로 설정하여 200 epoch 동안 학습하는 중 10 epoch동안 검증 손실의 성능 향상이 없으면 조기 종료하도록 하여 모델을 획득하였다. 음향 기반 낙상 분류 모델의 성능은 <표 5>와 같다.

모델	MobileNetV2	EfficientNet-B0
F1 score	0.9985	0.9982

<표 5> 음향 기반 낙상 분류 모델의 분류 성능

학습한 두 음성 분류 모델 모두 충분한 성능을 보여준다고 판단하였다. 그러나 MobileNetV2의 모델 구조 중 `has_torch_function_unary`와 `hardtanh` 함수가 TensorRT에서 최적화가 지원되지 않아 EfficientNet-B0을 플랫폼에 사용하였다.

4.1.2. 열화상 동영상 기반 낙상 분류 모델 성능 평가



<그림 33> 열화상 동영상 기반 낙상 분류 모델의 손실 그래프

각 데이터는 5초간 10FPS로 촬영한 50장의 열화상 이미지 프레임으로 구성된다. 마찬가지로 각 이미지를 모델에 동일한 크기로 전달하기 위해 (1, 50, 64, 64) 형태로 크기를 조절하였다. 모델 학습 시 하이퍼파라미터 및 조기 종료 조건은 음성 모델과 동일하다.

모델	Conv3D	S3D	R3D-18
F1 score	0.9095	0.8613	0.8613

<표 6> 열화상 동영상 기반 낙상 분류 모델의 분류 성능

성능 평가 결과 사전 학습 후 미세 조정을 진행한 모델인 S3D와 R3D-18의 성능이 낮게 나왔다. 모델의 깊이에 비해 직접 구축한 데이터셋이 단순하여 파라미터들이 원하는 방향으로 학습되지 않았다. 가장 좋은 성능을 보인 Conv3D 모델을 플랫폼에 사용하였다.

4.1.3. 딥러닝 모델 추론 속도 최적화 성능 평가

PyTorch 모델을 TensorRT로 변환하여 최적화한 경우와 그렇지 않은 경우의 성능 비교를 위해 추론 속도를 측정하였다. Lazy loading을 방지하기 위해 추론 연산을 먼저 1회 진행한 뒤, 5번에 걸쳐 측정한 후 평균을 계산하였다.

Step	1	2	3	4	5	Average
PyTorch	0.0403	0.0380	0.0377	0.0406	0.0395	0.0392
TensorRT	0.0056	0.0053	0.0052	0.0053	0.0049	0.0053

<표 7> 음향 기반 낙상 분류 모델의 추론 시간 비교

Step	1	2	3	4	5	Average
PyTorch	0.0280	0.0225	0.0215	0.0215	0.0217	0.0231
TensorRT	0.0062	0.0063	0.0060	0.0062	0.0059	0.0061

<표 8> 열화상 동영상 기반 낙상 분류 모델의 추론 시간 비교

음향 모델의 경우 7.4배, 열화상 동영상 모델의 경우 3.8배 빠른 결과를 보였다. 두 경우 모두에서 최적화를 적용한 TensorRT가 그러지 않은 PyTorch 모델보다 추론 속도가 빨랐으며, 모델 추론 속도가 데이터를 전송하는 시간 간격인 5초보다 짧으므로, 실시간 서비스에 적합함을 확인하였다.

4.2. 플랫폼

4.2.1. 비회원 기능

로그인 및 회원가입

회원가입 시 회원의 ID, 비밀번호, 이름, 전화번호, 병원 식별 ID를 입력해 서비스에 계정을 등록할 수 있고, 로그인을 통해 인증 토큰을 발급받을 수 있다.

전국 안전사고 현황 조회

전국적으로 발생한 안전사고 통계를 조회할 수 있는 기능이다. 시기/유형별 안전사고의 빈도 파악하여 앞으로의 사고 예방 및 관리에 활용할 수 있다.

4.2.2. 회원 기능

로그인한 사용자(병원 관리자)가 이용할 수 있는 기능이다.

병실 실시간 모니터링

관리자가 특정 병실의 상황을 실시간으로 스트리밍하여 조회할 수 있는 기능이다. 이를 통해 관리자는 병실 내 상황을 실시간으로 모니터링할 수 있다.

사고 발생 시 알람 수신

로그인한 계정과 연동된 병원에서 사고가 발생할 경우 알람을 제공하여 신속하게 병원 관리자에게 알림을 제공하여 빠른 대처가 가능하게 한다. 이때 알람 내용에는 사고 전후 10초 분량의 영상을 다운로드할 수 있는 URL 정보가 포함된다.

Toast

안전사고가 감지될 경우 웹 페이지 좌측 상단에 알림을 표시한다.

SMS 메시지

안전사고가 감지될 경우 병원 관리자의 연락처로 SMS 메시지를 송신한다.

감지된 사고를 라벨링하여 등록

사고가 감지되었을 때 사용자가 해당 사고가 발생한 사유 및 피해자의 연령대 등 부수적인 정보를 입력하고 데이터셋에 저장할 수 있다.

병원 내 안전사고 통계 조회

병원 내에서 발생한 안전사고의 발생 추이를 도표로 나타내어 한눈에 파악할 수 있도록 한다. 추가로 지금까지 발생한 사고들의 사고 영상을 비롯한 정보를 열람할 수 있다.

4.2.3. 핵심 기능 구현 상세

사용자 식별을 위한 로그인 & 회원가입

JWT 토큰 기반 인증 방식을 사용. 로그인 시 백엔드 서버로부터 발급받은 토큰을 Redux 및 Cookie를 통해 관리한다. Axios의 interceptor 기능을 활용하여 권한이 필요한 API를 호출할 때 HTTP 요청 헤더의 Authorization 속성에 토큰을 포함하도록 구현하였다.

열화상 카메라 실시간 스트리밍

이미지 정보를 포함하는 메시지를 발행하는 STOMP 채널을 구독하여 열화상 카메라에 녹화된 실시간 영상을 수신한다.

사고 발생 시 알람 기능

사고 발생 시 메시지를 발행하는 STOMP 채널을 구독하여 사고가 감지될 때마다 알람이 표시되도록 구현하였다.

사고 영상 제공

서버로부터 받은 사고 영상 URL을 활용해 사고 영상을 보여주는 뷰어를 화면에 표시하였다.

사고 통계 출력

MUI에서 제공하는 막대그래프 UI를 활용해 일정 기간 발생한 사고 발생 추이를 보여주는 UI를 구현하였다.

4.2.4. 플랫폼 배포

S3에서는 웹 호스팅 기능을 제공하고 있기 때문에, 애플리케이션 빌드 파일을 S3에 저장하면 애플리케이션을 외부에 배포할 수 있다. 이에 소스코드가 업데이트될 때마다 Github Actions에서 애플리케이션을 빌드하고, 빌드 파일들을 S3에 전송하도록 하였다.

이를 CloudFront에 연동하고, Route 53에서 발급받은 도메인 주소를 연결하여 <https://teameffective.link> 으로 배포한 애플리케이션에 접근할 수 있게 하였다.

4.2.5. API

이름	Role	Method	URL
로그인	ALL	POST	/auth/login
로그아웃	ALL, ROLE_ADMIN	POST	/auth/logout
아이디 중복 확인	ALL	GET	/auth/login-id/{login-id}
회원탈퇴	ROLE_ADMIN	DELETE	/auth/signout
회원가입(병원가입)	ALL	POST	/auth/signup

<표 9> Auth 도메인의 API 목록

이름	Role	Method	URL
병원 검색	ALL	GET	/hospitals?=keyword
내 정보 조회	ROLE_ADMIN	GET	/auth/members/me

<표 10> Hospital 도메인의 API 목록

이름	Role	Method	URL
병원에 설치된 카메라 목록 조회	ROLE_ADMIN	GET	/hospitals/\${hospitalId}/cameras
병원에서 모니터링 하고 있는 카메라 목록 조회	ROLE_ADMIN	GET	/hospitals/\${hospitalId}/monitors
병원에서 모니터링 하고 있는 카메라 변경	ROLE_ADMIN	PATCH	/hospitals/\${hospitalId}/monitors

<표 11> Camera 도메인의 API 목록

이름	Role	Method	URL
병원 내 사고 목록 조회	ROLE_ADMIN	GET	/accidents/hospitals/{hospital-id}?pageNumber=&pageSize=
처리되지 않은 사고 정보 조회	ROLE_ADMIN	GET	/accidents/hospitals/{hospitalId}/unprocessed
나잇대 목록 조회	ROLE_ADMIN	GET	/accidents/ages
사고원인 목록 조회	ROLE_ADMIN	GET	/accidents/types

<표 12> Accident 도메인의 API 목록

이름	Role	Method	URL
전국사고 건 수 통계 조회(월별)	ALL	GET	/statistics/month?year=
전국사고 건 수 통계 조회(연도별)	ALL	GET	/statistics/year

병원 내 사고 건 수 통계 조회(월별)	ROLE_ADMIN	GET	/hospitals/{hospitalId}/statistics /month?year=
병원 내 사고 건 수 통계 조회(연도별)	ROLE_ADMIN	GET	/hospitals/{hospitalId} /statistics/year
전국사고 성과 통계 조회	ALL	GET	/statistics/performance
병원 내 사고 성과 통계 조회	ROLE_ADMIN	GET	/statistics/performance /hospitals/{hospital-id}
통계가 존재하는 연도 목록 조회	ALL	GET	/statistics/exist/years
병원 내 통계가 존재하는 연도 목록 조회	ROLE_ADMIN	GET	/statistics/exist/years /hospital/{hospitalId}

<표 13> Statistic 도메인의 API 목록

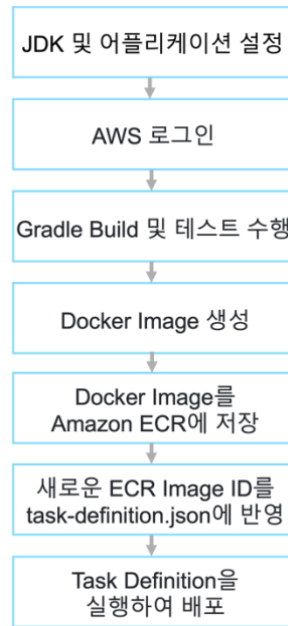
4.3. 공통

4.3.1. 코드 버전 관리 및 CI/CD

Git & GitHub

코드의 버전 관리 도구인 Git과 GitHub를 사용하여 서비스 제작에 필요한 소스 코드를 관리하고 협업했다.

GitHub Actions



<그림 34> CI/CD 구축 순서도

GitHub Actions는 GitHub 리포지토리에서 직접 CI/CD 파이프라인을 구축할 수 있는 자동화 도구이다. <그림 32>와 같이 자동 CI/CD를 위한 워크플로우를 정의하여 애플리케이션 빌드부터 배포까지의 과정을 자동화하여 빠른 주기의 배포 속도를 가질 수 있게 했다.

4.3.2. Notion을 활용한 문서화 및 협업

팀 활동을 위한 Notion 페이지를 생성해 회의록을 포함한 과제 수행을 위해 필요한 자료들을 기록으로 남기고 공유하였다. 특히 개발에 필요한 API 명세를 일관성 있게 문서화함으로써 프론트엔드/백엔드 간에 협업을 원활히 수행할 수 있었다.

5. 결론 및 향후 연구 방향

본 과제에서는 음성 및 열화상 정보를 딥러닝을 이용하여 노인복지시설의 안전사고 발생 여부를 실시간으로 탐지하여 사고 영상을 병원 관계자 및 보호자에게 스트리밍하는 플랫폼을 개발하였다. 이를 통해 사람이 직접 CCTV 영상을 주시하지 않아도 사고 발생을 인지할 수 있도록 하였으며, 스트리밍되는 영상을 통해 빠른 초동 대처를 유도하였으며 향후 환자 - 간호사 간 의료 분쟁 발생 시 증거 자료로 채택될 수 있도록 사고 발생 및 사후 대처 과정을 충실히 녹화하고자 하였다. 본 과제를 통해 차세대 노인복지시설에 적용할 수 있는 AIoT 기술의 프로토타입을 구체적인 구현과 함께 제시하였다.

향후 연구 방향으로 대규모 열화상 동영상 데이터셋을 이용한다면 더욱 정밀한 안전사고 탐지가 가능할 것이다. 또한 이번 과제에서는 멀티 모달 모델이 단순히 음향 또는 열화상 동영상에서의 안전사고 발생 여부를 취합하기만 하고 있으나, 각 모달리티가 더욱 다양한 정보를 전달한다면 이들을 복합적으로 이용하여 안전사고뿐만 아니라 다양한 행동을 인지할 수 있을 것으로 예상된다.

6. 참고 문헌

- [1] Gul, Sania, and Muhammad Salman Khan. "A survey of audio enhancement algorithms for music, speech, bioacoustics, biomedical, industrial and environmental sounds by image U-Net." IEEE Access (2023).
- [2] Mishra, Pratik K., Alex Mihailidis, and Shehroz S. Khan. "Skeletal Video Anomaly Detection Using Deep Learning: Survey, Challenges, and Future Directions." IEEE Transactions on Emerging Topics in Computational Intelligence (2024).
- [3] Baldewijns, Greet, et al. "Bridging the gap between real-life data and simulated data by providing a highly realistic fall dataset for evaluating camera-based fall detection algorithms." Healthcare technology letters 3.1 (2016): 6-11.
- [4] Xu, Yufei, et al. "Vitpose: Simple vision transformer baselines for human pose estimation." Advances in Neural Information Processing Systems 35 (2022): 38571-38584.
- [5] Gatt, Thomas, Dylan Seychell, and Alexiei Dingli. "Detecting human abnormal behaviour through a video generated model." 2019 11th International Symposium on Image and Signal Processing and Analysis (ISPA). IEEE, 2019.
- [6] Sandler, Mark, et al. "Mobilenetv2: Inverted residuals and linear bottlenecks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.
- [7] Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." International conference on machine learning. PMLR, 2019.
- [8] Howard, Andrew G. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." arXiv preprint arXiv:1704.04861 (2017). [6] KAY, Will, et al. The kinetics human action video dataset. arXiv preprint arXiv:1705.06950, 2017.
- [9] Kay, Will, et al. "The kinetics human action video dataset." arXiv preprint arXiv:1705.06950 (2017).
- [10] Xie, Saining, et al. "Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification." Proceedings of the European conference on computer vision (ECCV). 2018.
- [11] Tran, Du, et al. "A closer look at spatiotemporal convolutions for action recognition." Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. 2018.
- [12] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.