네트워크 상태와 영상 특성 기반 실시간 적응형 비디오 스트리밍 시스템

중간 보고서

부산대학교 정보컴퓨터공학부

지도교수 : 김종덕

분과명 : 네트워크/통신(C)

팀명 : BitFlow

팀원 : 202155515 김남희

202155553 박은재

목차

1. 목표

(1) 실시간 네트워크 분석 및 영상 특성 분석

- 서버측은 영상 특성을 분석하여 AI 모델에 적용해 영상 인코딩을 최적화

- 클라이언트 측은 처리량, 버퍼 상태를 기반으로 네트워크 분석을 수행

(2) 영상 인코딩 파라미터 최적화를 위한 AI 모델 개발

- 해당 모델은 영상 특성을 입력 값으로 받고 최적화된 인코딩 파라미터를 출력하여 영상 특성에 맞추어 최적화하여 인코딩

(3) 실시간 적응형 스트리밍 웹 서비스 제공

- dash.js를 활용하여 스트리밍 서비스를 제공하고 시청자들이 웹을 통하여 스트리밍 서비스의 다양한 기능을 사용할 수 있게 함

2. 요구조건 및 제약 사항 분석에 대한 수정사항

2.1 요구조건 수정사항

(1) 안전성 확보 : 사용자의 네트워크 속도가 느려지거나 불안정해지면, 실시간 스트리밍 서비스는 영상의 해상도를 자동으로 낮춘다. 해상도를 낮출수록 영상 데이터의 전송량(비트레이트)이 줄어들어, 갑작스럽게 대역폭이 줄어드는 상황에서도 영상이 끊기지 않고 안정적으로 재생될 수 있다.

(2) 지연 시간 최소화 : 실시간 스트리밍에서 발생할 수 있는 지연 시간을 최소화하기 위해 영상 세그먼트 길이를 조정하고, 영상 특성(움직임 정도)를 AI 모델에 입력하여, 서버에서 각 세그먼트에 대해 예측된 최적의 인코딩 파라미터(예: CRF, max rate)을 동적으로 적용한다. 이로써 엔드-투-엔드 지연 시간을 10초 이하로 유지한다.

(3) 사용자 편의성 : 실제 넷플릭스 웹 UI를 참고하여 재생/일시정지, 볼륨 조절, 전체 화면 모드, 화질 수동 선택, 재생 목록 관리, 회원 관리, 사용자의 선호도에 따른 영상 추천, 영화 인기 차트 등 사용자가 다양한 기능을 편리하게 사용할 수 있도록 구현한다.

2.2 제약 사항 분석 수정사항

(1) 실시간성 구현의 어려움 (지연 시간 최소화)

제약 사항 : AI 모델의 예측 및 적용 과정에서 추가적인 지연을 유발하여, 실시간으로 변화하는 네트워크 환경에 대한 즉각적인 반응 확보가 어렵다는 점에서 난이도가 높다.

대책 : 세그먼트의 길이가 길수록 이에 대한 움직임을 분석하는 데 시간이 많이 걸리므로, AI 모델의 예측 및 적용 시간을 줄이기 위하여 세그먼트의 길이를 1~2초 대로 짧게 유지하여 적용한다. 또한 AI 모델은 비트레이트의 최적화에만 집중하고 프레임 레이트는 고정하는 방안을 고려한다.

(2) 학습 데이터 부족으로 인한 AI 정확도 저하

제약 사항 : AI 모델의 성능은 양질의 충분한 학습 데이터에 크게 의존한다. 그러나 현재 확보된 영상은 약 200여개 정도 밖에 되지 못하여 과적합이 우려되며 일반화 성능이 좋지 못하다.

대책 : 공개된 대규모 영상 데이터셋(Yutube-8M 등)과 네트워크 데이터셋을 활용하여 사전 학습(Pre-training)을 수행하도록 한다. 학습 데이터가 충분히 확보되기 전에는 RandomForest와 같은 머신러닝 모델을 작동시켜 과적합을 방지한다.

(3) 훈련 데이터 생성 및 모델 훈련 시간 초과

제약 사항 : 로컬 환경에서 훈련 데이터를 생성하기 위해 영상의 모션 벡터를 분석하거나, VMAF 품질 평가를 실시하는 데 시간이 지나치게 오래 걸린다.

대책 : Google Colab 환경에서 GPU 혹은 TPU를 사용하여 영상 분석과 모델 훈련 속도를 높일 수 있다. Google Colab은 100개의 컴퓨팅 단위를 무료로 제공하나 이로는 부족하여 Pro+를 구입하여 모델 훈련에 사용한다.

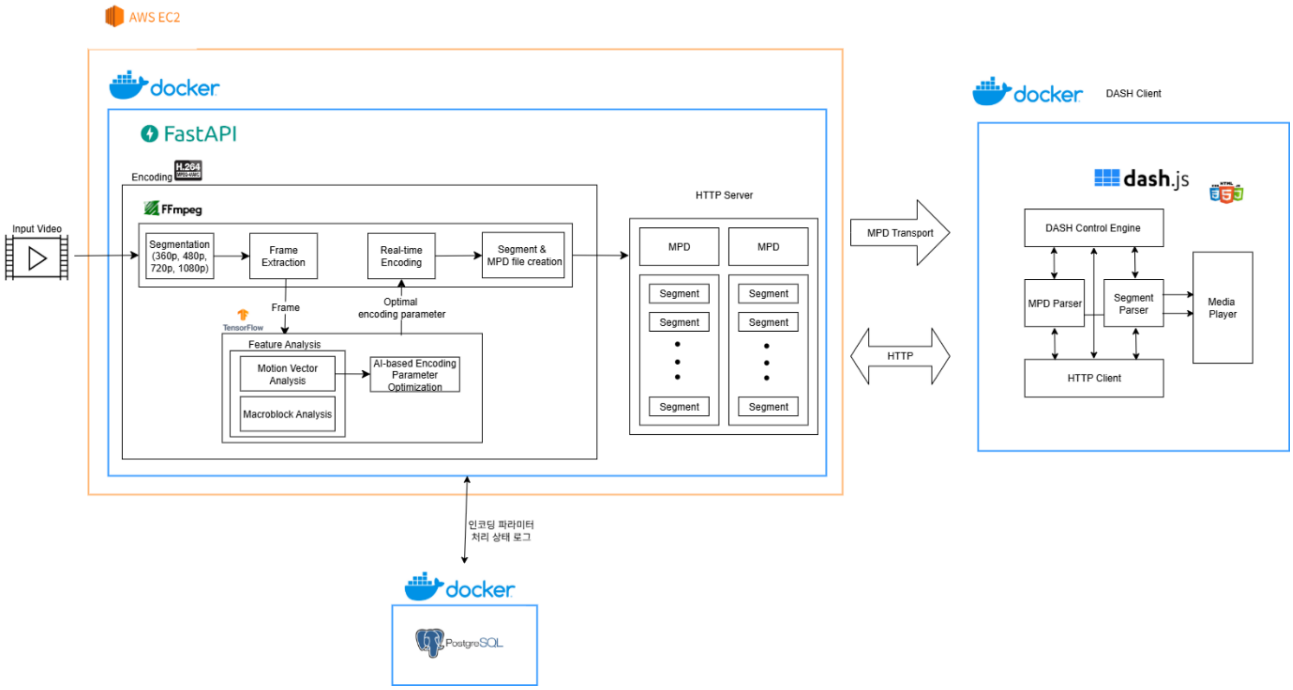
(4) 저작권 이슈로 영상 소스 확보의 어려움

제약 사항 : 넷플릭스 같은 스트리밍 서비스를 구현하는데 실제 영화, 드라마 등의 영상 콘텐츠는 저작권 문제로 인해, 영상 소스를 확보하기 어렵다.

대책 : 무료 영상 여러 개를 준비해서 실시간 적응형 스트리밍을 수행한다. 무료 영상 외에 다양한 영상은 유뷰트에 있는 짧은 영상으로 대체한다. 이 방식은 TMDB API를 활용하여, 영화 정보, 별점, 줄거리 등의 정보를 함께 제공해주기 때문에 실제 스트리밍 서비스와 유사해진다. 단, 유튜브 영상으로는 적응형 스트리밍 수행이 불가능하므로, 적응형 스트리밍의 기능은 소수의 영상에 한정한다.

3. 설계 상세화 및 변경 내역

(1) 전체 구성도

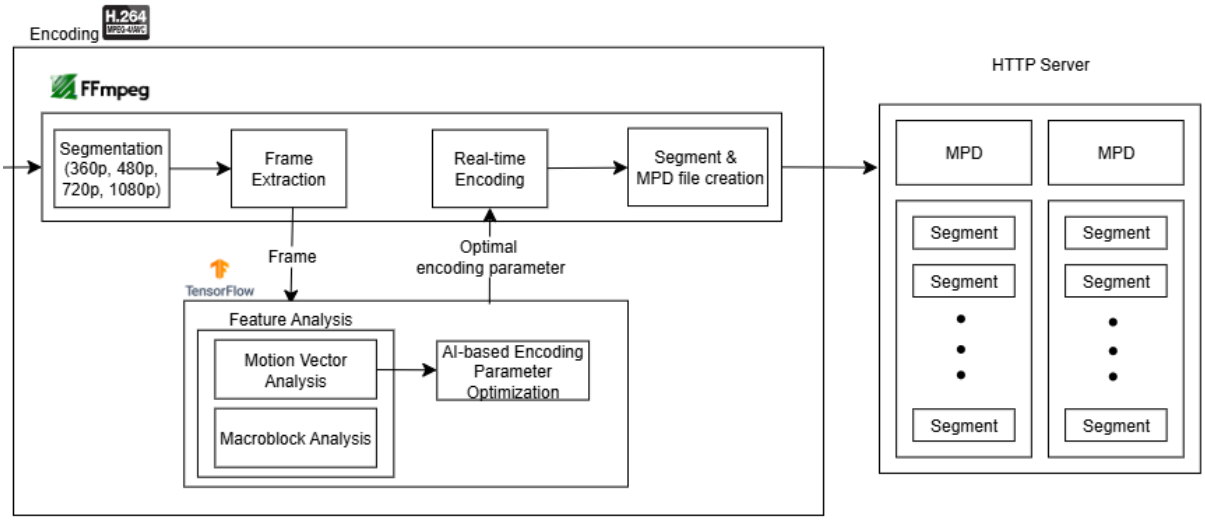


1. 구성도는 서버 측과 클라이언트 측으로 구분
2. 서버는 AWS EC2 인스턴스 상에 구축되고, 클라이언트는 사용자의 웹 브라우저에서 동작
3. 좌측의 영상은 사전에 서버에 업로드 된 콘텐츠
4. 사용자가 특정 영상을 시청하길 원할 때, 서버는 해당 영상을 처리하여 스트리밍
5. 서버 동작
   A. AI모델이 영상의 특성을 분석하고, 이 결과를 바탕으로 최적의 파라미터를 예측
   B. AI 모델이 예측한 인코딩 파라미터를 기반으로, FFmpeg을 사용해 H.264 형식으로 영상을 인코딩
   C. 이 과정에서 영상을 여러 개의 세그먼트로 분할하고, 그 구조 정보를 담은 MPD 파일을 함께 생성
   D. 생성된 MPD 파일과 세그먼트는 FastAPI를 통해 HTTP 서버로 제공
6. 클라이언트 동작
   A. MPD 파일을 HTTP로 요청해서 스트리밍을 진행
   B. dash.js 내부에서 받은 MPD 파일을 파싱하고 네트워크 상태에 따라 최적의

세그먼트를 선택해 재생

C. 전체 시스템은 Docker 환경에서 컨테이너 단위로 구성되어 있어, 구성 요소 간의 독립성 과 이식성을 확보

(2) 서버측 구성도



1. 세그먼트 생성
   A. 원본 영상을 1~2초 단위로 분할하여 세그먼트를 생성한다.
2. 세그먼트의 영상 특성 추출
   A. 모션 벡터 크기의 평균, 최대, 표준편차, 모션 벡터의 개수, 프레임 비율 (I/P/B) : 오픈 소스 라이브러리인 motion vector extractor를 사용하여 분석
   B. 매크로 블록 비율(Inter, Intra, Skip) : 오픈 소스 소프트웨어인 FFmpeg을 사용하여 분석 가능
3. 영상 인코딩 최적화 모델
   A. 세그먼트의 영상 특성을 입력하여 최적화된 CRF, Max rate을 출력
   B. CRF는 유동적으로 비트레이트를 조절하여 영상의 일정한 품질을 유지하도록 하며, Max rate는 갑작스러운 비트 전송량의 증가에도 버퍼링이 발생하지 않도록 최대 비트 전송량을 제한
4. 최적화 인코딩 파라미터를 적용하여 세그먼트를 인코딩
   A. 각 세그먼트는 최적화된 CRF, Max rate로 인코딩
   B. 해상도별로 각 세그먼트를 인코딩 – 360p, 480p, 720p, 1080p
   C. 인코딩 된 파일은 m4s 포맷으로 세그먼트에 저장

D. 각 비디오의 첫번째 세그먼트는 init-stream으로 저장

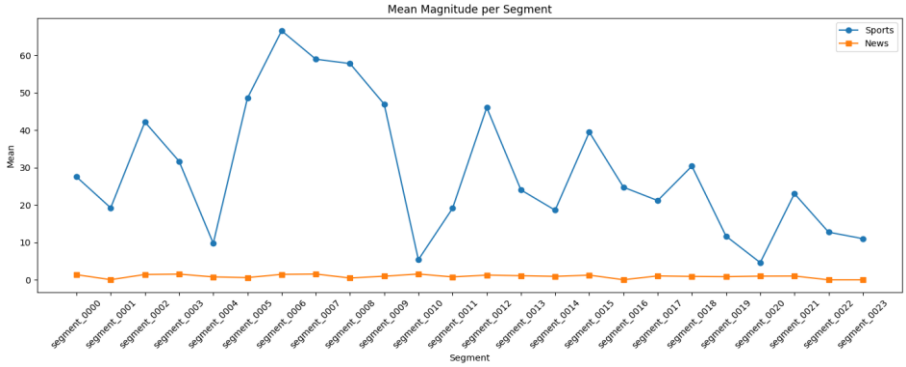
5. MPD 파일과 세그먼트가 서버에 저장
   A. MPD 파일은 세그먼트들의 정보를 담고 있으며, 클라이언트가 어떤 세그먼트를 요청할 지 판단하는 데 활용
   B. 세그먼트는 미리 최적화 파라미터로 인코딩 되어 클라이언트의 요청에 따라 Fast API를 통해 HTTP 서버로 제공

<영상 인코딩 효율 최적화 모델>

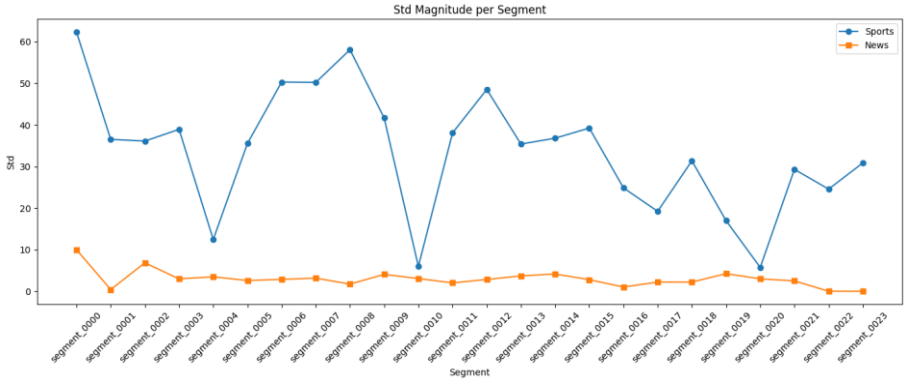
1. 모델 개요
   A. 기존 CRF 방식은 모든 영상을 동일한 비트레이트로 인코딩한다. 그러나 뉴스와 같이 움직임이 적은 영상은 게임과 같이 움직임이 많은 영상보다 적은 비트레이트로 사람 눈에 거의 차이가 나지 않도록 인코딩이 가능하다.
   B. 이러한 점을 이용하여 영상을 일정한 간격으로 잘라 영상 특성을 분석한 다음, AI 모델에 적용하여 최적화된 인코딩 파라미터(비트레이트)를 얻을 수 있다.

2. 데이터셋 : YouTube-UGC Dataset (https://media.withyoutube.com/)
   A. 주로 20초 길이의 동영상 클립(총 1,500개)로 구성된 데이터셋. 원본 데이터는 대부분 YUV(4:2:0) 포맷의 raw 비디오 파일로 제공
   B. 실제 YouTube 사용자들이 업로드한 UGC(비전문, 일상 콘텐츠) 기반의 데이터로, 인위적/이상적인 영상이 아닌 현실적 품질을 다양한 장르, 다양한 해상도로 포함한다.
   C. 각종 노이즈, 블러, 밴딩, 비디오 품질 저하 현상을 고르게 반영하며, 15개 콘텐츠 카테고리, 다양한 해상도, 다양한 환경과 상황에서 촬영된 영상들로 구성되어 모델의 일반화 능력을 키우기 이상적이다.

3. 입력 데이터
   A. 세그먼트별 모션 벡터의 크기의 분포(평균, 분산, 최댓값)
      i. 스포츠 영상과 뉴스 영상의 모션 벡터 분포 비교
         1. 모션 벡터의 평균 크기 : 스포츠 영상은 뉴스 영상에 비해 움직임이 크므로 모션 벡터의 평균 크기가 뉴스 영상보다 크게 나타나는 경향을 보인다.

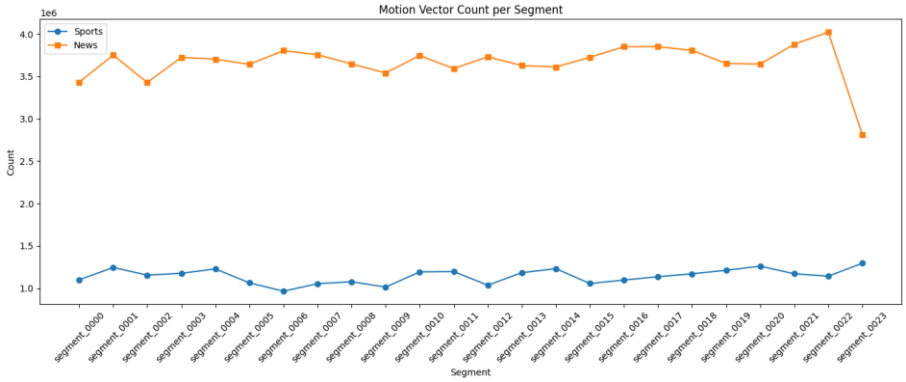


2. 모션 벡터의 분산 : 스포츠 영상은 선수, 공, 카메라 이동이 복합적으로 일어난다. 즉, 움직임의 다양성이 크고 역동적이어서 모션 벡터의 크기가 매우 다양하게 분포하는 편이다. 반면 뉴스 영상은 전체적으로 고정된 화면에서 앵커만 약간 움직이므로 표준 편차가 작은 편이다.



B. 세그먼트별 모션 벡터의 개수

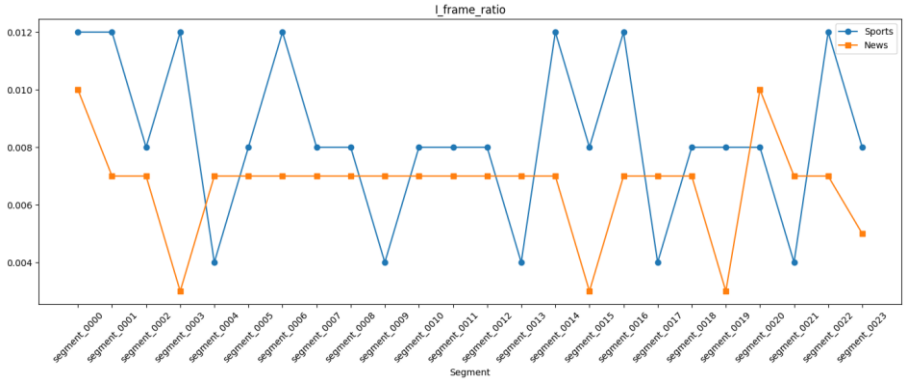
i. 스포츠 영상과 뉴스 영상의 모션 벡터 개수 비교 : 스포츠 영상에서는 실제로 움직이는 객체는 선수, 공 등 화면의 일부에 국한된다. 따라서 움직임이 크더라도 벡터 개수는 상대적으로 작은 편이다. 반면 뉴스 영상은 작은 변화(배경, 자막)가 반복적이고 미세한 픽셀 변화가 많아 프레임마다 감지되는 모션 벡터의 개수가 많은 편이다.

1. 

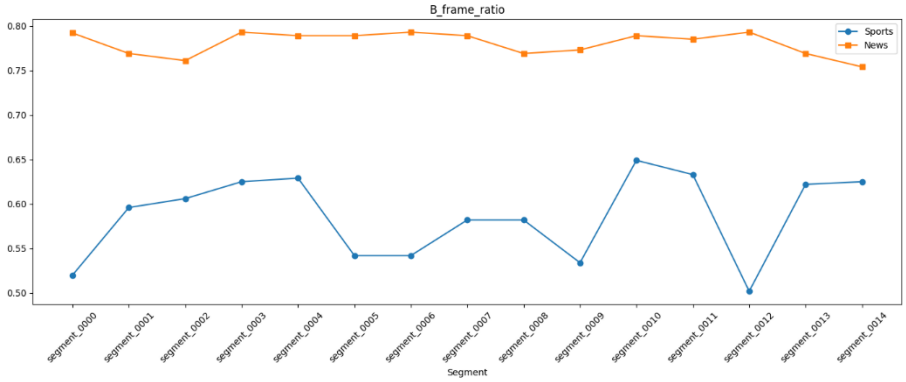
C. 세그먼트별 프레임 비율(I/P/B)

i. 스포츠 영상과 뉴스 영상의 프레임 비율 비교

1. I-프레임 : 복잡한 움직임이 많고 예측이 어려운 스포츠 영상이 뉴스 영상에 비해 I-프레임 비율이 비교적 높다.



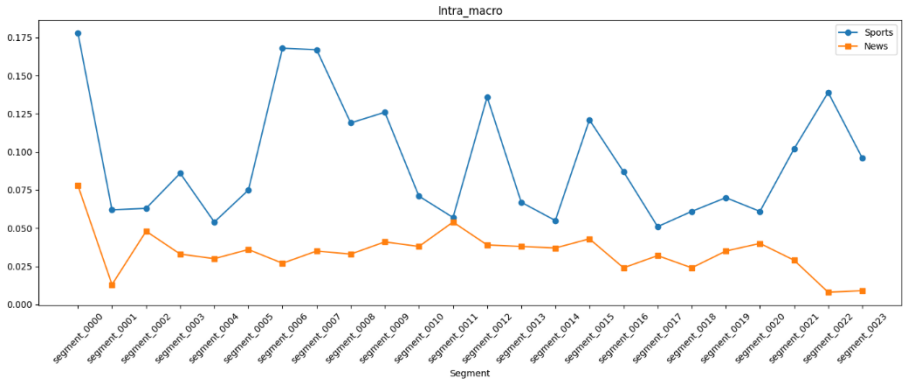
2. B-프레임 : 움직임이 거의 없어 예측이 쉬운 뉴스 영상이 스포츠 영상보다 B-프레임의 비율이 높다.



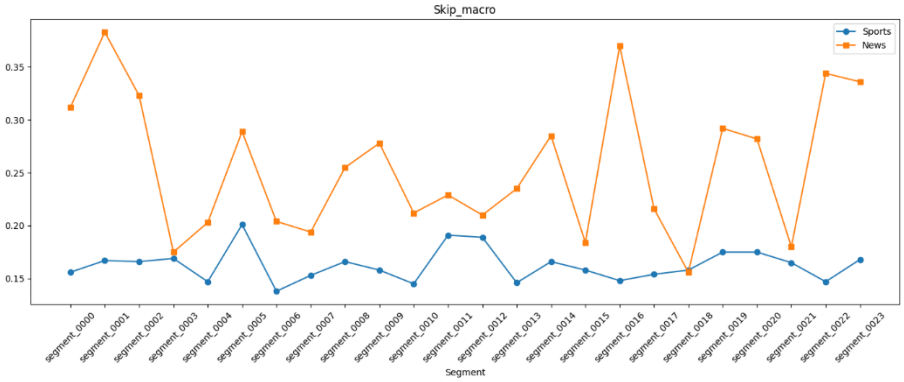
D. 세그먼트별 매크로 블록 비율(Intra/Inter/Skip)

i. 스포츠 영상과 뉴스 영상의 매크로 블록 비율 비교

1. Intra 매크로 블록 : 장면 전호나이 잦은 스포츠 영상이 장면 전환이 거의 없는 뉴스 영상에 비해 Inter/Intra 매크로 블록의 비율이 높다.

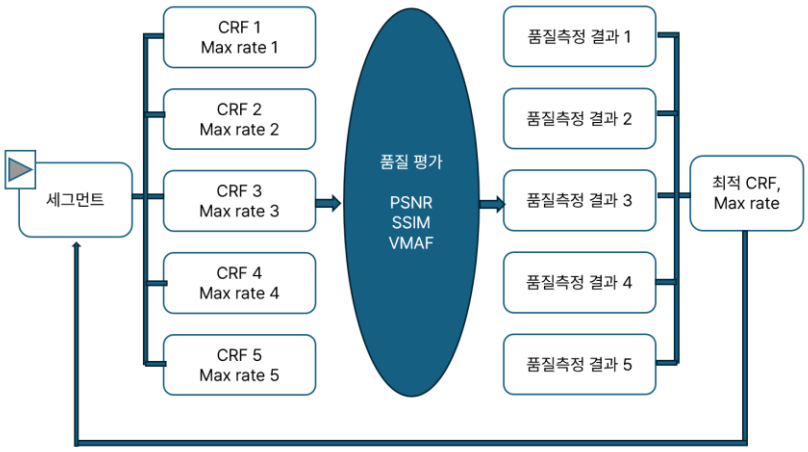


2. Skip 매크로 블록 : 움직임이 거의 없는 뉴스 영상에서 Skip 매크로 블록의 비율이 높다.



4. 출력 데이터
   A. CRF : Constant Rate Factor, 압축 강도 결정
      i. CRF는 일정한 영상 품질을 유지할 수 있으나, 네트워크 상황은 고려하지 않아서 순간적으로 비트 배분이 높아질 때 네트워크 과부하와 버퍼링을 방지하기 위해 Max rate가 필요하다.
   B. Max rate : 세그먼트에 허용 가능한 최대 비트레이트

5. 훈련 데이터 준비
   A. 원본 영상은 CBR 방식으로 인코딩되었으므로 이를 CRF 방식으로 인코딩하여 최적의 CRF, Max rate를 찾아야만 모델 훈련에 사용할 수 있다.
   B. 영상을 일정 구간마다 모든 CRF 값과 Max rate의 조합으로 인코딩하여 영상 품질을 측정하며 각 세그먼트마다 가장 적절한 CRF 값과 Max rate 값을 저장하고, 이를 모델 학습에 사용한다.



최적 인코딩 파라미터 설정

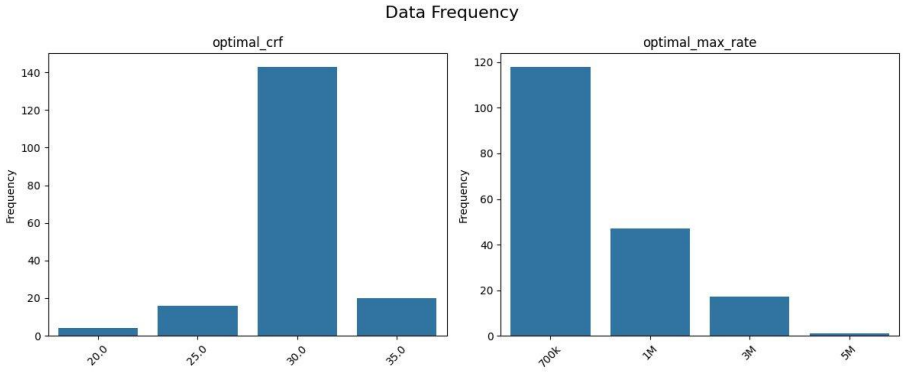
C. 품질 평가 지표 : VMAF (Video Multimethod Assessment Fushion)

i. 넷플릭스에서 개발한 품질 평가 지표로 기존의 품질 평가 방법을 결합하고 머신 러닝을 활용하여 영상의 품질을 측정해 weighted sum으로 계산한다.

ii. 기존 품질 평가 지표 중 사람이 인지하는 화질 수준과 가장 유사하고, 인간의 시각적 측면을 잘 반영한다.

D. 품질 평가 결과

i. VMAF 점수가 최소 90점을 넘기되 원본 파일의 크기를 고려하여 최적 인코딩 파라미터를 선정



6. 네트워크 및 AI 모델 아키텍쳐

A. 영상 인코딩 최적화를 위한 회귀 기반 MLP(다층 퍼셉트론, Multi-Layer Perceptron) 모델을 사용한다. 고전적인 fully-connected feed-forward 신경망이지만 입력 특징량이 적기 때문에 복잡한 구조 없이도 충분하다 또한 모션 벡터 추출 등에 많은 시간이 걸리기 때문에 추론 속도가 빠른 MLP를 사용한다.

B. Input (10차원의 입력 벡터) -> Dense(64) + ReLU -> Dense(32) + ReLU -> Dense(16) + ReLU -> Dense(2) + ReLU -> [CRF, Max rate]

i. 손실 함수 : MSELoss (MSE Squared Error)

ii. 옵티마이저 : Adam

iii. 정규화 : MinMaxScaler

7. 전체적인 DASH 흐름

A. 원본 영상을 1~2초 단위의 세그먼트(mp4)로 분할

B. 각 세그먼트에 대해서 모션벡터, 프레임, 매크로블록 등의 영상 특성을 추출

C. AI 모델을 적용하여 세그먼트별 최적화 인코딩 파라미터를 출력

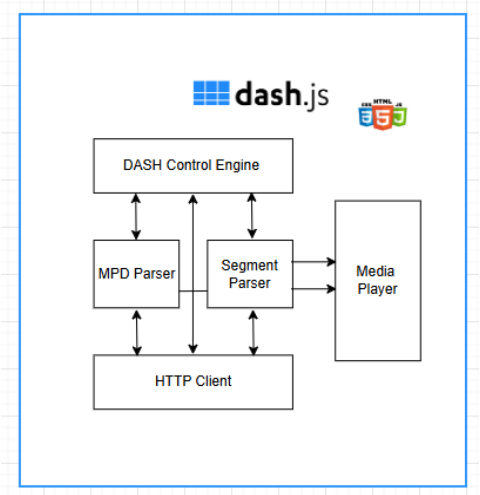
D. 각 세그먼트를 최적 CRF, Max rate를 적용하고, 해상도는 4개로 인코딩

i. 세그먼트는 총 4개의 m4s 파일로 생성(chunk-stream0-0.m4s)
ii. 첫번째 세그먼트는 init-stream.mp4를 만들어 영상 구조를 전달

E. 세그먼트 파일(m4s)과 init-stream 파일을 포함하는 manifest 파일을 생성
i. Mpd 파일은 총 4개의 represantation으로 구성

F. 과정이 완료되면 각 비디오 폴더에는 (세그먼트 개수*해상도 개수) 만큼의 세그먼트 m4s 파일들, (해상도 개수) 개의 init-stream.mp4 파일, 마지막으로 manifest.mpd 파일이 저장

(3)클라이언트측 구성도



1. dash.js 라이브러리 기반 영상 스트리밍 수행
   A. 서버에서 MPD파일 다운로드
   B. MPD파일을 파싱하여, 현재 사용 가능한 Representation 목록과 세그먼트 위치 등의 정보를 추출
   C. dash.js 내부의 ABR 알고리즘이 네트워크 처리량과 버퍼 상태를 고려하여, 가장 적절한 품질의 세그먼트를 선택
   D. 선택된 세그먼트는 URL을 통해 다운로드 되며, Media Source Extensions(MSE)을 통해 해당 세그먼트를 video buffer에 추가
   E. 세그먼트는 H.264로 압축된 인코딩 영상이므로, 브라우저는 이 버퍼에 들어온 데이터를 자동으로 디코딩

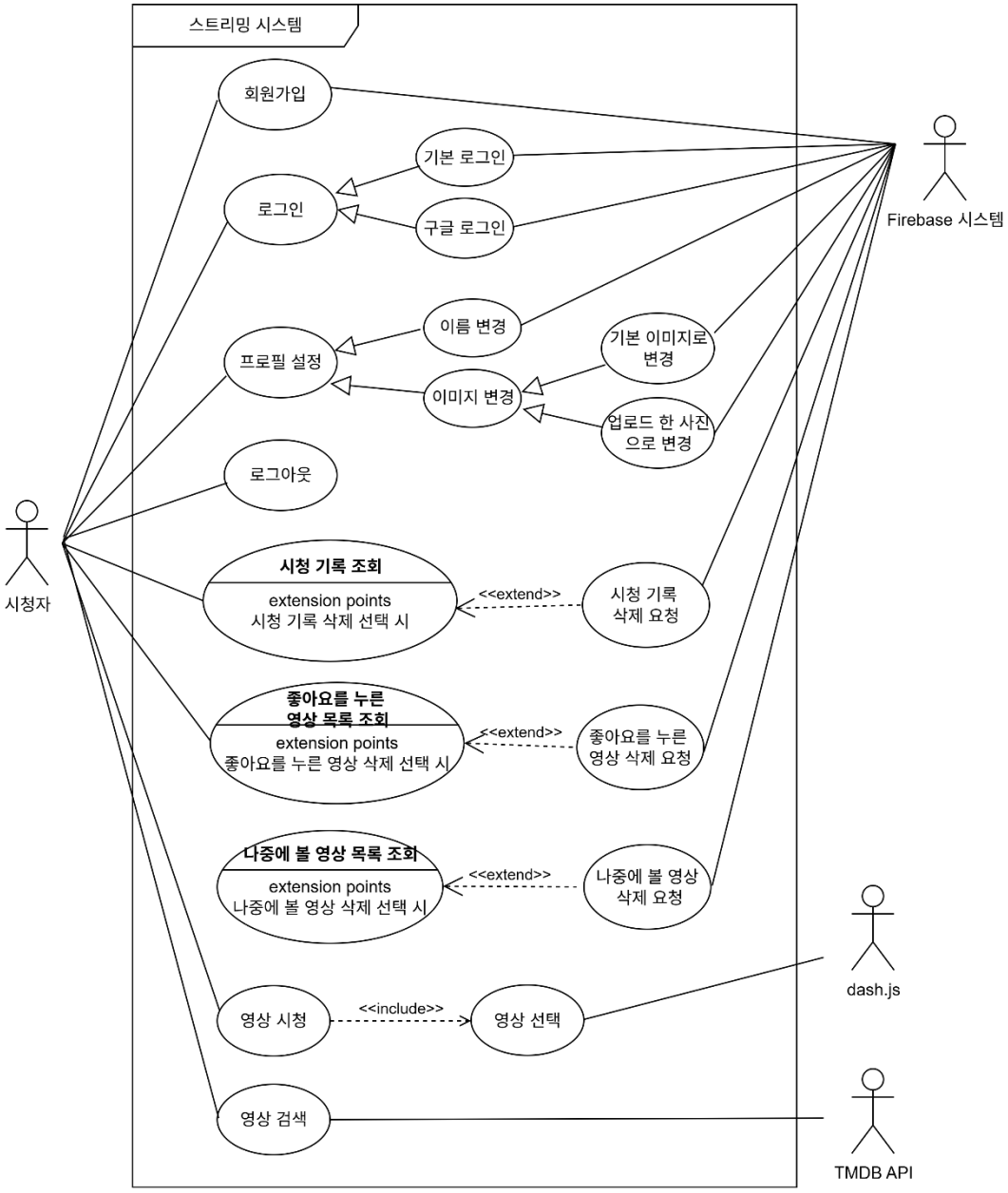
2. 세그먼트 구성 방식
   A. 우리 시스템은 기존 DASH처럼 Representation 하나가 고정된 품질(해상도,비

트레이트)로 구성된 구조가 아니라, 각 세그먼트마다 영상의 움직임을 분석해 예측한 최적의 CRF와 Maxrate 값으로 인코딩 된 구조

B. 한마디로 서버 측에서 영상 품질에 따라 비트레이트를 결정하고, 클라이언트 측은 네트워크에 따라 해상도를 결정하는 양방향 시스템
C. 서로 다른 CRF/Maxrate 세그먼트들을 하나의 DASH 스트림으로 묶기 위해, 외부 툴인 MP4Box를 사용해 개별 mp4세그먼트를 m4s 조각과 MPD로 패키징
D. dash.js에서 CRF/Maxrate 최적화 세그먼트를 일반 DASH 세그먼트처럼 읽어서, 실시간 적응형 스트리밍을 정상적으로 수행

(4) 유스케이스 다이어그램

(유스케이스 다이어그램과 시퀀스 다이어그램 중 복잡한 부분만 설명 추가)

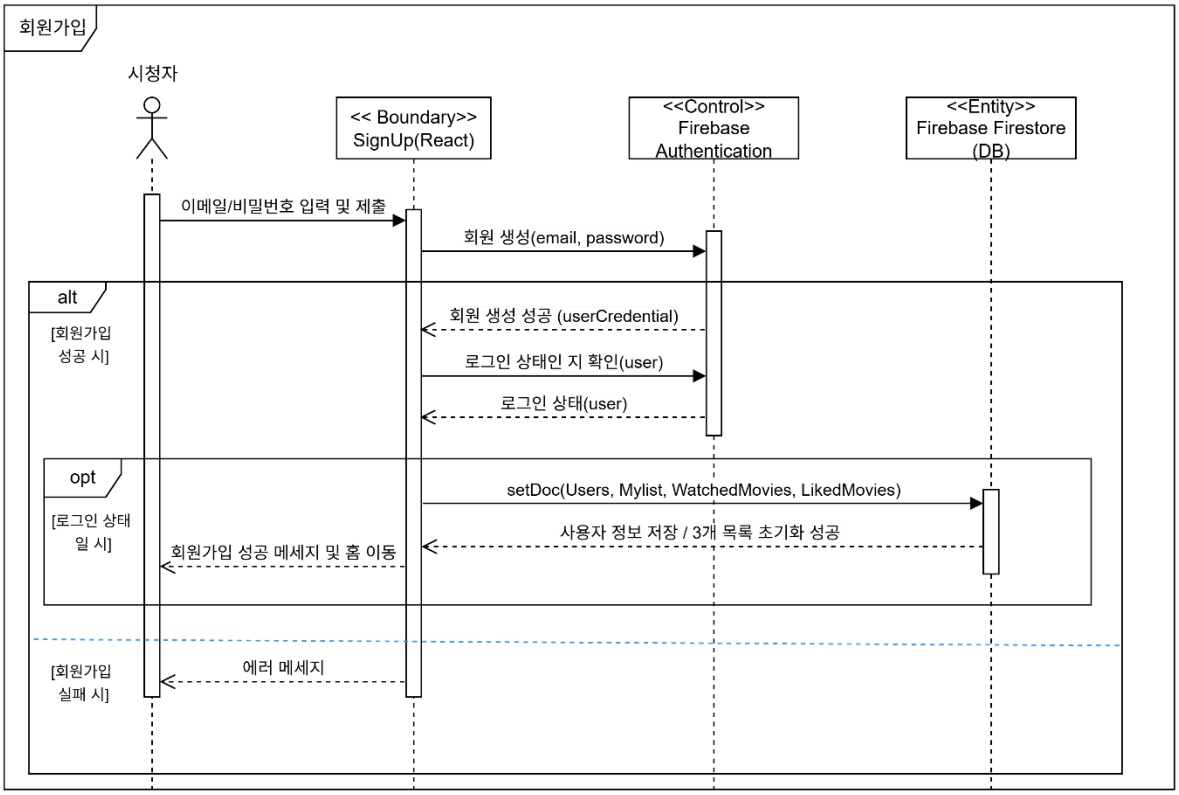


1. 영상 시청

- 영상 선택
- 영상 인코딩 최적화 모델 적용
- dash.js 기반 적응형 스트리밍

(5) 시퀀스 다이어그램

<회원가입>



1. 시청자가 회원가입 요청

- 이메일, 비밀번호를 입력한 뒤, 회원가입 버튼 클릭

2. 회원 생성

- 이메일, 비밀번호 정보를 바탕으로 회원 생성

3-1. 회원가입 성공시

- 새로 생성된 사용자 정보와 인증 관련 데이터를 담고 있는 userCredential 객체 반환
- user 정보를 바탕으로 회원가입 이후, 사용자가 실제 로그인 된 상태인지 확인

3-1-1. 로그인 상태일 시

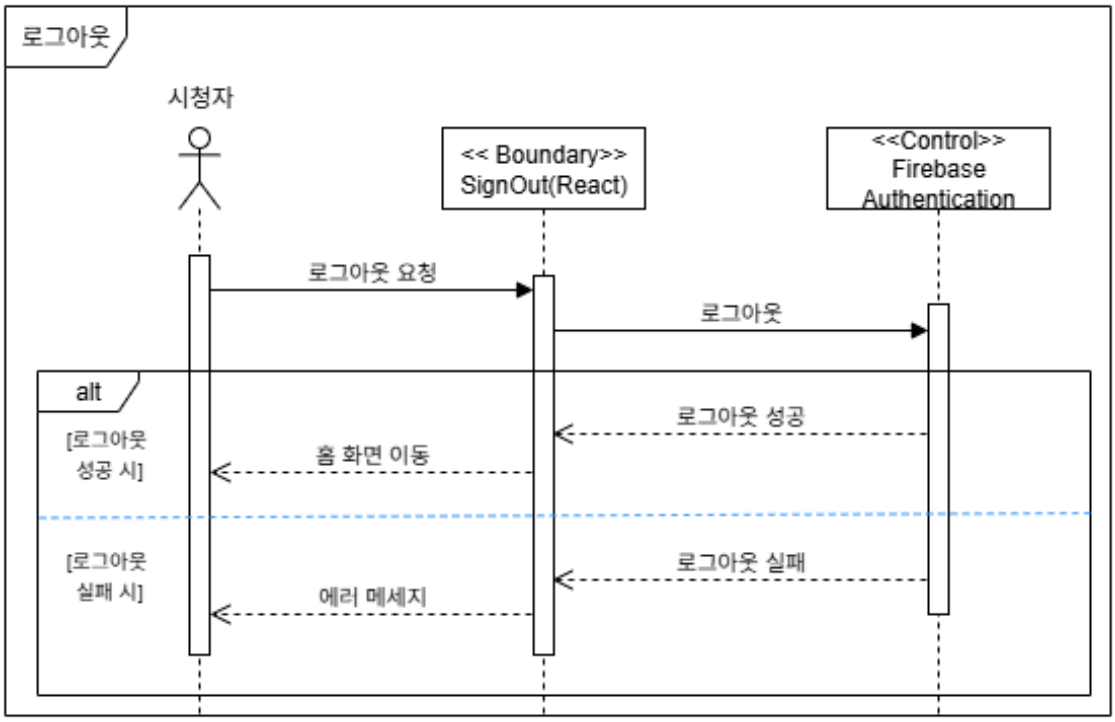
- 사용자 정보(email,uid) 저장

- 사용자 전용 MyList, WatchedMoives, LikedMovies 목록을 빈 배열로 초기화 (나중에 사용자가 각 목록에 맞는 영화를 추가하거나 삭제할 수 있게 해주는 작업)
- 회원가입 성공 메시지 및 홈 이동

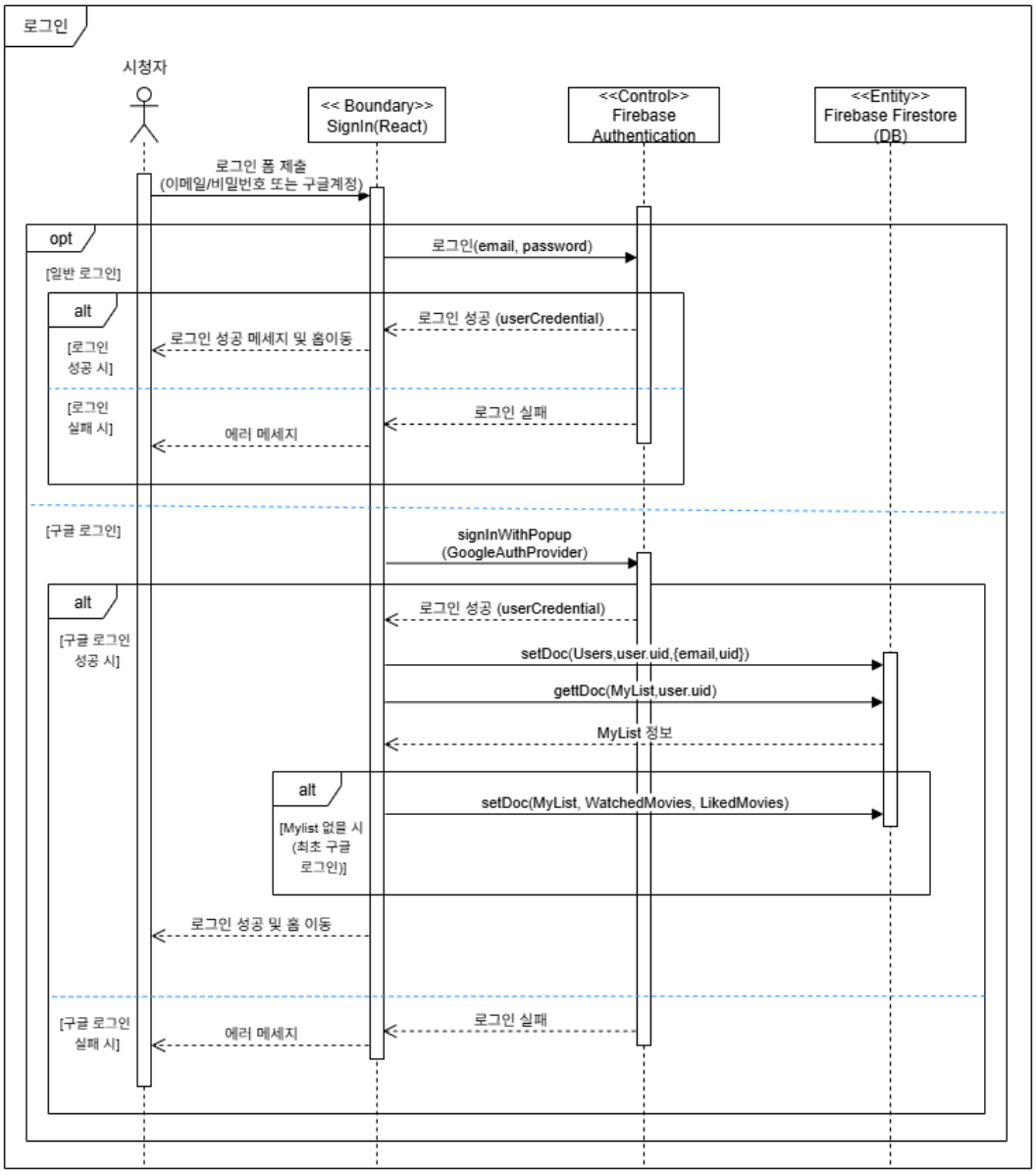
3-2. 회원가입 실패 시

- 에러 메시지

<로그아웃>



<로그인>



1. 시청자가 로그인 요청

- 이메일/비밀번호 (일반 로그인) 또는 구글 계정 (구글 로그인) 폼 제출

2-1. 일반 로그인

- email, password 정보를 바탕으로 로그인

2-1-1. 일반 로그인 성공 시

- 로그인 성공 메시지 및 홈 이동

2-1-2. 일반 로그인 실패 시

- 에러 메시지

2-2. 구글 로그인

- 구글 로그인 팝업을 사용하여 로그인 처리

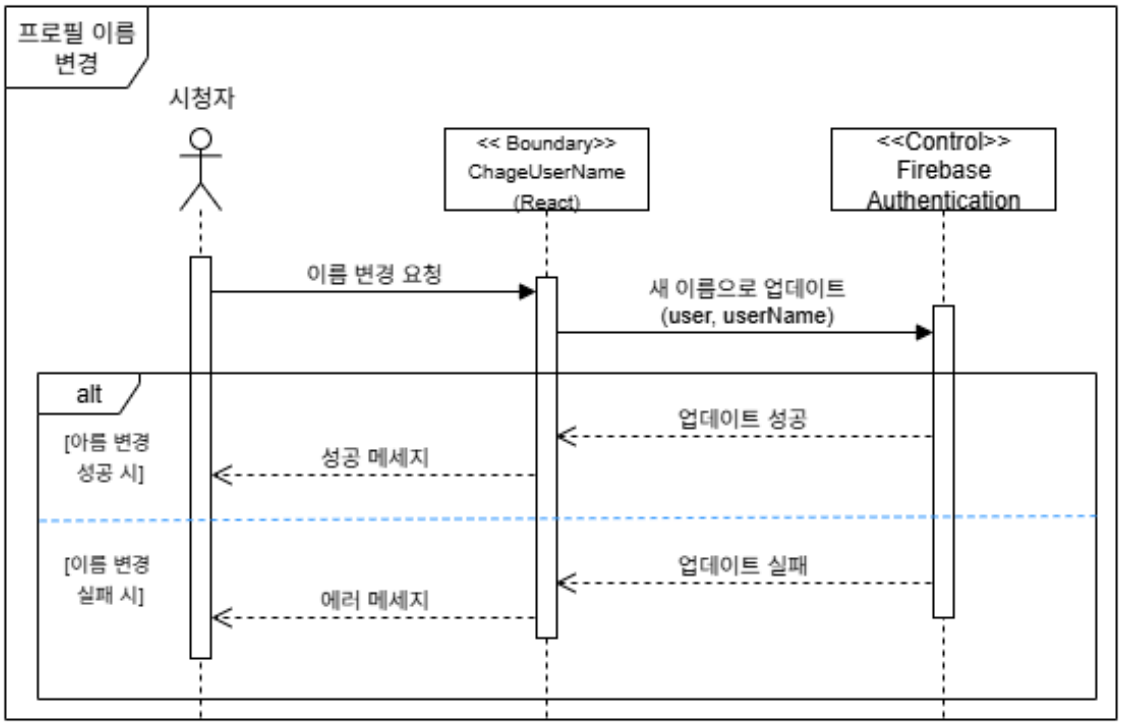
2-2-1. 구글 로그인 성공 시

- result (userCredential과 동일한 구조) 반환
- 사용자 정보 저장(email, uid)
- 해당 사용자의 MyList 정보 가져와서 존재하지 않으면, (최초 구글 로그인 시) Mylist, WatchedMovies, LikedMovies 목록 초기화
  (구글 로그인은 회원가입시 수행하는 사용자 별 3가지 목록을 초기화하지 않았기 때문에 최초 로그인 시 해당 작업 수행)
- 로그인 성공 및 홈 이동

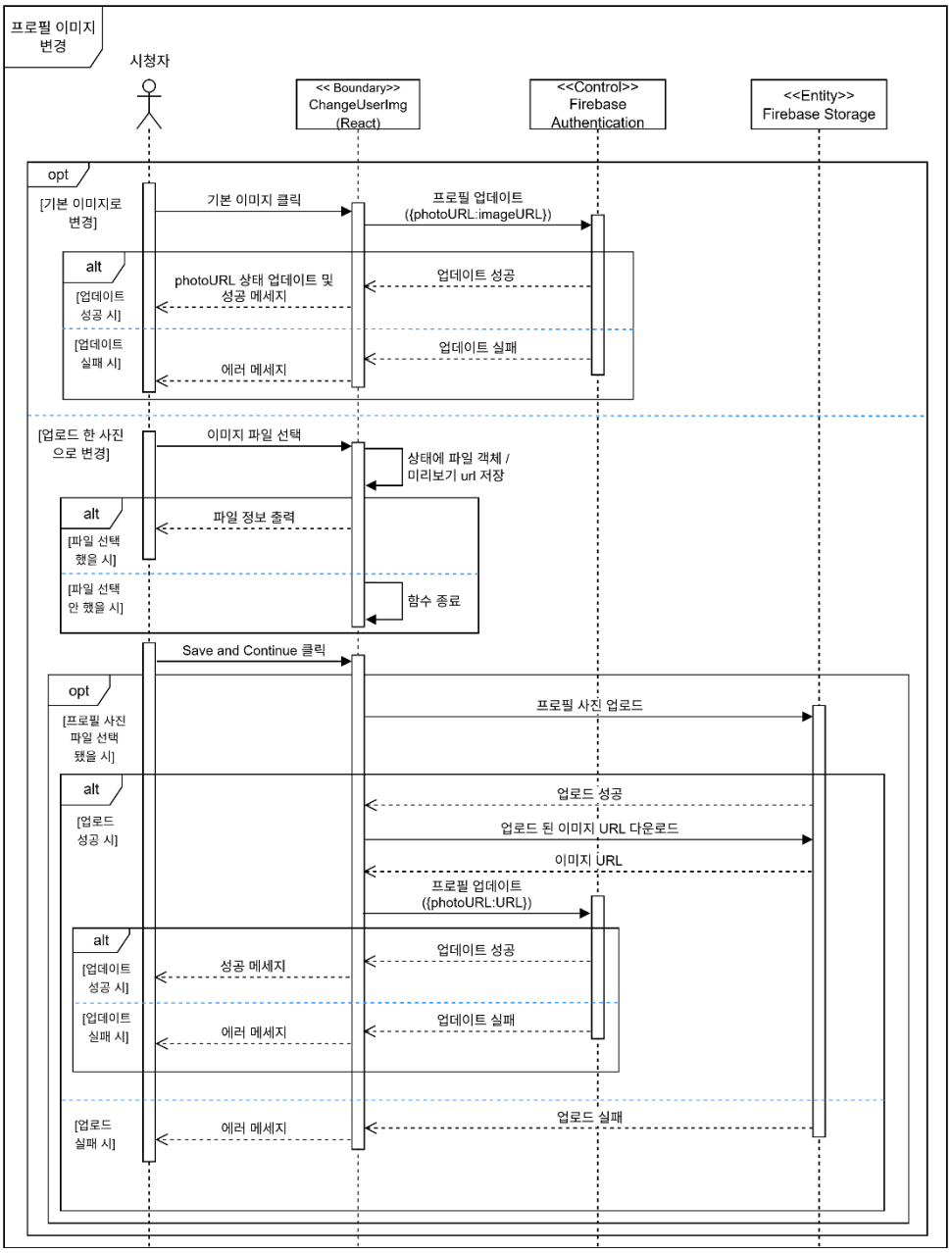
2-2-2. 구글 로그인 실패 시

- 에러 메시지

<프로필 이름 변경>



<프로필 이미지 변경>



1-1. 기본 이미지로 변경

- 제시되어 있는 4개의 기본 이미지 중 프로필 사진으로 바꿀 이미지 1개 선택
- 프로필 업데이트 (imageURL을 받아와서 firebase auth에 업데이트 -> 프로필에 반영)

1-1-1. 업데이트 성공 시

- photoURL 상태 업데이트 및 성공 메시지

1-1-2. 업데이트 실패 시

- 에러 메시지

1-2. 업로드 한 사진으로 변경

- 사용자가 자신의 컴퓨터에 저장된 이미지 파일 중 업로드 할 파일 선택
- 상태에 파일 객체 / 미리보기 url 저장 (업로드 한 사진을 미리 보여주기 위함)
- 파일을 선택했을 시 파일 정보 출력
- 파일을 선택 안 했을 시 함수 종료

2. 사용자가 Save and Continue 버튼 클릭

3. 프로필 사진 파일이 선택됐을 시

- Firebase Storage에 새 프로필 사진 업로드

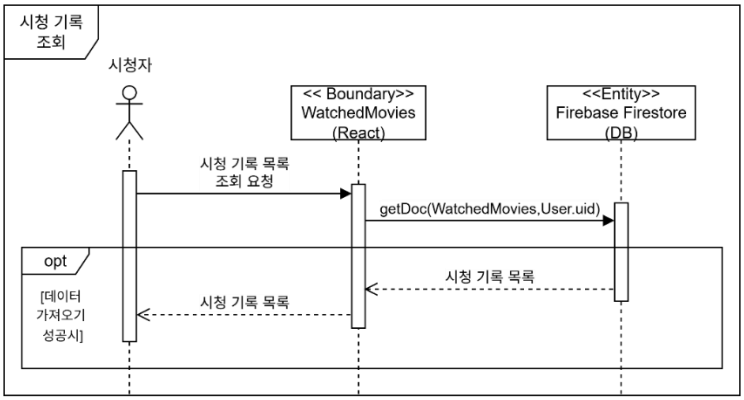
3-1. 업로드 성공 시

- 업로드 된 이미지 URL 다운로드 (실제 접근 URL 가져오기)
- Firebase auth의 사용자 정보 중 photoURL 값을 새로 받은 이미지 주소로 업데이트
- 업데이트 성공 시 성공 메시지
- 업데이트 실패 시 에러 메시지

3-2. 업로드 실패 시

- 에러 메시지

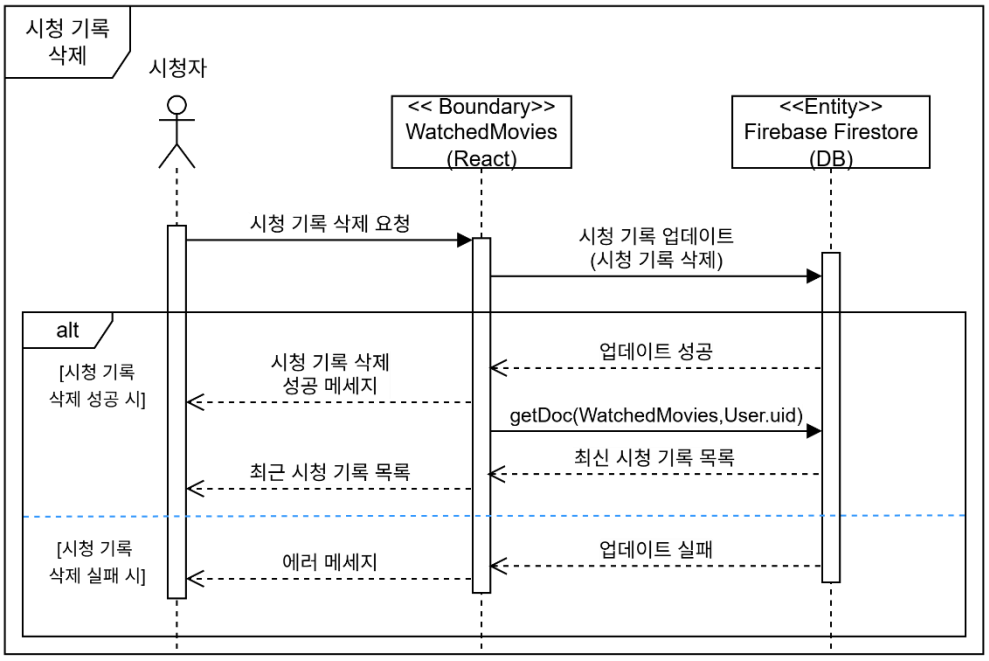
<시청 기록 조회>



1. 사용자가 지금까지 시청한 영상 목록 조회를 요청

- 사용자 별 WatchedMovies 목록 가져옴
- 데이터를 성공적으로 가져오면 시청 기록 목록을 보여줌

<시청 기록 삭제>



1. 사용자가 지금까지 시청한 영상 시청 기록 중 특정 영상의 시청 기록 삭제 요청

- 특정 영상의 시청 기록을 삭제하고, 시청 기록 업데이트 (UI에 반영)

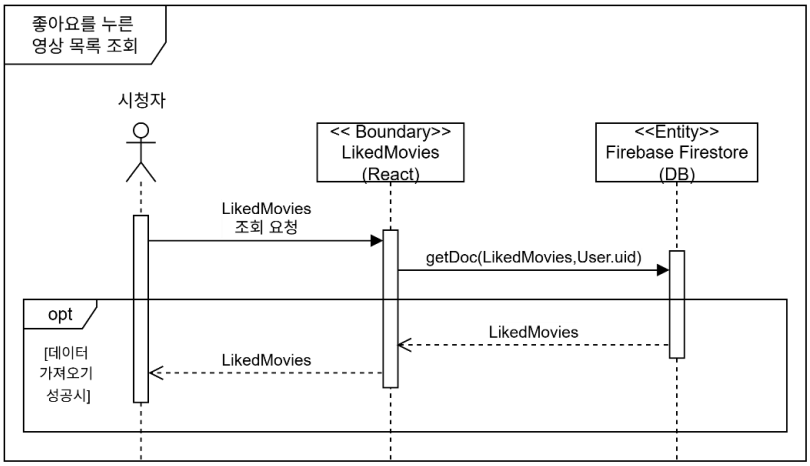
2-1. 시청 기록 삭제 성공 시

- 시청 기록 삭제 성공 메시지
- 사용자 별 WatchedMovies 목록을 가져와서 업데이트 된 최신 시청 기록을 보여줌

2-2. 시청 기록 삭제 실패 시

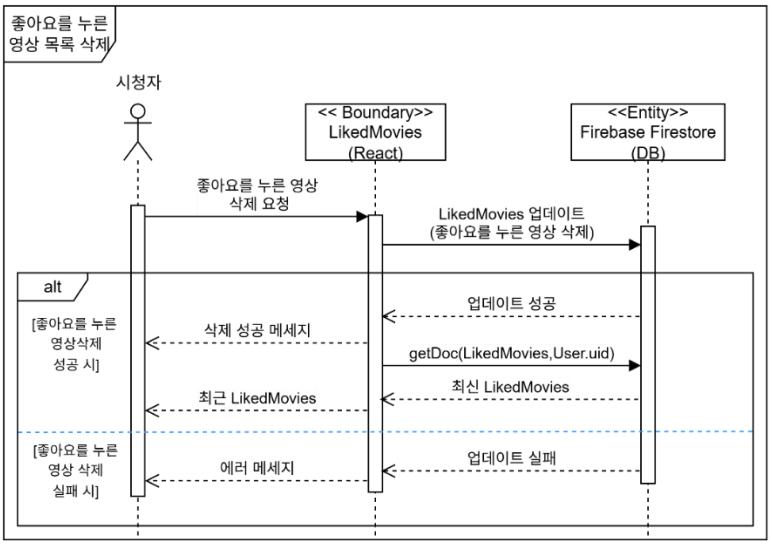
- 에러 메시지

<좋아요를 누른 영상 목록 조회>



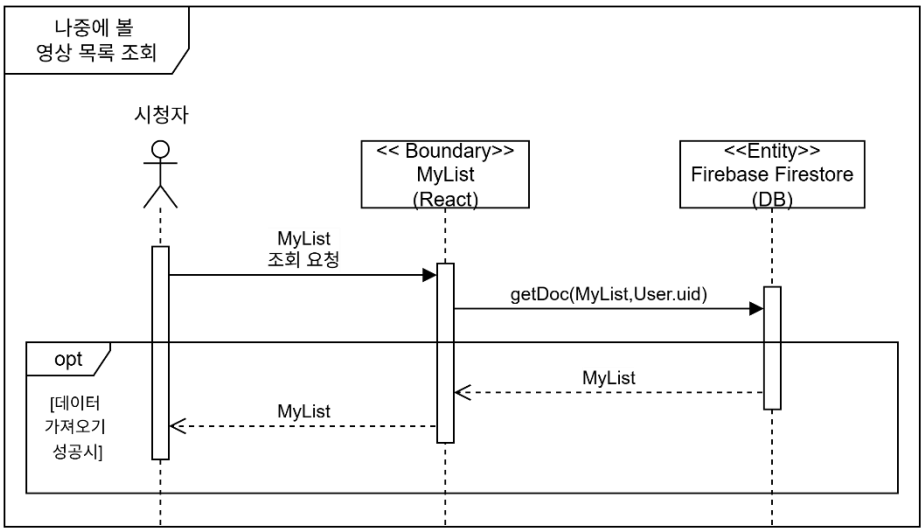
(시청 기록 조회 기능의 시퀀스 다이어그램 구조와 유사)

<좋아요를 누른 영상 목록 삭제>



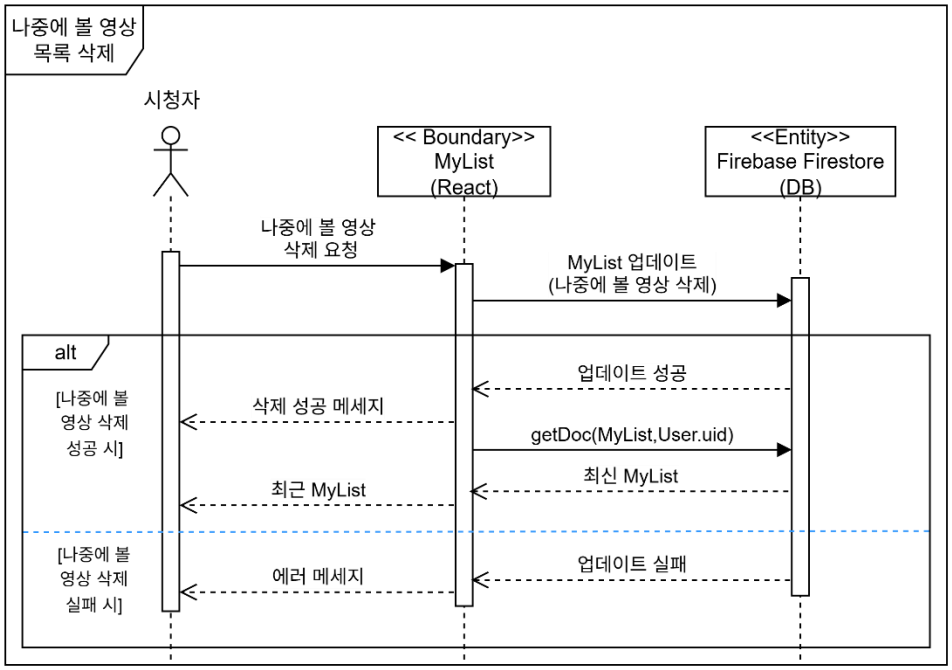
(시청 기록 삭제 기능의 시퀀스 다이어그램 구조와 유사)

<나중에 볼 영상 목록 조회>



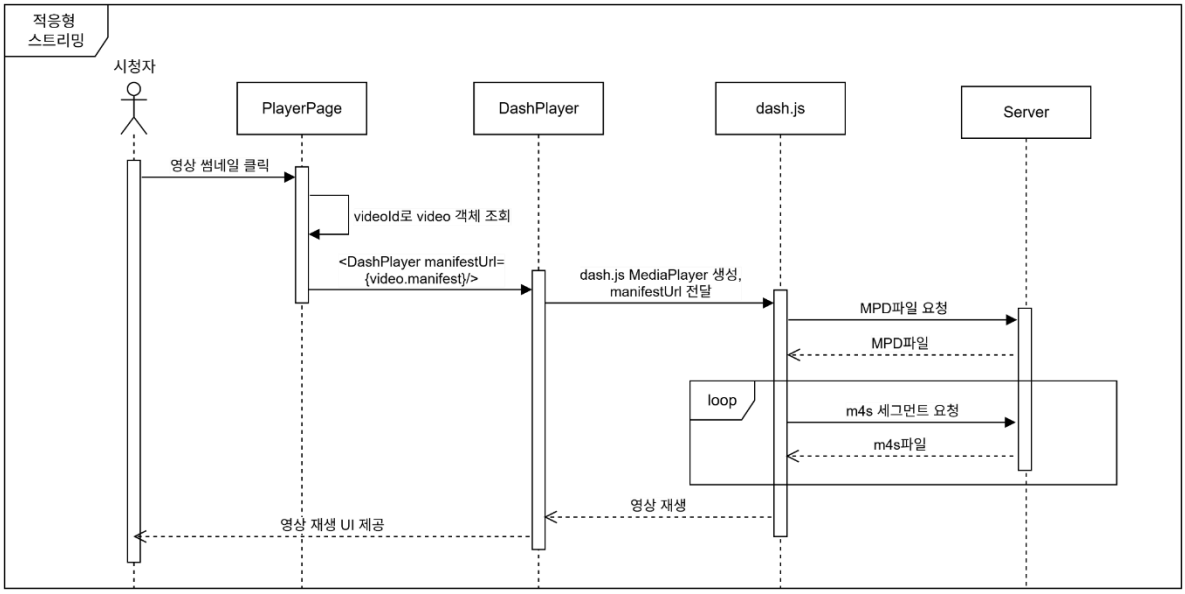
(시청 기록 조회 기능의 시퀀스 다이어그램 구조와 유사)

<나중에 볼 영상 목록 삭제>



(시청 기록 삭제 기능의 시퀀스 다이어그램 구조와 유사)

<적응형 스트리밍>



1. 스트리밍을 원하는 영상의 썸네일 클릭

2. PlayerPage

- URL 파라미터에서 videoId 추출
- 해당 videoId에 맞는 video객체 찾기
- 해당 video 객체의 manifest 찾아서 DashPlayer에 전달

3. DashPlayer

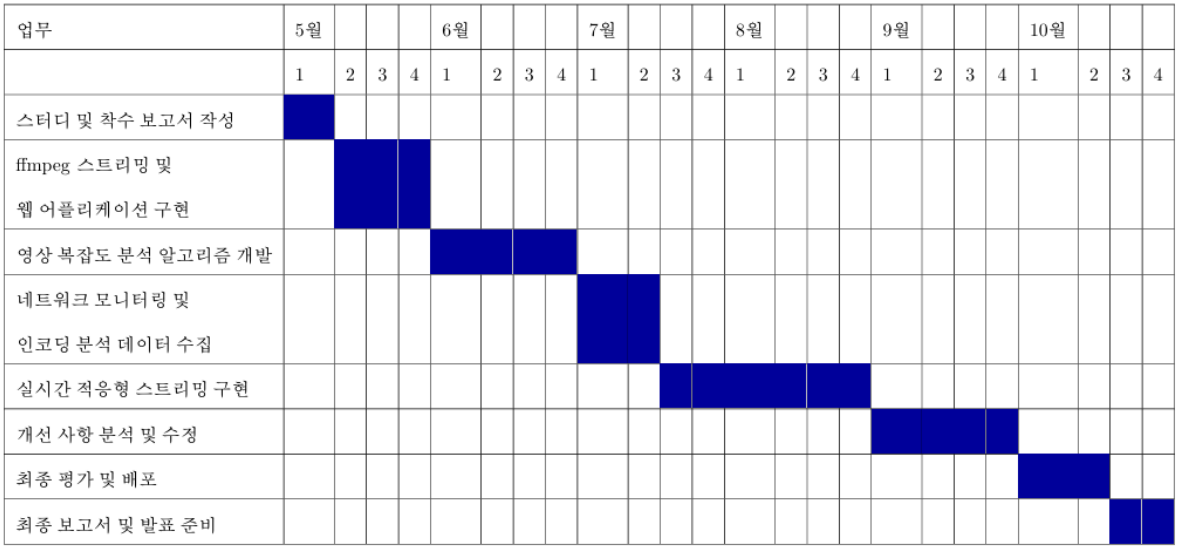
- dash.js MediaPlayer 생성
- manifestUrl 전달
- ABR 자동 품질 조절 활성화

4. dash.js

- MPD 파일 요청
- m4s 세그먼트 반복 요청

5. 영상 재생

4. 갱신된 과제 추진 계획



5. 구성원별 진척도

학번	이름	구성원별 진척도
202155515	김남희	훈련 데이터 생성 1. 모션 벡터 분석 ● 오픈 소스 라이브러리 Motion-vector-extractor을 활용해 프레임에서 모션 벡터 추출 ● 세그먼트 별 모션 벡터 크기의 통계 분석 (평균, 표준편차, 최대) ● 세그먼트 별 모션 벡터의 개수 분석 2. 프레임 비율 분석 ● I/P/B 프레임의 비율을 분석 3. 매크로 블록 분석 ● Inter/Intra/Skip 매크로 블록 비율을 분석 4. VMAF 품질 평가 ● 각 세그먼트의 최적 CRF, Max rate 조합

		을 분석 모델 훈련 및 평가 1. RandomForest 기반의 AI 모델을 훈련 2. 교차 검증을 통해 모델 성능 평가 모델 적용 1. 실시간 스트리밍 서비스에 모델 적용 2. 각 세그먼트별 인코딩 파라미터 최적화 3. 세그먼트 인코딩 후 DASH 스트림 생성
202155553	박은재	DASH 기반 영상 스트리밍 웹 서비스 1. 회원가입 2. 로그인 ● 기본 로그인 ● 구글 로그인 3. 이름 변경 4. 이미지 변경 ● 기본 이미지로 변경 ● 업로드한 사진으로 변경 5. 로그아웃 6. 시청 기록 조회 7. 시청 기록 삭제 8. 좋아요를 누른 영상 목록 조회 9. 좋아요를 누른 영상 삭제 10. 나중에 볼 영상 목록 조회 11. 나중에 볼 영상 삭제 12. DASH 기반 영상 스트리밍 13. 영상 검색

6. 보고 시점까지의 과제 수행 내용 및 중간 결과

6.1 영상 인코딩 최적화 모델

1. 훈련 데이터 생성 (input)

- 모션 벡터 크기의 분포 : 평균, 최대, 표준편차

- 모션 벡터의 개수

- I/P/B 프레임의 비율

- Inter/Intra/Skip 매크로 블록의 비율

2. VMAF 품질 평가 : 훈련 데이터의 최적 파라미터를 선정 (target)

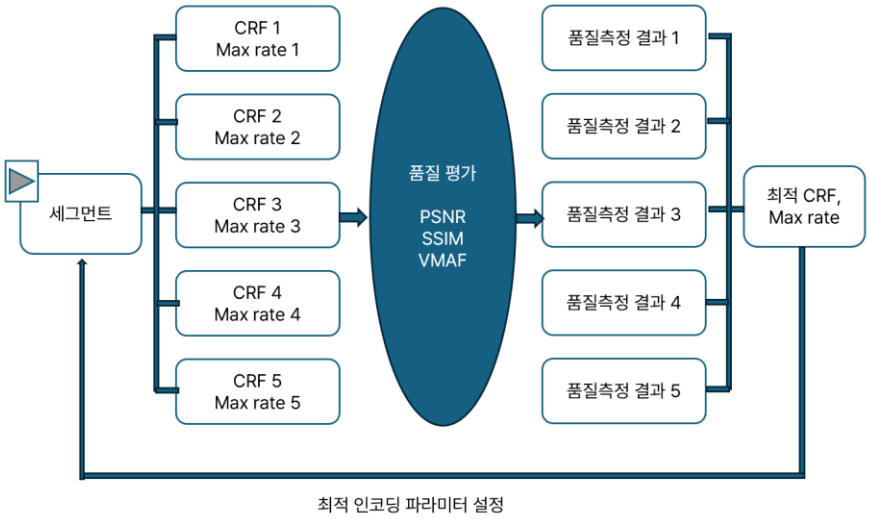
- 원본 영상을 10초 단위로 분할하여 세그먼트를 생성

- CRF는 [20, 25, 30, 35], Max rate는 [700k, 1M, 3M, 5M]를 기준

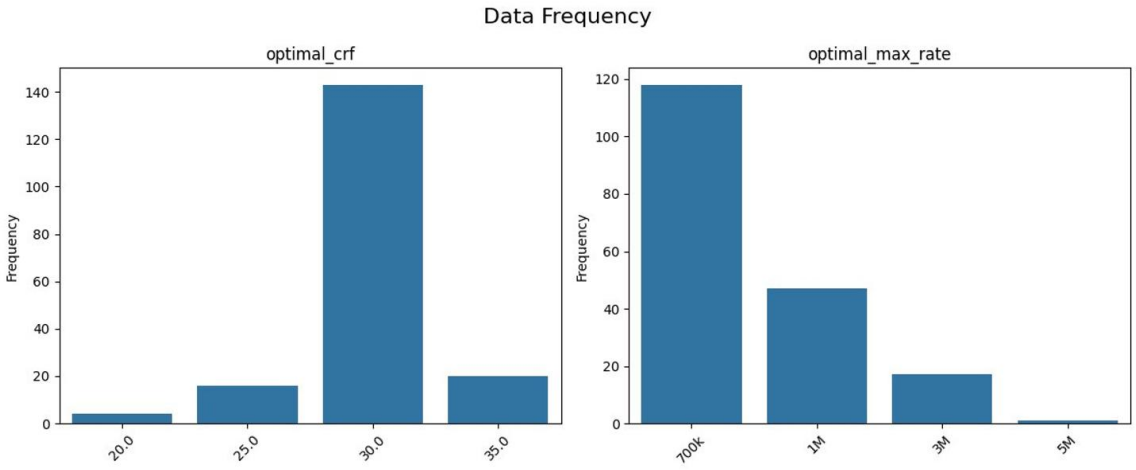
- 세그먼트를 다양한 CRF, Max rate 조합으로 인코딩

- 인코딩된 세그먼트의 품질을 VMAF로 품질 평가

- (품질 점수 / 파일 크기)가 가장 큰 조합을 찾아 최적화 파라미터로 선정



- VMAF 품질 평가 과정



- VMAF 품질 평가 결과

3. 모델 훈련 및 평가

- RandomForestRegressor 기반으로 모델 학습

- k-fold 교차검증을 통해 모델 성능을 평가

4. 모델 적용

- 세그먼트 단위로 최적화 인코딩 파라미터 예측

5. 추후 개선 사항

- Yutube 8M과 같은 대규모 데이터셋을 통해 다양한 학습 데이터를 확보하여 과적합을 방지하고 모델의 일반화 성능을 향상

- GAN 기반의 딥러닝 모델로 확장하여 대규모 데이터셋에도 잘 동작하는 AI를 개발

- 플로우 차트, 표 등을 통해 AI 처리 구조나 AI 모델의 성능을 시각적으로 설명

- 스트리밍 중에 비트레이트, 해상도의 변화를 시각적으로 확인할 수 있는 기능을 추가

6.2 DASH 기반 영상 스트리밍 웹 서비스

1. 전체적인 구조

    -GitHub에 공개된 넷플릭스 클론 UI를 참고하여 틀을 먼저 구현한 뒤, 그 위에 필요한 기능들을 수정하고 추가하면서 우리 프로젝트 목적에 맞는 형태로 개발

2. 현재 상황

    - dash.js를 활용하여 서버에서 받아온 MPD 파일을 기반으로 적응형 스트리밍이 가능하도록 구현했고, UI 디자인을 일부 수정한 상태

3. 추후 개선 사항

    - 저요금제 모드, 인기 영상 순위 제공, 한국어 버전, 프로필별 비밀번호 잠금 기능 등 다양한 기능 추가 예정

4. 저요금제 모드

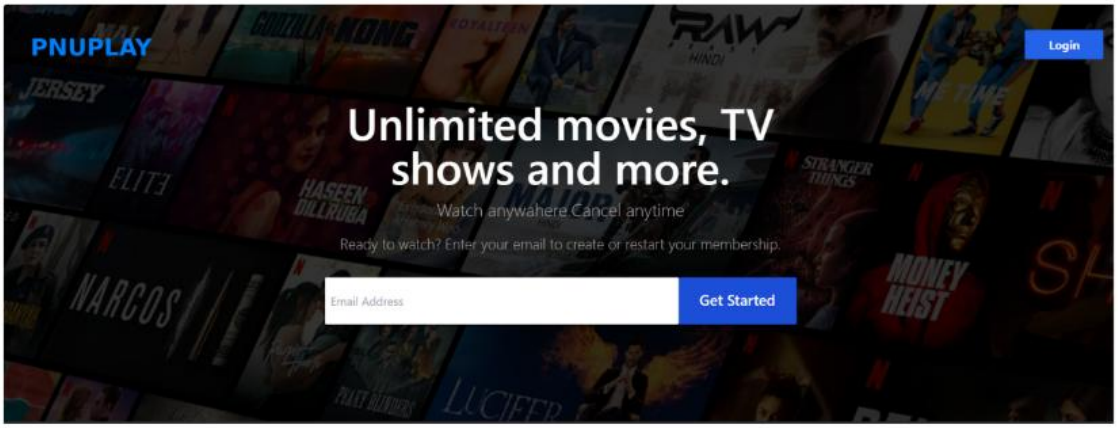
    - 넷플릭스 : 네트워크가 느릴 때 자동으로 저화질로 전환해주는 기능 존재

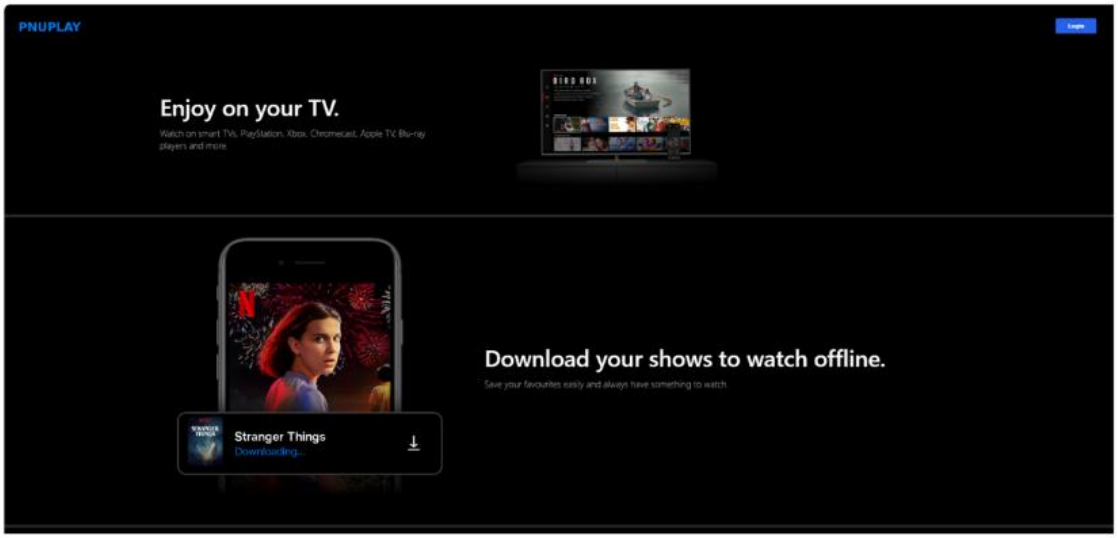
    - 저요금제 모드 : 화질을 낮추는 것을 넘어서, 데이터가 부족한 상황에서 일정한 품질로 끊김 없이 영상 시청 가능

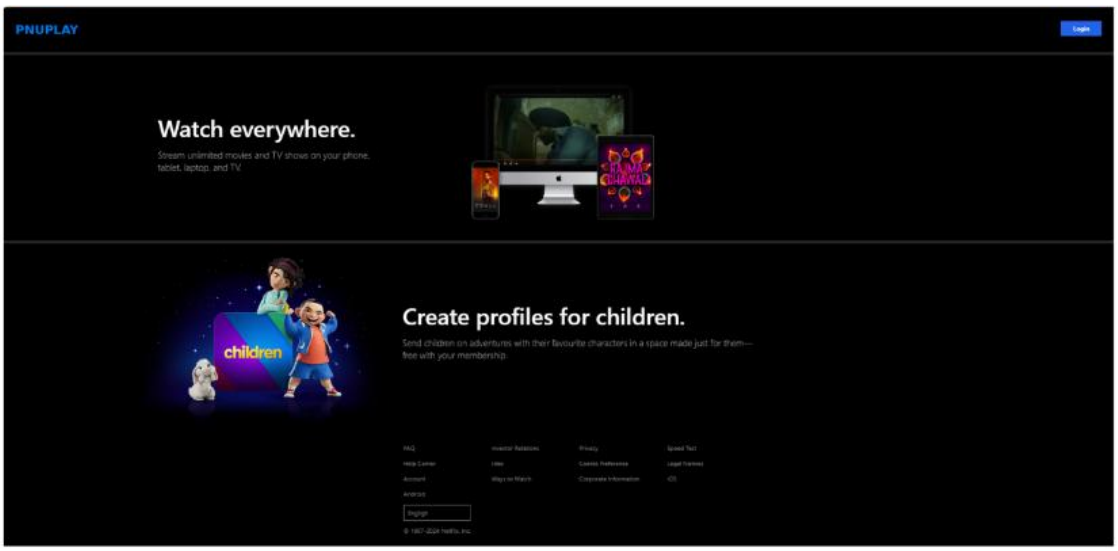
    - 네트워크 상태와 영상 특성을 동시에 고려하여 스트리밍 품질을 최적화하는 우리 시스템에 적합한 기능

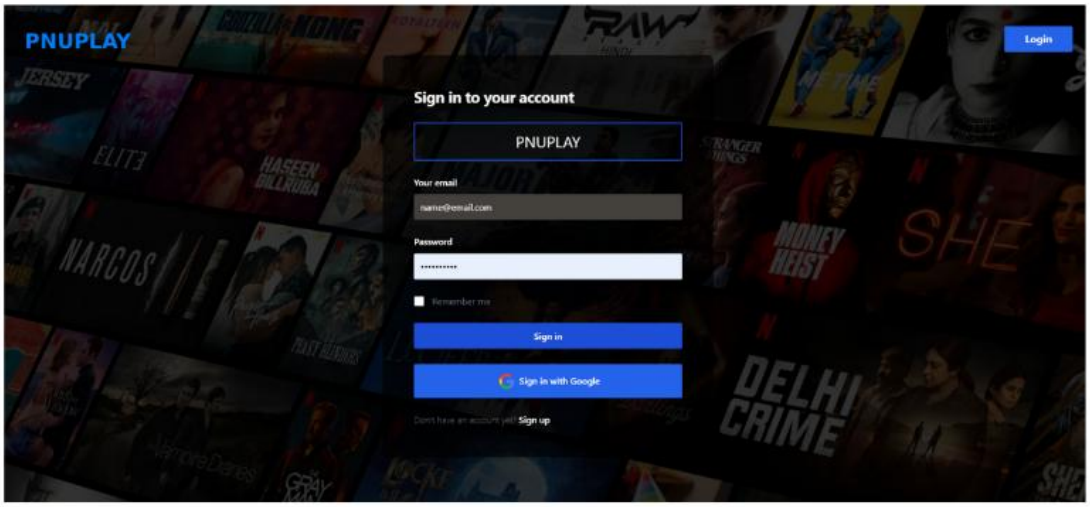
    - 같은 데이터 용량으로 더 많은 시간을 스트리밍할 수 있게 해주고, 데이터 제약이 있는 사용자에게 유용한 기능

5. 지금까지 구현한 스트리밍 웹 서비스 사진 첨부

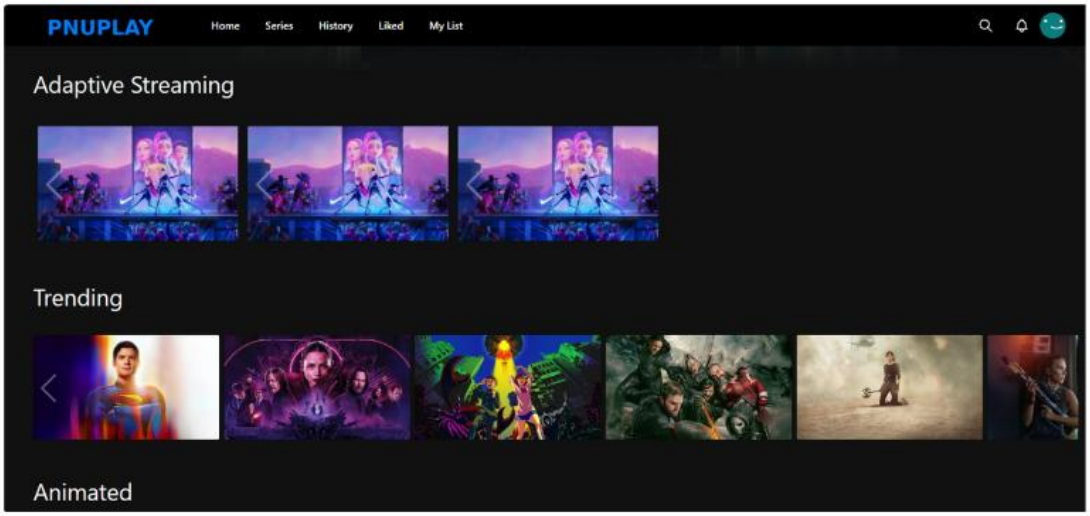

PNUPLAY
Login
Unlimited movies, TV shows and more.
Watch anywhere Cancel anytime
Ready to watch? Enter your email to create or restart your membership.
Email Address
Get Started

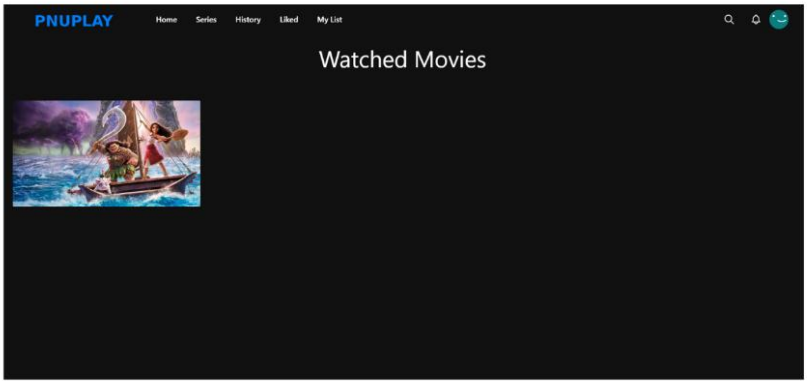

PNUPLAY
Enjoy on your TV.
Watch on smart TVs, PlayStation, Xbox, Chromecast, Apple TV, Blu-ray players and more.
Stranger Things
Downloading...
Download your shows to watch offline.
Save your favourites easily and always have something to watch.

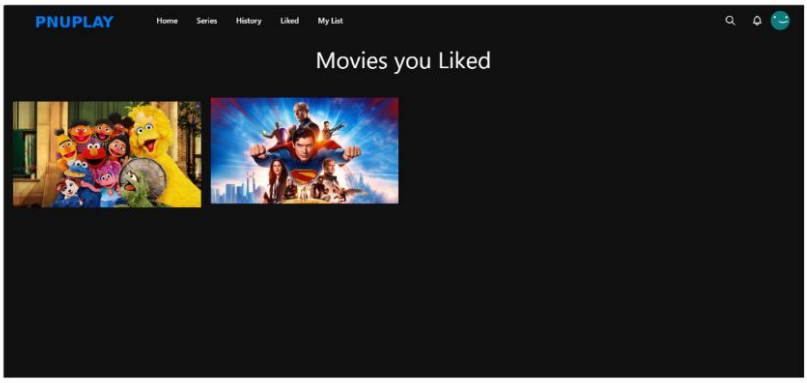

PNUPLAY
Watch everywhere.
Stream unlimited movies and TV shows on your phone, tablet, laptop, and TV.
children
Create profiles for children.
Send children on adventures with their favourite characters in a space made just for them—free with your membership.

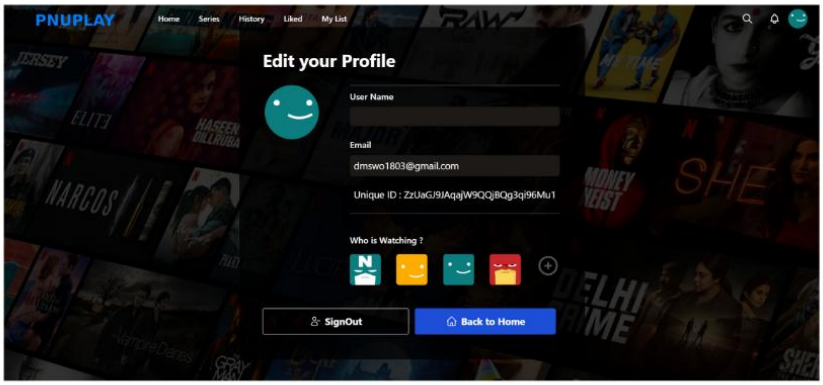

PNUPLAY
Login
Sign in to your account
PNUPLAY
Your email
Password
Remember me
Sign in
Sign in with Google
Sign up
NARCOS
MONEY HEIST
DELHI CRIME

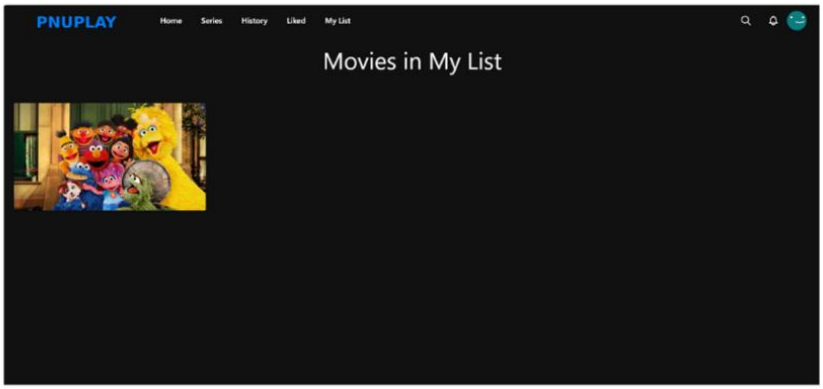

PNUPLAY
Home
Series
History
Liked
My List
Brick
2025-07-09
HD
When a mysterious brick wall encloses their apartment building overnight, Tim and Olivia must unite with their wary neighbors to get out alive.
Play
More Info


PNUPLAY
Home
Series
History
Liked
My List
Adaptive Streaming
Trending
Animated


PNUPLAY
Home
Series
History
Liked
My List
Watched Movies


PNUPLAY
Home
Series
History
Liked
My List
Movies you Liked


PNUPLAY
Home
Series
History
Liked
My List
Edit your Profile
User Name
Email
dmswo1803@gmail.com
Unique ID : ZzUaGJ9JAqajW9QQjBQg3qi96Mu1
Who is Watching ?
SignOut
Back to Home


PNUPLAY
Home
Series
History
Liked
My List
Movies in My List

7. 멘토 의견서 기반 보완방안

(1) 시각 자료 활용

- 시스템 구성도, AI 처리 구조를 시각적으로 제시

- 유스케이스 다이어그램과 시퀀스 다이어그램 추가

- 스트리밍 중 비트레이트와 해상도의 변화를 시각적으로 확인할 수 있도록 구현 예정

(2) 결과물에 대한 명확한 정의

- AI 모델 성능 평가 보고서 : 현재 프로토타입이라 k-fold 교차 검증만 수행, 이후 딥러닝 모델로 확장하여 다양한 평가 기법을 통해 AI 모델의 손실 등 평가를 진행할 예정

- 실시간 스트리밍 성능 평가 보고서 : 네트워크가 변화하여 클라이언트의 해상도 선택이 달라질 때마다 브라우저의 콘솔창에 출력. 이후 이를 로그로 출력하여 네트워크의 변화나 영상의 움직임 변화에 따른 비트레이트, 해상도, 발생 latency의 변화를 분석할 예정

(3) AI 모델의 구체화

- AI 모델의 학습 데이터를 어떻게 구성할 것인지 구체화하였다. GAN을 사용하여 가상 데이터를 생성해도 되나, 기존에 있는 데이터셋을 먼저 활용하여 모델을 만든 다음, 모델의 성능이 떨어지면 가상 데이터를 활용할 예정

(4) 성능 평가 기준 필요

- 지연 시간 6초 미만

- CBR 방식 대비 비트레이트 전송량의 감소 비율

8. 결론 및 기대효과

(1) 네트워크 환경이 갑자기 불안정해지더라도 dash.js를 활용해 해상도와 비트레이트를 자동 조절하여 안정적으로 스트리밍 서비스를 제공한다.

(2) 불안정한 환경에서 저품질의 영상을 제공하여 사용자는 데이터 비용을 절감한다.

(3) 움직임 정도에 따라 품질을 자동 조절할 수 있다. 즉 움직임이 적은 구간은 저품질로, 복잡한 구간은 고품질로 인코딩함으로써, 전체 데이터 사용량을 줄였다. 이 때 사용자가 품질 저하를 인식하지 못하도록 구현하였다.

(4) 이러한 품질 최적화는 서버 저장 공간과 인코딩 시간을 줄인다. 결과적으로 시청자 수가 많아지더라도 서버 트래픽 부담을 줄이는 효과를 기대할 수 있다.