목 차

1. 요구 조건 및 제약 사항 분석에 대한 수정사항

1.1 요구조건

이번 과제의 목표는 간접 광고가 있는 영상에서 기업의 로고가 노출된 시간을 도출하는 AI 모델의 제작이다. 이 모델은 웹 페이지를 통해 보여줄 계획이며, 입력으로 들어간 영상과 검출할 로고를 사용해 라벨링 한 새로운 영상으로 그 결과를 보여줄 예정이다.

1.2 제약 사항 분석에 대한 수정사항

객체 검출은 크게 1-stage 모델과 2-stage 모델로 나뉜다. 1-stage 모델은 검출 작업에서 객체의 위치 검출과 분류를 동시에 한다. 이런 이유로 빠른 동작 시간과 낮은 정확도를 보인다. 이에 반해 2-stage 모델은 검출 작업 도중 위치 검출과 분류를 동시에 하지 않는다. 그러므로 1-stage 방식에 비해 낮은 속도를 보이지만 그만큼 정확한 성능을 보인다.

여러 모델 사이 YOLOv5를 선택하게 된 이유는 빠른 시간 내에 영상에서의 객체 검출이 가능한 1-stage 모델이기 때문이다. 그러나 제약사항에 따라 흐름도가 바뀌게 되어 추가적인 유사도 비교 단계가 추가되었다. 이는 1-stage 모델을 2-stage로 사용하게 되는 것이다.

1-stage 모델을 사용하기 위해서는 특정한 로고가 있는 datasets만 사용해 훈련해야 한다. 그러나 사용자가 어떤 로고를 찾을지 모르는 상황 아래에 진행되어야 하므로 미리 특정한 로고를 훈련 시킬 수 없다.

이를 해결하기 위해 착수 보고서에서는 학습 데이터 생성을 위해 하나의 로고 이미지와 여러 뒷배경 이미지를 사용하여 합성을 통해 무작위로 생성된 훈련 데이터를 만든다고 기술되어 있었다. 그러나 사용자의 관점에서 입력 후 데이터 생성부터, 학습, 객체 검출까지 너무 많은 시간이 걸릴 것으로 예상되어 이 방법은 사용하지 않기로 했다.

2. 설계 상세화 및 변경 내역

2.1 설계 상세화

위의 제약 사항에 따른 수정사항이 적용된 전체적인 프로젝트의 순서도는 아래와 같다.

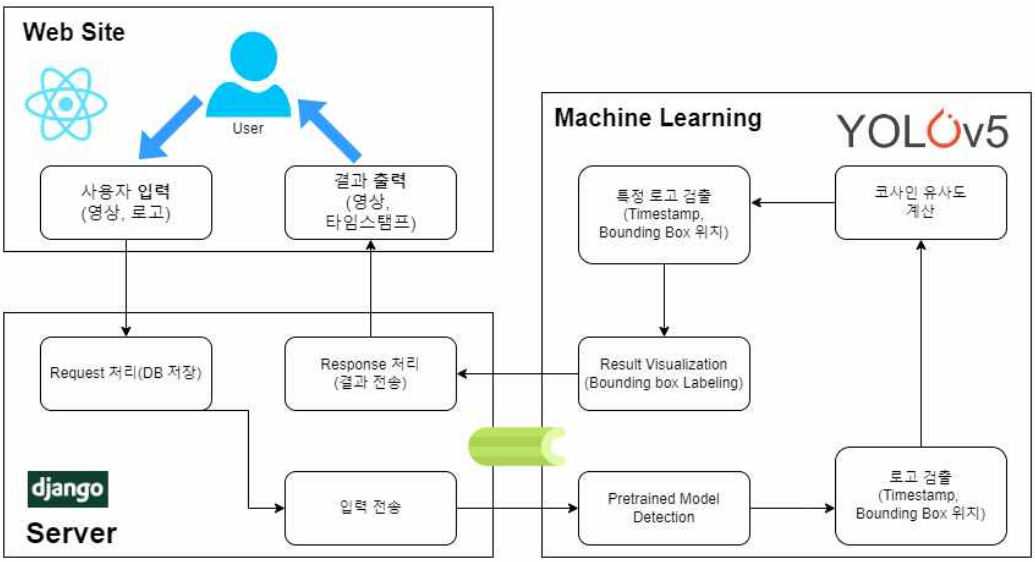

그림 1 - 전체 흐름도.

사용자로부터 웹 페이지를 통해 영상과 검출할 로고의 원본 이미지를 입력으로 받는다. 그 후 이를 서버에 전송하여 미리 제작된 pretrained 모델을 통해 영상에서 모든 로고 객체를 검출한다. 이 후 객체 검출 결과를 로고 원본 이미지와 유사한지 코사인 유사도를 계산한다. 유사도 값이 일정 threshold 이상일 경우, 찾으려는 브랜드의 로고로 결론을 내린 후 이들의 결과를 취합한다.

결과는 새로 라벨링 한 영상과 타임 스탬프의 형식으로 보여준다. 웹에서 결과들을 http 통신의 response로 받은 후 영상을 띄워 사용자에게 전달한다. 동시에 타임 스탬프는 버튼으로 구현하여 눌렀을 때 영상이 그 시간대로 이동할 수 있게 한다.

Pretrained 모델에 사용할 데이터셋은 LogoDet-3K을 사용한다. Dataset은 카테고리와 브랜드별로 나뉘어 있으며, 총 3천 개의 브랜드와 약 15만 개의 이미지를 가지고 있다. Pretrained 모델에서는 각 브랜드 별로 클래스를 나눌 필요 없이, 로고별로 같은 객체로 취급해 학습한다. 그러므로 학습에서 인자로 들어갈 클래스의 수는 **“Logo”** 1개이다.

AI 모델은 현재 가장 높은 성능을 보이고 있는 YOLOv5를 사용한다. YOLOv5는 모델 학습뿐만 검출 결과를 사용된 영상에 bounding box를 그려 제공한다. 이번 프로젝트에 목적과 가장 적합한 모델이라 판단해 사용중에 있다. 또한 YOLOv5는 크기에 따른 여러 모델을 제공하고 있으므로 사용 가능한 모델 중 성능이 가장 좋은 모델을 결정하기 위해 시도 후 비교하고 있다.

2.2 변경내역

먼저 로고 이미지를 모은 데이터셋을 이용해 미리 학습된 모델인 Pretrained 모델을 사용하기로 했다. 사용자의 입장에서는 학습 및 검출 작업까지 기다리는 것 대신, 검출 작업만 기다리면 되므로 더욱 빠른 결과를 받을 수 있다.

colab 환경에서 무료로 제공하는 ram용량에 한계가 있기 때문에 LogoDet-3K 데이터셋의 모든 이미지를 한번에 학습에 사용할 수 없으므로, LogoDet-3K를 작은 크기로 나누어 학습에 이용하기로 하였다.

또한 웹 프레임워크에서 사용될 예정이였던 Vanila html 대신, React.js를 사용해 웹 페이지를 구현할 예정이다. React.js는 기존의 웹 프레임워크에서 효율적인 상태 관리를 내세운 기술 스택으로서 현재 많은 반응형 웹 페이지에 사용되는 대표적인 프레임워크이다. 이번 프로젝트에서도 효율적인 상태 관리가 필요하며 또한 detection 작업에서 오랜 시간이 걸릴 것이므로 비동기 구현이 필수적이다. React.js는 redux-toolkit을 이용해 상태 관리와 비동기 구현을 동시에 가능하므로 이번 프로젝트 구현에 있어 적합하다.

3. 갱신된 과제 추진 계획

<table>
<tr><th colspan="3">6월</th><th colspan="4">7월</th><th colspan="5">8월</th><th colspan="4">9월</th></tr>
<tr><td>13</td><td>20</td><td>27</td><td>4</td><td>11</td><td>18</td><td>25</td><td>1</td><td>8</td><td>15</td><td>22</td><td>29</td><td>5</td><td>12</td><td>19</td><td>26</td></tr>
<tr><td colspan="3">Datasets 모집</td><td></td><td></td><td></td><td colspan="2">유사도 계산
알고리즘 개발</td><td></td><td></td><td colspan="3">최종 테스트</td><td></td></tr>
<tr><td colspan="4">웹 플랫폼 개발</td><td></td><td></td><td></td><td></td><td colspan="3">검출 결과 출력
및 시각화</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td colspan="4">검출 모델 개발 및
테스트</td><td></td><td></td><td colspan="4">웹 연동 및 배포 관리</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td colspan="2">중간 보고</td><td></td><td></td><td></td><td></td><td></td><td></td><td colspan="4">최종 보고서
작성 및 발표
준비</td></tr>
</table>

표 1 - 갱신된 과제 추진 계획

4. 구성원 별 진척도

이름	작업 완료	작업 예정
전승윤	웹 프레임워크 개발	Github 버전 관리 Docker 컨테이너 개발 및 배포 관리
유동운	리펙토링 및 테스트 담당	모델 학습 및 검증
강태환	학습 데이터 생성 및 전처리	평가 지표 계산 및 평가
공통	필요한 개념 학습 보고서 작성 및 발표 준비	

표 2 - 구성원 별 진척도

5. 보고 시점까지의 과제 수행 내용 및 중간 결과

5.1 Datasets 전처리 작업

먼저 LogoDet-3K를 학습에서 사용할 수 있게 전처리가 필요하다. LogoDet-3K에서는 각 사진과 라벨 정보(객체의 위치)가 같은 이름으로 jpg와 xml 파일로 저장되어 있다. YOLOv5의 라벨 데이터는 지정된 형식의 txt 파일로 받아야 하므로 xml 파일을 읽어 parsing한 후 지정된 형식으로 변경해주어야 한다.

LogoDet-3K의 라벨 데이터는 픽셀로 표시하고 있다. 그러나 YOLOv5에서 필요한 라벨 데이터는 절대적이 픽셀값이 아닌 사진 전체의 길이를 1로 보고 표현하는 상대적인 값이므로 이에 알맞게 바꿔줘야 한다. Python의 ElementTree를 이용해 xml 파일을 읽은 후 각각 계산해주어 YOLOv5가 사용할 수 있게 바꿔준다.

5.2 YOLOv5 모델 학습

Yolov5를 이용하여 객체 검출 모델을 만들기 위해서 먼저 Yolov5 모델 학습을 진행하였다.

```
python train.py --img 500 --batch 60 --epochs 50 --data Det-B.yaml —weights yolov5m.pt
```

Yolov5에서 학습은 다음과 같이 train.py를 이용하여 진행할 수 있으며 학습에 사용되는 인자는 다음과 같은 역할을 수행한다.

인자	설명
img	학습에 사용되는 이미지의 크기
batch	학습에 사용되는 batch size
epochs	반복 학습 횟수
data	학습 데이터 경로를 가진 yaml파일
weights	이미 학습된 pretrained model

표 3 - YOLOv5의 학습 인자

또한 학습에 사용한 Datasets도 여러 집단이 존재한다(A ~ D). 각 Datasets의 내용 및 구성은 아래의 표로 정리한다.

Datasets	Data 수(개)	Train - Validation 비율	Category
A	8000	4 : 1	Sport, Medical
B	10000		전체
C	12000		
D	14000		

표 4 - 사용한 Datasets의 특징

위의 인자와 Datasets의 조합을 이용해, 각 sets 별로 최적의 인자값을 찾아 비교하였다. 비교한 결과는 Yolov5에서 지원하는 모델 개발 지원툴인 Wandb(Weight & Biases)를 이용해 시각화했다. 아래의 사진은 Wandb의 결과를 시각화 하는 모습을 캡처한 것이다.

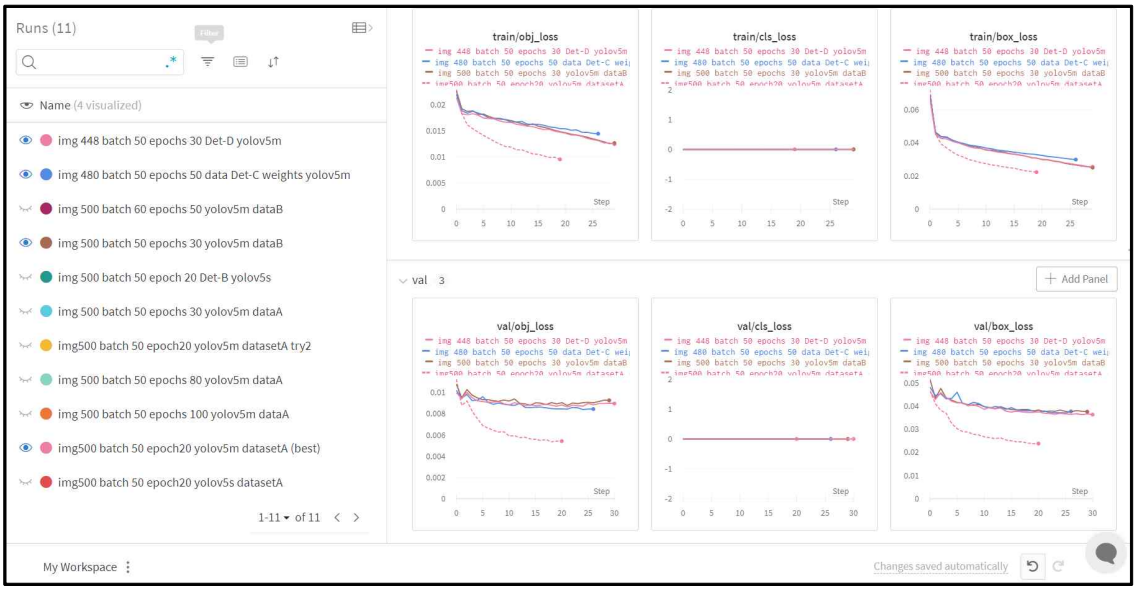

그림 2 - WanDB - Train And Validation metrics

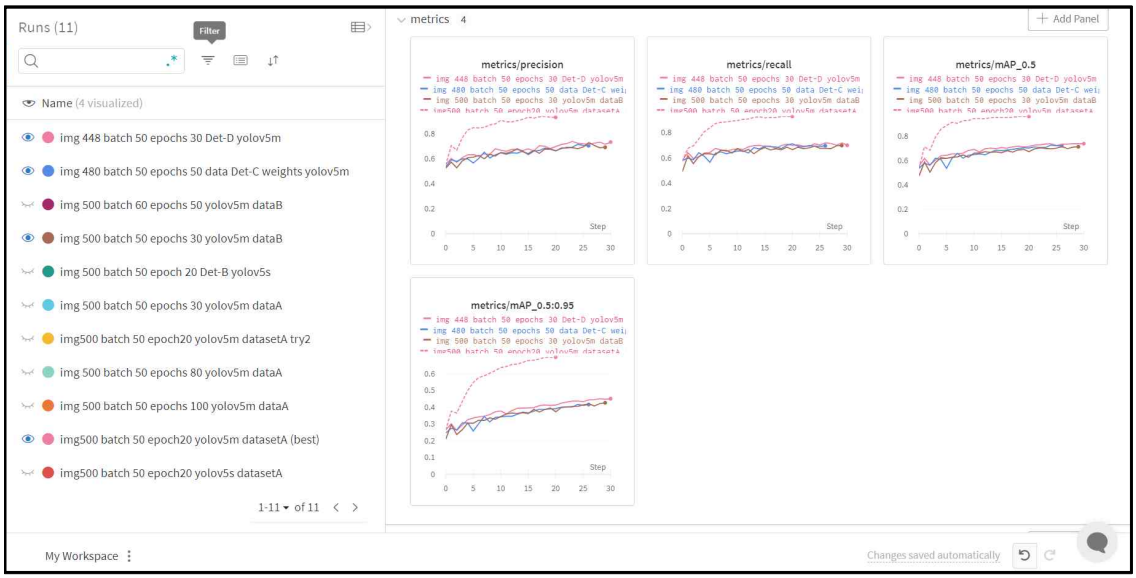

그림 3 - WanDB - Test metrics

그림 4 - Datasets A의 Labeling

그림 5 - Datasets D의 Labeling

분석한 결과, Datasets A가 가장 좋은 지표와 성능을 보이는 것을 확인할 수 있었다. 그러나 이 차이는 Validation sets의 구성이 다른 sets과 차이가 나기 때문이다. Datasets A는 LogoDet-3K의같은 서브카테고리 내에서 Train-Validation set을 나눴다. 이에 반해 Datasets B~D는 LogoDet-3K의 전체 이미지에서 랜덤하게 뽑은 이미지로만 Train-Validation을 나누었기 때문에 A에 비해 Train-Validation set 사이의 유사도가 다소 부족하다. 또한 같은 Train 또는 Validation set이라도 각 이미지 별로 유사도가 부족하다.

하지만 영상 sample을 이용한 결과 분석에서는 가장 많은 데이터를 가지는 Datasets D에서 가장 많은 로고를 탐지해냈다. Yolov5 모델은 이미지에 있는 로고 객체 검출에 사용하고 결과를 이용해서 유사도 분석을 진행한다. 이러한 이유로 가장 많은 로고 객체를 학습한 Datasets D가 가장 많은 로고를 찾아 좋은 성능을 보일 수 있었다.

그림 6 - Detection Result of Datasets A

그림 7 - Detection Result of Datasets D

5.3 React.js 프레임워크 작성

React.js 작성은 최대한 모든 웹 사이트의 구성 요소를 잘게 나누어 각 Component 별로 각 하나의 역할만을 담당하게 하는 방향으로 작성되었다. 이에 따라 최대한 재활용할 여지가 있는 Component들은 common에 위치하게 하며 나머지들은 각각 사용되는 역할에 따라 분류하였다.

현재 개발된 React.js 플랫폼 파일 구조는 아래와 같다.

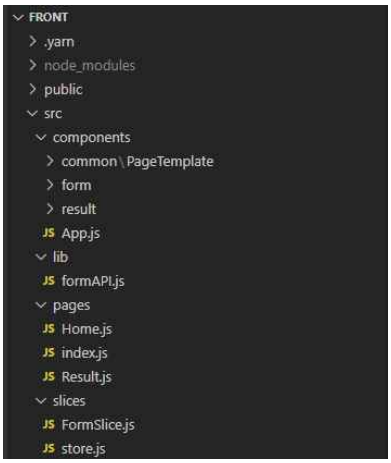

그림 8 - React.js 파일 구조 1

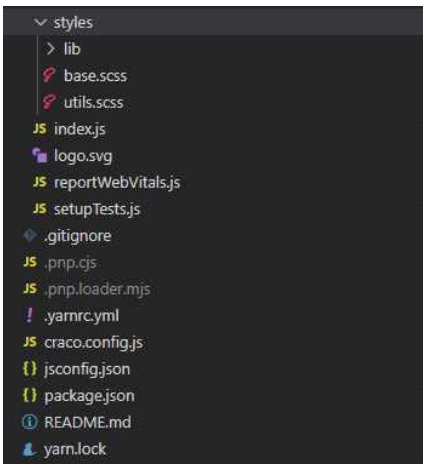

그림 9 - React.js 파일 구조 2

또한 기존의 무겁고 의존성 문제를 가지고 있던 npm 대신 yarn으로 package를 관리하였다. yarn은 npm과 달리 패키지들을 cache로 관리함에 따라 npm에 비해 1/4의 용량만을 차지한 채 패키지를 관리할 수 있다.

먼저 사용자가 처음 보게 되는 페이지이다.

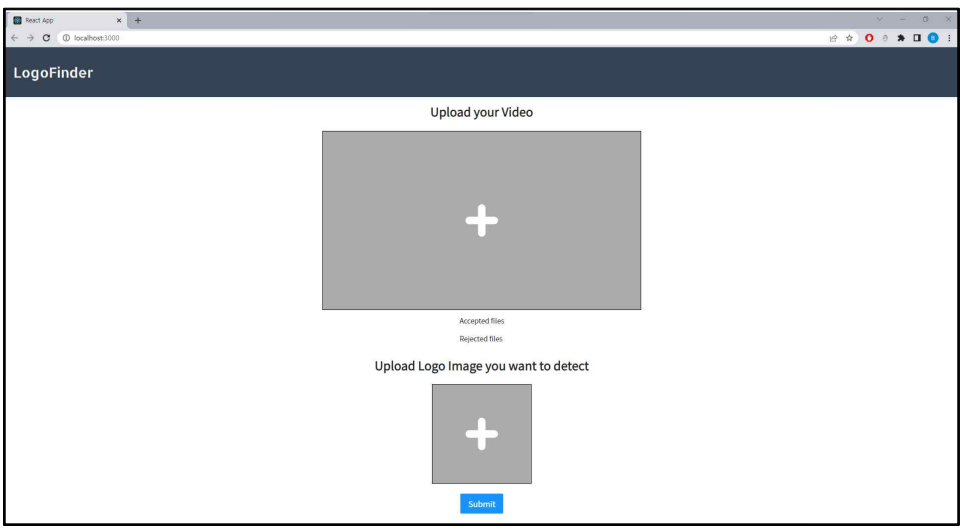

그림 10 - 메인 홈페이지

사용자는 이 페이지에서 비디오와 로고 이미지를 입력할 수 있다. 각각 구역은 Dropzone으로 구현되어 있으므로 파일 탐색기에서 드래그 앤드 드롭 하는 것으로 파일을 넣을 수도 있다.

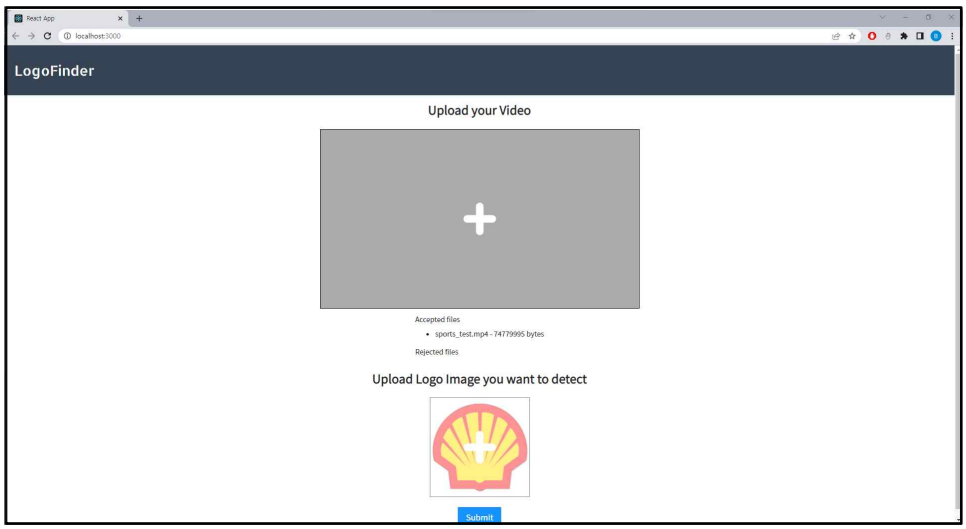

그림 11 - 메인 홈페이지, 파일을 입력했을 때

파일을 넣을 때 위의 사진과 같이 어떤 파일을 넣었는지 알 수 있다. 필요한 파일을 모두 입력한 다음 Submit 버튼을 눌러 결과를 확인할 수 있다. Detection 작업이 상대적으로 오래 걸리기 때문에 결과를 받을 때까지 기다려야 한다. 이때 페이지가 멈춰져 사용자의 불편을 느끼는 것을 방지하기 위해 비동기로 처리한다.

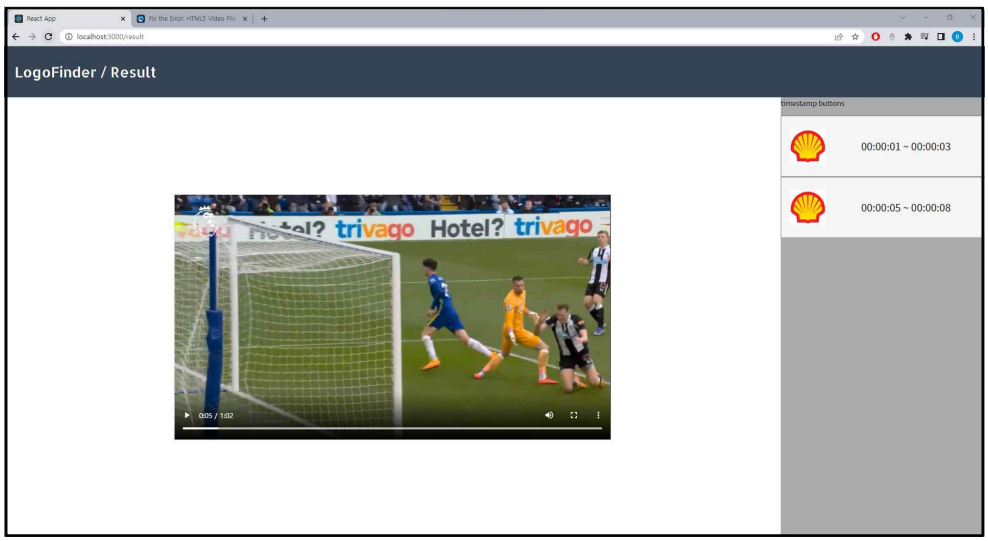

그림 12 - 결과 공개 페이지

결과 페이지에서는 Detection 결과를 Bounding box로 그린 영상과 각 로고의 타임 스탬프로 갈 수 있는 버튼을 보여준다. 현재 버튼에 보이는 결과는 임의로 서버에서 보내는 타임 스탬프의 값이다. 각각 누르면 버튼에 명시되어 있는 시간대로 이동할 수 있다.