

11조

Sign_Language.txt

계속반 계획서



과목명	졸업프로젝트2
담당교수	김두현 교수님
지도교수	임창훈 교수님
학과	컴퓨터공학과
팀원	201815004 이찬민
	201611205 박승민
	201814119 문지영

목차

1. 시작반 프로토타입 분석	3
2. 변경(개선) 사항	6
3. 변경된 대표 시연 시나리오.....	7
4. 시스템 구축 계획	9
개발 환경 구축 계획	9
시스템 구축 계획	10
5. 구현 계획.....	11
화면.....	11
채팅.....	11
웹캠.....	11
예측.....	11
DB.....	11
6. 시험 계획.....	12
7. 팀원 역할 분담	13
8. 일정	14

1. 시작반 프로토타입 분석

수어 통역 채팅 프로그램을 하는 데 제일 중요했던 문제는 성능 리스크, 즉 “얼마나 수어 인식을 잘 할 수 있는가?” 였다.

모델을 만들고 딥러닝을 수행할 때 한 단어 당 몇번의 학습을 시키는 것이 제일 적절할 지가 가장 큰 관건이었다.

많은 횟수를 학습시키면 정확도는 확실히 올라가나, 모델을 생성하는 시간이나 데이터가 기하급수적으로 커지는 문제점이 있었고,

적게 학습시키면 정확도가 떨어져서 시연 시 제대로 인식이 안되거나, 혹은 동작이 비슷한 다른 단어로 오인식을 하는 경우가 많이 생겼다.

예시) 유사한 동작을 가진 수어들



감사합니다



사과하다

비슷한 단어를 구별하면서도 데이터를 최소화하기 위한 실험을 반복해 본 결과, 모델을 만들 때 약 30번 정도 촬영하는 것이 가장 적절하다는 결론에 도달했다.

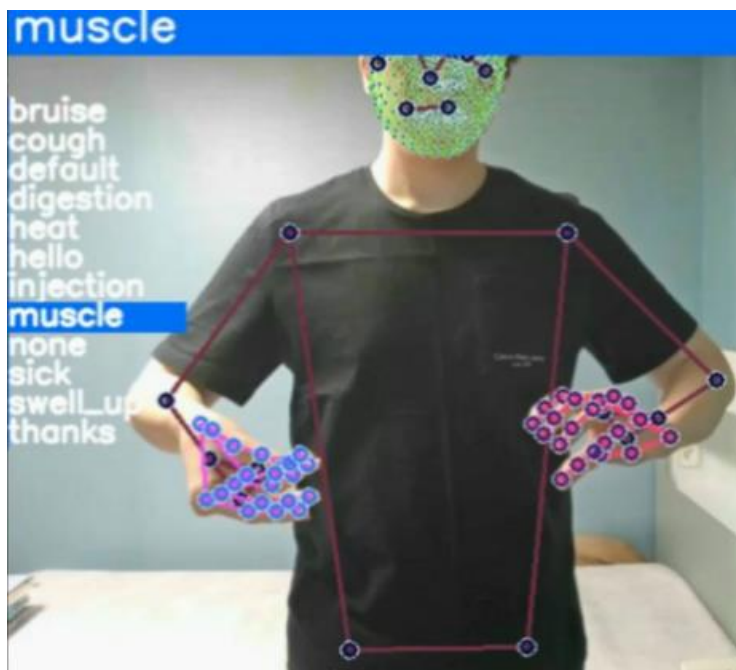
또한 촬영한 모델이 다른 사람이 시연했을 때도 잘 번역되는지, 즉 범용성 리스크도

짚고 넘어가야 할 문제였다.

우리 모두의 신체 조건이 다르므로 모델을 촬영한 사람의 신체조건과 상당한 차이가 있는 사용자가 사용할 경우 무슨 차이가 있는지 반드시 짚고 넘어가야 했다.

이를 위해서 조원들의 지인들에게 시연을 부탁한 결과, 서로 다른 사용자가 시연해도 문제없이 잘 인식함을 확인하였다.

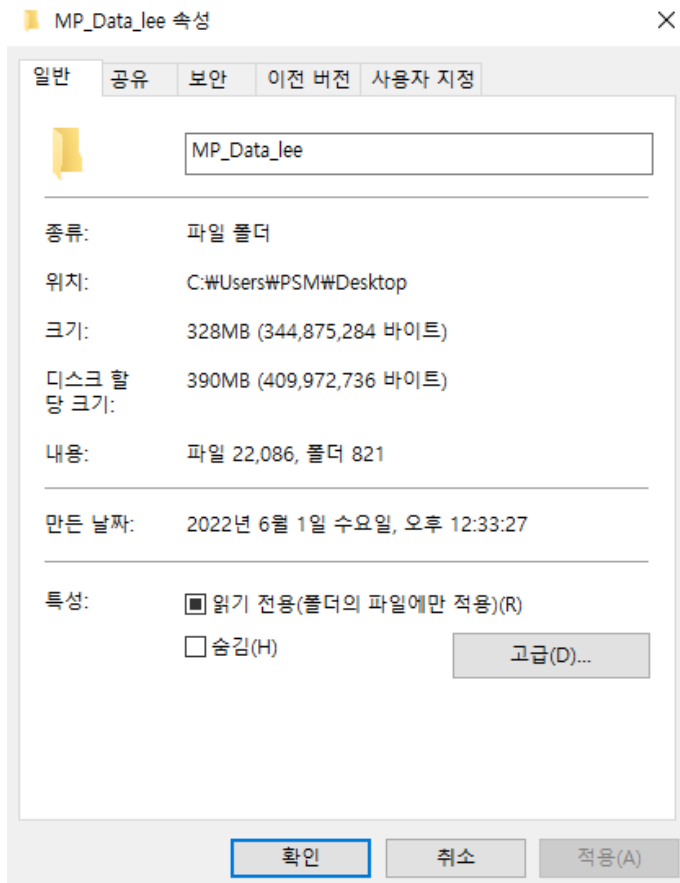
다만 사용자의 차이보다는 카메라각도, 카메라와 사용자 간의 거리, 촬영범위 등에 인식이 좌우된다는 것도 확인하였다.



반복적 시연 결과 이미지와 같이 상반신이 다 나오게 일어서서 시연해야 미디어파이프의 포인트가 잘 찾아가서 인식이 원활하게 이루어진다는 결론을 내렸다.

그리고 시연 종료 후 손 또한 카메라 앵글 밖으로 벗어나지 않는 것이 좋았다. 손이 급격 벗어나지 않게 카메라를 설정하거나, 미리 공지하는 방식으로 사용자에게 알려주는 것이 좋을 것 같다.

수어가 다양한 만큼 데이터 리스크도 존재하는데 앞서 설명한 바와 같이 많은 단어를 입력하려면 데이터가 기하급수적으로 많이 필요하다



15개의 단어를 30번 반복촬영한 데이터가 약 390MB로 수백개의 단어를 쓴다고 가정하면 수집기가 이상의 용량이 필요하다.

이를 기반으로 딥러닝을 수행해서 모델을 생성할 경우, 2000번 반복학습 기준으로 약 15분 정도가 소요되는데, 단어가 많이 늘어나면 늘어날수록 시간이 가늠할 수 없을 정도로 늘어나게 된다는 문제도 있다.

그래서 미리 상정한 시나리오에서 정말 필수적으로 필요한 어휘 및 일상 용어 위주로 (ex. 병원에서 주사, 항생제, 통증 등등) 단어를 선별해서 이용할 예정이다.

2. 변경(개선) 사항

청각장애인이 수어를 평소의 속도로 하게 된다면 너무 빨라 keypoints들이 제대로 그려지지 않을 수 있기 때문에 사용자의 수어 속도를 늦춰야 한다는 부분이 추가될 것이다. 또한 인식 성능이 비슷한 수어의 구별은 완벽하지 않기 때문에 성능 요구사항에서 재시도 하는 횟수를 최대 3회로 하되 비슷한 수어에 한하여 최대 5회까지 늘릴 수 있도록 한다는 부분이 변동되었다.


이에 따라 본래 단어의 선택폭에 어느정도 자율성을 주기 위해 자음, 모음을 넣을 생각이었으나, 프로그램의 성능을 감안하였을 때 현실적으로 시연 시간이 너무 길어져서 일단 포함하지 않는 방향으로 갈 생각이다.

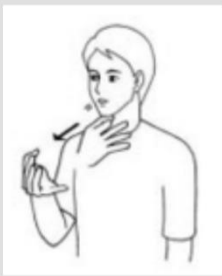
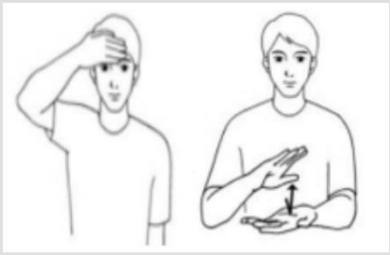

하드웨어 적인 성능을 생각보다 많이 요구해서(16GB 램 기준 71%까지 사용률이 치솟음) 이를 최소화하는 방향을 모색해 봤으나, 일단 프로그램 자체의 성능을 우선으로 보고 하드웨어의 확충을 해서 프로젝트를 진행할 것이다.

3. 변경된 대표 시연 시나리오

기존의 병원 시나리오를 채팅 형식으로 시연을 했습니다.

오른쪽(초록색)이 의사이고 왼쪽(회색)이 청각장애인인 환자입니다.

 <p>안녕하세요.</p> <p>(안녕하세요.)</p> <p>'안녕하세요' 동작</p>	 <p>어디가 아프셔서 오셨나요?</p> <p>(목, 코를 가리킨다.)</p> <p>아픈 부위를 가리키며 '아프다' 동작</p>
---	---

<div data-bbox="240 226 748 533">  </div> <div data-bbox="448 584 748 669"> <p>어떤 증상이 있으세요?</p> </div> <div data-bbox="245 750 549 835"> <p>(기침)</p> </div> <div data-bbox="199 1059 343 1095"> <p>'기침' 동작</p> </div>	<div data-bbox="845 226 1353 533">  </div> <div data-bbox="1050 584 1350 669"> <p>어떤 증상이 있으세요?</p> </div> <div data-bbox="850 750 1153 835"> <p>(기침)</p> </div> <div data-bbox="850 857 1153 943"> <p>(열)</p> </div> <div data-bbox="805 1059 917 1095"> <p>'열' 동작</p> </div>
<div data-bbox="240 1104 748 1411">  </div> <div data-bbox="448 1462 748 1563"> <p>네 목감기 걸리셨네요. 약 3일치 처방전 드릴게요.</p> </div> <div data-bbox="245 1630 549 1715"> <p>(감사합니다.)</p> </div> <div data-bbox="199 1939 422 1975"> <p>'감사합니다' 동작</p> </div>	<div data-bbox="1050 1131 1350 1216"> <p>안녕하세요.</p> </div> <div data-bbox="850 1234 1153 1319"> <p>(안녕하세요.)</p> </div> <div data-bbox="1050 1361 1350 1447"> <p>어떤 증상이 있으세요?</p> </div> <div data-bbox="850 1464 1153 1550"> <p>(기침)</p> </div> <div data-bbox="850 1572 1153 1657"> <p>(열)</p> </div> <div data-bbox="1050 1702 1350 1803"> <p>네 목감기 걸리셨네요. 약 3일치 처방전 드릴게요.</p> </div> <div data-bbox="850 1832 1153 1917"> <p>(감사합니다.)</p> </div> <div data-bbox="805 1939 999 1975"> <p>전체 대화 내용</p> </div>

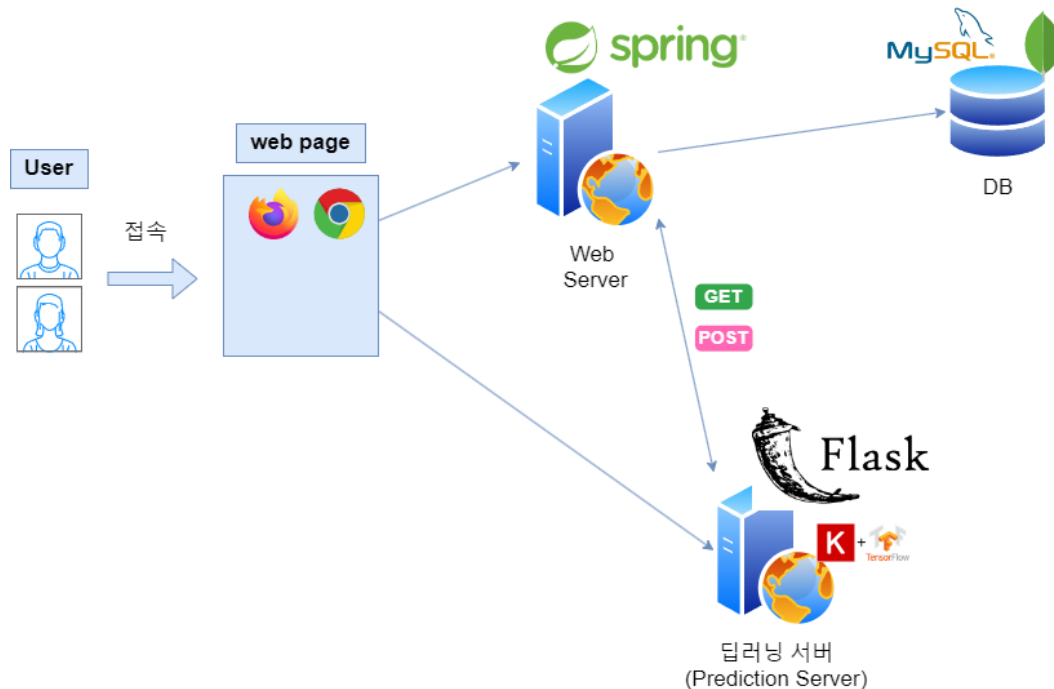
4. 시스템 구축 계획

개발 환경 구축 계획

- 운영체제: Windows 10
- 형상관리 소프트웨어: Git을 이용하여 형상관리를 할 계획이다.
- 웹서버: Spring을 이용하여 전반적인 웹과 채팅서버
- 딥러닝 서버: Flask를 이용하여 수어 예측 및 결과 반환하도록 구성할 계획이다.
- Web Application Server: Tomcat(초기엔 내장 톰캣을 이용할 계획이다.)
- 딥러닝 서버: 화면에 웹 캠 띄우기 및 모델을 통한 예측, 결과값 웹서버로 전송을 Flask 프레임워크를 이용하여 수행할 계획이다.
- 데이터베이스: MySQL 또는 MongoDB
- 테스트 도구: Spring의 Junit
- IDE: IntelliJ, VSCode

시스템 구축 계획

시스템 구축도



- 웹페이지의 경우에는 html, css, javascript를 이용하여 구축할 계획이다.
- Spring boot를 이용하여 웹 서버를 구축하여 대부분의 유저의 request를 처리할 계획이다. 웹서버의 경우에는 수어 예측을 제외한 채팅, 영상 등을 처리한다.
- 딥러닝 서버는 Flask 프레임워크를 이용하여 구축할 계획이다.
- 웹 서버와 딥러닝 서버 간의 통신은 딥러닝 서버를 REST API처럼 구성하여 GET, POST를 이용하여 커뮤니케이션 할 계획이다.
- DB는 MongoDB 또는 MySQL을 사용하여 스프링에 연결하여 데이터를 관리할 예정이다.

5. 구현 계획

화면

- 메인 화면 - Bootstrap을 이용하여 구현
- 병원 페이지 화면 - html과 css를 이용하여 구현

채팅

- 채팅방 생성 - RoomController에서 POST를 통해 생성 가능하도록 구현
- 채팅방 입장 - RoomController에서 GET을 통해 목록을 받아와 화면에 출력하도록 구현
- 채팅창 생성 - 채팅방 입장 시 채팅이 가능하도록 구현
- 클라이언트 간의 통신 - STOMP를 이용하여 여러 개의 채팅방 생성 및 채팅이 가능하도록 구현하고 StompChatController를 통해 메시지 전달이 이루어질 수 있도록 구현
- 예측 결과 출력 - 딥러닝 서버로부터 받아온 결과를 출력하도록 구현

웹캠

- 영상 촬영 - Spring에서 웹캠의 내용을 blob로 만든 후 avi확장자로 변환하도록 구현
- 영상 전송 - 답러닝 서버에 Post를 통해 전송하도록 구현

예측

- 영상 획득 - Spring으로부터 avi확장자의 비디오를 획득하도록 구현
- 영상 예측 - 미리 학습된 모델을 통해 받아온 비디오로부터 예측결과를 생성하도록 구현
- 웹서버로 결과 전송 - Http 메소드를 이용하여 Spring 웹서버에 전송하도록 구현

DB

- 채팅방 이름과 목록 저장

6. 시험 계획

Testcase	#	조건	기대 결과	Usecase
수어 동작 하기	1	수어 동작을 한다.	수어 정보를 학습된 모델로 전송한다.	수어 입력
텍스트 입력하기	2	채팅에서 텍스트를 입력한다.	입력된 텍스트를 서로의 화면에 표시한다.	텍스트로 입력
버튼 선택하기	3	자주 사용하는 표현은 버튼을 누른다.	선택된 버튼의 텍스트를 서로의 화면에 표시한다.	텍스트로 입력
수어 학습 모델 정확도 확인하기	4	충분한 양의 데이터를 학습시킨다.	정확도가 90%이상이고 categorical_crossentropy의 손실함수가 정상적으로 그려졌는지 확인한다.	
학습된 모델 성능 확인하기	5	학습시킨 단어의 수어 동작을 입력한다.	정확한 수어 예측값이 나온다.	
	6	동작이 유사한 단어의 수어를 입력한다.	정확한 수어를 예측 시간이 2초 이내이다.	
	7	제3자의 수어 동작을 입력한다.	정확한 수어 예측값이 나온다.	
수어를 텍스트로 번역하기	8	사용자로부터 학습된 수어 데이터를 받는다.	학습된 모델로 예측을 하고 예측된 결과를 웹서버로 전송한다.	수어를 텍스트로 번역
	9	사용자로부터 학습되지 않은 수어 데이터를 받는다.	사용자에게 수어 동작을 다시 요구한다.	
번역한 텍스트 출력하기	10	수어 예측값을 웹서버에 전송한다.	서로의 화면에 예측값을 텍스트로 보여준다.	번역한 텍스트 출력

7. 팀원 역할 분담

- 이찬민 (팀장)

- 모델학습을 위한 데이터 생성
- 모델 생성 및 학습
- 모델 테스트
- 웹페이지 생성
- 모델 비교
- 웹서버, 딥러닝 서버 구축

- 박승민

- 모델학습을 위한 데이터 생성
- 모델 테스트
- 모델 비교
- 웹페이지 구성

- 문지영

- 모델학습을 위한 데이터 생성
- 시나리오 생성
- 모델 테스트
- 채팅방 DB 관리
- 수어 예측 결과 DB 관리

8. 일정

	프로젝트 진행 계획	보고서 제출 일정
3월	- 수어를 글로 번역하는 주제 선정	- 프로젝트 신청서 제출
4월	- 특정 목적(예, 병원)을 위한 수어 데이터를 수집 - 사용할 오픈소스를 탐색	- 요구사항 분석 - 요구사항 분석서 작성 - 시스템 설계
5월	- 딥러닝을 이용한 영상 처리	- 시스템 설계서 작성 - 프로토타입 설계 및 구현 - 프로토타입 테스트
6월	- 딥러닝을 이용한 영상 처리	- 프로토타입 구현결과 발표 - 기말보고서 작성
7월	- 웹페이지 및 서버 개발 - 인식속도 및 인식률 개선	
8월	- 웹페이지 및 서버 개발	
	-	
9월	- 웹 서버에 OpenCV를 올리기 - 웹페이지에 채팅 및 버튼 구현	- 프로젝트 계획서 - 시스템 구조 설계서 수정
10월	- 추가 시나리오를 위한 단어로 모델 학습	- 풀 스케일 시스템 구축
11월	- 디버깅 - 최종 완성 및 시연	- 시험 계획 수립 및 시험 시행 - 시험 결과 분석을 통한 디버깅 및 성능/기능 최종 개선
12월	- 코드 리팩토링	- 최종발표