

[캡스톤디자인]

최종 보고서

Language Ribbon

: Authentic Voice-Preserving Speech
Translation App

지도 교수 : 김 대 원 교 수 님

조 원 : 김소현 (60201868) 조장

김민희 (60200860)

이동혁 (60182196)

채기웅 (60201710)

최지현 (60211904)

2023.12.19.



명지대학교
MYONGJI UNIVERSITY

목 차

1. 문제 정의	3
1.1. 제안 배경	3
1.2. 기대 효과	3
1.3. 애플리케이션 개요	3
2. 시스템 아키텍처	4
2.1. 시스템 구성도	4
2.2. ERD	5
2.3. 데이터베이스 설계도	6
2.4. 메뉴 구성도	6
3. 주요 기술	7
3.1. 프론트엔드	7
3.2. 알고리즘	15
3.2.1. Speech To Text, STT	15
3.2.2. Large Language Models, LLM	16
3.2.3 Text To Speech, TTS	16
3.2.4. Voice Conversion, VC	16
3.3. 백엔드	16
3.3.1. 서버 구축과정	16
3.3.2. 회원 정보 관리	17
3.3.3. 초기 목소리 수집	18
3.3.4. 음성 변환	18
4. 버그 히스토리	20
5. 검증 시나리오 및 결과	25
5.1. MOS 테스트	25
5.2. 사용성 테스트	29
6. 최종 결과 분석 및 한계	31
참고 문헌	31

1. 문제 정의

1.1. 제안 배경

최근 인공지능(AI) 기술의 급속한 발전은 번역 분야에서 혁명적인 변화를 가져오고 있다. 특히, AI를 활용한 번역기 및 ChatGPT와 같은 언어 모델의 등장으로 언론은 통번역사의 역할이 줄어들거나 사라질 것으로 예측하고 있다. 그러나 현실적으로 AI 번역은 아직 사용자 경험과 효율성 측면에서 한계를 가지고 있다. 구글 번역기, 파파고, 딥엘(DeeL)과 같은 대표적인 번역기는 음성 서비스를 제공한다. 실시간 통역에 가장 가까운 기술을 제공하고 있지만, 여전히 짧은 문장에 적합하며 음성 인식 과정에서 오류가 발생할 수 있다.

거대 언어 모델 기반의 서비스인 ChatGPT는 관용구와 속담을 포함한 함축적인 의미를 정확하게 해석하며, 이를 효과적으로 번역하는 능력을 갖추고 있다. 또한 정리되지 않은 구어체 발화에 대한 통역에서 문장 내 맥락을 파악하는 데 뛰어난 능력을 보여주어, 기존 기계 번역 서비스의 한계를 극복할 수 있는 가능성을 제시하고 있다.

1.2. 기대 효과

본 연구는 거대 언어 모델(Large Language Model, LLM)을 기반으로 한 AI 번역과 기존의 통역이나 기계 번역 서비스에서 불가능했던 음성 변환(Voice Conversion, VC)을 적용한 서비스를 제안한다. 해당 서비스가 기존 기계 번역 서비스의 한계를 극복하고, 음성 변환을 통해 화자 간의 직접적인 의사소통 과정에서 더 큰 유대감을 형성할 수 있을 것으로 기대한다. 또한 의사소통 목적 외에도 발음 및 억양을 교정하는 교육적 목적으로도 활용할 수 있을 것으로 기대한다.

1.3. 애플리케이션 개요

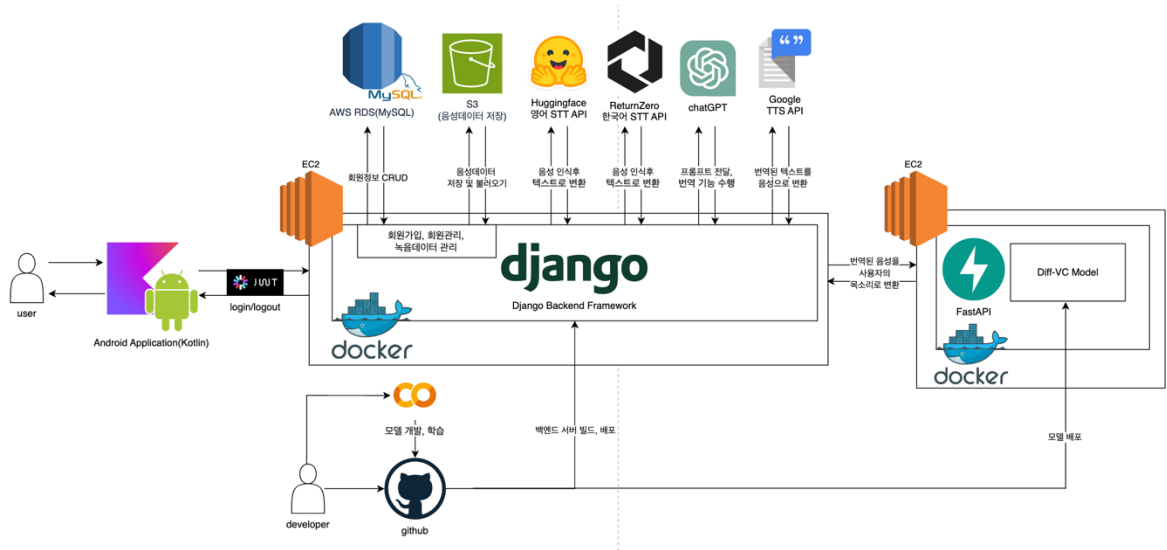
본 연구는 “Language Ribbon”이라는 음성 번역 어플리케이션의 개발을 목표로 설정하였다. 이 어플리케이션의 주요 목표는 사용자의 음성 특성을 유지한 채로, 해당 언어에 능숙하지 않은 사용자도 본인이 직접 말하듯이 대화를 진행할 수 있게 하는 것이다. 타겟 사용자는 영어에 익숙하지 않은 사람들로, 이들이 외국인 친구를 사귀거나 영어로 회의나 발표를 진행할 때 이 어플리케이션을 활용하도록 설계하였다.

어플리케이션의 주요 기능은 (1) 사용자의 한국어 또는 영어 발화 음성을 입력, (2) 사용자가 번역하고자 하는 언어를 선택, (3) 번역하기를 원하는 음성 입력, (4) 번역된 음성과 텍스트를 실시간으로 출력하는 통역 어플리케이션 기능이다.

이 어플리케이션의 활용을 통해 사용자들은 낯선 외국인과의 대화를 더욱 쉽게 진행할 수 있게 되며, 기계음이 아닌 사용자의 본래 음성을 유지한 번역을 통해 보다 높은 유대감을 형성하는 데 도움이 될 것으로 기대된다.

2. 시스템 아키텍처

2.1. 시스템 구성도



시스템 구성도는 다음과 같이 크게 안드로이드 어플리케이션, AWS EC2 인스턴스로 구성된 백엔드 서버와 Diff-VC 모델을 서빙 서버 그리고 백엔드 서버 및 모델 개발, 배포 부분으로 이루어져있다.

안드로이드 어플리케이션: Kotlin으로 작성된 안드로이드 어플리케이션을 통해 사용자(user)는 서비스 회원가입, 로그인, 통역 서비스 이용이 가능하다.

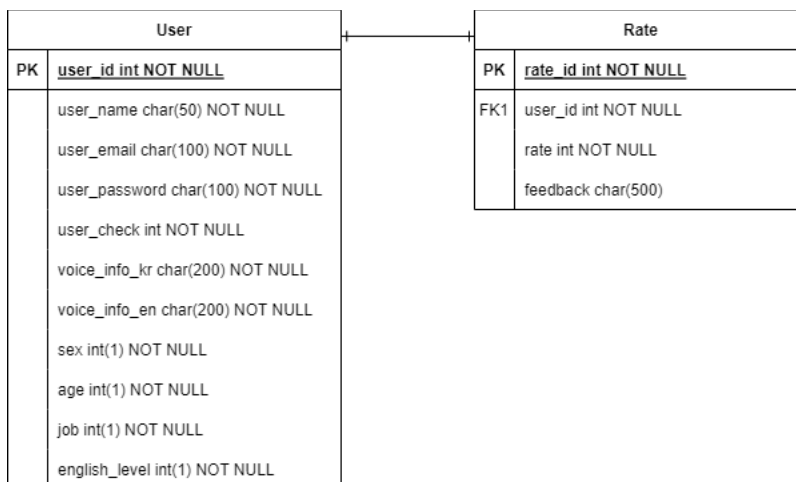
백엔드 서버: AWS EC2 인스턴스위에 도커 컨테이너로 구축된 백엔드 서버는 파이썬 기반의 백엔드 프레임워크인 django로 작성된다. 이 백엔드 컨테이너를 통해 사용자는 회원가입, 로그인, 서비스 이용을 수행하며, 백엔드 서버는 7개의 다른 서비스들과 API를 통해 연결되어, 회원가입, 로그인, 통역 서비스를 프론트 단의 안드로이드 어플리케이션에 제공한다. 백엔드 서버는 MySQL이 포함된 AWS RDS(DB)와 상호작용하며 테이블을 생성, 마이그레이션하고 기본적인 DB CRUD(Create, Read, Update, Delete)작업을 수행할 수 있다. AWS S3는 AWS에서 제공하는 가용성 높은 객체스토리지 서버로, 이 스토리지 서버를 통해 회원가입시 녹음된 초기음성 파일을 저장하고, 저장경로를 AWS RDS에 저장하여 음성파일 사용이 필요할 때 RDS를 통해 S3에 저장된 음성파일 경로를 획득하여 S3에서 원하는 음성파일을 가져올 수 있다. 이 두 서비스와 백엔드 서버가 결합하여 회원가입, 회원관리, 녹음데이터 관리를 수행한다. 백엔드 서버와 연결된 나머지 서비스들은 사용자음성을 통역하여 사용자의 목소리로 번역된 음성을 제공하는 통역서비스에 사용된다. Huggingface 영어 STT(Speech to Text) API, returnzero 한국어 STT API 서비스는 사용자가 서비스 이용시 녹음한 음성을 해당 서비스로 전송하여 STT 결과인 인식된 텍스트를 받아오게 하는 역할을 수행한다. chatGPT는 백엔드 서버에서 STT API를 통해 전달받은 텍스트를 번역하는데 사용되며, 백엔드 서버는 번역 프롬프트를 추가하여 chatGPT로 전송하고, 번역 결과를 받아와 Google TTS API를 통해 한국어 음성으로 변환한다. 이렇게 통역된 음성과

회원가입시 녹음한 사용자 초기 음성을 Diff-VC 서버에 전송하여 사용자 목소리의 통역 음성으로 변환하여 사용자에게 통역 서비스를 제공한다.

Diff-VC 서버: Diff-VC 서버는 diffusion voice conversion(diffusion 알고리즘을 이용해 입력음성을 목표음성과 비슷한 스타일의 음성으로 변형)을 수행하는 Diff-VC 모델을 FastAPI를 통해 API로써 서빙한 모델 추론 서버이다. Diff-VC 서버는 GPU 사용을 지원하는 도커 상에 컨테이너의 형태로 작동한다. Diff-VC 서버는 입력음성과 목표 음성 파일을 입력받아, 입력음성의 목소리 스타일을 목표음성의 목소리 스타일로 변형한 음성파일을 결과로 제공한다.

백엔드 서버 및 모델 개발, 배포 : 개발자는 colab을 통해 음성모델 학습을 진행하고, github를 통해 개발한 백엔드서버 코드와 학습된 모델을 백엔드 서버로 배포한다.

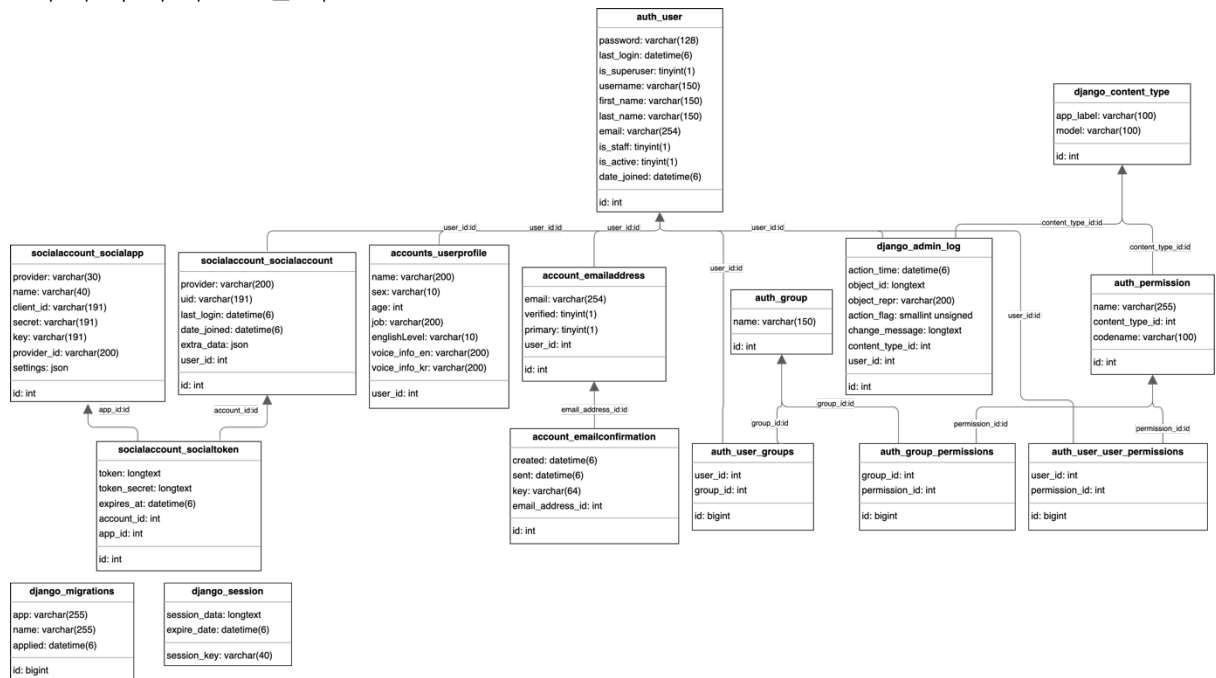
2.2. ERD



1. User 테이블은 사용자 정보를 관리하는 주요 테이블로, 각 사용자는 고유한 user_id를 가지며, 이는 Primary Key로 설정되어 있다. 사용자의 이름, 이메일, 비밀번호, 성별, 나이, 직업, 영어 수준 등의 정보는 각각 user_name, user_email, user_password, sex, age, job, english_level 필드에 저장된다. 또한, user_check 필드를 통해 사용자 계정의 이용 약관 동의 상태를 확인할 수 있으며, 사용자의 한국어와 영어 초기 음성 정보는 각각 voice_info_kr, voice_info_en 필드에 저장된다.

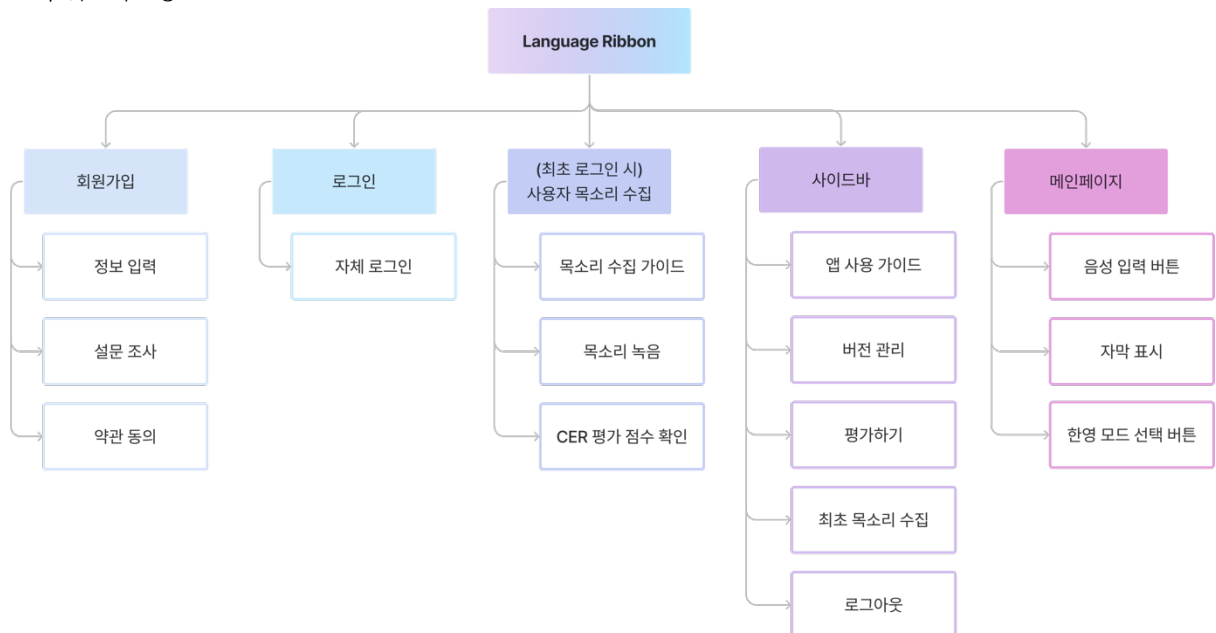
2. Rate 테이블은 사용자의 서비스 이용에 대한 평가와 피드백을 관리한다. Rate 테이블은 고유한 rate_id를 Primary Key로 가지고, user_id는 User 테이블과 연결된 Foreign Key로 설정되어 있다. 이를 통해 각 평가와 피드백이 어떤 사용자에게 속하는지 쉽게 파악할 수 있다. 사용자의 평가 점수는 rate 필드에, 그리고 사용자의 자유 형식 피드백은 feedback 필드에 저장된다.

2.3. 데이터베이스 설계도



Django ORM을 활용하여 회원 관리와 서비스 관리를 위한 데이터베이스 스키마를 구축했다.

2.4. 메뉴 구성도



3. 주요 기술

3.1. 프론트엔드

3.1.1. 회원가입

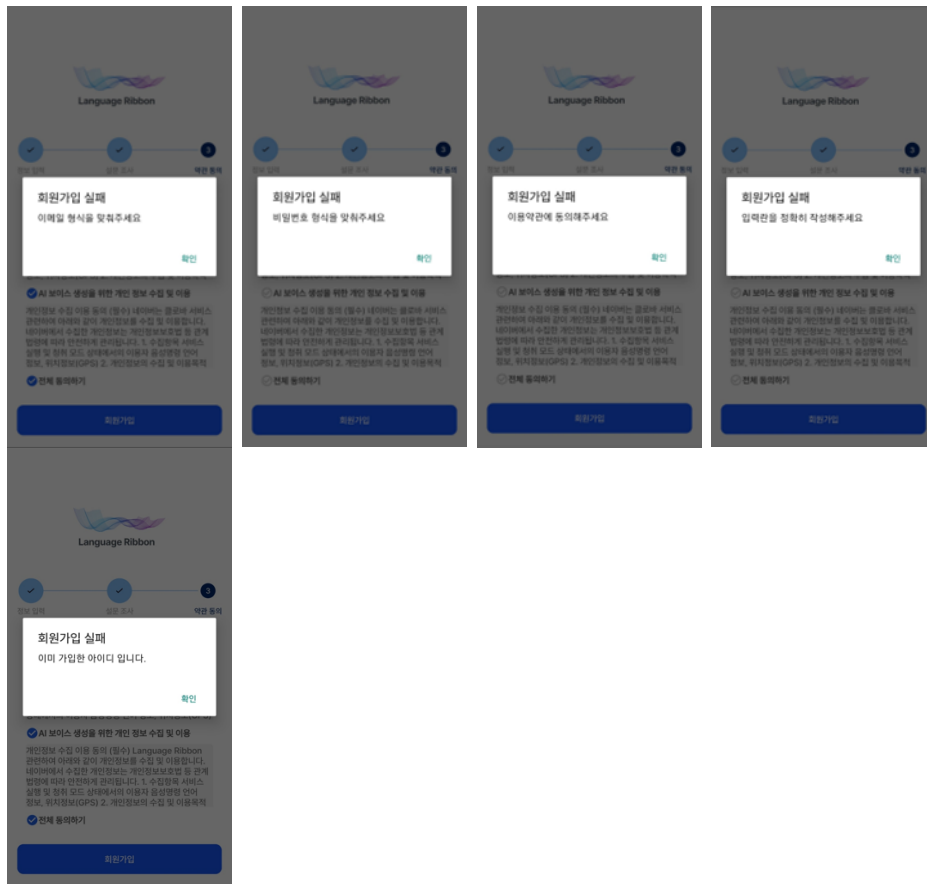
회원가입은 총 3단계(정보입력, 설문조사, 약관 동의)로 이루어진다. 각 요소는 모두 입력해야 회원가입 post가 서버로 전달된다. 회원가입을 성공하면 로그인 페이지로 이동하고 이메일 형식, 비밀번호 형식, 이용약관 미동의, 입력 미완료, 이미 가입된 아이디와 같은 회원가입 실패시 오류 alert가 화면에 띄워진다. 로그인 버튼을 누르면 로그인 페이지로 회원가입 없이 이동할 수 있다.

- 회원가입 이미지

The registration process consists of three steps:

- Step 1: 정보 입력 (Information Input)**
 - Language Ribbon을 사용하기 위해 몇 가지 정보가 필요합니다.
 - 이름:
 - 이메일: @ **직업 선택**
 - 비밀번호(특수기호 포함 8자 이상):
 - 비밀번호 확인:
 - 성별: ☒ 남성 ☐ 여성
 - 로그인
 - Next
- Step 2: 설문 조사 (Survey)**
 - Language Ribbon을 사용하기 위해 몇 가지 정보가 필요합니다.
 - 사용자의 연령대를 알려주세요: **연령대 입력**
 - 사용자를 가장 잘 나타내는 문항을 골라주세요.
☒ 학생 ☐ 직장인 ☐ 기타
 - 사용자의 외국어(영어) 실력을 골라주세요.
☒ 상 ☐ 중 ☐ 하
 - 다음
- Step 3: 약관 동의 (Terms of Service)**
 - ☒ **Language Ribbon 사용 동의**
개인정보 수집 이용 동의 (필수) 네이버는 클로바 서비스 관련하여 아래와 같이 개인정보를 수집 및 이용합니다. 네이버에서 수집한 개인정보는 개인정보보호법 등 관계 법령에 따라 안전하게 관리됩니다. 1. 수집항목 서비스 실행 및 청취 모드 상태에서의 이용자 음성명령 언어 정보, 위치정보(GPS) 2. 개인정보의 수집 및 이용목적
 - ☒ **AI 보이스 생성을 위한 개인정보 수집 및 이용**
개인정보 수집 이용 동의 (필수) 네이버는 클로바 서비스 관련하여 아래와 같이 개인정보를 수집 및 이용합니다. 네이버에서 수집한 개인정보는 개인정보보호법 등 관계 법령에 따라 안전하게 관리됩니다. 1. 수집항목 서비스 실행 및 청취 모드 상태에서의 이용자 음성명령 언어 정보, 위치정보(GPS) 2. 개인정보의 수집 및 이용목적
 - ☒ **전체 동의하기**
 - 회원가입

- 회원가입 오류



3.1.2. 로그인

로그인은 자체 로그인으로 진행되고 회원가입한 이메일과 비밀번호를 입력받고 로그인 버튼을 클릭하면 “로그인 중”이라는 toast message와 함께 서버에 post된다. 성공시 메인페이지로 이동하고 실패시 “로그인에 실패”라는 toast message가 화면에 나타난다.

- 로그인 이미지



- 로그인 중



zyzu@naver.com

.....

로그인

로그인 중입니다.

- 로그인 실패



zyzu@naver.com

.....

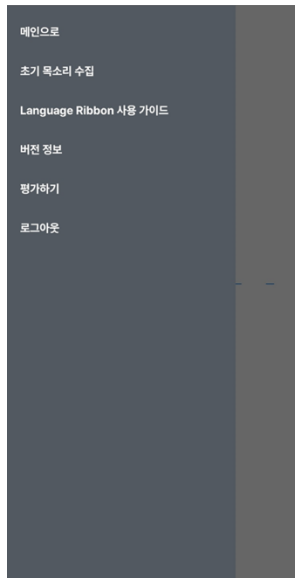
로그인

로그인 실패

3.1.3. 메뉴

각 페이지는 사이드 메뉴를 통해 이동할 수 있다. 각 메뉴 클릭시 이전 fragment는 제거되고 해당 fragment 생성된다. 로그아웃을 클릭시 로그아웃이 되고 회원가입 페이지로 이동된다.

- 메뉴 이미지



3.1.4. 메인화면


메인화면의 기능은 사용자가 본인의 언어로 음성을 입력하면 본인의 음성을 살린 번역언어가 출력되는 것이다. 초기 목소리 설정을 하지 않았을시 녹음 버튼을 눌러도 초기목소리 수집 페이지로 이동한다 초기목소리 설정을 완료한 후 녹음 버튼을 누르면 메인기능이 이용가능하다. 녹음 시작 후 녹음 완료 버튼을 클릭하면 사용자 언어, 타겟언어, 음성파일이 전송되고 음성이 생성중이라는 텍스트가 화면에 표시된다. 서버로부터 값을 받아오면 음성이 재생되고 화면에도 영어/한국어 텍스트가 나타난다.

- 메인이미지




- 스크립트 녹음 단계

Language Ribbon




녹음 버튼을 누른 후
천천히 스크립트를 읽어 주세요.

저는 Language Ribbon에 제 목소리를 등록하여,
이를 통해 제 Language Ribbon 계정에
제 목소리를 사용할 수 있는 권한을 부여함을
동의합니다.
2023-12-15




다음



Language Ribbon



녹음 버튼을 누른 후
천천히 스크립트를 읽어 주세요.

저는 Language Ribbon에 제 목소리를 등록하여,
이를 통해 제 Language Ribbon 계정에
제 목소리를 사용할 수 있는 권한을 부여함을
동의합니다.
2023-12-15




재녹음하기


다음

Language Ribbon




녹음 버튼을 누른 후
천천히 스크립트를 읽어 주세요.

I register my voice with Language Ribbon
and agree to grant my Language Ribbon
account
the rights to use my voice.
2023-12-16




다음



Language Ribbon



녹음 버튼을 누른 후
천천히 스크립트를 읽어 주세요.

I register my voice with Language Ribbon
and agree to grant my Language Ribbon
account
the rights to use my voice.
2023-12-15







재녹음하기

다음


- CER 단계

Language Ribbon



목소리 녹음 정확도를 평가한
결과입니다.

영어 CER




20 %

* CER(Character Error Rate)

녹음을 성공적으로 마쳤습니다.

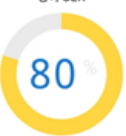
초기 목소리 설정 완료

Language Ribbon



목소리 녹음 정확도를 평가한
결과입니다.

영어 CER




80 %

* CER(Character Error Rate)

녹음을 성공적으로 마쳤습니다.


초기 목소리 설정 완료

Language Ribbon



목소리 녹음 정확도를 평가한
결과입니다.

한국어 CER



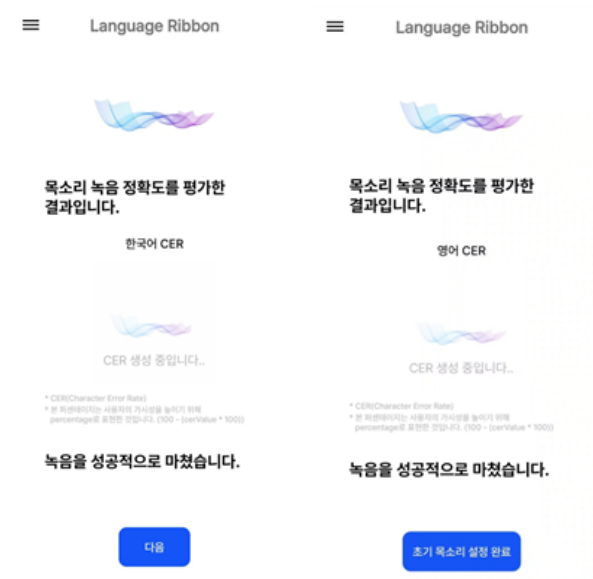
100 %

* CER(Character Error Rate)

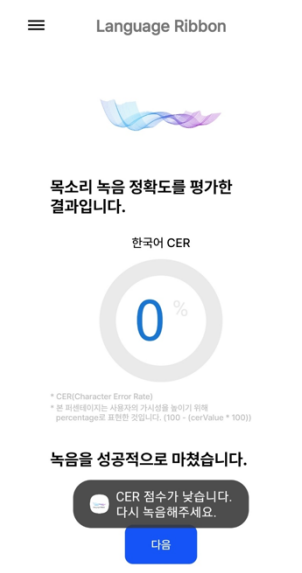
녹음을 성공적으로 마쳤습니다.

다음

- CER 생성 로딩



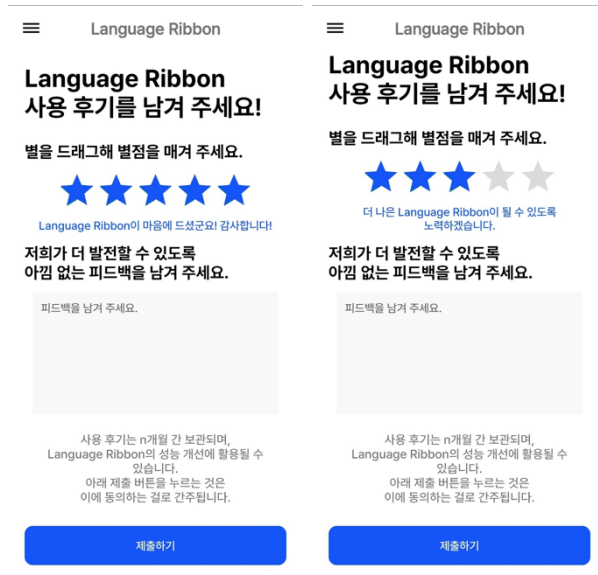
- CER 점수 낮을 시 toast message



3.1.6. 평가하기 페이지

평가하기 페이지는 사용자가 어플리케이션을 이용해보고 평가하는 값을 받는 페이지이다. 별점과 텍스트로 피드백을 입력받고 별점은 총 5개의 별로 0.5단위로 입력 받는다. 제출하기 버튼 클릭시 제출된다.

- 평가하기 이미지



3.1.7. 사용 가이드 페이지

사용 가이드를 확인할 수 있다.

- 사용가이드 이미지



3.1.8. 버전정보 페이지

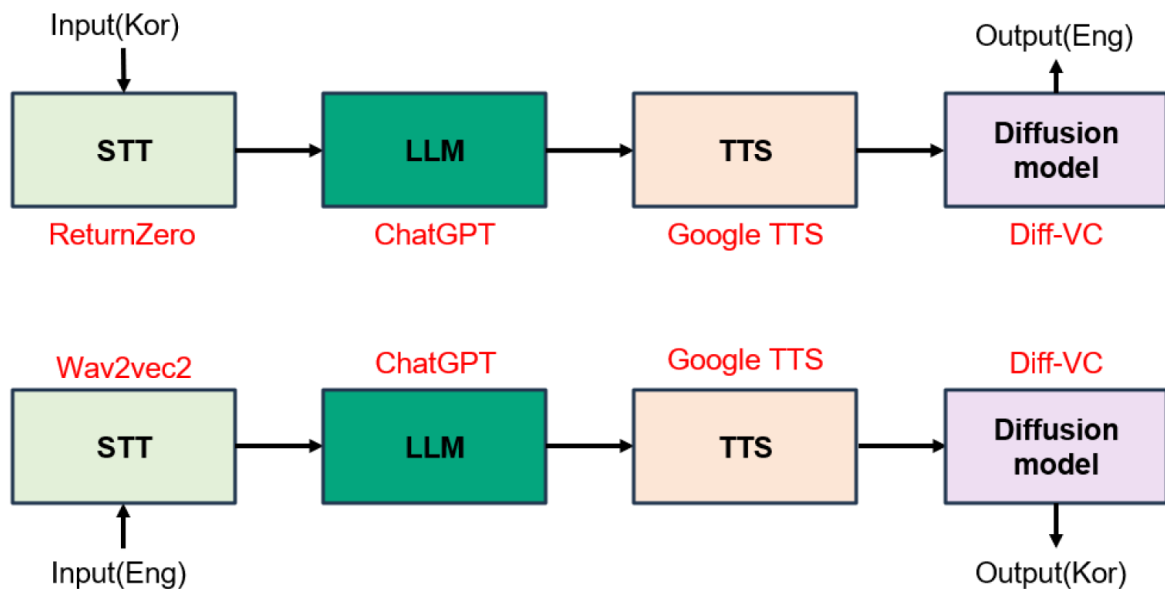
버전 정보를 확인할 수 있다.

- 버전정보 이미지

≡ Language Ribbon



3.2. 알고리즘



3.2.1. Speech To Text, STT

첫 번째 단계는 음성을 텍스트로 변환하는 과정이다. 각 입력 음성의 언어에 따라 다른 STT 모델을 선택한다. 한국어 음성에는 국내 STT 스타트업인 리턴제로사의 모델을, 영어 음성에는 PyTorch에서 제공하는 미리 학습된 Wav2Vec 2.0을 활용한다. Wav2Vec 2.0은 영어 음성에 탁월한 인식률을 보이지만, 한국어 음성에 대해서는 낮은 인식률과 알파벳만을 출력하는 한계가 있다.

따라서 한국어 음성은 높은 인식률과 한글 출력을 위해 한국어 특화 STT 모델인 리턴제로사의 모델을 활용한다. STT 모델은 음성 입력을 받아 해당 언어의 텍스트를 생성하며, 이 텍스트는 다음 모델로 전달된다.

한국어 음성 STT 모델로 koSpeech도 고려해보았다. koSpeech 모델은 입력 음성 ‘밥 먹었어?’를 ‘게 그 요밥 먹었어?할’로 출력하였다. 이는 koSpeech 모델이 음성의

녹음 환경, Background Noise, 화자의 정보를 포함한 모든 측면을 분석하는 특성이 있기 때문이다. 오디오를 있는 그대로 출력하지 않고 reconstruct 과정을 거치는 koSpeech모델은 오디오의 텍스트만 있는 그대로 출력하는 데에 있어 낮은 성능을 냈기 때문에 해당 모델을 사용하지 않고 리턴제로사의 모델을 사용하였다.

3.2.2. Large Language Models, LLM

두 번째 단계에서는 이전 단계에서 생성된 텍스트를 번역하는 과정이다. 가장 유명한 LLM 어플리케이션 중 하나인 ChatGPT 의 GPT-4를 활용한다. 선행연구에서 보았듯이 GPT-4는 이전 맥락을 고려하여 대화가 자연스럽게 이어지도록 한다. 또한 GPT-3.5 보다 높은 정확도와 빠른 속도를 제공한다. LLM 은 이전 모델의 출력 텍스트를 입력으로 받아 해당 입력을 번역하고 새로운 텍스트를 생성하여 다음 모델에 전달한다.

3.2.3. Text To Speech, TTS

세 번째 단계는 텍스트를 음성으로 변환하는 과정이다. 대표적인 TTS 모델인 FastSpeech 2.0 은 Tacotron2에 비해 빠른 추론 속도를 보여준다. 그러나 FastSpeech 2.0 은 한글 처리가 어렵다는 한계가 있다. 대표적인 문제점 두가지로는 한국어 띄어쓰기를 제대로 인식하지 못한다는 점과 숫자를 제대로 읽지 못하는 것이다. 예를 들어 ‘1번출구’와 ‘1번만 용서해줘’라는 텍스트가 있다면 이 두 텍스트의 숫자 1을 모두 ‘한’이라고 읽는 한계가 있었다. 따라서 본 단계에서는 한국어와 영어 음성 생성이 모두 가능한 구글사의 모델을 활용한다. TTS 모델은 이전 모델에서 출력된 텍스트를 입력으로 받아 해당 입력에 대한 음성을 생성하며, 이 음성을 다음 모델로 전달한다.

3.2.4. Voice Conversion, VC

마지막 단계는 VC 모델을 통한 음성 변환이다. 기존의 VC 모델들은 많은 초기 음성을 요구하고 훈련해야한다. 하지만 실제 발화 환경에서 사용 가능하려면 one-shot 형태의 빠른 추론 속도를 가진 모델이어야만 한다. 따라서 이 조건을 만족하는 오픈소스 모델인 Diff-VC 를 VC 모델로 채택하였다. Diff-VC 는 모델의 인코더(Encoder)와 강력한 확산 기반 디코더(Decoder)는 도메인이 다른 이전에 본 적 없는 화자에 대해서도 화자 유사성과 음성 자연스러움 면에서 우수한 결과를 달성하였다. VC 모델은 이전 단계에서 생성된 음성과 어플리케이션 초기설정에서 얻은 음성을 입력으로 받아, 이전 음성의 내용을 유지한 채 어플리케이션 초기설정에서 받은 음성의 특징을 복원하여 새로운 음성을 출력한다.

3.3. 백엔드

3.3.1. 서버 구축과정

백엔드 서버 구축: 크게 백엔드 회원가입, 로그인 등의 주요 백엔드 기능을 담당하는 AWS EC2 서비스를 이용해 t2.micro 유형 인스턴스를 할당받아 SSH 키를 등록하여 로그인 후, docker, docker-compose를 설치하여 docker 기반의 컨테이너를 구축할 수 있는 환경을 구성한다. 이후 해당 인스턴스에서 git clone을 통해 백엔드 레포지토리 소스 코드를 복제한 뒤, docker-compose 명령어를 이용해 백엔드 서버 운영 환경을 구축한다. 백엔드 구성은 python 이미지에 django를 설치하여 구성하며, 컨테이너 생성과정에서 원활한 django 서버 운영과 mp3를 wav로

변환하는 등의 서비스 내부 로직 수행에 필요한 각종 패키지를 설치하도록 Dockerfile 스크립트를 구성한다.

Diff-VC 서버 구축: Diff-VC 서버는 voice conversion 모델인 Diff-VC 모델을 open API 형태로 서빙하는 서버로, Diff-VC 모델 추론을 위해 고성능 GPU 자원이 지원되는 인스턴스가 필요하다. 따라서 AWS에 GPU 지원 인스턴스 서버를 사용할 수 있는 권한을 요청하여, 고성능 GPU를 지원하는 p4 시리즈 인스턴스 사용 권한을 획득한다. 이후 nvidia사의 머신러닝용 고성능 GPU인 v100을 지원하는 p4.xlarge 유형 인스턴스를 EC2 서비스를 통해 할당받고, SSH키 등록후 로그인하여 Diff-VC 레포지토리를 해당 서버로 클론한다. 그 다음 해당 서버에 nvidia GPU 드라이버를 설치한 다음, 컨테이너가 인스턴스의 GPU 자원을 활용할 수 있도록 하는 nvidia-docker를 설치한다. 이후 미리 작성한 docker-compose 스크립트를 docker-compose 명령을 통해 실행시켜 자동적으로 서버를 구축할 수 있도록 한다.

3.3.2. 회원 정보 관리

사용자의 어플리케이션 사용을 위해 사용자의 계정을 관리하고, 사용자의 정보를 수집하는 회원 관리 기능이 필요하다. 회원 정보 관리 기능은 회원 가입, 로그인, 로그아웃 기능으로 나뉜다.

1. 회원가입 기능은 signup 함수를 통해 구현한다. 만약 사용자가 이미 로그인한 상태에서 signup 함수를 호출하면, 메인 페이지로 리다이렉트한다. POST 방식으로 요청이 오면, Django의 SignupForm을 사용하여 회원가입 폼을 처리한다. 폼의 유효성 검사를 통과하면, 사용자 정보를 저장하고, 사용자 프로필을 생성한다. 그 후, 사용자의 ID와 함께 회원가입이 성공적으로 이루어졌음을 나타내는 메시지를 JSON 형태로 응답한다. 만약 폼의 유효성 검사를 통과하지 못하면, 에러 메시지를 반환한다.

사용자의 프로필 정보는 UserProfile 모델을 통해 관리된다. 이 모델은 Django의 User 모델과 일대일 관계를 가지며, 사용자의 이름, 성별, 나이, 직업, 영어 능력 수준, 그리고 사용자의 초기 목소리 정보를 포함한다. 초기 목소리 정보는 후술할 초기 목소리 수집 기능에서 설명한다. 이 모델을 통해 사용자의 프로필 정보를 효과적으로 저장하고 관리할 수 있다.

2. 로그인 기능은 login 함수를 통해 구현한다. 사용자가 이미 로그인한 상태에서 이 함수를 호출하면, 메인 페이지로 리다이렉트한다. POST 방식으로 요청이 오면, Django의 AuthenticationForm을 사용하여 로그인 폼을 처리한다. 폼의 유효성 검사를 통과하면, 사용자의 프로필 정보를 조회하고, 이를 JSON 형태로 응답한다. 만약 폼의 유효성 검사를 통과하지 못하면, 에러 메시지와 함께 400 상태 코드를 반환한다.
3. 로그아웃 기능은 logout 함수를 통해 구현한다. 사용자가 로그인한 상태에서 이 함수를 호출하면, 사용자를 로그아웃시키고, 로그아웃이 성공적으로 이루어졌음을 나타내는 메시지를 JSON 형태로 응답한다. 그 후, 로그인 페이지로 리다이렉트한다.

이렇게 구현된 회원 정보 관리 기능을 통해, 사용자는 자신의 회원 정보와 을 안전하게 관리할 수 있으며, Language Ribbon 어플리케이션의 음성 번역 기능을 원활하게 이용할 수 있다.

3.3.3. 초기 목소리 수집

Language Ribbon 어플리케이션에서 음성 통역 및 변조 기능을 제공하기 위해, 사용자의 초기 목소리를 수집하고 처리하는 기능이 필요하다. 이를 위해 uploadvoice 함수로 초기 목소리 수집 기능을 구현한다.

uploadvoice 함수는 사용자로부터 한국어, 영어 음성 데이터를 받아 처리하는 기능을 담당한다. 요청 방식이 POST가 아니거나, 요청에서 'audio' 파일이 누락되었거나, 'lang' 값이 'kr'(한국어) 또는 'en'(영어)이 아닌 경우에는 잘못된 요청임을 알리는 메시지를 JSON 형태로 응답한다. 'lang' 값이 'kr'인 경우, 사용자가 한국어 초기 음성을 수집하는 단계다. 사용자가 한국어 스크립트를 읽은 한국어 음성이 넘어오면, 해당 음성을 텍스트로 변환한 텍스트 결과물과 원래의 스크립트를 비교해 CER(Character Error Rate)를 계산한다. 계산된 CER이 0.3 이하면 잘 녹음된 초기 목소리라고 판단한 후 DB에 저장한다.

'lang' 값이 'en'일 경우, 사용자가 영어 초기 음성을 수집하는 단계다. 한국어 초기 목소리 수집 단계와 마찬가지로 CER을 체크한 후 DB에 저장된다.

3.3.4. 음성 통역 및 변조

사용자의 음성을 다른 언어로 통역하고, 사용자의 목소리 특징을 바탕으로 변조하는 기능이다. 음성을 입력 받아 텍스트로 변환한 후, 그 텍스트를 다른 언어로 번역하고, 번역된 텍스트를 다시 음성으로 변환한 뒤 디퓨전 모델의 서버를 거쳐 사용자의 목소리로 음성을 변조시킨다. 현재 한국어에서 영어, 영어에서 한국어 통역이 가능하다.

음성 통역 및 변조 기능은 POST 방식의 요청을 처리한다. 사용자로부터 'lang'과 'target-lang' 파라미터를 받아오는데, 이 파라미터는 각각 원본 언어와 대상 언어를 나타낸다.

1. lang이 한국어, target-lang이 영어인 경우

사용자가 말하는 음성이 한국어이고, 통역을 거쳐 출력하고자 하는 음성이 영어인 경우다. 사용자가 말하는 한국어 음성은 리턴제로 API를 거쳐 텍스트로 변환된다. 그 결과물은 LLM인 ChatGPT를 거쳐 상황과 맥락에 맞게 번역되며, 번역된 텍스트는 Google TTS API를 거쳐 출력된다.

이후, diffusion voice conversion을 위해, 백엔드에서 S3로부터 사용자 초기음성과 Google TTS API 거쳐 출력된 번역 음성을 Diff-VC 추론서버로 전송한다. 이때, Diff-VC 추론서버는 입력음성을 WAV 형식으로 입력받으므로, Google TTS API가 반환한 MP3 형식의 음성파일을 pydub 모듈을 이용해 WAV형식으로 변환한다. Diff-VC 서버는 입력음성에 해당하는 'file1'과 목표음성에 해당하는 'file2'를 POST 방식의 form 형태로 입력받는다. Diff-VC 서버로 file1 파라미터에 번역된 음성을 넣고, file2 파라미터에 S3로부터 가져온 사용자 초기음성을 넣어 Diff-VC 서버로 전송하게 되면, 서버 내부에서 1차적으로 입력음성과 목표음성의 잡음을 제거하고, 이후 훈련된 가중치 파일(.pt)를 이용해 사전에 정의한 추론 함수를 이용해 GPU상에서 추론(Inference)을 진행하여 입력음성에 대해 목표음성의 스타일을 반영한 음성파일을 생성한다. 이후 생성된 음성을 streaming 방식으로 응답하여 백엔드서버에 최종 결과 음성파일을 전달한다. 최종적으로 백엔드 서버는 응답받은 파일을 처음 요청한 사용자에게 자신의 목소리로 변조된 음성파일을 전달하게

되는데, 이때 텍스트 번역 결과도 함께 제공하기 위해 응답 헤더에 “X-Json-Response” 헤더를 추가하고, 헤더의 값으로 텍스트 번역 결과와 응답메세지를 담은 JSON 데이터를 담아 사용자에게 함께 전송한다. 이때 ‘X-Json-Response’헤더의 값은 직렬화 과정에서 base64 인코딩을 거쳐 반환된다. 결과적으로 사용자는 사용자의 목소리로 번역된 음성과 ‘X-Json-Response’헤더에 해당 번역 텍스트를 응답받게 되고, Language Ribbon 어플리케이션이 ‘X-Json-Response’의 디코딩된 JSON문자열과 최종 번역 음성을 어플리케이션 상에 적절하게 표시한다.

2. lang이 영어, target-lang이 한국어인 경우

사용자가 말하는 음성이 영어고, 통역을 거쳐 출력하고자 하는 음성이 한국어인 경우다. 사용자가 말하는 영어 음성은 HuggingFace의 Wav2Vec2 API를 거쳐 텍스트로 변환된다. 그 결과물은 LLM인 ChatGPT를 거쳐 1) 경우와 동일하게 상황과 맥락에 맞게 번역되며, 번역된 텍스트는 Google TTS API를 거쳐 출력된다.

lang이 영어, target-lang이 한국어인 경우인 경우도 lang이 한국어, target-lang이 영어인 경우와 동일한 방식으로 Diff-VC 서버를 거쳐 사용자의 음성 스타일로 변형된 번역음성과 번역 텍스트를 반환한다. Language Ribbon 어플리케이션은 번역 음성과 텍스트를 사용자의 어플리케이션 상에 적절하게 표시한다.

4. 버그 히스토리

4.1. 프론트엔드

기존 코드	<pre>binding.agreeAll.setOnCheckedChangeListener { _, isChecked -> binding.agree1.isChecked = isChecked binding.agree2.isChecked = isChecked } binding.agree1.setOnCheckedChangeListener { _, isChecked -> if (!isChecked) { binding.agreeAll.isChecked = false } else if (isChecked && binding.agree2.isChecked) { binding.agreeAll.isChecked = true } } binding.agree2.setOnCheckedChangeListener { _, isChecked -> if (!isChecked) { binding.agreeAll.isChecked = false } else if (isChecked && binding.agree1.isChecked) { binding.agreeAll.isChecked = true } } if(!binding.agreeAll.isChecked){ isAgree = false } else { isAgree = true }</pre>
오류 결과	'agreeAll' 체크박스가 선택되면 모든 체크박스가 선택되지만, 'agree1'과 'agree2' 중 하나만 선택되었을 때는 'agreeAll' 체크박스가 선택되지 않으며 일관성이 없는 동작이 발생한다.
해결	'agreeAll'의 상태 변경이 'agree1'과 'agree2'의 상태 변경을 일으키지 않도록 하기 위해 리스너를 잠시 해제하는 추가적인 로직을 포함하여 코드가 약간 복잡해지지만, 이벤트 핸들링이 더욱 세밀하게 제어한다.
변경 코드	<pre>val agreeAll = findViewById<CheckBox>(R.id.agreeAll) val agree1 = findViewById<CheckBox>(R.id.agree1) val agree2 = findViewById<CheckBox>(R.id.agree2) agreeAll.setOnCheckedChangeListener { buttonView, isChecked - > if (buttonView.isPressed) {</pre>

	<pre> agree1.isChecked = isChecked agree2.isChecked = isChecked } } val individualCheckListener = CompoundButton.OnCheckedChangeListener { _, _ -> agreeAll.setOnCheckedChangeListener(null) agreeAll.isChecked = agree1.isChecked && agree2.isChecked agreeAll.setOnCheckedChangeListener { buttonView, isChecked -> if (buttonView.isPressed) { agree1.isChecked = isChecked agree2.isChecked = isChecked } } } agree1.setOnCheckedChangeListener(individualCheckListener) agree2.setOnCheckedChangeListener(individualCheckListener) </pre>
--	--

2.2. 알고리즘

기존 코드	<pre> def kor_preprocess(text): g2p=G2p() phone = g2p(text) phone = h2j(phone) phone = list(filter(lambda p: p != ' ', phone)) phone = '{' + '}'.join(phone) + '}' phone = re.sub(r'W{[^WwWs]?W}', '{sil}', phone) phone = phone.replace('{}', ' ') sequence = np.array(text_to_sequence(phone, hp.text_cleaners)) sequence = np.stack([sequence]) return torch.from_numpy(sequence).long().to(device) </pre>
오류 결과	한국어 TTS 과정에서 띄어쓰기 및 문장 부호에 {sli} 이라는 의문의 소리가 들어가 어색하게 느껴진다. 또한 띄어쓰기를 인식하지 못한다.
해결	해당 소리를 공백으로 처리하고 띄어쓰기별로 끊어준다.
변경 코드	<pre> def kor_preprocess(text): g2p=G2p() phone = g2p(text) phone = h2j(phone) phone = re.sub(r'{sli}', ' ', phone) phone = re.sub(r'[^a-zA-Z0-9가-힣Ws]', ' ', phone) phone = ' '.join(phone.split()) sequence = np.array(text_to_sequence(phone, hp.text_cleaners)) </pre>

	<pre>sequence = np.stack([sequence]) return torch.from_numpy(sequence).long().to(device)</pre>
--	--

2.3. 백엔드

기존 코드	<pre>@csrf_exempt def get_transcription_status(jwt_token, transcribe_id): url = f"https://openapi.vito.ai/v1/transcribe/{transcribe_id}" headers = { 'accept': 'application/json', 'Authorization': f'Bearer {jwt_token}' } response = requests.get(url, headers=headers) response_data = json.loads(response.content.decode('utf-8')) return response_data</pre>
오류 결과	<p>해당 부분은 한국어 STT 과정에서 transcribe_id로 전사 결과를 조회하는 과정이다. 하지만 조회했을 때 전사 작업이 아직 진행 중인 경우('transcribing' 상태인 경우), STT 결과 문장이 아닌 transcribing 상태 코드가 반환된다.</p>
해결	<p>전사 작업이 완료될 때까지 반복해서 조회한다. 다만 서버에 과도한 요청을 보내는 것을 막기 위해 time.sleep(0.1) 코드를 삽입해 일정 시간 간격으로 요청을 보낸다.</p>
변경 코드	<pre>@csrf_exempt def get_transcription_status(jwt_token, transcribe_id): url = f"https://openapi.vito.ai/v1/transcribe/{transcribe_id}" headers = { 'accept': 'application/json', 'Authorization': f'Bearer {jwt_token}' } while True: response = requests.get(url, headers=headers) response_data = json.loads(response.content.decode('utf-8')) if response_data.get('status') != 'transcribing': break time.sleep(0.1) return response_data</pre>
기존 코드	<pre>def get_response_based_on_cer(request, lang_type, file_path, cer): if cer <= 0.3: try: user_id = request.POST.get('user.id')</pre>

	<pre> upload_path = f"/voices/{user_id}_{lang_type}.wav" print(upload_path) s3.upload_file(file_path, bucket_name, upload_path) user_profile = UserProfile.objects.get(user_id=user_id) if lang_type == "en": user_profile.voice_info_en = upload_path elif lang_type == "kr": user_profile.voice_info_kr = upload_path return JsonResponse({"uploadSuccess": True, "confirm": True, "message": "초기 목소리 데이터 수집에 성공했습니다.", "metric": {"cer": cer}}) except Exception as e: print(e) return JsonResponse({"uploadSuccess": False, "confirm": False, "message": "초기 목소리 데이터 저장에 실패했습니다. 다시 시도해주세요.", "metric": {"cer": cer}}) </pre>
오류 결과	S3 스토리지 서버에 사용자의 초기 녹음 음성파일을 업로드하지 못한다.
해결	S3 스토리지에 서버를 저장 완료하는 코드 추가한다.
변경 코드	<pre> def get_response_based_on_cer(request, lang_type, file_path, cer): if cer <= 0.3: try: user_id = request.POST.get('user.id') upload_path = f"/voices/{user_id}_{lang_type}.wav" print(upload_path) s3.upload_file(file_path, bucket_name, upload_path) user_profile = UserProfile.objects.get(user_id=user_id) if lang_type == "en": user_profile.voice_info_en = upload_path elif lang_type == "kr": user_profile.voice_info_kr = upload_path user_profile.save() return JsonResponse({"uploadSuccess": True, "confirm": True, "message": "초기 목소리 데이터 수집에 성공했습니다.", "metric": {"cer": cer}}) except Exception as e: print(e) return JsonResponse(</pre>

	<pre> {"uploadSuccess": False, "confirm": False, "message": "초기 목소리 데이터 저장에 실패했습니다. 다시 시도해주세요.", "metric": {"cer": cer}} </pre>
--	---

5.검증 시나리오 및 결과

5.1. MOS (Mean Opinion Score) 테스트

TTS 모델인 fastSpeech2에서 MOS를 진행할 때 20명을 대상으로 진행하고, tocotron2에서는 15명을 대상으로 진행하므로 MOS 검증 평가자의 수를 15-20명으로 선정한다. MOS를 통해 음성 모델이 출력한 결과물의 품질을 평가하고자 한다. 평가 요소는 다음과 같다.

- Naturalness: 음성 모델이 생성한 음성이 자연스럽게 들리는지, 인간의 목소리와 얼마나 유사한지를 평가하는 요소이다. 예를 들어 발음, 억양, 강세 등이 자연스러운지 평가하는 요소이다.
- Similarity: 음성 모델이 생성한 음성이 특정 인물의 목소리와 얼마나 유사한지 평가하는 요소이다. 음성 모델이 생성한 음성이 해당 인물의 목소리와 얼마나 비슷하게 느껴지는지 평가하는 요소이다.

Google Form과 Notion을 이용하여 설문을 진행한다. Notion을 왼쪽에 배치해 화자의 음성, Ground-truth (1번 음성), 번역 제외 음성(2번 음성), 본 어플리케이션에서 생성한 음성(3번 음성)을 오디오로 재생할 수 있게 만든다. 오른쪽에는 Google Form을 배치한다.

MOS 설문조사

- 시작 전 드리는 말씀
 - Notion 앱이 브라우저보다 음성의 로딩속도가 빠르므로 Notion 앱으로 진행하는 것을 권장합니다.
 - Google 설문지는 Google 로그인 후에 진행하는 것을 권장합니다.

음성 모델 퀄리티 평가 A1

화자의 음성

▶

—

🔊

⋮

1번 음성

▶

—

🔊

⋮

2번 음성

▶

—

🔊

⋮

3번 음성

▶

—

🔊

⋮

▶ 음성 모델 퀄리티 평가 A2

▶ 음성 모델 퀄리티 평가 A3

자연스러운 통역을 위한 음성 모델 평가

안녕하세요. 명지대학교 캡스톤디자인 2조 수강생(김소현, 김민희, 이동혁, 채기웅, 최지현)입니다.

저희는 사용자의 목소리를 그대로 보존해 자연스러운 통역을 제공하기 위한 어플리케이션을 개발하고 있습니다.

본 설문지의 목적은 저희가 개발한 모델의 성능을 평가하기 위함이며, 설문 대상은 영어영문학과 학생 15명, 영어에 능통한 교수님 5명, 일반 학생 10명을 목표로 설정했습니다.

설문지 작성 시간은 10분 내외며, 귀하의 소중한 응답은 저희가 개발한 모델 평가에 큰 도움이 될 것입니다.

설문지 응답 내용은 평가가 종료된 이후 개인정보는 파기됩니다.

[조사 기간: 2023.12.07. ~ 2023. 12. 10.]

mink141416@gmail.com [계정 전환](#)

📧 비공개

* 표시는 필수 질문임

25

- 섹션 1: 설문조사의 목적과 대상을 밝히고 조사자가 속한 그룹을 선택한다.

자연스러운 통역을 위한 음성 모델 평가

×

⋮

안녕하세요. 명지대학교 캠퍼스디자인 2조 수강생(김소현, 김민희, 이동혁, 채기웅, 최지현)입니다.

저희는 사용자의 목소리를 그대로 보존해 자연스러운 통역을 제공하기 위한 어플리케이션을 개발하고 있습니다.

본 설문지의 목적은 저희가 개발한 모델의 성능을 평가하기 위함이며, 설문 대상은 영어영문학과 학생 15명, 영어에 능통한 교수님 5명, 일반 학생 10명을 목표로 설정했습니다.

설문지 작성 시간은 10분 내외며, 귀하의 소중한 응답은 저희가 개발한 모델 평가에 큰 도움이 될 것입니다.

설문지 응답 내용은 평가가 종료된 이후 개인정보는 파기됩니다.

[조사 기간: 2023.12.07. ~ 2023. 12. 10.]

귀하가 속한 그룹을 선택해 주세요. *

☐ 교수
 ☐ 영어영문학과 및 영미권 거주 경험이 있는 학생
 ☐ 위 두 항목을 제외한 모든 학생

- 섹션 2: 조사자의 신변과 속한 그룹이 명확한 대상에게만 설문지를 전달하였음에도 불구하고 익명은 결과에 신뢰를 얻기 어려울 것이라 판단해 설문자의 개인정보를 입력받는다.

21 중 2 섹션

개인정보 입력

×

⋮

설문자의 개인정보를 입력해주세요.

성함 *

단답형 텍스트

학번 *

단답형 텍스트

소속학과 *

단답형 텍스트

- 섹션 3: 음성 모델이 출력한 결과물의 품질을 평가하는 요소를 설명해 조사자가 설문조사에 원활하게 참여할 수 있게 한다.

음성 모델이 출력한 결과물의 품질 평가

본 섹션은 음성 모델이 출력한 결과물의 품질을 평가하기 위해 구성되었습니다.
총 여섯 차례의 평가가 진행되오니, 왼쪽 Notion 페이지에 있는 음성 번호를 잘 확인하신 후 설문에 참여해 주세요.
평가 요소는 다음과 같습니다.

1. Naturalness
2. Similarity

- **Naturalness**는 음성 모델이 생성한 음성이 자연스럽게 들리는지, 인간의 목소리와 얼마나 유사한지를 평가하는 요소입니다. 예를 들어 발음, 억양, 강세 등이 자연스러운지 평가해 주세요.
- **Similarity**는 음성 모델이 생성한 음성이 특정 인물의 목소리와 얼마나 유사한지 평가하는 요소입니다. 음성 모델이 생성한 음성이 해당 인물의 목소리와 얼마나 비슷하게 느껴지는지 평가해 주세요.

- 섹션 4-12: 영어 음성을 입력 값으로 받았을 때 출력되는 한국어 음성에 대한 평가이다. 1번 문항은 Ground-Truth 음성의 Naturalness, 2 번 문항은 Ground-Truth 음성과 화자 음성에 대한 Similarity 비교이다. 3, 4번 문항은 번역 제외 음성, 5, 6번 문항은 본 어플리케이션에서 생성한 음성이다.

음성 모델 평가 A1

왼쪽 Notion 페이지에 있는 음성 번호 'A1'를 활용해 주세요.
(대본은 다를 수 있습니다.)

1번 음성에 대한 Naturalness(자연스러움) *

	1	2	3	4	5	
Bad	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Excellent

화자의 음성과 1번 음성과의 Similarity(유사성) *

	1	2	3	4	5	
Bad	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Excellent

2번 음성에 대한 Naturalness(자연스러움)

	1	2	3	4	5	
Bad	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Excellent

화자의 음성과 2번 음성과의 Similarity(유사성)

	1	2	3	4	5	
Bad	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Excellent

3번 음성에 대한 Naturalness(자연스러움)

	1	2	3	4	5	
Bad	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Excellent

화자의 음성과 3번 음성과의 Similarity(유사성)

	1	2	3	4	5	
Bad	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Excellent

- 섹션 13-21: 한국어 음성을 입력값으로 받았을 때 출력되는 한국어 음성에 대한 평가이다. 1번 문항은 Ground-Truth 음성의 Naturalness, 2번 문항은 Ground-Truth 음성과 화자 음성에 대한 Similarity 비교이다. 3, 4번 문항은 번역 제외 음성, 5, 6번 문항은 본 어플리케이션에서 생성한 음성이다. (항목은 섹션 4-12와 동일하다.)

평가자에게 주어진 MOS 의 범위는 1에서 5까지이며, 높은 점수일수록 음성 품질이 높다는 것을 의미한다. 평가자는 총 세가지 음성을 평가한다. 세가지 음성 중 Ground-truth 음성은 원본 화자의 음성을 의미하고, 번역 제외 음성은 언어 번역 파이프라인을 거치지 않은 원본 화자의 음성을 나타낸다. 마지막으로 본 어플리케이션의 출력 음성은 번역이 적용된 음성이다. 우리는 본 어플리케이션이 출력한 음성의 점수가 나머지 두 음성의 점수와 큰 차이가 없다면 성능이 좋은 것을 판단한다. 데이터 셋은 한국인 20대 남자 2명, 여자 1명의 한국어와 영어 발화 음성이다. 평가자는 총 18명으로 명지대학교에서 영어 교양을 가르치는 교수 4명, 영어영문학과 및 영미권 거주 경험이 있는 학생 6명과 타 전공 학생 7명으로 구성되어 있다.

	영어		한국어		전체	
	자연스러움	유사도	자연스러움	유사도	자연스러움	유사도
Ground truth	3.65	4.63	4.82	4.95	4.23	4.79
번역 제외	3.36	2.8	2.41	2.27	2.88	2.54
본 어플리케이션	3.38	2.82	2.11	2.1	2.75	2.46

표 1. MOS 결과.

번역 제외 음성과 본 어플리케이션 음성은 실제 화자의 음성(Ground-truth)과 유의미한 차이가 있음을 알 수 있다. 한국어 음성은 번역 제외 음성이 본 어플리케이션 음성보다 더 높은 점수를 획득하였다. 하지만, 영어 음성은 본 어플리케이션 음성이 번역 제외 음성보다 더 높은 점수를 획득하였다. 이를 통해 번역 과정이 유의미한 차이를 발생시키지 않음을 알 수 있다.

5.2. 사용성 테스트

MOS 테스트와 마찬가지로 Google Form을 이용하여 설문을 진행하였다. 조사자는 10명의 명지대학교 학생이다. 사용성 테스트 설문지는 총 10가지의 문항으로 구성되어 있으며, MOS 테스트와 마찬가지로 1점에서 5점으로 평가한다. 어플리케이션 apk 파일을 전달하여 조사자가 직접 안드로이드 모바일 환경에서 해당 어플리케이션을 사용한다. 회원가입부터 초기 목소리 수집, 통역 기능을 스스로 이용하며 각각 문항을 평가한다.

- 섹션 1: 설문조사의 목적과 대상을 밝히고 조사 대상이 해당 어플리케이션 사용 역력이 있어야 함으로 명시한다.

The screenshot shows the first section of a Google Form titled "Language Ribbon 어플리케이션 사용성 테스트". The form includes a header with the title and a close button. Below the header, there is a paragraph of text explaining the purpose of the survey and the target audience. The text states that the survey is for the "Language Ribbon" application, which aims to improve the user experience by providing a more natural and accurate translation service. It mentions that the survey is for users who have used the application and are interested in providing feedback. The survey is scheduled for December 16, 2023, to December 17, 2023. At the bottom of the section, there is a question: "Language Ribbon 어플리케이션을 사용해 보신 적 있으신가요?" (Have you ever used the Language Ribbon application?). There are two radio button options: "예" (Yes) and "아니오" (No).

- 섹션 2: 문항 1-10까지 시스템 사용성 척도 설문을 실시한다. 문항 내용이 다르고 형식은 동일하다.

The screenshot shows the second section of a Google Form titled "Language Ribbon 어플리케이션을 사용해 보신 분들을 대상으로 진행되는 문항입니다." (This section is for users who have used the Language Ribbon application). The form includes a header with the title and a close button. Below the header, there is a paragraph of text explaining the purpose of the survey and the target audience. The text states that the survey is for users who have used the application and are interested in providing feedback. The survey is scheduled for December 16, 2023, to December 17, 2023. At the bottom of the section, there is a question: "나는 이 어플리케이션을 자주 사용하고 싶을 것 같다." (I want to use this application frequently). There are five radio button options: "매우 그렇다" (Very much), "그렇다" (Much), "보통이다" (Average), "아니다" (Not), and "매우 아니다" (Not at all).

System Usability Scale (SUS)는 시스템 사용성을 평가하기 위해 사용되는 척도이다. SUS는 사용자들의 시스템에 대한 인식과 만족도를 측정하는데 사용되며, 사용자 경험과 시스템의 유용성에 대한 평가를 제공한다. SUS는 1980년대에 개발된 척도로, 간단하고 효과적인 사용성 평가 도구로 널리 사용되고 있다. SUS 설문은 10개의 질문 항목으로 구성되어 있으며, 이를 통해 사용자가 시스템에 대한 인식과 만족도를 평가할 수 있다. SUS는 각 질문 항목에 대한 5단계 리커트 척도(1부터 5까지의 점수)를 사용합니다. 사용자는 각 항목에 대해 자신의 경험에 따라 적절한 점수를 선택하게 됩니다. 선택된 점수는 총점으로 계산되어 최종적인 사용성 점수를 도출한다.

SUS는 상대적으로 간단하고 빠르게 사용자의 인식을 파악할 수 있는 장점이 있다. 또한, 다양한 종류의 시스템에 적용할 수 있으며, 사용성 평가 결과를 비교하고 개선할 수 있는 기준을 제공한다.

1. 이 어플리케이션을 자주 사용하고 싶을 것 같다.	4.7
2. 이 어플리케이션이 불필요하게 복잡하다고 생각했다.	1.5
3. 이 어플리케이션을 사용하려면 기술 지원자(안내 가이드, 도우미 등)의 도움이 필요할 것 같다.	1.8
4. 이 어플리케이션이 사용하기 쉽다고 생각했다.	4.7
5. 이 어플리케이션을 사용하는 것이 매우 불편하다고 느꼈다.	1.3
6. 이 어플리케이션을 시작하기 전에 많은 것을 배워야 했다.	1.3
7. 이 어플리케이션의 다양한 기능이 잘 통합되어 있다고 생각했다.	4.3
8. 이 어플리케이션이 불안정하다고 생각했다.	1.6
9. 대부분의 사람들이 이 어플리케이션을 매우 빠르게 배울 수 있을 것이라고 생각한다.	4.5
10. 이 어플리케이션을 사용하는 데 매우 자신감이 있다.	4.7

표 2. 사용성 검사 문항 및 결과

본 어플리케이션의 점수가 높을수록 긍정적으로 해석할 수 있는 문항으로는 1, 3, 7, 9, 10번이다. 특히 ‘자주 사용하고 싶다.’, ‘사용하기 쉽다.’, ‘사용하는데 자신감이 있다.’는 문항에서 최고 점수를 얻었다. 반대로 점수가 낮을수록 긍정적으로 해석할 수 있는 문항에서는 어플리케이션을 사용하기 불편하다는 문항과 어플리케이션을 시작하기 전에 많은 것을 배워야 한다는 문항이 최저 점수를 얻었다. 따라서 어플리케이션이 사용하기 편하고 많은 숙련도를 요구하지 않는다는 것을 알 수 있다.

6. 최종 결과 분석 및 한계

본 연구에서는 현실적이고 유용한 형태의 음성 번역 서비스를 제안하였다. 거대 언어 모델과 음성 변환 기술의 통합으로 사용자들에게 우수한 번역 경험을 제공하는 데 성공하였다. 음성과 텍스트 간의 자연스러운 전환과 화자 특징 복원을 통해 번역된 음성의 품질을 높일 수 있었다. 이 서비스는 일상 대화나 교육적인 목적으로 활용 가능할 것으로 기대된다.

그러나, 한국어와 영어의 언어적 구조의 차이로 인해 한국어 음성의 자연스러운 처리에 어려움이 있었다. 이를 극복하기 위해 한국어의 언어적 특성을 고려한 음성 변조 모델을 훈련시키는 것이 필요하며, 이를 통해 더 나은 성능을 기대할 수 있다.

참고 문헌

- 김유리. "챗 GPT 로 사라질 위기 직업군 1 위, 번역가, 통역사, 세무사, 회계사 4 위," Tax Times, Apr. 12, 2023. [Online]. Available: <http://www.taxtimes.co.kr/news/article.html?no=259163>
- 배문정. "통역 사용자를 대상으로 한 통역 모드 선호도 조사: 인간 통역, AI 통역, 자막비교," 번역학연구, vol. 24, no. 3, pp. 591-614, 2023.
- 박미정. "생성형 AI 와 기계번역 - 챗 GPT 번역을 통한 한일통역교육 고찰" 통번역학연구 27, no.3 (2023) : 27-56.doi: 10.22844/its.2023.27.3.27
- 민소연, 이광형, 이동선 and 류동엽. "한국어 특성 기반의 STT 엔진 정확도를 위한 정량적 평가방법 연구" 한국산학기술학회논문지 21, no.7 (2020) : 699-707.doi: 10.5762/KAIS.2020.21.7.699
- 권철홍. "한국어 TTS 시스템에서 딥러닝 기반 최첨단 보코더 기술 성능 비교" 문화기술의 융합 6, no.2 (2020) : 509-514.
- 임철홍. "LLM(Large Language Model) 속성과 성능 연관성 연구" 정보화연구 20, no.3 (2023) : 257-266.doi:10.22865/jita.2023.20.3.257
- V. Popov, I. Vovk, V. Gogoryan, T. Sadekova, M. Kudinov, and J. Wei, "Diffusion-Based Voice Conversion with Fast Maximum Likelihood Sampling Scheme," arXiv.org, Aug. 04, 2022. <https://arxiv.org/abs/2109.13821> (accessed Dec. 11, 2023).
- OpenAI, "GPT-4 Technical Report," arXiv:2303.08774 [cs], Mar. 2023, Available: <https://arxiv.org/abs/2303.08774>
- 권세영, 맹지연, 백예슬, and 김영국, "개인화된 TTS구현을 위한 음성합성 딥러닝 모델 비교," in Proceedings of KIIT Conference, 2021, pp. 759-762.