국민대학교
전자정보통신대학
컴퓨터공학부

캡스톤 디자인 I

종합설계 프로젝트

프로젝트 명	*아나운서 준비생을 위한 맞춤형 AI 스피치 연습 애플리케이션, Loro(로로)*
팀 명	*커비 (8조)*
문서 제목	결과보고서

Version	2.0
Date	2024-MAY-23

팀원	안지원 (20203095, 팀장)
	김필모(20191579)
	신민경(20203090)
	윤하은(20203110)

문서 정보 / 수정 내역

Filename	결과보고서-아나운서 준비생을 위한 맞춤형 AI 스피치 연습 애플리케이션, Loro(로로).docx
원안작성자	김필모, 신민경, 안지원, 윤하은
수정작업자	김필모, 신민경, 안지원, 윤하은

수정날짜	대표수정자	Revision	추가/수정 항목	내 용
2024-05-16	신민경	1.0	초안 작성	• 추진 배경 및 필요성 추가 • 프론트엔드 활용 기술 추가
2024-05-17	안지원	1.1	아키텍처 추가	• 아키텍처 이미지 및 설명 추가
2024-05-17	김필모	1.2	연구/개발 추가	• 사용자 맞춤 가이드 음성 파트 추가 • 정확도 피드백 파트 추가
2024-05-19	신민경	1.3	기능 요구사항 추가	• 회원가입/로그인, 홈, 설정, 기록, 스크립트 부분 추가
2024-05-20	김필모	1.4	비기능 요구사항 추가	• 제품에 대한 요구사항 추가 • 조직 요구사항 추가 • 외부 요구사항 추가

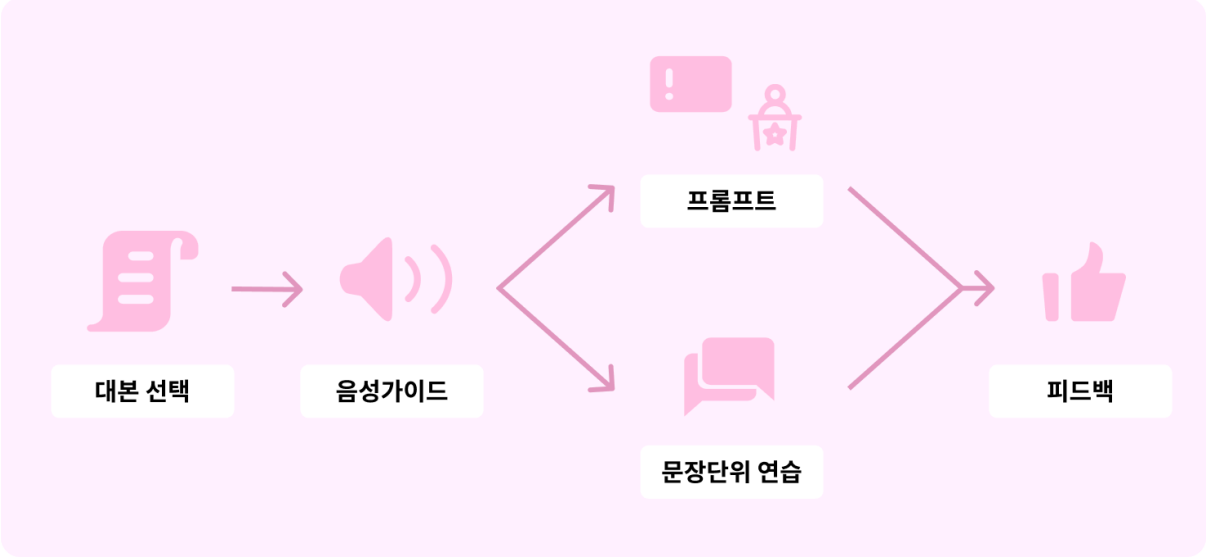
2024-05-21	윤하은	1.5	참고 문헌 추가	• TTS 관련 논문 추가 • STT 관련 논문 추가 • 의존성 라이브러리 기술문서 추가
2024-05-21	신민경	1.6	기능 요구사항 추가	• 검색, 문장단위연습, 프롬프트 부분 추가
2024-05-21	김필모	1.7	운영자 메뉴얼/ 배포 가이드 추가	• ubuntu 기준 배포 가이드 작성 • 모델 재빌드 관련 메뉴얼 추가 • 운영 시 생길 수 있는 이슈들과 해결 방안 작성
2024-05-22	안지원	1.8	현실적 제한요소/ 해결방안 추가	• 백엔드 현실적 제한요소 추가 • 비용 최적화 방법 추가
2024-05-22	신민경	1.9	테스트 케이스 / 활용 기술 추가	• 테스트 케이스 추가 • 대본 생성 관련 연구/개발 내용 추가 • 대본 생성 관련 활용 기술 추가
2024-05-23	윤하은	1.10	모델 추가 / 전체 내용 검토 및 수정	• 모델 연구/개발 내용 추가 • 자기평가 추가 • 전체 내용 검토 및 수정
2024-05-23	신민경	1.11	개요 / 사용자 매뉴얼 추가	• 개요 추가 • 사용자 매뉴얼 추가 • 전체 내용 검토 및 수정
2024-05-23	안지원	2.0	결과물 목록 추가	• 결과물 목록 추가

목 차

1 개요

1.1 프로젝트 개요

우리가 매일 뉴스에서, TV 프로그램에서 볼 수 있는 아나운서들은 어떤 과정을 거쳐 그 자리에 서는 걸까요? 저희는 아나운서들이 갖추어야 할 가장 중요한 능력인 '발성'과 '프롬프트' 연습을 돕기 위해 AI 기술을 활용한 스피치 트레이닝 어플리케이션을 개발했습니다.



1. 대본 생성
   사용자가 연습할 수 있는 다양한 예시 대본을 제공합니다. 또한 사용자가 직접 대본을 업로드하거나 실시간으로 원하는 대본을 생성할 수 있는 기능도 제공합니다.

2. '아나운서'라는 직업적 특성을 반영한 연습 방법
   문장단위 연습에서는 한 문장씩 꼼꼼히 발성을 연습할 수 있습니다. 충분한 연습 후 대본에 익숙해지면, 실제 방송 환경처럼 구성한 프롬프트 연습을 진행할 수 있습니다.

3. 맞춤형 음성 가이드 제공
   개인화 TTS 모델을 이용해 사용자의 목소리와 아나운서의 특성을 반영한 맞춤형 음성 가이드를 제공합니다. 이를 통해 사용자는 가이드 음성에 맞춰 발음, 억양 등을 개선할 수 있습니다. 또한 STT 모델을 활용해 사용자의 발음과 억양에 대한 실시간 피드백을 제공하여 학습 효과를 극대화할 수 있습니다.

1.2 추진 배경 및 필요성

1.2.1 문제 정의

비용적 측면

고가의 등록비 및 수업료: 대부분의 아나운서 학원은 높은 등록비와 수업료를 요구합니다. 이로 인해 경제적으로 부담이 있는 학생들은 학원 수강을 포기하거나, 부모님의 경제력에 따라 선택을 할 수밖에 없는 경우가 있습니다.

추가 비용 부담: 정규수업 외에도 특별 강의, 개인지도 등을 위한 추가 비용이 발생하는데 이로 인해 전체적인 비용 부담이 더 커지는 경우가 있습니다.

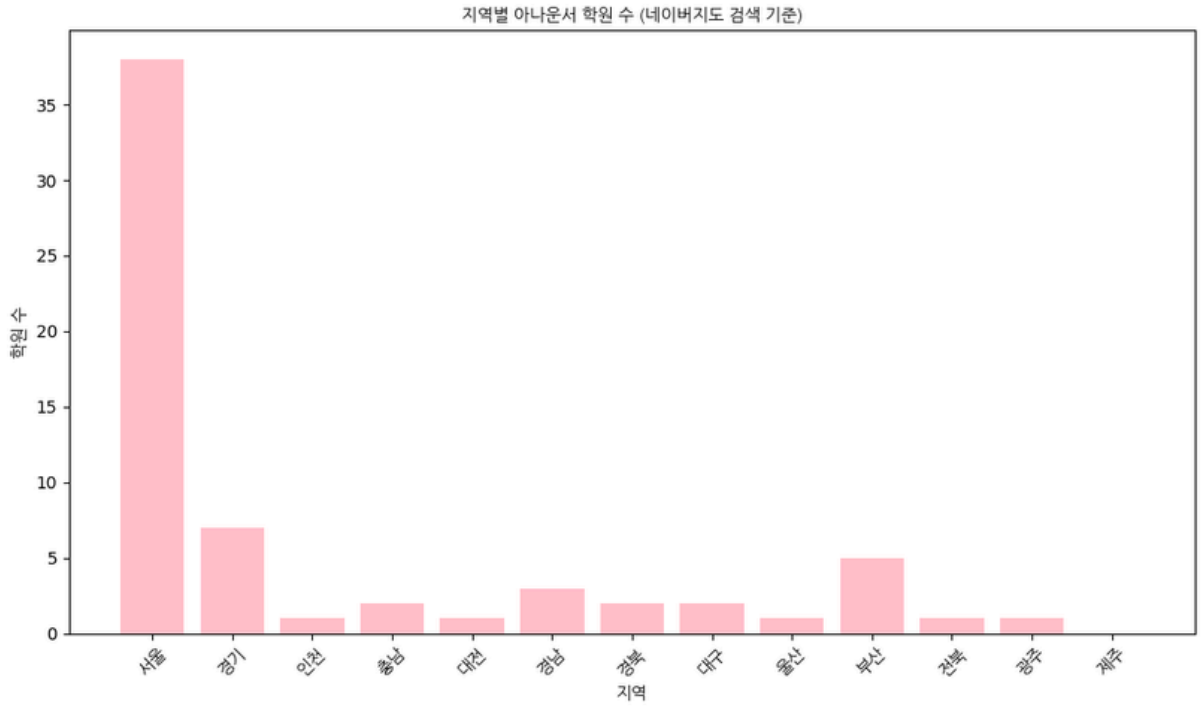
- 아나운서 학원 비용 관련 뉴스 기사 목록
  - 아나운서 되려면 학원 필수, 학원비 1000만원이라니
  - "아나운서 되고 싶어요"…1년 학원비만 1000만원 훌쩍 - 머니투데이

아카데미명	강의횟수	수강료	최대정원
투비앤아나운서아카데미학원	50	4,900,000	5
아이비스피치	50	6,000,000	3
봄온 아카데미	40	5,200,000	6
스포티비아나운서스피치아카데미	40	5,200,000	3
아나레슨	10	1,800,000	1
MBC 아카데미	10	1,350,000	1

직접 조사한 결과를 바탕으로 작성한 표(2024년도 3월 기준)

지리적 측면

대부분의 아나운서 학원은 대도시나 번화가에 위치해 있기 때문에, 지방이나 외곽 지역 거주자들에게는 접근성이 떨어집니다. 이로 인해 지방 거주자들은 학원까지 이동하는데 추가적인 교통비와 시간을 들여야 합니다.



직접 조사한 결과를 바탕으로 작성한 그래프(네이버지도 검색 기준)

1.2.2 유사 서비스 조사

다음은 저희 앱과 유사한 기존의 서비스들을 벤치마킹한 결과입니다.

서비스	설명
Teleprompter°	프롬프터 기능 • 미러링, 속도 변경 및 글꼴 변경 기능이 있는 사용하기 쉬운 텔레프롬프터로 대본, 가사, 연설문을 읽을 수 있습니다.
Parrot Teleprompter	프롬프터 • 사용자가 스크립트를 업로드할 수 있습니다. • 스톱워치, 글씨 크기, 속도 변경 등을 지원합니다.
스픽 (Speak)	AI 영어 스피킹 • 문장 단위로 음성 가이드를 제공합니다. • 사용자의 발음을 피드백해줍니다. • AI와 프리토킹할 수 있습니다.
말해보카	영어 학습 • 어휘 학습, 문법 학습, 회화 연습이 가능하며, 자세한 설명과 예시를 제공합니다. • 영어 듣기, 말하기로 발음과 억양 섬세하게 체크 가능합니다. • 최근 7일 내 배운 문장들을 반복해서 들을 수 있어요.
바름(Bareum)	한국어 발음 교정 • 본인의 발음을 들리는 그대로 한글로 표기해주어 틀린 발음을 눈으로 인지하고 교정할 수 있습니다. (ex. “좋은 아침이야” -> “조은 아치미야”) • 연습할 문장의 발음 기호를 보여줍니다. • 음소 단위로 분석한 발음 정확도를 실시간으로 확인할 수 있습니다.
한글발음연습에 진심	한글 발음 연습 • 잰말놀이와 같이 발음이 어려운 문장들을

	제공한다. • 사용자가 연습한 문장의 일치율을 알려주고, 몇 회 연습했는지 표시해준다.
세종학당 어휘학습 초급·중급	초·중급 한국어 학습자를 위한 한국어 공부 • 단어, 짧은 문장에 대해 발음을 연습할 수 있습니다. • 음성 가이드와 사용자의 발음을 비교해볼 수 있습니다. • 낱말 퍼즐과 같은 게임을 제공합니다.

2 개발 내용 및 결과물

2.1 목표

저희는 Loro를 통해 비용적, 지리적 부담 없이 언제, 어디서나 아나운서나 앵커의 발성을 연습하고 피드백 받을 수 있도록 하고자 합니다.

이때 사용자마다 사용자의 음성을 기반으로 한 아나운서 억양 개인화 TTS 모델을 만들어 이를 바탕으로 실제 학원에서 배우는 것과 유사하게 발성을 연습하고 피드백 받을 수 있습니다.

2.2 연구/개발 내용 및 결과물

2.2.1 연구/개발 내용

예시 대본 제공 및 사용자 대본 생성 지원

넷플릭스 드라마 '더 글로리'에는 '박연진'이라는 기상 캐스터가 대본을 직접 작성하는 부분이 나오는데요. 이처럼 비단 기상 캐스터 뿐아니라 아나운서 대부분은 직접 대본 원고를 작성하고 이를 바탕으로 방송을 합니다. 그렇다고 자유 형식으로 적는 것이 아니라 큰 틀이 정해져 있고 이를 바탕으로 작성합니다. 예를 들어 기상 캐스터가 원고를 쓴다고 가정을 해보면 오프닝 - 구름사진 - 일기도 - 개황 - 기온 - 해상 - 주간날씨 순서로 원고를 작성합니다.

Loro는 사용자에게 예시 대본을 제공합니다. 공공데이터포털에서 csv 형태로 제공하는 뉴스 기사 데이터셋을 활용했습니다. csv 파일로부터 제목, 카테고리, 원문 url 정보를 추출합니다. Selenium을 활용해 기사 원문을 가져온 후, OpenAI LLM으로 '아나운서의 뉴스 대본'에 맞게 내용을 가공하고 문체를 수정합니다. 이때 질 좋은 대본을 생성할 수 있도록 LangChain을 사용해 정교한 입력 프롬프트를 작성했습니다. 적절하게 가공된 제목, 카테고리, 내용은 파이어스토어에 저장됩니다. 생성하고 싶은 대본의 개수를 입력하고 구동시키면 위 과정을 거쳐 앱에서 곧바로 확인 가능하도록 파이프라인을 구축했습니다. 약 2만여개의 여분 데이터를 보유하고 있어 사용자에게 지속적으로 새로운 대본을 제공할 수 있습니다.

예시 대본 외에도 사용자가 직접 대본을 생성할 수 있습니다. Loro에서는 이를 '사용자 대본'이라고 칭합니다. 사용자가 직접 대본 원고를 작성할 때 도움을 주기 위해 제목과 카테고리를 입력하면 OpenAI LLM 모델을 통해 원하는 원고를 작성해줍니다.

프롬프터 화면 제공

아나운서 시험은 방송사별로 다르지만 공통적으로 1차 카메라, 2차 필기, 3차 직무실기, 4차 면접으로 진행됩니다. 이때, 카메라 테스트는 원고를 주고 검은색 프롬프터 화면보며 낭독을 하게 합니다. KBS는 뉴스 원고 하나를 낭독하게 하고 MBC는 뉴스, MC, 나레이션 등 다양한 원고를 시키는데 “프롬프터"를 보면서 스피치하는 것에 익숙하지 않은 준비생들은 이 1차 테스트에서 많이 떨어진다고 합니다. 눈동자의 움직임도 자연스러워야 하며 스크립트가 스크롤되는 것에 맞춰 속도와 리듬을 맞춰야 하기 때문입니다. 기존에 종이 대본을 보고 연습을 하거나 유튜브 영상으로 준비하던 연습생들을 위해 Loro는 저희가 방송사들의 프롬프터 환경에 대해 파악해 만든 실제 방송 환경에 맞춘 프롬프터 환경을 제공합니다.

사용자 맞춤 가이드 음성 제공

김광석이 부르는 ‘스물 다섯 스물 하나(원곡: 자우림)’, 임재범이 부르는 ‘Hype boy(원곡: 뉴진스)’같은 ai 커버를 보면 실제로 그 가수가 부르지는 않았지만 ai를 이용해 특정 가수의 음색, 억양을 입힌 노래를 들을 수 있습니다. 이러한 예시처럼 Loro는 여러 방송사의 남녀 아나운서 데이터로 학습된 모델이 사용자의 억양, 발음을 학습해 맞춤형 가이드 음성을 제공합니다.

이는 단순히 기존에 이미 쓰여진 대본을 가지고 아나운서, 앵커, 기자 목소리를 모방하는 것이 아닌 자신이 직접 만든 대본을 가지고 내 목소리로 전문 아나운서의 발음, 억양, 속도를 가지고 말했을 때의 음성을 듣고 이를 통해 피드백을 받을 수 있게 합니다.

정확도 피드백 제공

피드백은 현재 상황과 문제점에 대해 객관적으로 전달하고 무엇을 개선해야 할 지 알려주기 때문에 학습 시에 정말 중요합니다. Loro는 아나운서 준비생에게 정확하고 객관적인 피드백을 주기 위해서 음성과 텍스트 모두를 고려해 정확도 점수를 계산합니다. 사용자가 대본을 읽고 발음한 음성과 가이드 음성은 의미 있는 음성 특징인 mfcc로 변환되어 유사도를 계산하는 데 사용됩니다. 이때, 사용자가 발음한 음성의 길이와 가이드 음성의 길이가 같다는 보장을 할 수 없습니다. 따라서 이 두 음성 간의 시간 축 매핑이 필요합니다. 저희는 이 문제를 다차원데이터에 적용할 수 있는 DTW 알고리즘을 사용해 해결했습니다.

하지만 mfcc 만으로는 정확도 점수를 측정하기에는 부족하다고 판단했습니다. 아나운서 억양에 대한

피드백은 할 수 있지만 **발음**에 대한 정보는 정확히 제공하지 못하기 때문입니다. 그래서 음성을 텍스트화 하는 STT(speech to text) 기술을 도입해 사용자가 발음한 내용을 보여주는 동시에 이 정보를 활용해 텍스트 간의 유사도를 추가로 계산해 발음 정확도에 대한 내용도 점수 계산식에 포함하게 되었습니다.

일반적으로 한국어에서 텍스트 간의 유사도는 WER(Word Error Rate)이 아닌 CER(Character Error Rate)을 사용합니다. WER과 CER의 계산 식은 아래와 같이 동일합니다. 띄어쓰기로 구분되는 토큰들의 총개수에 대비되는 insertion, deletion, substitution의 수가 얼마나 많은 지를 계산하는 것입니다.

$$WER = \frac{S+D+I}{N} = \frac{S+D+I}{S+D+C}\ ,\ CER = \frac{S+D+I}{N} = \frac{S+D+I}{S+D+C}$$

토큰의 단위가 단어이냐 아니면 문자이냐의 차이입니다. 왜 한국어에서 CER을 사용하는 지 알기 위해 A: “나는 로로의 친구” 와 B: “나는 뽀로로친구" 라는 두 문장이 있다고 할 때, WER은 에러율 100%지만 CER에서는 문자 단위로 판단하므로 42%로 더 정확한 것을 알 수 있습니다.

저희 서비스는 단순 CER보다 더욱 정확한 점수를 제공하기 위해 여기에 텍스트를 국제 음성 기호인 IPA로 변환해 비교하는 로직을 추가했습니다. 이렇게 되면 발음에 관해서 더욱 정확 비교가 가능해집니다.

예를들어 “나는 집에 갔다"라는 음성이 있다고 가정하겠습니다. 이를 제대로 발음하면 “나는 지베 갇따"로 발음해야 합니다. 이는 ipa로 nˈɐɯntɕˈibe gˈɐdtɐ 입니다. 이때, 사용자가 ‘갔다’의 “ㅆ” 받침 발음을 실수 해 “가따”로 발음했다고 합시다. 이는 ipa로 nˈɐɯntɕˈibe gˈɐtɐ 이고 이 두 텍스트를 단순히 위와 같이 CER을 적용하면 비슷하게 발음을 했음에도 34%의 큰 에러율이 나오나 ipa로 CER을 측정하면 6% 의 에러율이 계산됩니다.

개인화 가이드 음성 제공

VITS2: Text to Speech

Stochastic Duration Predictor with Time Step-wise Conditional Discriminator

- Adversarial learning을 적용하여 generator와 동일한 입력을 제공하는 conditional discriminator로 지속 예측기를 훈련합니다. 가변 길이의 입력을 적절하게 판별하기 위해

모든 토큰의 예측된 지속 시간 각각을 개별적으로 판별하는 time step-wise로 판별하며, 손실 함수는 adversarial learning을 위한 최소 제곱 손실 함수 및 평균 제곱 오차 손실 함수를 사용합니다.

$$L_{adv}(D) = \mathbb{E}_{(d,z_d,h_{text})}\Big[(D(d,h_{text})-1)^2 + (D(G(z_d,h_{text}),h_{text}))^2\Big], \quad (1)$$

$$L_{adv}(G) = \mathbb{E}_{(z_d,h_{text})}\Big[(D(G(z_d,h_{text}))-1)^2\Big], \quad (2)$$

$$L_{mse} = MSE(G(z_d,h_{text}),d) \quad (3)$$

-

Monotonic Alignment Search with Gaussian Noise

- 가능한 모든 단조 정렬 중에서 가장 높은 확률을 갖는 텍스트와 오디오 간의 정렬을 산출하고, 모델은 그 확률을 최대화하도록 훈련합니다. 특정 정렬을 검색하고 최적화한 후, 더 적절한 다른 정렬을 찾기 위한 탐색에 제한이 있다는 문제점을 해결하기 위해 계산된 확률에 작은 가우시안 노이즈를 추가합니다.

$$P_{i,j} = \log \mathcal{N}(z_j;\mu_i,\sigma_i) \quad (4)$$

$$Q_{i,j} = \max_A \sum_{k=1}^{j} \log \mathcal{N}(z_k;\mu_{A(k)},\sigma_{A(k)}) = \max(Q_{i-1,j-1},Q_{i,j-1}) + P_{i,j} + \epsilon \quad (5)$$

-

Normalizing Flows using Transformer Block

- 장기 종속성을 캡처할 수 있도록 소규모의 트랜스포머 블록을 잔차 연결과 함께 normalizing flow에 추가합니다.

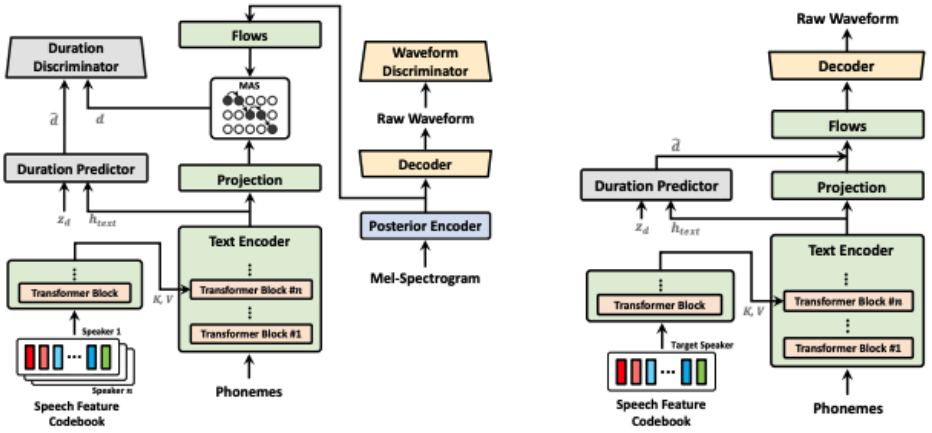


Figure 3: (a) Training procedure of TTS model. (b) Inference procedure of TTS model.

Speaker-Conditioned Text Encoder

- SFEN에서 만들어진 화자의 speech feature codebook을 텍스트 인코더의 세 번째 트랜스포머 블록에서 화자 벡터에 conditioning합니다.

SFEN: Speech Feature Encoding Network

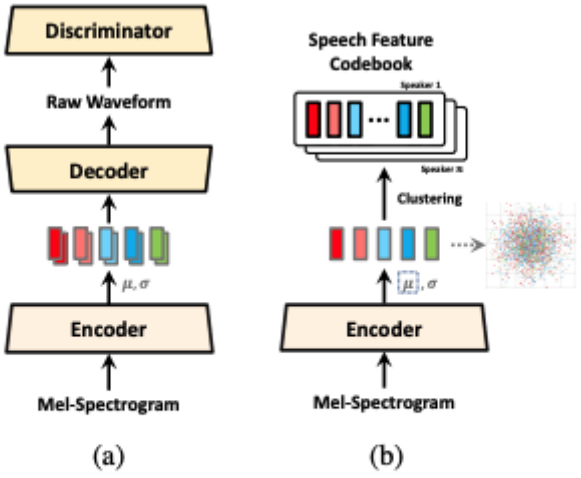


Figure 1: (a) Training procedure of SFEN.
(b) Inference procedure of SFEN.

Loro는 사용자의 음성 특성을 반영한 가이드 음성을 제공합니다. 따라서, Multi speaker 모델처럼 화자의 전반적인 특성을 세부적으로 표현하기 위해서 speech feature를 인코딩하고, 이를 음성 합성 모델인 VITS2에 conditioning합니다. 화자의 음성은 콘텐츠에 따라 음색과 운율이 매우 다르게 표현될 수 있기 때문에, 화자들의 음성에서 다양한 speaker feature을 밀집되고 연속적인 분포로 인코딩합니다. 그런 다음, 이러한 음성 특징들을 클러스터링하여 이산화된 대표 지점을 얻습니다.

Input mel spectogram으로부터 인코딩 및 디코딩을 통해 원시 파형을 재구성하도록 훈련된 오토인코더를 사용합니다.

- Target speaker의 오디오에서 생성된 mel spectogram을 인코더에 입력하고 출력으로 분포 매개변수인 μ와 σ를 얻습니다.
- 모든 Target speaker 오디오의 μ 값을 수집하고 k-means++를 사용하여 값들을 클러스터링합니다.
- 클러스터의 센트로이드를 speaker의 잠재 음성 특징 코드북으로 사용합니다.

이러한 특징은 유한 개수의 비연속 벡터입니다. 그러나 이러한 벡터들은 음성 합성 모델에 조건을 걸 때 query 및 key로서 VITS2의 text encoder의 intermediate feature와 Multi-head Attention을 통해 선형 결합되어 연속 공간에서 점을 샘플링하는 것과 유사한 효과를 낳습니다.

디코더로는 원시 파형을 재구성하는 데 우수한 성능을 보인 HiFi-GAN을 사용합니다. 또한 재구성 성능을 높이기 위해 적대적 학습 메커니즘과 판별자를 도입했습니다. Latent space의 조합을 통해 학습된 공간에서 점을 샘플링할 가능성을 높이기 위해 variational 접근법 GAN을 사용하여 unit Gaussian prior에 맞춥니다.

2.2.2 시스템 기능 요구사항

분류	요구사항명	상세 설명	상태	변경 사항
회원가입/ 로그인	구글 로그인	• 구글 로그인을 통해 사용자의 구글 계정을 등록한다. • 계정 미등록 시, 구글 로그인 API를 통해 회원가입을 진행한다. • 계정이 등록되어있으면서 사용자의 정보가 저장되어 있지 않을 시, 닉네임/캐릭터 설정 페이지로 이동한다. • 계정과 사용자 정보 모두 등록되어있을 시, 홈화면으로 이동한다.	완료	
	애플 로그인	애플 로그인을 통해 사용자의 애플 계정을 등록한다.	미완료	
	이용약관	서비스를 이용하는 데 있어 사용자와 서비스 제공자 간의 규정된 권리와 의무를 명시한다.	완료	

	개인정보 처리방침	• 개인정보를 수집, 보유, 이용, 제공하는 과정에서 이용자의 개인정보를 보호하고 안전하게 처리하기 위한 정책이나 규정을 명시한다.
		• 수집하는 개인정보의 항목
		  ○ 구글 계정
		  ○ 목소리 녹음파일
			완료	

	닉네임 설정	• 키보드를 통해 입력받을 수 있어야 한다. • 닉네임은 8자를 초과해선 안된다. • 닉네임에 특수문자는 포함될 수 없다.	완료	
	캐릭터 설정	• 제시된 4가지 캐릭터 중 사용자가 원하는 캐릭터를 선택한다. • 캐릭터는 무료로 상업적 이용이 가능한 디자인으로 선정한다.	완료	

	사용자 음성정보 수집	• 사용자에 맞는 가이드 음성을 생성하기 위해 세 종류의 텍스트에 대한 음성정보를 수집한다.
		• 사용자 편의성을 위한 progress bar를 배치한다.
		  ○ 0-100%로 표시한다.
		• 마이크 아이콘 버튼을 눌러 사용자 음성 녹음을 시작한다.
		• 재생/중지 버튼을 눌러 녹음 진행 및 중지가 가능해야 한다.
		• 휴지통 아이콘 버튼을 눌러 진행 중인 녹음을 삭제한다.
		• 완료 아이콘 버튼을 눌러 녹음을 완료한다.
		• 완료된 녹음의 재생 버튼을 통해 청취할 수 있어야 한다.
		  ○ 재생바를 움직여 원하는
			완료	

		구간을 확인한다. • 초 단위로 시간을 표시한다. • 음성정보 수집이 완료되면, 데이터베이스에 사용자 정보 저장 후 홈화면으로 이동한다.		
네비게이션 바	네비게이션 바	• 기록/홈/스크립트로 구성한다. • 기본값은 홈으로 한다. • 홈 클릭 시, 홈화면으로 이동한다. • 기록 클릭 시, 예시 대본 연습 기록 목록으로 이동한다. • 스크립트 클릭 시, 예시 대본 목록으로 이동한다.	완료	
홈	사용자 캐릭터	사용자가 선택한 캐릭터를 표시한다.	변경	사용자의 연습 추이에 따른 캐릭터 성장 제거
	히스토리	마지막으로 연습한 대본을 표시한다.	변경	progress bar 제거
	연속 공부일 수	• 연속으로 연습한 일수를 표시한다. • 마지막 접속 시간을 기준으로 24시간이 넘어가면 리셋하고, 재접속 시 다시 1일로 표시한다.	변경	디자인적으로 기존 홈화면의 최하단에서 중앙으로 위치 변경
	설정으로 이동	앱 상단바에 톱니바퀴 모양의 설정 아이콘을 통해 설정 페이지로 이동한다.	완료	

설정	사용자 음성정보 재설정	초기에 설정한 사용자 음성정보를 변경한다.	미완료	
	이용약관	서비스를 이용하는 데 있어 사용자와 서비스 제공자 간의 규정된 권리와 의무를 명시한다.	완료	

	개인정보 처리방침	• 개인정보를 수집, 보유, 이용, 제공하는 과정에서 이용자의 개인정보를 보호하고 안전하게 처리하기 위한 정책이나 규정을 명시한다.
		• 수집하는 개인정보의 항목
		  ○ 구글 계정
		  ○ 목소리 녹음파일
			완료	

	로그아웃	• dialog를 통해 사용자에게 확인 후 수행한다. • 로그아웃 후 회원가입/로그인 화면으로 이동한다.	완료	
	탈퇴	• dialog를 통해 사용자에게 확인 후 수행한다. • 사용자 계정, 음성, 정보 모두 삭제 후 회원가입/로그인 화면으로 이동한다.	완료	
기록 목록	대본 종류에 따라 News / User로 분류	• 예시 대본 연습 기록은 News 탭에서, 사용자 대본 연습 기록은 User 탭에서 확인한다. • 선택된 탭은 진하게	완료	

		표시한다.		
	카테고리 설정	• 주어진 카테고리 목록 중 한 가지 선택해 대본을 필터링할 수 있어야 한다. ○ 카테고리 목록 : 전체, 정치, 경제, 사회, 스포츠, 생활/문화, IT/과학, 세계 • 좌우 스크롤이 가능해야 한다.	완료	
	연습한 대본 목록	• 대본의 제목, 카테고리, 내용 일부분을 표시한다. • 해당 대본의 마지막 프롬프트 연습 점수를 표시한다. ○ 아직 프롬프트 연습 이력이 없는 경우엔 점수를 표시하지 않는다. • 상하 스크롤이 가능해야 한다. • 대본을 클릭하면, 해당 대본 상세 기록 페이지로 이동한다.	변경	• 언론사 표시 제거 • 마지막 점수가 100점일 때, 메달이 아닌 점수로 표시
기록 상세 페이지	뒤로가기 버튼	왼쪽 화살표 아이콘을 통해 이전 화면으로 이동한다.	완료	
	카테고리 / 제목 표시	대본의 카테고리와 제목을 표시한다.	변경	언론사 표시 제거
	스크랩한 문장 목록	• 문장 단위 연습에서 사용자가 스크랩한 문장 목록을 표시한다.	변경	계획서에 없는 새로운 기능

		• 좌우로 슬라이드가 가능해야 한다. • 문장 개수를 점으로 나타내고, 현재 문장 인덱스는 점을 진하게 표시한다.		
	프롬프트 정확도 추이 그래프	• 프롬프트 연습 기록을 선형 그래프로 표시한다. • X축: 연습 시각, Y축: 정확도 • 그래프의 각 점을 누르면 해당 연습 시각과 점수를 표시한다.	변경	계획서에 없는 새로운 기능
	다시 연습하기 버튼	버튼을 누르면 해당 대본의 연습 방법 선택 화면으로 이동한다.	변경	기존엔 기록 페이지에서 바로 연습 가능했으나, 연습 페이지로 이동해 연습 가능하도록 변경

	문장 단위 연습 기록	• 각 문장별 연습 기록을 확인한다.
		  ○ 음성 가이드, 사용자 음성, 피드백
		• 다시 연습하기 버튼을 통해 해당 페이지에서 다시 연습 후 피드백을 받을 수 있어야 한다.
		• 다음 버튼을 눌러 대본 내 기록된 다음 문장으로 이동 가능해야 한다.
		• 마지막 문장일 경우 완료 버튼을 표시한다.
		  ○ 완료 버튼 클릭 시, 대본 페이지로 이동한다.
			변경	기획 변경으로 인한 기능 삭제

	프롬프트 연습 기록	• 이전에 연습했던 프롬프트 기록 확인한다. ○ 음성 가이드, 사용자 음성, 피드백 • 다시 연습하기 버튼을 통해 해당 페이지에서 다시 연습 후 피드백을 받을 수 있어야 한다.	변경	기획 변경으로 인한 기능 삭제
스크립트 목록	대본 종류에 따라 News / User로 분류	• 예시 대본 연습 기록은 News 탭에서, 사용자 대본 연습 기록은 User 탭에서 확인한다. • 선택된 탭은 진하게 표시한다.	완료	
	검색 버튼	돋보기 아이콘 버튼을 클릭하면 검색 페이지로 이동한다.	변경	계획서에 없는 새로운 기능
	카테고리 설정	• 주어진 카테고리 목록 중 한 가지 선택해 대본을 필터링할 수 있어야 한다. ○ 카테고리 목록 : 전체, 정치, 경제, 사회, 스포츠, 생활/문화, IT/과학, 세계 • 좌우 스크롤이 가능해야 한다.	완료	
	연습한 대본 목록	• 대본의 제목, 카테고리, 내용 일부분을 표시한다. • 상하 스크롤이 가능해야 한다. • 대본을 클릭하면, 해당 대본 상세 기록 페이지로 이동한다.	변경	언론사 표시 제거

	나만의 대본 만들기 버튼	• User 탭 하단에 버튼을 배치한다. • 사용자가 직접 대본을 생성하는 페이지로 이동한다.	완료	
검색	뒤로가기 버튼	왼쪽 화살표 아이콘을 통해 이전 화면으로 이동한다.	변경	계획서에 없는 새로운 기능
	검색 키워드 입력	• 입력창 클릭 시, 키보드가 활성화 되어야 한다. • 키보드를 통해 입력받을 수 있어야 한다.		
	대본 종류에 따라 News / User로 분류	• 예시 대본 연습 기록은 News 탭에서, 사용자 대본 연습 기록은 User 탭에서 확인한다. • 선택된 탭은 진하게 표시한다.		
	검색어에 따른 대본 표시	• 검색어가 제목에 포함된 대본을 표시한다. • 대본의 제목, 카테고리, 내용 일부분을 표시한다. • 상하 스크롤이 가능해야 한다. • 대본을 클릭하면, 해당 대본 상세 기록 페이지로 이동한다.		
사용자 대본 생성	뒤로가기 버튼	왼쪽 화살표 아이콘을 통해 이전 화면으로 이동한다.	완료	
	제목 입력	키보드를 통해 입력받을 수 있어야 한다.	완료	
	카테고리 지정	주어진 카테고리 목록 중 한	완료	

		가지 선택한다. ● 카테고리 목록 : 전체, 정치, 경제, 사회, 스포츠, 생활/문화, IT/과학, 세계		
	내용 입력	키보드를 통해 입력받거나 AI를 통해 자동 입력되어야 한다.	완료	
	AI로 생성하기 버튼	● 제목, 카테고리를 기반으로 gpt를 통해 대본 내용을 생성한다. ● 제목, 카테고리 값이 있는지 확인한다. ○ 제목이 비어있으면, 제목란에 에러 메시지를 표시한다. ○ 카테고리가 선택되지 않았으면, dialog를 통해 안내한다. ● AI로 내용을 생성하는 동안 로딩 메시지가 나타나야 한다. ● 내용이 생성되면, 로딩 메시지를 지우고 내용 표시한다.	완료	
	완료 버튼	● 제목, 카테고리, 내용의 값이 있는지 확인한다. ○ 제목 또는 내용이 비어있으면, 비어있는 칸에 에러 메시지를 표시한다. ○ 카테고리가 선택되지	완료	

		않았으면, dialog를 통해 안내한다. • 모든 값이 있으면, 대본 내용 수정 페이지로 이동한다.		
사용자 대본 내용 수정	뒤로가기 버튼	왼쪽 화살표 아이콘을 통해 이전 화면으로 이동한다.	완료	
	제목/카테고리 표시	대본 생성 페이지에서 사용자가 입력한 제목과 카테고리를 표시한다.	완료	
	내용 표시 및 수정	• 대본 생성 페이지에서 사용자가 입력한 내용을 문장 단위로 표시한다. • 각 문장 블록을 눌러 내용 수정이 가능해야 한다. • 문장 블록은 추가/삭제가 가능해야 한다.	완료	
	저장 후 나가기 버튼	• 빈 블록이 있다면, dialog를 통해 빈 블록에 내용 추가 또는 빈 블록을 삭제하도록 안내한다. • 내용이 비어있다면, dialog를 통해 안내한다. • 위 사항 모두 만족 시, 데이터베이스에 대본 저장 후 사용자 대본 목록 화면으로 이동한다.	완료	
	연습하기 버튼	• 빈 블록이 있다면, dialog를 통해 빈 블록에 내용 추가 또는	완료	

		빈 블록을 삭제하도록 안내한다. ● 내용이 비어있다면, dialog를 통해 안내한다. ● 위 사항 모두 만족 시, 데이터베이스에 대본 저장 후 연습 방법 선택 화면으로 이동한다.		
대본 상세 페이지	뒤로가기 버튼	왼쪽 화살표 아이콘을 통해 이전 화면으로 이동한다.	완료	
	스크랩 버튼	연습하고 싶은 대본을 스크랩한다.	변경	해당 페이지에서 스크랩 기능 삭제
	제목/카테고리 표시	대본의 제목과 카테고리를 표시한다.	변경	언론사 표시 제거
	내용 표시	● 대본의 내용을 문장 단위 블록으로 표시한다. ● 상하 스크롤 가능해야 한다.	완료	
	연습하기/기록보기 버튼	● 이전에 연습한 이력이 있는 대본은 기록보기 / 연습하기 버튼을 나란히 배치한다. ○ 기록보기 버튼을 클릭하면, 해당 대본의 연습 기록 상세 페이지로 이동한다. ● 이전에 연습한 기록이 없는 대본은 연습하기 버튼만 배치한다. ● 연습하기 버튼을 클릭하면, 연습 방법 선택 화면으로	완료	

		이동한다.		
연습 방법 선택	취소 버튼	버튼을 누르면 이전 페이지로 이동한다.	변경	기존엔 '문장단위연습' 버튼과 '프롬프트' 버튼 외의 화면을 클릭하면 이전 페이지로 이동했으나, 사용자 편의성을 위해 명시적으로 표기
	문장단위연습 버튼	버튼을 누르면 문장 단위 연습 페이지로 이동한다.	완료	
	프롬프트 버튼	버튼을 누르면 프롬프트 가이드 페이지로 이동한다.	완료	
문장단위연습	progress bar	문장 수에 따라 진행률을 표시한다. ● 0-100%로 표시한다.	변경	계획서에 없는 새로운 기능
	대본 제목/ 카테고리 표시	현재 연습 중인 대본의 제목과 카테고리를 표시한다.	완료	
	스크랩 버튼	책갈피 아이콘 버튼을 눌러 해당 문장을 스크랩 또는 스크랩 취소한다. ● 기록 페이지에서 스크랩한 문장 목록에 추가/삭제된다.	변경	계획서에 없는 새로운 기능
	연습 문장 표시	연습할 문장을 표시한다.	완료	
	음성 가이드	● 개인화 TTS 모델을 이용해 사용자의 목소리와 아나운서의 특성을 반영해 연습 문장에 대한 음성 가이드를 생성한다. ● 재생/정지 버튼을 눌러 음성 가이드 청취 재생 및 중지가 가능해야 한다.	완료	

		○ 재생바를 움직여 원하는 구간을 확인한다. ○ 초 단위로 시간을 표시한다.		
	음성 녹음	● 마이크 아이콘 버튼을 눌러 음성 녹음을 시작한다. ● 재생/정지 버튼을 눌러 녹음을 진행 및 정지할 수 있다. ● 휴지통 아이콘 버튼을 눌러 진행 중인 녹음을 삭제한다. ● 완료 버튼을 눌러 녹음을 완료한다. ● 완료된 녹음의 재생 버튼을 통해 청취한다. ○ 재생바를 움직여 원하는 구간을 확인한다. ● 초 단위로 시간을 표시한다.	완료	
	정확도 피드백	● 음성 녹음이 완료된 후 표시한다. ● 아래 항목을 종합한 정확도 점수를 제공한다. ○ 가이드 음성과 사용자 음성 간 유사도 ○ 전사된 텍스트 간 유사도 ● 0-100 사이의 숫자로 표시한다.	변경	기존에는 음성 가이드와 유사도가 일정 범위 이상 벗어나는 부분은 빨간색으로 표시해 피드백을 제공하기로 했으나, 음성 유사도와 발음 유사도를 모두 포함하는 피드백 방법으로 변경
	사용자 발음 표시	● 음성 녹음이 완료된 후		

		표시한다. • STT 모델을 이용해 사용자 음성 녹음을 전사하여 제공한다.		
	저장하기 버튼	버튼을 누르면 해당 문장 연습 기록 저장 및 기록 페이지에서 확인 가능해야 한다.	변경	기획 변경으로 인한 기능 삭제
	다음 버튼	버튼을 누르면 다음 연습 문장으로 이동한다. • 음성 녹음 미완료 시, 녹음을 완료하라는 안내 메시지가 뜨고 화면이 넘어가지 않는다.	완료	
프롬프트 연습	가이드 음성 듣기/연습하기 중 선택	• 가이드음성 듣기 버튼을 눌러 음성 가이드를 듣는다. • 연습하기 버튼을 눌러 연습을 바로 시작한다. • 어떤 버튼을 선택하든 가이드 음성은 생성되어야 한다.	완료	
	3초 카운트다운	3-2-1 카운트다운 후 자동으로 프롬프트 화면으로 이동한다.	완료	
	가이드 음성 듣기	• 가이드 음성을 재생하며 대본이 프롬프트 화면처럼 자동으로 스크롤되어 내려간다. • 재생/정지 버튼을 눌러 음성 가이드 청취 재생 및 정지가 가능해야 한다. • 음성 가이드를 다 들은 후에는 다시 가이드 음성 듣기/연습하기 중 선택할 수 있어야 한다.	완료	

	프롬프트 화면 구성	실제 뉴스 프롬프트와 같은 환경을 구성한다. ● 검정 바탕에 흰 글씨로 대본 전문을 표시한다. ● 가로모드에서 시작되도록 한다. ● 자동으로 화면이 위로 올라가야 한다.	완료	
	사용자 음성 녹음	● 마이크 아이콘 버튼을 눌러 음성 녹음을 시작한다. ● 재생/정지 버튼을 눌러 녹음을 진행 및 정지한다. ● 휴지통 아이콘 버튼을 눌러 진행 중인 녹음을 삭제한다. ● 완료 버튼을 눌러 녹음을 완료한다. ● 완료된 녹음의 재생 버튼을 통해 청취할 수 있어야 한다. ○ 재생바를 움직여 원하는 구간을 확인한다. ● 초 단위로 시간을 표시한다.	완료	
프롬프트 피드백	대본 제목/ 카테고리 표시	연습한 대본의 제목과 카테고리를 표시한다.	완료	
	대본 전문 표시	연습한 대본의 전문을 표시한다.	완료	
	음성 가이드	● 개인화 TTS 모델을 이용해 사용자의 목소리와 아나운서의 특성을 반영해 연습 대본 전문에 대한 음성 가이드를 생성한다.	완료	

		• 재생/정지 버튼을 눌러 음성 가이드 청취 재생 및 정지가 가능해야 한다.
		  ○ 재생바를 움직여 원하는 구간을 확인한다.
		  ○ 초 단위로 시간을 표시한다.		

	녹음된 사용자 음성	• 프롬프트 연습에서 녹음된 사용자 음성을 표시한다.
		• 재생/정지 버튼을 눌러 음성 가이드 청취 재생 및 정지가 가능해야 한다.
		  ○ 재생바를 움직여 원하는 구간을 확인한다.
		  ○ 초 단위로 시간을 표시한다.
			완료	

	정확도 피드백	• 음성 녹음이 완료된 후 표시한다.
		• 아래 항목을 종합한 정확도 점수를 제공한다.
		  ○ 가이드 음성과 사용자 음성 간 유사도
		  ○ 전사된 텍스트 간 유사도
		• 0-100 사이의 숫자로 표시한다.
			변경	기존에는 음성 가이드와 유사도가 일정 범위 이상 벗어나는 부분은 빨간색으로 표시해 피드백을 제공하기로 했으나, 음성 유사도와 발음 유사도를 모두 포함하는 피드백 방법으로 변경

	연습 기록 자동 저장	연습을 마치고 나면 데이터베이스에 연습 기록을	변경	기존에는 사용자가 저장하기 버튼을 누른

		자동으로 저장한다. ● 저장 목록 ○ 연습 시각 ○ 정확도		경우에만 연습 기록 저장 ● 저장 목록 ○ 음성 가이드 ○ 사용자 음성 ○ 피드백
	저장하기 버튼	버튼을 누르면 해당 프롬프트 연습 기록 저장 및 기록 페이지에서 확인 가능해야 한다.	변경	기획 변경으로 인한 기능 삭제
	다음 버튼	버튼을 누르면 자동으로 홈화면으로 이동한다.	완료	

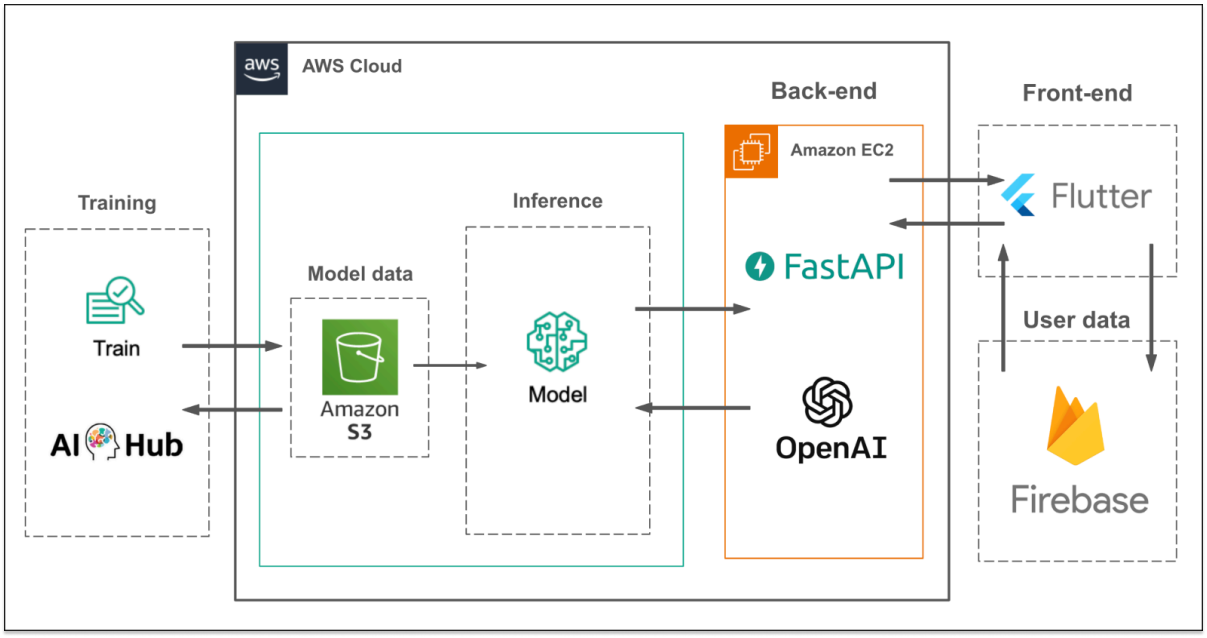
2.2.3 시스템 비기능(품질) 요구사항

대분류	요구사항 명	상세 설명
제품 요구사항: 제품의 품질에 대한 비기능적 요구 사항	**사용성**: 사용자가 어떻게 쉽게 사용할 수 있는 가	카테고리 필터링 기능을 지원해 사용자가 대본을 쉽게 찾을 수 있도록 한다.
		회원가입 시 사용자 약관을 제공한다.
		사용 중 에러가 발생한다면, 해결방법이 포함된 에러 메시지를 보여준다.
	효율성(성능): 특정 기능이 특정시간내에 실행	가이드 음성 생성은 최소 4초 이내로 이루어져야 한다.
		정확도 피드백은 최소 2초 이내로 이루어져야 한다.
	효율성(공간): 특정 기능 수행 시 메모리를 최대 얼마까지 사용할 수 있는 가	모든 모델의 추론연산은 g4dn instance의 메모리인 16GB 이내에서 이루어져야 한다.
		저장하는 음성 데이터는 125GB 를 초과해서는 안된다.
	신뢰성: 특정한 기능을 실행할 때의 가능성	시스템은 24시간 운영되어야 하며, 99.9% 이상의 시간 동안 서비스가 중단되지 않아야 한다.
	이식성: 소프트웨어가 다양한 플랫폼에서 작동하는가	android 및 ios 환경에서 동작해야 한다.
조직 요구사항: 소프트웨어 개발과 관계되는 조직들에 대한 비기능적 요구 사항	**배포:** 소프트웨어를 어떻게 배포할 것인가	Android에서 서비스를 이용할 수 있어야 한다.
	구현: 소프트웨어를 어떤 방법론을 사용해 구현할 것인가	프론트 엔드는 Dart, 백엔드는 Python 언어를 사용한다.
		Agile 방법론을 사용해 반복적이고 점진적으로

		소프트웨어 개발을 진행한다.
		Git을 이용해 코드 버전관리를 하며 모든 코드의 병합은 code review를 마친 pr merge로만 이루어져야 한다.
		문서화를 철저히 하여, 코드와 시스템 구조, API 명세서 등이 Confluence에 명확히 기록되도록 한다.
외부 요구사항: 소프트웨어에 영향을 미치는 외부에 대한 비기능적 요구사항	윤리: 소프트웨어 개발 및 운영에서 윤리적 기준을 준수하는 요구사항	소프트웨어의 데이터 수집, 사용, 공유에 대해 사용자에게 명확한 정보를 제공하고, 사용자가 이를 쉽게 이해할 수 있도록 한다.
		소프트웨어는 사용자의 개인정보를 보호해야 하며, 관련 법규를 준수해야 한다.
	법적: 소프트웨어가 준수해야하는 법적 요구사항	소프트웨어 개발 시 사용되는 모든 라이브러리와 툴의 저작권 및 라이선스 조건을 준수한다.

2.2.4 시스템 구조 및 설계도

시스템 아키텍처



이러한 서비스를 제공하기 위한 Loro의 시스템 아키텍처는 다음과 같습니다.

Loro는 사용자 맞춤형 음성 피드백을 위해 아나운서 억양을 학습한 모델로 개인화 TTS 기능을 제공합니다. 이때 모델은 AI Hub에서 제공하는 아나운서 음성 데이터셋을 이용해 직접 학습시켰습니다. 학습 완료 후 만들어진 모델은 S3에 저장해 이후 추론 단계에서 다시 가져와 사용합니다. 프론트로부터 사용자 음성 데이터와 텍스트를 받아와 모델에 입력으로 전달하고, 추론을 거쳐 해당 텍스트에 대한 가이드 음성을 생성합니다.

스크립트 생성 기능을 위해 OpenAI를 활용하고, AI와 프론트 사이에서 데이터를 주고받는 과정은 FastAPI를 이용해 백엔드 서버를 구축하여 수행합니다.

사용자 정보는 파이어베이스의 Authentication을 사용하여 관리하고, 사용자별 음성 데이터와 피드백 기록은 Firestore와 Storage를 이용해 저장합니다. 마지막으로 프론트는 안드로이드와 iOS 모두 개발 가능한 크로스 플랫폼 프레임워크인 Flutter를 사용하여 개발했습니다.

2.2.5 활용/개발된 기술

VITS2: TTS 모델

최근 논문들이 end-to-end TTS를 제안하고는 있는데 이러한 모델들은 single stage 학습과 병렬 샘플링을 가능하게 하지만, 그 샘플 품질은 현재의 두 단계 파이프 라인의 TTS 시스템과는 맞지 않았습니다. vits는 현재의 two-stage 모델보다 자연스러운 음성을 생성하는 **병렬 end-to-end TTS** 방법을 제안하는데 normalizing flows로 augment된 variational inference과 adversarial training을 채택하여 생성 모델링의 표현력을 향상시킵니다.

또 입력 텍스트에서 다양한 리듬으로 음성을 합성하기 위해 stochastic duration predictor를 제안했는데 잠재 변수에 대한 불확실성 모델링과 stochastic duration predictor를 통해 입력 텍스트를 다양한 피치와 리듬등 여러 가지 방식으로 말하는 자연스러운 일대다 관계를 표현합니다.

저희 팀은 vits 모델을 선택할 때, MOS 점수를 기준으로 성능을 파악했는데 LJ Speech 데이터셋 기준 MOS 점수가 성능 좋은 공개 TTS 시스템들을 능가하며 그라운드 트루스와 유사한 MOS를 달성한다는 결과를 보여 선택하게 되었습니다.

특히 vits2는 multi-speaker modeling의 경우, 다른 모델들이 화자 확장을 하려면 학습을 추가적으로 하거나 Zero-shot approach를 하지만 이들은 화자 공간이 sparse 한 문제가 있어 들어봤을 때, 화자의 전반적인 발화 특징, 예를들어 감정, 강조 같은 것이 표현이 되지 않습니다. 그래서 “음성 각 부분의 발화 특성을 모두 추출해서 보관 후 합성 시 각 부분에 맞도록 입혀주는 것이 가능하다면, 해당 인물의 모든 contents에서의 발화 특성을 모두 표현할 수 있는 것이 아닐까?” 라는 아이디어에서 출발했습니다.

음성 오디오의 분포를 알 수 없기 때문에 예측하고 다룰 수 있는 분포로 Transform하고 Target Speaker의 다양한 오디오로부터 분포에 매핑된 데이터 포인트를 찍는데, 클러스터링을 통해 여기서 대표되는 feature를 뽑는 방식을 사용했습니다. 구체적으로는 클러스터링 한 센트로이드들을 모아서 화자의 음성 특징이라 정의합니다.

연속공간에서 데이터가 분포되어 있었을 것이기 때문에 중간점이 있어야 음성을 복원할 수 있는데 이는 Sum to one이 되는 weight를 사용하는 Weighted Sum을 사용함으로써 해결했습니다.

SFEN: Speech Feature Encoding Network

Multi speaker speech synthesis는 각 화자의 특성을 유사하게 표현하고 화자 간 공통 정보를 공유할 수 있게 하여, 각 화자를 개별적으로 훈련시키는 것보다 상대적으로 적은 훈련 데이터로도 높은 품질의 다중 화자 음성을 합성할 수 있게 합니다.

그러나, 음성 오디오에서 표현되는 음색과 운율은 그 내용과 일치하며, 짧은 음성에서는 화자의 음성 특성의 일부분만 드러난다는 문제점이 있습니다. 짧은 참조 오디오로부터 얻은 벡터는 주어진 콘텐츠에 따라 달라지는 speaker의 음성 특성의 작은 부분만을 나타냅니다. 따라서 SFEN은 화자의 음성을 작은 단위로 분할해서 분포에 매핑시킨 후, 이러한 음성 특징들을 클러스터링 하여 이산화된 대표 지점을 얻습니다. 이러한 대표 지점들로 speech feature codebook을 만들어 화자의 전반적인 음성 특성을 모델링함으로써 해결했습니다.

또한, 음색이나 음성 특성이 유사한 speaker들은 실제로는 비슷함에도 불구하고 과도하게 멀리 떨어지도록 학습될 수 있으며, speaker vector를 위한 학습된 공간은 희소하고 불연속적일 수 있다는 문제가 있습니다. 이러한 특징들은 학습된 공간에서 음성 합성을 위해 적절한 speaker vector를 얻는 것을 제한할 수 있으며, 합성 모델은 각 벡터를 공간이 아닌 독립적인 조건으로 학습하는 경향이 있습니다. 이는 Target speaker와 유사한 음성 특성을 표현하는 데 실패하고 정확한 음성을 합성하는 데 실패하는 결과를 초래합니다. 따라서 SFEN은 input mel spectogram으로부터 인코딩 및 디코딩을 통해 원시 파형을 재구성하도록 훈련된 오토인코더를 이용하여 latent space를 구성합니다. 화자들의 음성에서 다양한 speaker feature을 밀집되고 연속적인 분포로 인코딩합니다.

jamotools, g2pk

Loro가 제공하는 피드백 중 발음에 대한 피드백 평가는 사용자의 음성을 텍스트화(Speech to Text)해서 텍스트 간의 유사도를 계산해 구현했습니다. 이때, 단순히 한글, 한글 간의 CER(Character Error Rate)를 구하는 것이 아닌 텍스트를 발음 기호인 IPA(International Phonetic Alphabet)으로 바꿔 비교하게 됩니다. 예를 들어 기준 음성의 올바른 텍스트가 “나는 집에 갔다"이고, 사용자가 기준 음성과 상당히 비슷하게 발음한 “나는 지베 가따"의 경우 둘 간의 CER은 약 50%로 에러율이 굉장히 크지만 ipa로 변환 후 “nˈɐnɯntɕˈibe gˈɐdtɐ” 와 “nˈɐnɯntɕˈibe gˈɐtɐ” 를 비교하게 되면 94% 에러율로 훨씬 정확하게 피드백 하는 것을 알 수 있습니다. 따라서 이렇게

text -> ipa 변환 로직이 필요하게 됐고 이에 자소 단위 추출을 위한 jamotools, 자소를 음소로 변환하는 g2pk를 이용하게 되었습니다.

Whisper-Jax

Loro는 가이드 음성을 제공하고 이와 사용자가 연습한 음성과의 비교를 통해 피드백을 합니다. 피드백을 위해 두가지 정보를 활용하는 데 그 중, 텍스트 간 유사도를 계산을 하기 위해 STT 모델을 사용합니다. 한국어 STT 모델은 허깅페이스에서 찾아 볼 수 있는데, 그 중 저희는 whisper-medium 모델을 사용하기로 했습니다.

하지만 그냥 whisper 모델을 로드하고 추론을 했을 때, 추론시간이 너무 오래걸리는 문제가 있어 저희가 사용 중인 g4dn instance가 gpu 칩이 있음을 활용해 시간을 줄일 수 있는 방법이 없을 지 고민하게 되었고, 그래서 google의 jax를 이용한 whisper-jax 오픈소스 코드를 사용하기로 했고 결과적으로 10배 빠른 속도를 얻을 수 있게 되었습니다. jax의 JIT(just in time compile) 기능을 이용해 비교적 시간이 오래걸리는 모델을 로드하고 초기화하는 과정을 코드레벨 캐싱하기로 했고 jax가 기존 numpy 연산보다 빠른 gpu 가속연산을 한다는 점을 통해 계산속도를 줄이고자 했습니다.

GPU EC2 instance - g4dn

Loro가 STT, TTS 모델을 사용하며 각각 모델의 추론 코드는 gpu를 활용한 병렬 연산에 최적화되어 있습니다. 따라서 사용자에게 빠른 응답을 제공하기 위해 gpu 칩이 포함된 instance를 선택하기로 했습니다.

S3 storage

아나운서 데이터셋을 학습한 모델을 저장하고 이를 관리하기 위해 아마존 의 s3 서비스를 이용했습니다. 특히 s3는 버전관리를 지원해 모델의 체크포인트를 기록해놓고 테스트 하기 편한 것이 큰 장점으로 다가왔습니다.

FastAPI

- 간단한 설치와 조작법으로 Python 프레임워크 중 가장 빠르고 쉽게 사용할 수 있어 선택했습니다.
- Swagger API 문서를 자동으로 생성합니다. 스키마 확인이 용이하고, 간편하게 request/response를 테스트해볼 수 있습니다.
- FastAPI는 비동기 프로그래밍에 기반한 동시성 제어 모델을 사용합니다. Uvicorn을 사용해 Thread가 대기 시간동안 다른 일을 처리할 수 있어 자원을 효율적으로 사용할 수 있습니다.

Selenium

예시 대본 생성을 위한 데이터를 스크래핑하기 위해 사용했습니다. Selenium을 활용해 뉴스 데이터셋의 원문 url로부터 뉴스 원문 내용만 가져옵니다. Selenium은 동적 크롤링을 지원하고 있어 AJAX를 사용하는 웹 사이트를 스크래핑하는 데 적합했습니다. Python을 제공하며 간편한 사용법으로 별도의 학습 없이 활용할 수 있어 선택했습니다.

LangChain

프롬프트에 항상 같은 명령을 입력해도 출력이 달라져 직접 확인 후 추가적으로 가공해야 하는 문제가 있었습니다. 또한 결과가 기대에 미치지 못하는 경우가 많았습니다. 그래서 정교한 프롬프트 입력을 위해 LangChain을 적용했습니다. LangChain의 ChatPromptTemplate을 활용해 system에 '아나운서 대본 작성' 이라는 역할을 부여하고, 한층 질 좋은 결과물을 도출했습니다.

OpenAI LLM

- 뉴스 방송에서 사용될 대본이기 때문에 흐름, 문체 등 문장의 매끄러움이 중요하다고 판단했습니다. 사용자에게 높은 품질의 콘텐츠를 제공하기 위해, 성능이 우수한 Open AI LLM을 사용하여 대본 생성 기능을 구현하기로 결정했습니다. 이를 위해 추가 비용이 발생할 수 있더라도, 사용자 경험과 대본 품질을 우선 고려한 결정입니다.
- OpenAI LLM 사용의 주 목적은 텍스트 가공과 텍스트 콘텐츠 생성입니다. 이를 수행할 수 있는 빠르고 비용 효율적인 모델인 gpt-3.5-turbo 모델을 활용했습니다. 이 모델은 텍스트의

의도를 이해하는 능력이 우수해 관련성 높은 결과를 도출해냅니다.

Flutter

- 소수 인원으로 단기간에 안드로이드와 iOS를 각각 개발하고 유지 보수하는 것은 무리라고 판단했습니다. 따라서 하나의 코드 베이스로 안드로이드와 iOS에서 모두 사용할 수 있는 크로스 플랫폼 프레임워크로 애플리케이션을 개발했습니다.
- Flutter와 React Native 모두 Hot reload를 제공하여 앱 초기화 없이 코드를 수정할 수 있어 빠른 개발이 가능하지만 Flutter는 UI와 로직 변경을 모두 지원하며 이를 빠른 속도로 반영하기 때문에 Flutter를 최종 선택했습니다.
- 다양한 패키지와 라이브러리를 제공하고 있어 서비스에서 필요한 오디오 플레이어, 선형 그래프, 상태 관리 등 다양한 기능을 쉽게 구현할 수 있었습니다. 뿐만 아니라 활발한 커뮤니티를 통해 마주한 문제를 해결하고 여러 오픈소스 자원들을 활용했습니다.

Firebase(Firestore, Authentication, Storage)

백엔드를 직접 구축할 경우 보안 문제, 시스템 성장 시 확장성과 성능을 고려해 봤을 때 서비스를 오래 유지하는 것에 어려움이 있을 것이라 판단했습니다. 따라서 Baas 중 대용량 데이터 처리에 적합한 데이터베이스이며, Flutter와의 연동성이 뛰어난 Firebase를 데이터베이스로 선택했습니다.

- Firestore

  NoSQl 기반의 데이터베이스이며, Firebase의 Realtime Database도 Firestore와 동일하게 데이터베이스의 기능을 제공하지만 쿼리 시 정렬과 필터링 조건문을 동시에 사용해야 하기 때문에 해당 기능을 제공하는 Firestore를 최종적으로 선택했습니다. 비용 책정 부분에서도 큰 단위의 데이터 요청이 자주 발생하는 경우 Realtime Database 보다 유리하게 작용합니다.

- Authentication

  Firebase에서 제공하는 인증 서비스를 이용하여 로그인 기능을 구현하였습니다. 인증된 사용자인지 확인하는 세션 처리에서 그 세션으로 데이터베이스와 저장소에 접근해도 문제가 없는지 확인하는 보안 처리, 비밀번호 찾기, 아이디 찾기, 비밀번호 변경 등이 모두 제공되고 있어 완성도 높은 서비스의 구현이 가능하다고 판단했습니다.

- Storage

  Firestore를 사용하고 있어 같은 생태계에 속해 연동이 용이한 스토리지 서비스인 Firebase Storage를 사용자의 음성을 저장하는 스토리지로 사용했습니다.

2.2.6 현실적 제한 요소 및 그 해결 방안

지원금을 받기는 하지만 학생으로서 서버에 투자할 비용에 한계가 있다 보니 모델 학습, 추론 시간이 오래 걸리는 문제가 있었습니다. 특히 추론시간의 증가는 응답 시간과 직결되어 저희가 가지고 있는 자원 안에서 최대한 좋은 성능을 내기 위해서 여러 최적화 방법을 모색하게 되었습니다.

JAX를 이용한 코드 레벨 캐싱

jax는 python에서 JIT(just in time) compile을 지원해서 파이썬 함수를 먼저 분석하고 그 연산을 최적화된 기계어 코드로 변환할 수 있게 해주는데 이를 통해 원래 파이썬이 가지고 있는 인터프리터 오버헤드를 크게 줄이고 계산을 줄일 수 있고 모델 로드, 연산의 결과값을 사용자의 매 요청마다 하는 것이 아니라 캐싱을 통해 빠르게 얻어 속도를 크게 줄일 수 있었습니다.

Half-precision

gpu instance를 최대한 활용하기 위해서 32비트 연산의 single precision이 아닌 16비트 부동소수점 연산인 half precision을 사용해 응답시간을 줄였고 저희가 가진 인스턴스가 적은 메모리를 가지고 있음에도 효율적으로 실행 가능하도록 했습니다.

I/O bound job 의 효율적 처리를 위한 비동기적 I/O 사용

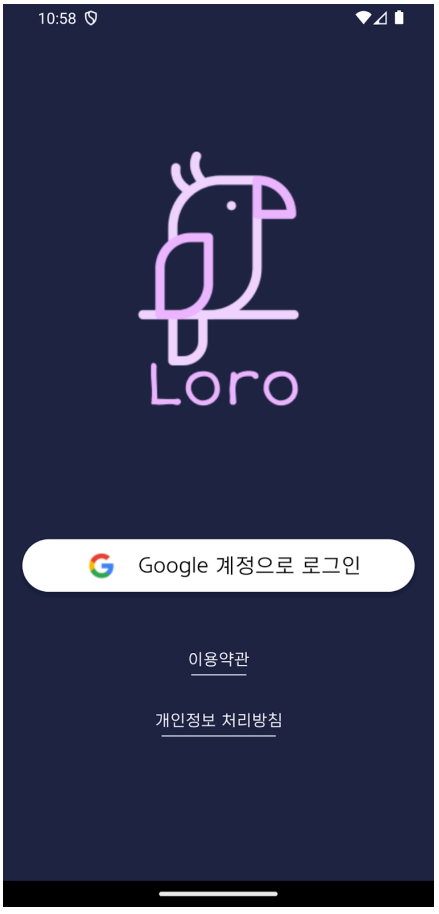
저희 loro 프로젝트에서는 음성 파일을 모델 추론의 입력, 그리고 또 결과 값으로 사용하며 임시 디렉토리에 저장하거나 실제로 스토리지 내에 저장하기도 하는데 동시성을 위한 라이브러리를 통해서 비동기 코루틴을 사용하여 I/O 작업이 블로킹되는 동안 다른 요청을 처리할 수 있도록 했습니다.

python에서는 병렬성/동시성을 위한 고수준 api로 asyncio, multiprocessing, thread,

subinterpreter 등이 있는데, 저희가 가진 인스턴스 자원으로는 멀티 프로세싱을 사용하기에는 메모리 리소스에 한계가 있었고 멀티 스레딩 개발 경험이 부족했습니다.

그래서 동시 실행을 위한 asyncio를 사용하기로 했습니다. asyncio는 이벤트 루프와 코루틴을 사용하여 I/O 바운드 작업을 효율적으로 처리하는데 기본적으로 단일 스레드에서 실행되고 await 키워드를 사용한 부분에서 다른 작업으로 컨텍스트를 전환할 수 있는 특징이 있어 이점을 이용해 개발을 진행했습니다.

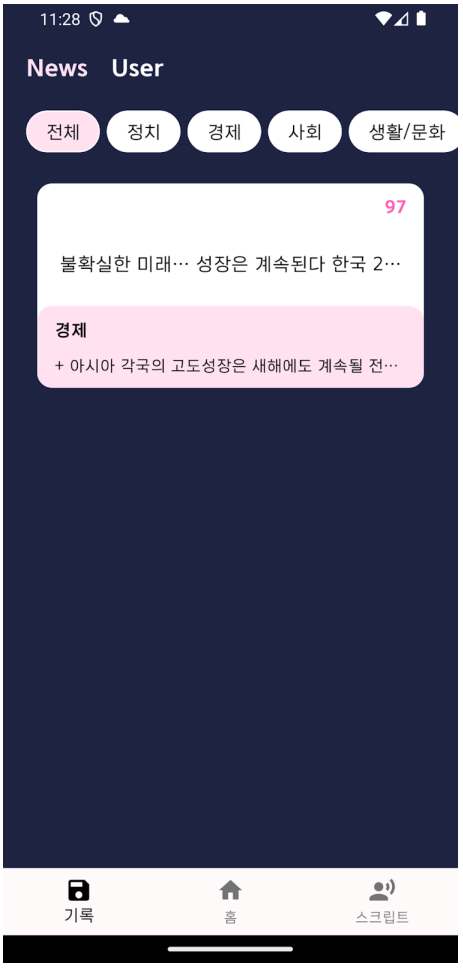
2.2.7 결과물 목록

대분류	소분류	기능	결과물 모습	기술 문서 유무
회원가입/로그인	구글 로그인	구글 로그인을 통해 계정을 등록한다.		무

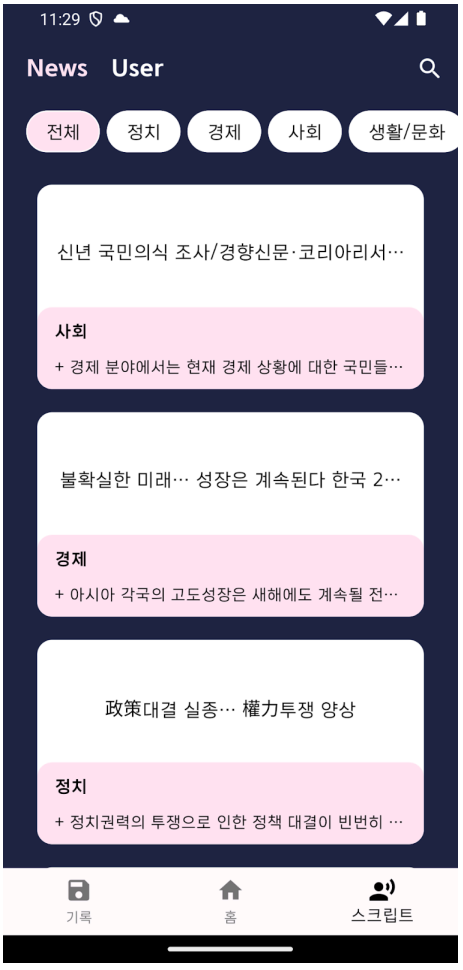
	닉네임 설정	닉네임을 입력한다.		무
	캐릭터 설정	캐릭터를 선택한다.		무
	사용자 음성정보 수집	음성을 녹음한다.		무
		다음 버튼을 통한 화면 이동		무

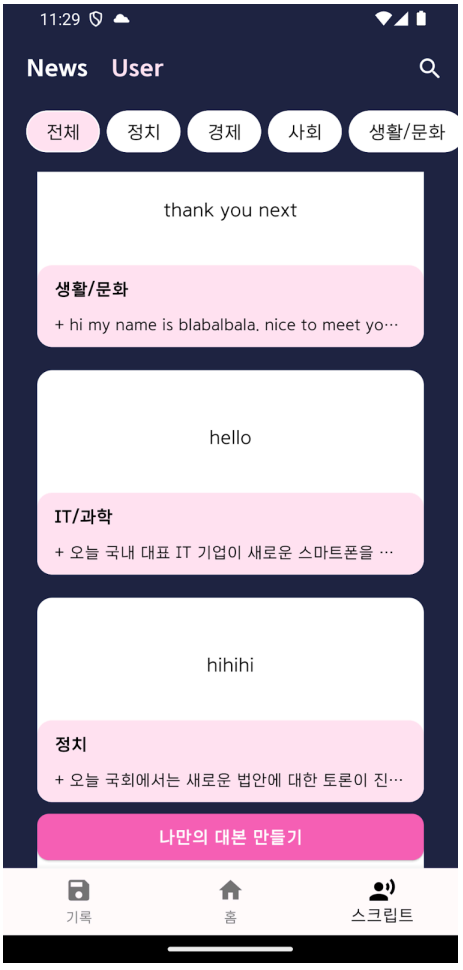
<table>
<tr><td></td><td></td><td></td><td> 
</td><td></td></tr>
<tr><td>네비게이션 바</td><td>네비게이션 바</td><td>네비게이션 바를 클릭해 화면을 이동한다.</td><td rowspan="3"> 
</td><td>무</td></tr>
<tr><td rowspan="2">홈</td><td>히스토리</td><td>마지막으로 연습한 대본을 확인한다.</td><td>무</td></tr>
<tr><td>설정 아이콘 버튼</td><td>설정 화면으로 이동한다.</td><td>무</td></tr>
</table>

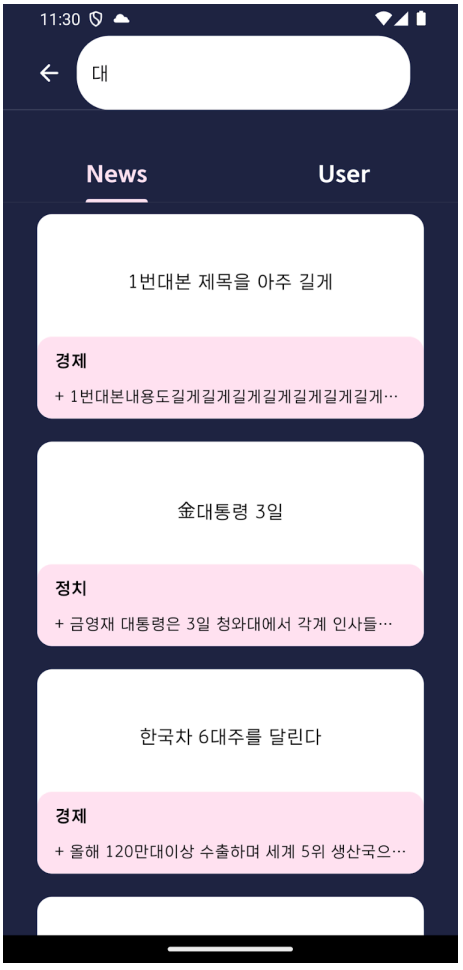
설정	로그아웃	서비스에서 로그아웃한다.		무
	탈퇴하기	서비스에서 탈퇴한다.		무

기록 목록	News/User 탭바	탭을 클릭해 대본 종류를 전환한다.		무
	카테고리	카테고리를 선택해 대본을 필터링한다.		무
	대본 목록	기록 상세 페이지로 이동한다.		무

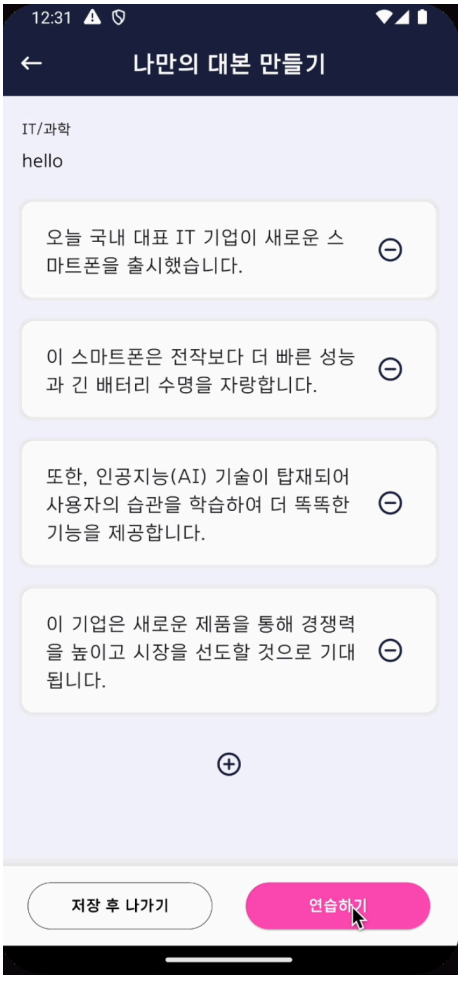
기록 상세 페이지	뒤로가기	이전 페이지로 이동한다.		무
	스크랩한 문장 목록	스크랩한 문장 목록을 확인한다.		무
	프롬프트 정확도 추이 그래프	프롬프트 연습 기록을 확인한다.		무
	다시 연습하기 버튼	다시 연습하기를 클릭해 연습 방법 선택 화면으로 이동한다.		무

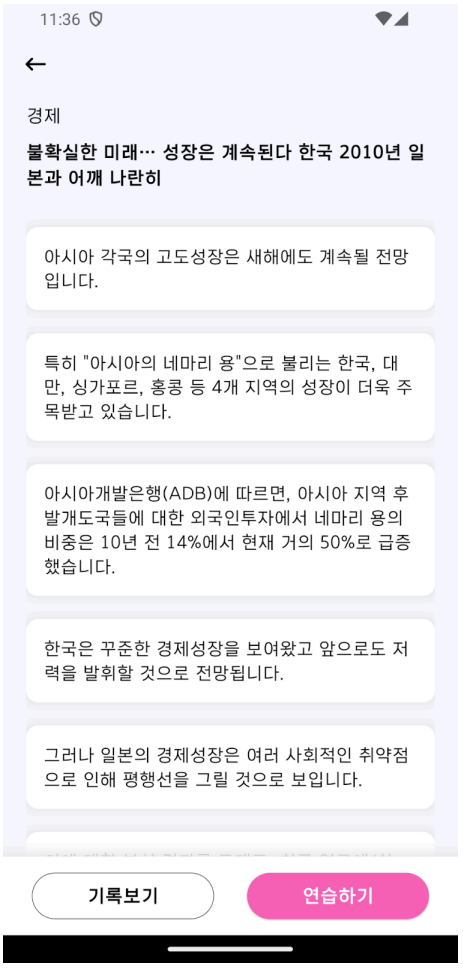
스크립트 목록	News/User 탭바	탭을 클릭해 대본 종류를 전환한다.		무
	카테고리	카테고리를 선택해 대본을 필터링한다.		무
	대본 목록	대본 상세 페이지로 이동한다.		무
	검색 아이콘 버튼	검색 화면으로 이동한다.		무

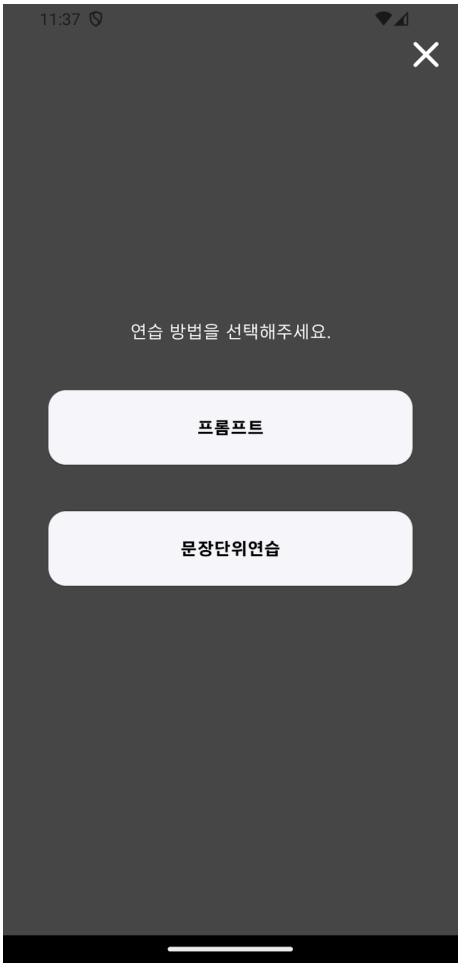
	나만의 대본 만들기 버튼	대본 생성 화면으로 이동한다.		무

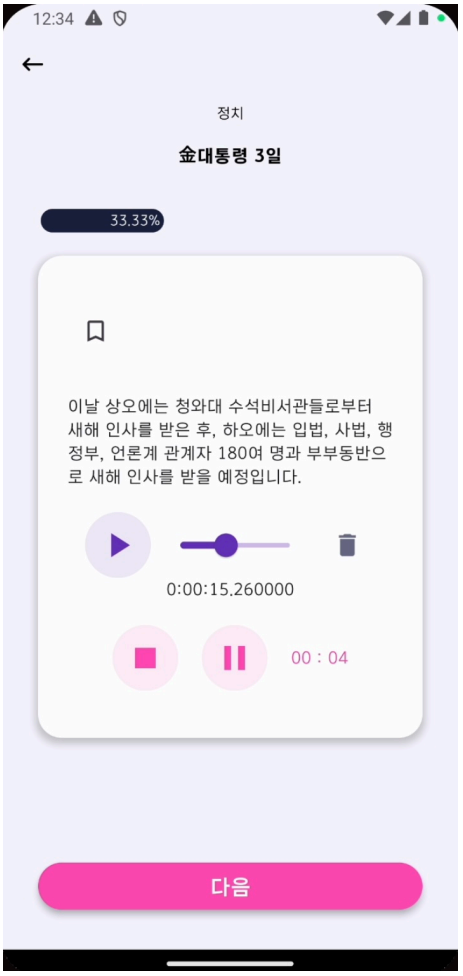
검색	뒤로가기	이전 페이지로 이동한다.		무
	검색 키워드 입력	검색어를 입력한다.		무
	News/User 탭바	탭을 클릭해 대본 종류를 전환한다.		무
	대본 목록	대본 상세 페이지로 이동한다.		무

사용자 대본 생성	뒤로가기	이전 페이지로 이동한다.		무
	대본 구성요소 입력	대본의 제목을 입력한다.		무
		대본의 카테고리를 지정한다.		무
		대본의 내용을 입력한다.		무
	AI로 생성하기 버튼	AI로 대본 내용을 생성한다.		무
	완료 버튼	사용자 대본 내용 수정 페이지로 이동한다.		무

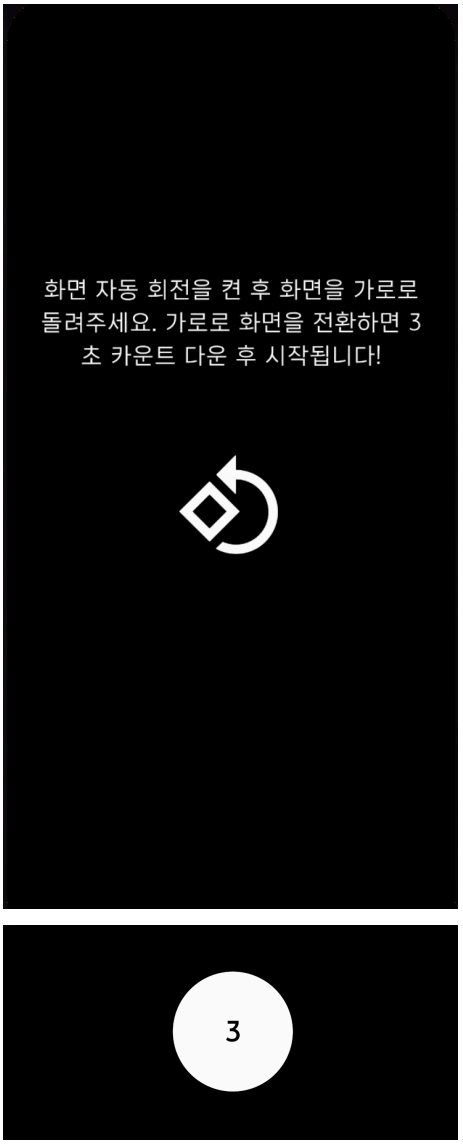
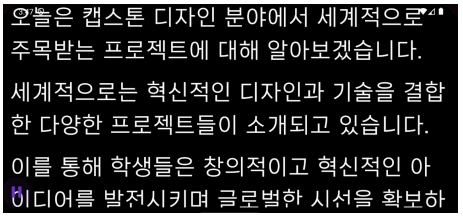
사용자 대본 내용 수정	뒤로가기	이전 페이지로 이동한다.		무
	내용 블록	앞에서 입력한 대본 내용을 수정할 수 있다.		무
	저장 후 나가기 버튼	스크립트 목록 화면으로 이동한다.		무
	연습하기 버튼	연습 방법 선택 화면으로 이동한다.		무

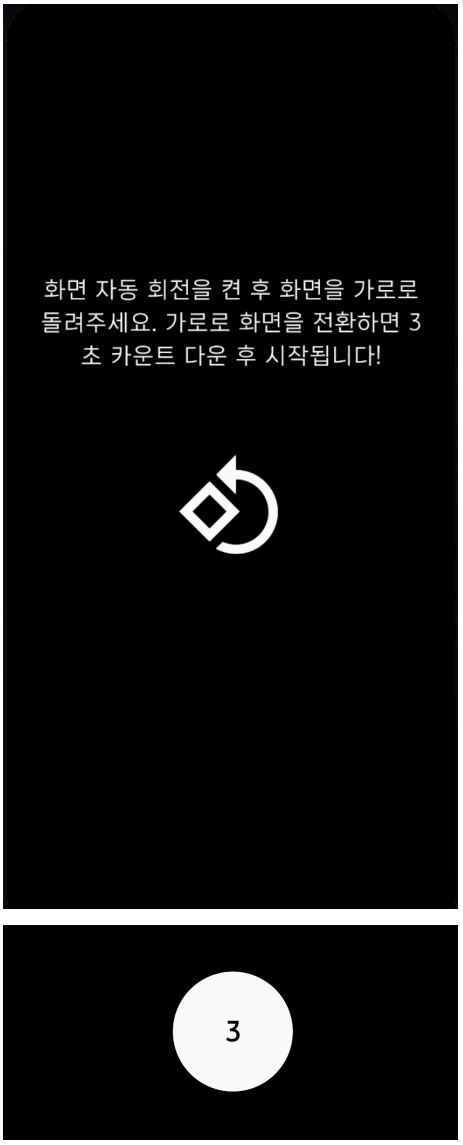
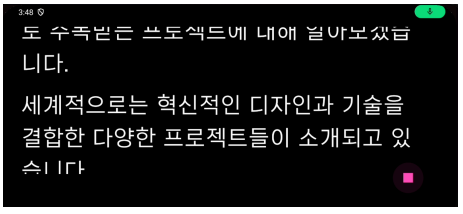
대본 상세 페이지	뒤로가기	이전 페이지로 이동한다.		무
	연습하기 버튼	연습 방법 선택 화면으로 이동한다.		무
	기록보기 버튼	기록 상세 페이지로 이동한다.		무

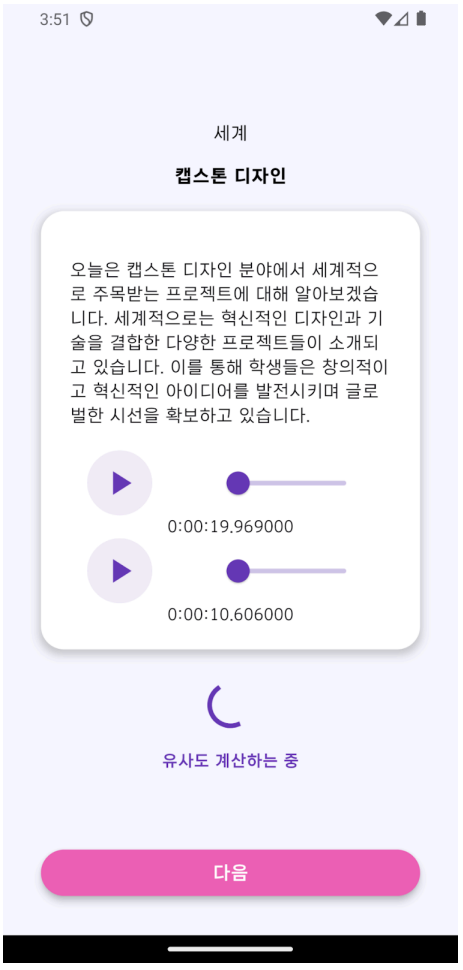
연습 방법 선택	뒤로가기	이전 페이지로 이동한다.		무
	문장단위연습 버튼	문장단위연습 화면으로 이동한다.		무
	프롬프트 버튼	프롬프트 연습 화면으로 이동한다.		무

문장단위연습	스크랩 버튼	다시 연습하고 싶은 문장을 스크랩한다.		무
	음성 가이드	음성 가이드를 들을 수 있다.		무
	연습 음성 녹음	연습 음성을 녹음한다.		무
	정확도 피드백	완료된 녹음에 대해 정확도 피드백이 나타난다.		무
	사용자 발음 표시	완료된 녹음의 발음이 표시된다.		무
	다음 버튼	다음 버튼을 클릭해 화면을 이동한다.		무

프롬프트 가이드 음성	가이드 음성 생성 로딩	해당 대본에 대한 가이드 음성을 생성한다. 완료하면 카운트다운 화면으로 전환된다.		유

	카운트 다운	만약 화면이 현재 가로 모드가 아닐 경우 가로로 화면을 전환해달라는 안내 메시지가 나온다. 가로 모드라면 3초 카운트 다운 뒤 프롬프트 연습이 시작된다.	 	무
	가이드 음성 재생	가이드 음성을 재생하며 대본이 프롬프트 화면처럼 자동으로 스크롤되어 내려간다.		무

프롬프트 연습	카운트 다운	만약 화면이 현재 가로 모드가 아닐 경우 가로로 화면을 전환해달라는 안내 메시지가 나온다. 가로 모드라면 3초 카운트 다운 뒤 프롬프트 연습이 시작된다.	 	무
	연습 음성 녹음	연습 음성을 녹음한다.		무

프롬프트 피드백	음성 가이드	음성 가이드를 들을 수 있다.		무
	연습 음성	연습 음성을 들을 수 있다.		무
	정확도 피드백	완료된 녹음에 대해 정확도 피드백이 나타난다.		무
	다음 버튼	다음 버튼을 클릭해 홈화면으로 이동한다.		무

2.3 기대효과 및 활용방안

'Loro'를 통해 사용자는 언제 어디서나 전문적인 발성 연습을 할 수 있는 환경을 제공받을 수 있습니다. 이 서비스는 비용적, 지리적 제약을 크게 줄이면서도 전문적인 피드백을 제공하여 사용자가 발성 능력을 효과적으로 개선할 수 있도록 합니다.

사용자마다 개인화된 아나운서 억양의 TTS(Text-to-Speech) 모델을 사용함으로써, 개인의 음성 특성과 목표에 맞춘 맞춤형 피드백을 받을 수 있고 전통적인 교육 방식에서 경험할 수 없는 개인적인 학습 경험을 가능하게 합니다.

3 자기평가

김필모(20191579)

저희 팀은 아나운서가 되기 위해 필요한 핵심적인 능력인 '발성'과 '프롬프트' 연습을 지원하는 AI 기반 스피치 트레이닝 어플리케이션을 개발했습니다. 이 프로젝트를 통해 저희는 실제 방송 환경을 모방한 훈련과 객관적인 피드백을 제공함으로써 사용자들이 전문 아나운서처럼 발성과 억양을 연습할 수 있도록 하였습니다.

대본 생성 기능의 평가

저희가 개발한 대본 생성 기능은 사용자가 실시간으로 원하는 스크립트를 생성할 수 있게 함으로써, 연습의 다양성과 접근성을 크게 향상시켰습니다. 이를 통해 사용자는 보다 실제와 가까운 상황에서 연습할 수 있을 것으로 기대합니다.

맞춤형 음성 가이드의 효과성

개인화된 TTS 모델을 통해 제공된 맞춤형 음성 가이드는 사용자가 자신의 발음과 억양을 보다 효과적으로 개선할 수 있도록 도왔습니다. STT 모델을 활용한 실시간 피드백은 사용자에게 구체적인 개선점을 지적해주어 학습 과정에서의 동기부여를 강화할 것으로 기대합니다.

개선점 및 미래 방향

저희 어플리케이션은 아직 몇 가지 개선해야 할 방향이 있습니다. 예를 들어, 더 다양한 언어를 지원하여 국제적인 사용자들도 이용할 수 있게 할 수 있고 데이터셋이나 모델 알고리즘 개선을 통해 더 좋은 음성 품질을 제공하도록 할 수 있습니다. 또한, UI/UX 디자인을 개선하여 더 직관적으로 보이도록 해야 합니다.

신민경 (20203090)

Loro는 실제 아나운서 학원에서 진행되는 교육의 핵심인 발성과 프롬프터에 초점을 맞추어 다양한 기능을 제공합니다. 한 문장씩 꼼꼼히 연습할 수 있는 문장단위 연습과 실전과 같은 환경의 프롬프트 연습은 아나운서 준비생이 갖추어야 할 능력을 키우는 데 도움을 줍니다. 물론 학원만큼 많은 부분을 커버하지는 못하며, 대본, 음성 가이드, 피드백 등 제공하는 기술의 퀄리티가 월등히 뛰어나진 않습니다. 그럼에도 비용 부담없이 언제 어디서든 맞춤형 연습을 할 수 있는 환경을 제공하고, 기술을 포함한 앱 전반이 무궁무진하게 발전해나갈 수 있다는 점에서 충분히 가치 있다고 생각합니다. 또한 ‘발성’을 중점으로 다양한 분야로 확장이 가능하다는 것도 상당한 장점이라고 생각합니다. 발음 교정, 말하기 훈련 등 여러 컨텐츠로 뻗어나갈 수 있으며, 다양한 연령층과 직업군에서 유용하게 사용할 수 있는 앱으로도 발전할 여지가 있습니다.

안지원 (20203095)

문제를 정의한 결과, 경제적 부담과 지리적 제약으로 인해 많은 학습자들이 전문적인 아나운서 교육에 접근하기 어려운 상황임을 확인했습니다. 이에 따라 'Loro'를 개발하고자 했습니다.

'Loro'는 사용자가 비용적 부담 없이 언제 어디서나 전문적인 발성 연습을 할 수 있는 환경을 제공합니다. 아나운서 학원의 높은 등록비와 수업료, 추가 비용 부담을 고려할 필요 없이, 'Loro'를 통해 경제적 부담을 줄일 수 있습니다.

또한, 대도시나 번화가에 위치한 아나운서 학원에 접근하기 어려운 지역 거주자들에게도 'Loro'를 통해 지리적 제약을 해소할 수 있습니다. 사용자마다 개인화된 아나운서 억양의

TTS(Text-to-Speech) 모델을 사용하여 맞춤형 피드백을 제공함으로써, 사용자가 개인의 음성 특성과 목표에 맞는 발성 교육을 받을 수 있도록 돕고 있습니다.

이러한 'Loro'의 목표 설정과 기능들은 비용적, 지리적 제약으로 인한 문제를 해결하고, 사용자들에게 편리하고 효과적인 발성 교육을 제공할 수 있다는 점에서 성공적이라고 자평합니다.

윤하은 (20203110)

이번 프로젝트의 결과물인 'Loro(로로)'는 AI를 이용해 사용자가 원하는 주제로 대본을 생성하고, 문장 단위나 전체 대본을 연습할 수 있는 기능을 제공합니다. 또한, AI가 개별 사용자의 목소리에 아나운서의 억양이 반영된 가이드 음성을 생성하여 제공해줍니다. 사용자가 연습을 진행하면, 그 음성과 가이드 음성 간의 유사도를 계산해 피드백을 제공하며, STT 기능을 통해 사용자의 발음을 확인할 수 있습니다.

이러한 프로세스는 꼭 아나운서 준비생만을 대상으로 하지 않아도, 일반인, 외국인, 아이들 등 다양한 사용자층에게도 유용하며, 사업성이 큽니다. 예를 들어, 로로는 아이들의 발음 교정을 위한 도구로도 유용할 수 있습니다. 어린 나이부터 올바른 발음 습관을 길러주는 데 도움을 주며, 부모들이 자녀의 말하기 능력을 향상시키는 데 중요한 역할을 할 수 있습니다. 이로 인해 교육 시장에서도 활용도가 높아질 것입니다.

팀원들과의 원활한 협업 덕분에 창의적인 아이디어를 실제로 구현할 수 있었으며, 포인트 제도와 같은 게임 요소를 도입하여 사용자 참여를 높일 수 있는 가능성을 발견했습니다. 이러한 점들을 종합적으로 평가해볼 때, 로로는 지속적으로 개선하고 확장할 잠재력이 있는 애플리케이션입니다.

4 참고 문헌

번호	종류	제목	출처	발행년도	저자	기타
1	논문	Conditional Variational Autoencoder with Adversarial Learning for End-to-End Text-to-Speech	arxiv: 2106.06103	11 June 2021	Jaehyeon Kim, Jungil Kong, Juhee Son, et al	VITS
2	논문	“Matcha-TTS: A fast TTS architecture with conditional flow matching”	arxiv: 2309.03199	6 Sep 2023	Shivam Mehta, Ruibo Tu, et al	
3	논문	“OverFlow: Putting flows on top of neural transducers for better TTS”	arxiv: 2211.06892	13 Nov 2022	Shivam Mehta, Ambika Kirkland, et al	
4	논문	“VITS2: Improving Quality and Efficiency of Single-Stage Text-to-Speech with Adversarial Learning and Architecture Design”	arxiv: 2307.16430	31 July 2023	Jungil Kong, Jihoon Park, Beomjeong Kim, et al	VITS2
5	논문	“Robust Speech Recognition via Large-Scale Weak Supervision”	arXiv: 2212.04356	6 Dec 2022	Alec Radford, Jong Wook Kim, et al	
6	논문	“Encoding Speaker-Specific Latent Speech Feature for Speech Synthesis”	arXiv: 2311.11745 l	31 July 2023	Jungil Kong, Junmo Lee, et al	SFEN
7	논문	“Sound Design Strategies for Latent Audio Space Explorations Using Deep Learning Architectures”	arXiv: 2305.15571	24 May 2023	Kıvanç Tatar, Kelsey Cotton, Daniel Bisig	
8	기술 문서	“Just-in-time compilation”	https://jax.readthedocs.io/en/latest/	1 May 2023	Jake Vanderplas	

			jit-compilation.html			
9	기술 문서	“FastAPI, Request Files”	https://fastapi.tiangolo.com/tutorial/request-files/	12 July 2022	Hasan Sezer Taşan	
10	기술 문서	“eSpeak NG user guide”	https://github.com/espeak-ng/espeak-ng/blob/master/docs/guide.md	22, July 2020	Valdis Vitolins	

5 부록

5.1 사용자 매뉴얼

설치 가이드

Loro가 구글 플레이스토어에 배포되었음을 가정하고 설치 방법을 작성했습니다.

1. 구글 플레이 스토어에 접속해 Loro를 검색합니다.
2. 아래와 같은 로고를 가진 Loro 애플리케이션을 다운로드 받습니다.

3. 설치 후 애플리케이션 목록에서 확인하실 수 있습니다.

사용 가이드

Loro를 사용하기 위한 메뉴얼입니다. 사용자가 직접 생성한 대본으로 프롬프트 연습을 진행한다고 가정하고 시나리오를 작성했습니다.

로로를 본격적으로 사용하기 앞서 먼저 구글 로그인을 통해 계정을 등록합니다.

구글 로그인

1. 앱 설치 후 처음 실행하면 화면 중앙에 [Google 계정으로 로그인] 버튼이 표시됩니다. 해당 버튼을 클릭합니다.
2. 구글 계정이 등록되지 않은 경우, 구글 로그인 API를 통해 회원가입이 진행됩니다.
3. 계정이 이미 등록되어 있지만 사용자 정보가 저장되지 않은 경우, 닉네임/캐릭터 설정 페이지로 이동합니다.
4. 계정과 사용자 정보가 모두 등록된 경우, 홈 화면으로 이동합니다.
5. 구글 계정 등록이 이미 완료된 경우, 처음 실행이 아니라면 바로 2 또는 3단계로 이동합니다.

그 후, 닉네임을 입력하고 캐릭터를 설정합니다.

닉네임 설정

1. [닉네임을 입력해주세요.] 창을 클릭하면 키보드가 활성화됩니다.
2. 원하는 닉네임을 입력합니다.
    - 닉네임이 8자를 초과할 경우 자동으로 입력이 제한됩니다.
    - 특수문자는 입력할 수 없습니다.

캐릭터 설정

1. 제공된 4가지 캐릭터 중 원하는 캐릭터를 클릭합니다.
2. 중앙 박스의 캐릭터가 선택한 캐릭터로 변경됩니다.

그후, 사용자 맞춤 음성을 위해 음성 정보를 제공해야 합니다.

사용자 음성 정보 수집

사용자 음성 정보는 총 3번에 걸쳐 진행됩니다.

1. [마이크] 아이콘을 클릭하고 제시된 지문을 따라 읽습니다.
2. [재생] 또는 [중지] 아이콘을 눌러 녹음을 재생하거나 중지합니다.
3. [완료] 아이콘을 눌러 녹음을 완료하면, 녹음된 음성을 재생할 수 있습니다.
4. 녹음된 음성 우측의 [휴지통] 아이콘을 눌러 현재 녹음을 삭제할 수 있습니다. 삭제 시 다시 처음 단계로 돌아갑니다.
5. 음성 녹음이 완료된 경우, 다음 화면으로 이동합니다.

    5.1. 음성 녹음이 완료되지 않은 경우, 녹음을 완료하라는 안내 메시지가 표시되고 화면이 이동하지 않습니다.

    5.2. 세 번째 음성 녹음까지 완료되면, 수집된 음성을 저장한다는 안내 메시지와 함께 자동으로 홈화면으로 이동합니다.

이제 사용자 대본을 생성해보겠습니다. 사용자 대본 생성 페이지으로 이동하는 방법은 다음과 같습니다.

1. 하단 네비게이션 바의 우측에 있는 [스크립트]를 클릭해 대본 목록 화면으로 이동합니다.
2. 상단의 User 탭을 클릭합니다.
3. 네비게이션 바 위의 [나만의 대본 만들기] 버튼을 클릭해 대본 생성 페이지로 이동합니다.

대본 생성

대본 생성은 대본 구성요소(제목/카테고리/내용) 입력과 대본 내용 수정 두 단계에 걸쳐 진행됩니다.

먼저 대본 구성요소 입력 단계입니다.

1. [제목을 입력하세요.] 창을 클릭해 키보드가 활성화되면 제목을 입력합니다.
2. 중앙의 카테고리 목록 (전체, 정치, 경제, 사회, 스포츠, 생활/문화, IT/과학, 세계) 중 하나를 선택합니다.
3. [내용을 직접 입력하거나 AI로 생성해보세요.] 창을 클릭해 키보드가 활성화되면 내용을 입력합니다.

   3.1. 내용을 직접 입력하지 않아도 AI를 통해 생성할 수 있습니다. 좌측 하단의 [AI로 생성하기] 버튼을 눌러 자동으로 내용을 만들어보세요.
4. 모든 항목이 채워지면, 우측 하단의 [완료] 버튼을 눌러 대본 내용 수정 페이지로 이동합니다.

   4.1. 대본 구성요소(제목, 카테고리, 내용)가 하나라도 비어있다면 안내 메시지가 나타납니다.

다음으로 대본 내용 수정 단계입니다. 이 단계에서는 앞에서 입력한 대본의 내용을 수정하고, 대본 저장 후 연습 화면으로 이동할 수 있습니다.

1. 블록을 눌러 활성화되는 키보드로 텍스트를 수정할 수 있습니다.
2. 블록 우측의 [-] 버튼을 눌러 해당 블록을 삭제하거나 마지막 블록 하단의 [+] 버튼을 눌러 블록을 추가할 수 있습니다.
3. [저장 후 나가기] 버튼을 통해 대본 목록으로 이동하거나 [연습하기] 버튼을 눌러 연습을 바로 진행할 수 있습니다. 이 시나리오에서는 [연습하기] - [프롬프트]를 눌러 프롬프트 연습으로 이동해주세요.

   3.1. 내용이 비어있다면, 안내 메시지가 나타납니다.

프롬프트 연습은 [가이드 음성 듣기]와 [연습하기] 중 선택할 수 있습니다. [가이드 음성]을 선택할 경우, Loro에서 제공하는 사용자 맞춤형 음성 가이드가 제공됩니다. [가이드 음성 듣기]를 클릭하면, 카운트 다운 후 가이드 음성을 듣고 다시 카운트 다운 후 프롬프트 화면을 보며 연습을 하는 방식으로 진행됩니다. 연습 후 사용자는 프롬프트에 대한 정확도 피드백을 받을 수 있습니다.

카운트 다운

1. 화면이 현재 가로 모드가 아닌 경우 가로로 화면을 전환해달라는 안내 메시지가 표시됩니다.
2. 가로 모드로 전환되면 3초 카운트 다운 후 프롬프트 연습이 시작됩니다.

가이드 음성을 재생하면 대본이 실제 프롬프터처럼 자동으로 스크롤됩니다.

가이드 음성 재생

1. [재생] 아이콘을 눌러 가이드 음성을 재생합니다.
2. [일시 중지] 아이콘을 눌러 음성을 일시 중지할 수 있으며, [연습하기] - [프롬프트] 버튼을 클릭했을 때와 동일한 dialog가 나타납니다.

   2.1. [가이드 음성 듣기] 버튼을 클릭하면 가이드 음성을 다시 들을 수 있고, [연습하기] 버튼을 클릭하면 프롬프트 연습 화면으로 이동합니다.

프롬프트 연습하기

1. [마이크] 아이콘을 누르고 제시된 지문을 따라 읽습니다.
2. [재생] 또는 [중지] 아이콘을 눌러 녹음을 재생하거나 중지할 수 있습니다.
3. [완료] 아이콘을 눌러 녹음을 완료화면, 녹음된 음성이 나옵니다.
4. [휴지통] 아이콘을 눌러 현재 녹음 삭제할 수 있고, 삭제 시 처음 단계로 돌아갑니다.
5. [완료] 버튼을 눌러 프롬프트 피드백 화면으로 이동한다.

프롬프트 피드백 화면에서는 완료된 녹음에 대한 정확도 피드백이 나타납니다. 정확도 점수는 사용자의 발음과 억양을 분석해 제공됩니다. 또한 점수와 함께 사용자가 발음한 어떻게 발음했는 지 텍스트로 함께 확인할 수 있습니다.

5.2 운영자 매뉴얼

Loro API 서버 운영 가이드

Loro의 API 서버를 운영하기 위해 필요한 절차와 관리 방법에 대해 설명하겠습니다.

배포 가이드에 따라 서버를 성공적으로 배포한 후, 서버를 안정적으로 운영하기 위한 지침입니다. 또 ec2 인스턴스를 사용할 경우 생길 수 있는 문제도 다뤘습니다

시스템 모니터링을 위해서는 다음의 명령어를 사용합니다.

```
# 시스템의 CPU, 메모리, 스왑 사용량, 개별 프로세스에 대한 상세 정보를 실시간으로 제공합니다.
htop


# nvidia 드라이버가 설치된 환경에서 드라이버 정보를 보여줍니다.
nvidia-smi


# 시스템의 메모리 사용량을 보여줍니다.
free -h
```

사용자가 많아진다면 아무리 모델 추론 시 아무리 음성파일을 임시로 만든다 할 지라도 디스크 사용량이 커질 수 밖에 없습니다. 디스크 사용량을 확인 할 필요가 있습니다.

```
# 파일 시스템의 전체 디스크 사용량 확인
df -h

# 특정 디렉토리의 디스크 사용량 확인
du -h /var/log
```

Nginx 재시작:

설정 파일 변경이나 에러로 인한 재시작이 필요할 경우 다음 명령어를 실행합니다.

```
sudo systemctl restart nginx
```

TTS 모델의 MAS 재빌드

흔하지 않지만 tts 모델의 duration predictor의 알고리즘에 문제가 있다고 판단하면 수정 후 MAS를 재빌드 하기 위해선 setup.py를 다시 실행하면 됩니다.

```
cd ~/capstone-2024-08/backend/tts/monotonic_align
python setup.py build_ext --inplace
```

문제 해결

서버 접속 시 문제가 생긴다면 인스턴스를 재부팅 하는 방법이 있습니다. 주로 nvidia driver에 문제가 있을 경우 사용합니다.

```
sudo reboot
```

또 보안그룹의 문제가 있을 수 있습니다.EC2 인스턴스의 보안 그룹 설정에서 http의 경우 포트 80, https의 경우 443이 열려 있는지 확인합니다. 또 중요한 것은 nginx에 리버스 프록시 설정이 이에 맞춰 되어 있는 지도 확인해야 합니다.

Loro는 스토리지 사용의 상당히 많은 비율이 모델을 가져오는 데 사용되고 jax의 코드레벨 캐싱을 하는데 사용됩니다. 혹시나 jax 코드를 수정하여 배포할 경우 디스크 공간이 부족할 수 있습니다. 최선의 방법은 스토리지를 늘리는 것이지만 대안으로 불필요한 파일들을 삭제하거나 pip 캐싱을 해둔 것들을 삭제할 수 있습니다.

```
pip cache purge
sudo apt-get clean
```

다음으로는 ec2 인스턴스를 운영하며 생길 수 있는 이슈들과 그 해결법에 대해 설명하겠습니다.

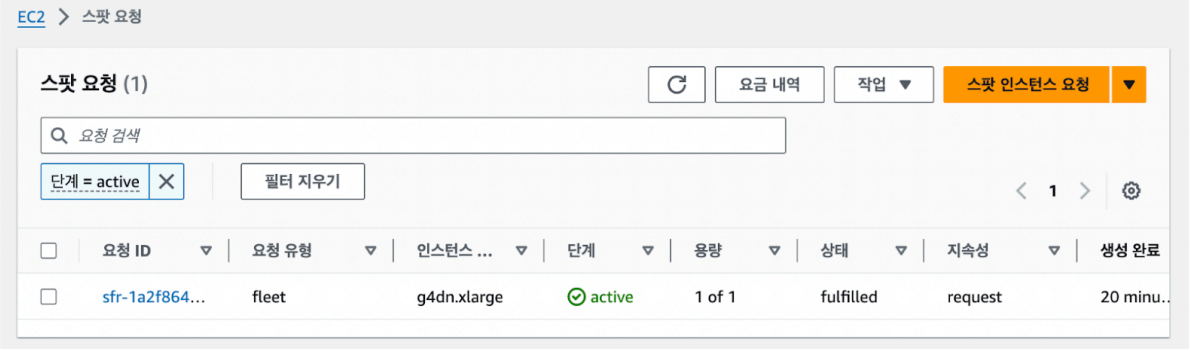
만약 gpu 인스턴스처럼 고비용의 인스턴스를 계속 켜두는 것이 부담스럽다면 스팟 인스턴스를 사용하는 방법이 있습니다. 물론 서비스할 때는 스팟 인스턴스는 삭제 될 위험이 있어 적합하지 않지만 개발 테스트 서버를 운영한다면 낮은 비용으로 테스트 할 수 있는 면에서 좋습니다. 다음은 저희 팀이 테스트 서버를 운영하기 위해 스팟 인스턴스를 사용하며 적은 이슈입니다.

스팟 인스턴스 사용

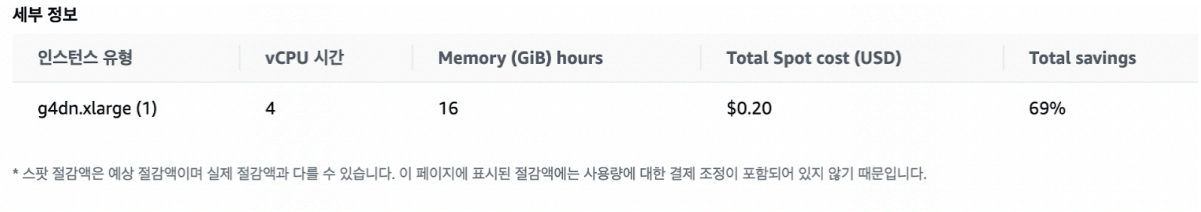
TTS 모델을 사용할 때 저희 프로젝트 특성 상 학습을 미리 시켜두고 추론을 중심으로 인스턴스 서버를 사용하기 때문에 스팟 인스턴스를 사용하기 적합하고 스팟 인스턴스 사용 시 기존보다 약 69% 비용이 절감될 것으로 확인되어 스팟 인스턴스를 사용하고자 합니다.

저희가 모델 추론을 위해 사용하는 인스턴스의 사양은 `**g4dn.xlarge**` 입니다.

스팟 인스턴스 요청



스팟 인스턴스 요청 후 온디맨드 대비 스팟 인스턴스 사용 비용 절감 결과



세부 정보

인스턴스 유형	vCPU 시간	Memory (GiB) hours	Total Spot cost (USD)	Total savings
g4dn.xlarge (1)	4	16	$0.20	69%

* 스팟 절감액은 예상 절감액이며 실제 절감액과 다를 수 있습니다. 이 페이지에 표시된 절감액에는 사용량에 대한 결제 조정이 포함되어 있지 않기 때문입니다.

5.3 배포 가이드

API 서버 배포 가이드

Loro의 api 서버를 배포하기 위해 필요한 소프트웨어 및 도구와 시스템의 요구 사항에 대해 설명하겠습니다.

서버를 위한 시스템은 다음과 같아야 합니다.

시스템 요구사항:

1. Ubuntu 20.04 이상의 nvidia gpu가 내장된 인스턴스
2. 최소 16GB RAM, 30GB이상의 디스크 공간
3. 최소 1 개 이상의 gpu, 4 개 이상의 cpu코어

먼저 loro의 시스템 설정을 위한 필수 패키지 설치합니다.

```
sudo apt update
sudo apt install python3-venv
sudo apt install espeak
sudo apt install ffmpeg
```

```
udo apt install nginx -y
```

이 가이드에서는 운영자가 ec2 인스턴스에서 (http 기준 80번 포트, https 기준 443 포트로 설정하여 ASGI서버를 실행하며 security group 설정을 통해 연결하고자 하는 클라이언트에 맞춰 inbound 규칙을 설정했다고 가정 후 진행하겠습니다.

위와 같은 가정 하에서는 리버스 프록시를 통한 포트 전달이 필요합니다. 이를 위해 설정 파일을 수정합니다.

nginx의 주 설정파일은 **/etc/nginx/nginx.conf**인데 이 설정파일에 include 문으로 포함된 **/etc/nginx/sites-enabled/**을 수정하여 변경하겠습니다.

vim 편집기를 이용해 파일을 만듭니다.

```
sudo vim /etc/nginx/sites-available/loro
```

위 파일에 다음과 같이 작성합니다.

```
server {
    listen 80;
    server_name {ip addr};
    location / {
        proxy_pass http://127.0.0.1:8000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

그 후 /etc/nginx/sites-available/loro 에 심볼릭 링크 걸어 줍니다. 이렇게 하는 이유는 여러 사이트 관리를 위한 모듈화를 하기 위함입니다.

```
sudo ln -s /etc/nginx/sites-available/loro /etc/nginx/sites-enabled/
```

그 후 설정 적용을 위한 nginx를 재시작 합니다.

```
sudo systemctl restart nginx
```

다음으로 Loro api 서버를 실행하는 방법에 대해 설명하겠습니다. python 가상환경에 모든 의존성을 설치하는 것 가정하겠습니다.

home dir 에 다음 명령어를 입력해 가상환경을 만들고 활성화 하겠습니다.

```
python3 -m venv loro
source ~/loro/bin/activate
```

api 서버는 github를 이용해 버전관리를 하고 있습니다. 아래 명령어로 프로젝트를 받고 의존성 라이브러리를 설치할 수 있습니다.

```
git clone https://github.com/kookmin-sw/capstone-2024-08.git
cd capstone-2024-08/backend
pip install -r requirements.txt
```

이번에는 TTS model을 위한 초기 build를 하는 과정을 설명하겠습니다. vits 모델은 음성과 텍스트간 alignment을 위해서 MAS(monotonic alignment search)를 사용합니다. 이를 사용하기 위해서 다음의 명령어를 입력합니다

우선 저희 프로젝트의 “capstone-2024-08/backend/tts/monotonic_align” 으로 이동 후 작업을 진행한다고 가정 후 설명하겠습니다.

```
pip install Cython
mkdir monotonic_align
sudo apt-get install build-essential
sudo apt-get install python3-dev
python setup.py build_ext --inplace
```

대본 생성을 위한 openai api 를 등록하기 위해 다음 작업을 수행합니다.

```
mkdir .config_secret
vim setting_local.json
```

이 파일에 api key를 넣어 줍니다. (key 내용은 공개하지 않겠습니다.)

```
{
    "openai_api_key" : "{key 입력}"
}
```

추가로 이 인스턴스에 nvidia driver를 설치하는 과정을 설명하겠습니다.

사용 가능한 NVIDIA 드라이버 버전을 확인을 합니다.

```
apt search nvidia-driver
```

저는 nvidia-driver-535 를 사용하겠습니다.

```
sudo apt install nvidia-driver-535
```

```
sudo reboot
```

올바르게 설치되었다면 다음 명령어에 gpu 설정에 관한 정보가 나옵니다.

```
nvidia-smi
```

stt 모델인 whisper-medium을 효율적으로 사용하기 위해서 jax를 사용합니다.

```
pip install -U "jax[cuda12]"
pip install torchaudio
pip install git+https://github.com/sanchit-gandhi/whisper-jax.git
```

s3에 저장된 TTS 모델 가져오기

직접 학습한 tts 모델이 s3에 저장되었다고 가정하고 설명하겠습니다.

위치를 /backend/tts로 옮긴 후 다음을 실행합니다.

```
pip install awscli
sudo apt-get install espeak-ng
aws configure  # AWS 설정하기 각자 다르므로 구체적인 정보는 설명하지 않겠습니다.
aws s3 cp s3://{s3 버킷 위치}  ./
```

api 서버 실행

서버는 다른 추가적인 명령어 없이 main.py를 실행시키면 동작하도록 구현되어 있습니다. 따라서 다음 명령을 통해 서버를 실행시킬 수 있습니다.

```
python3 main.py
```

5.4 테스트 케이스

대분류	소분류	기능	테스트 방법	기대 결과	테스트 결과
회원가입/로그인	구글 로그인	구글 로그인을 통해 계정을 등록한다.	앱 설치 후 최초 접속 시 화면 중앙의 [Google 계정으로 로그인] 버튼을 누르면, 1) 계정 미등록 시, 구글 로그인 API를 통한 회원가입 진행한다. 2) 계정이 등록되어 있으면서 사용자 정보가 저장되어있지 않을 시, 닉네임/캐릭터 설정 페이지로 이동한다. 3) 계정과 사용자 정보 모두 등록되어있을 시, 홈화면으로 이동한다. 1을 이미 수행한 상황에서 최초 접속이 아니라면 2 또는 3으로 곧장 이동한다.	구글 계정이 등록되며, 회원가입 진행 상황에 따라 자동으로 화면이 이동한다.	성공
	닉네임 설정	닉네임을 입력한다.	[닉네임을 입력해주세요.] 창을 클릭해 키보드가 활성화되면, 닉네임을 입력한다. ● 8자가 초과될 경우 자동으로 입력이 제한된다. ● 특수문자 입력은 허용되지 않는다.	입력한 닉네임이 저장된다.	성공
	캐릭터 설정	캐릭터를 선택한다.	제공된 4가지 캐릭터 중 원하는 캐릭터를 클릭하면, 중앙 박스의 캐릭터가 해당 캐릭터로 변경된다.	선택한 캐릭터가 저장된다.	성공
	사용자 음성정보 수집	음성을 녹음한다.	1) [마이크] 아이콘을 누르고 제시된 지문을 따라 읽는다. 2) [재생] 또는 [중지] 아이콘을 눌러 녹음을 재생 및 중지할 수 있다. 3) [중지] 아이콘을 눌러 녹음을 완료화면, 녹음된 음성이 나온다. 재생바를 움직여 원하는 구간을 확인할 수 있다.	음성을 녹음하고 유저 디비에 사용자 음성을 저장한다.	성공

			4) [휴지통] 아이콘을 눌러 현재 녹음 삭제할 수 있고, 삭제 시 1로 돌아간다.		
		다음 버튼을 통한 화면 이동	하단의 [다음] 버튼을 누르면, 1) 음성 녹음 완료 시, 다음 화면으로 이동한다. 2) 음성 녹음 미완료 시, 녹음을 완료하라는 안내 메시지가 뜨고 화면이 넘어가지 않는다. 3) 세번째 음성 녹음까지 완료하면, 수집된 음성을 저장한다는 안내 메시지 등장 후 자동으로 홈화면으로 이동한다.	녹음 완료 여부에 따라 안내 메시지가 나타나거나 화면을 이동한다.	성공
네비게이션 바	네비게이션 바	네비게이션 바를 클릭해 화면을 이동한다.	좌측의 [기록] 버튼 클릭 시, 기록 목록 화면으로 이동한다. 중앙의 [홈] 버튼 클릭 시, 홈 화면으로 이동한다. 우측의 [스크립트] 버튼 클릭 시, 스크립트 목록 화면으로 이동한다.	네비게이션 바를 클릭해 원하는 화면으로 이동한다.	성공
홈	히스토리	마지막으로 연습한 대본을 확인한다.	대본을 클릭하면, 해당 대본 상세 화면으로 이동한다.	마지막으로 연습한 대본으로 이동한다.	성공
	설정 아이콘 버튼	설정 화면으로 이동한다.	우측 상단의 [톱니바퀴] 아이콘을 클릭하면, 설정 화면으로 이동한다.	설정 화면으로 이동한다.	성공
설정	로그아웃	서비스에서 로그아웃한다.	[로그아웃] 버튼을 누르면, dialog가 나타난다. 1) [확인] 버튼을 누르면, 로그아웃 후 회원가입/ 로그인 화면으로 이동한다. 2) [취소] 버튼을 누르면, dialog가 사라진다.	서비스에서 로그아웃하고 로그인 화면으로 이동한다.	성공
	탈퇴하기	서비스에서 탈퇴한다.	[탈퇴] 버튼을 누르면, dialog가 나타난다.	서비스에서 탈퇴하고 로그인 화면으로 이동한다.	성공

			3) [확인] 버튼을 누르면, 계정 삭제 후 회원가입/ 로그인 화면으로 이동한다. 4) [취소] 버튼을 누르면, dialog가 사라진다.		
기록 목록	News/User 탭바	탭을 클릭해 대본 종류를 전환한다.	좌측 상단의 [News] 탭을 클릭하면, 연습 이력이 있는 예시 대본 목록이 나타난다. [News] 탭 우측의 [User] 탭을 클릭하면, 연습 이력이 있는 사용자 대본 목록이 나타난다.	선택한 탭에 따라 대본 종류가 전환된다.	성공
	카테고리	카테고리를 선택해 대본을 필터링한다.	News/User 탭 하단의 카테고리 목록 (전체, 정치, 경제, 사회, 스포츠, 생활/문화, IT/과학, 세계) 중 하나를 선택하면, 해당 카테고리의 대본만 나타난다. ● 카테고리 목록은 좌우로 스크롤할 수 있다.	선택한 카테고리의 대본만 나타난다.	성공
	대본 목록	기록 상세 페이지로 이동한다.	대본 목록 중 원하는 대본을 클릭하면, 해당 대본의 기록 상세 페이지로 이동한다.	선택한 대본의 기록 상세 페이지로 이동한다.	성공
기록 상세 페이지	뒤로가기	이전 페이지로 이동한다.	좌측 상단의 [왼쪽 화살표] 아이콘을 클릭하면, 이전 화면(기록 목록 화면 또는 대본 상세 페이지)으로 돌아간다.	이전 화면으로 돌아간다.	성공
	스크랩한 문장 목록	스크랩한 문장 목록을 확인한다.	‘스크랩한 문장 목록’ 하단의 하얀 박스를 좌우로 슬라이드해 문장 단위 연습에서 스크랩한 문장들을 확인한다.	스크랩한 문장 목록을 확인한다.	성공
	프롬프트 정확도 추이 그래프	프롬프트 연습 기록을 확인한다.	‘프롬프트 정확도 추이 그래프’ 하단 하얀 박스 내 그래프의 각 점을 클릭하면, 해당 연습 시각과 점수가 나타난다.	프롬프트 연습 시각과 정확도를 확인한다.	성공
	다시 연습하기 버튼	다시 연습하기를 클릭해	하단의 [다시 연습하기] 버튼을 클릭하면, 연습 방법 선택 화면으로 이동한다.	연습 방법 선택 화면으로 이동한다.	성공

		연습 방법 선택 화면으로 이동한다.			
스크립트 목록	News/User 탭바	탭을 클릭해 대본 종류를 전환한다.	좌측 상단의 [News] 탭을 클릭하면, 예시 대본 목록이 나타난다. [News] 탭 우측의 [User] 탭을 클릭하면, 사용자 대본 목록이 나타난다.	선택한 탭에 따라 대본 종류가 전환된다.	성공
	카테고리	카테고리를 선택해 대본을 필터링한다.	News/User 탭 하단의 카테고리 목록 (전체, 정치, 경제, 사회, 스포츠, 생활/문화, IT/과학, 세계) 중 하나를 선택하면, 해당 카테고리의 대본만 나타난다. ● 카테고리 목록은 좌우로 스크롤할 수 있다.	선택한 카테고리의 대본만 나타난다.	성공
	대본 목록	대본 상세 페이지로 이동한다.	대본 목록 중 원하는 대본을 클릭하면, 해당 대본의 상세 페이지로 이동한다.	선택한 대본의 상세 페이지로 이동한다.	성공
	검색 아이콘 버튼	검색 화면으로 이동한다.	우측 상단의 [돋보기] 아이콘을 클릭하면, 검색 화면으로 이동한다.	검색 화면으로 이동한다.	성공
	나만의 대본 만들기 버튼	대본 생성 화면으로 이동한다.	User 탭의 대본 목록 하단에 고정된 [나만의 대본 만들기] 버튼을 클릭하면, 대본 생성 화면으로 이동한다.	대본 생성 화면으로 이동한다.	성공
검색	뒤로가기	이전 페이지로 이동한다.	좌측 상단의 [왼쪽 화살표] 아이콘을 클릭하면, 이전 화면(스크립트 목록)으로 돌아간다.	스크립트 목록 화면으로 돌아간다.	성공
	검색 키워드 입력	검색어를 입력한다.	[검색할 키워드를 입력해주세요.] 창을 클릭해 키보드가 활성화되면, 검색할 키워드를 입력한다. 1) 검색어가 제목에 포함된 대본이 있다면 즉시 대본이 나타난다. 2) 입력창이 비어있거나 검색어가	입력한 검색어를 제목에 포함하는 대본 목록이 나타난다.	성공

			제목에 포함되는 대본이 없다면 대본 목록에 아무것도 나타나지 않는다.		
	News/User 탭바	탭을 클릭해 대본 종류를 전환한다.	좌측 상단의 [News] 탭을 클릭하면, 검색어를 제목에 포함하고 있는 예시 대본 목록이 나타난다. [News] 탭 우측의 [User] 탭을 클릭하면, 검색어를 제목에 포함하고 있는 사용자 대본 목록이 나타난다.	선택한 탭에 따라 대본 종류가 전환된다.	성공
	대본 목록	대본 상세 페이지로 이동한다.	대본 목록 중 원하는 대본을 클릭하면, 해당 대본의 상세 페이지로 이동한다.	선택한 대본의 상세 페이지로 이동한다.	성공
사용자 대본 생성	뒤로가기	이전 페이지로 이동한다.	좌측 상단의 [왼쪽 화살표] 아이콘을 클릭하면, 이전 화면(스크립트 목록)으로 돌아간다.	스크립트 목록 화면으로 돌아간다.	성공
	대본 구성요소 입력	대본의 제목을 입력한다.	[제목을 입력하세요.] 창을 클릭해 키보드가 활성화되면, 제목을 입력한다.	입력한 제목이 저장된다.	성공
		대본의 카테고리를 지정한다.	중앙의 카테고리 목록 (전체, 정치, 경제, 사회, 스포츠, 생활/문화, IT/과학, 세계) 중 하나를 선택한다.	선택한 카테고리가 저장된다.	성공
		대본의 내용을 입력한다.	[내용을 직접 입력하거나 AI로 생성해보세요.] 창을 클릭해 키보드가 활성화되면, 내용을 입력한다.	입력한 내용이 저장된다.	성공
	AI로 생성하기 버튼	AI로 대본 내용을 생성한다.	좌측 하단의 [AI로 생성하기] 버튼을 클릭하면, 1) 내용 입력창에 'AI로 대본을 생성하고 있습니다. 잠시만 기다려주세요.'라는 문구가 나타난다. 2) 몇 초 후 문구가 사라지며 AI가 생성한 대본 내용이 내용 입력창에 나타난다.	AI로 생성한 내용이 자동으로 대본 내용 입력창에 나타난다.	성공
	완료 버튼	사용자 대본 내용 수정	우측 하단의 [완료] 버튼을 클릭하면, 1) 대본 구성요소(제목, 카테고리,	대본 구성요소 전부 유효한 값이 있는지에 따라 안내 메시지가	성공

		페이지로 이동한다.	내용)가 하나라도 비어있다면, 안내 메시지가 나타난다. 2) 대본 구성요소가 모두 존재한다면, 사용자 대본 내용 수정 페이지로 이동한다.	나타나거나 사용자 대본 내용 수정 페이지로 이동한다.	
사용자 대본 내용 수정	뒤로가기	이전 페이지로 이동한다.	좌측 상단의 [왼쪽 화살표] 아이콘을 클릭하면, 이전 화면(사용자 대본 생성)으로 돌아간다.	사용자 대본 생성 화면으로 돌아간다.	성공
	내용 블록	앞에서 입력한 대본 내용을 수정할 수 있다.	앞의 대본 생성 페이지에서 입력한 내용을 문장 단위로 나누어 표시한다. • 각 문장 블록을 클릭하면, 키보드가 활성화되어 내용을 수정할 수 있다. • 문장 블록 우측의 [-] 아이콘을 클릭해 블록을 삭제할 수 있다. • 마지막 문장 블록 아래 [+] 아이콘을 클릭해 새 블록을 추가할 수 있다.	앞에서 입력한 대본 내용을 수정/삭제/추가할 수 있다.	성공
	저장 후 나가기 버튼	스크립트 목록 화면으로 이동한다.	좌측 하단의 [저장 후 나가기] 버튼을 클릭하면, 1) 내용이 아예 비어있거나 빈 내용 블록이 있다면, 안내 메시지가 나타난다. 2) 모든 블록에 내용이 있다면, 스크립트 목록 화면으로 이동한다. 방금 생성한 대본을 확인할 수 있다.	내용 유무에 따라 안내 메시지가 나타나거나 스크립트 목록 화면으로 이동한다.	성공
	연습하기 버튼	연습 방법 선택 화면으로 이동한다.	우측 하단의 [연습하기] 버튼을 클릭하면, 1) 내용이 아예 비어있거나 빈 내용 블록이 있다면, 안내 메시지가 나타난다. 2) 모든 블록에 내용이 있다면, 연습 방법 선택 화면으로 이동한다.	내용 유무에 따라 안내 메시지가 나타나거나 연습 방법 선택 화면으로 이동한다.	성공

대본 상세 페이지	뒤로가기	이전 페이지로 이동한다.	좌측 상단의 [왼쪽 화살표] 아이콘을 클릭하면, 이전 화면으로 돌아간다.	이전 화면으로 돌아간다.	성공
	연습하기 버튼	연습 방법 선택 화면으로 이동한다.	하단의 [연습하기] 버튼을 클릭하면, 연습 방법 선택 화면으로 이동한다.	연습 방법 선택 화면으로 이동한다.	성공
	기록보기 버튼	기록 상세 페이지로 이동한다.	해당 대본을 연습한 이력이 있을 때만, [기록보기] 버튼이 나타난다. [기록보기] 버튼을 클릭하면, 기록 상세 페이지로 이동한다.	해당 대본의 기록 상세 페이지로 이동한다.	성공
연습 방법 선택	뒤로가기	이전 페이지로 이동한다.	우측 상단의 [X] 아이콘을 클릭하면, 이전 화면으로 돌아간다. 이전 화면이 사용자 대본 내용 수정 페이지라면, 스크립트 목록 화면으로 이동한다.	이전 화면으로 돌아간다. (이전 화면이 사용자 대본 내용 수정 페이지라면, 스크립트 목록 화면으로 이동한다.)	성공
	문장단위연습 버튼	문장단위연습 화면으로 이동한다.	중앙의 [문장단위연습] 버튼을 클릭하면, 문장단위연습 화면으로 이동한다.	문장단위연습 화면으로 이동한다.	성공
	프롬프트 버튼	프롬프트 연습 화면으로 이동한다.	중앙의 [프롬프트] 버튼을 클릭하면, dialog가 나타난다. 1) [가이드 음성 듣기] 버튼을 클릭하면, 우선 가이드 음성을 생성하는 동안 로딩화면이 나타난다. 가이드 음성을 생성 완료한 후에는 프롬프트 가이드 음성 화면으로 이동한다. 2) [연습하기] 버튼을 클릭 시, 해당 대본에 대한 가이드 음성을 생성하지 않았다면 가이드 음성을 생성하는 동안 로딩 화면이 나타나고, 이미 가이드 음성이 있을 시 바로 프롬프트 연습 화면으로 이동한다.	1) 가이드 음성을 생성 완료한 후에는 프롬프트 가이드 음성 화면으로 넘어간다. 2) 해당 대본에 대한 가이드 음성을 생성하지 않았다면 가이드 음성을 생성하는 동안 로딩 화면이 나타나고, 이미 가이드 음성이 있을 시 바로 프롬프트 연습 화면으로 이동한다.	성공

문장단위연습	스크랩 버튼	다시 연습하고 싶은 문장을 스크랩한다	중앙의 하얀 박스 좌측 상단의 [책갈피] 아이콘을 클릭하여 스크랩을 할 수 있다. 1) 스크랩이 되어있지 않았다면, 색상이 채워지고 기록 상세 페이지의 스크랩한 문장 목록에 추가된다. 2) 스크랩이 되어있었다면, 색상이 비워지고 기록 상세 페이지의 스크랩한 문장 목록에서 삭제된다.	스크랩 여부에 따라 아이콘 색상이 변경되고, 기록 상세 페이지에서 스크랩한 문장 목록이 변경된다.	성공
	음성 가이드	음성 가이드를 들을 수 있다.	[재생] 또는 [중지] 아이콘을 눌러 녹음을 재생 및 중지할 수 있다. 재생바를 움직여 원하는 구간을 확인할 수 있다.	음성 가이드를 들을 수 있다.	성공
	연습 음성 녹음	연습 음성을 녹음한다.	1) [마이크] 아이콘을 누르고 제시된 지문을 따라 읽는다. 2) [중지] 또는 [일시 중지] 아이콘을 눌러 녹음을 중지하거나 일시 중지할 수 있다. 3) [중지] 아이콘을 눌러 녹음을 완료하면, 녹음된 음성이 나온다. 재생바를 움직여 원하는 구간을 확인할 수 있다. 4) [휴지통] 아이콘을 눌러 현재 녹음 삭제할 수 있고, 삭제 시 1로 돌아간다.	음성을 녹음하고 저장한다.	성공
	정확도 피드백	완료된 녹음에 대해 정확도 피드백이 나타난다.	[중지] 아이콘을 클릭하면, 몇초 후 '정확도:' 부분에 0-100 사이의 숫자로 정확도 피드백이 나타난다	연습 음성에 대한 정확도 피드백이 나타난다.	성공
	사용자 발음 표시	완료된 녹음의 발음이 표시된다.	녹음 [완료] 버튼을 클릭하면, 몇초 후 '발음:' 부분에 실제 사용자의 발음이 표시된다.	연습 음성에 대한 발음이 표시된다.	성공

	다음 버튼	다음 버튼을 클릭해 화면을 이동한다.	하단의 [다음] 버튼을 누르면, 1) 음성 녹음 완료 시, 다음 화면으로 이동한다. 2) 음성 녹음 미완료 시, 녹음을 완료하라는 안내 메시지가 뜨고 화면이 넘어가지 않는다. 3) 마지막 문장까지 연습을 완료하면, 자동으로 홈화면으로 이동한다. 홈화면의 마지막으로 연습한 대본이 해당 대본으로 변경된다.	녹음 완료 여부에 따라 안내 메시지가 나타나거나 화면을 이동한다.	성공
프롬프트 가이드 음성	카운트 다운	3초 카운트 다운 뒤에 프롬프트 연습을 시작한다.	만약 화면이 현재 가로 모드가 아닐 경우 가로로 화면을 전환해달라는 안내 메시지가 나온다. 가로 모드라면 3초 카운트 다운 뒤 프롬프트 연습이 시작된다.	가로 모드 전환 체크와 함께 3초 카운트 다운 뒤 프롬프트 연습이 시작된다.	성공
	가이드 음성 재생	가이드 음성을 재생하며 대본이 프롬프트 화면처럼 자동으로 스크롤되어 내려간다.	1) [재생] 아이콘을 눌러 가이드 음성을 재생한다. 2) [일시 중지] 아이콘을 눌러 음성 일시 중지할 수 있다. 또한 아이콘을 누르면 프롬프트 버튼 클릭 시와 동일한 dialog가 나타난다. [가이드 음성 듣기] 버튼을 클릭하면 가이드 음성을 다시 들을 수 있고, [연습하기] 버튼을 클릭 시, 프롬프트 연습 화면으로 이동한다. 3) 화면이 자동으로 스크롤이 되며 대본이 서서히 내려간다.	사용자의 선택에 따라 가이드 음성을 다시 듣거나 프롬프트 연습 화면으로 이동한다.	성공
프롬프트 연습	카운트 다운	3초 카운트 다운 뒤에 프롬프트 연습을 시작한다.	만약 화면이 현재 가로 모드가 아닐 경우 가로로 화면을 전환해달라는 안내 메시지가 나온다. 가로 모드라면 3초 카운트 다운 뒤 프롬프트 연습이 시작된다.	가로 모드 전환 체크와 함께 3초 카운트 다운 뒤 프롬프트 연습이 시작된다.	성공
	연습 음성 녹음	연습 음성을 녹음한다.	1) [마이크] 아이콘을 누르고 제시된 지문을 따라 읽는다. 2) [재생] 또는 [중지] 아이콘을 눌러 녹음을 재생 및 중지할 수 있다.	음성을 녹음하고 저장한다.	성공

			3) [완료] 아이콘을 눌러 녹음을 완료화면, 녹음된 음성이 나온다. 4) [휴지통] 아이콘을 눌러 현재 녹음 삭제할 수 있고, 삭제 시 1로 돌아간다. 5) [완료] 버튼을 눌러 프롬프트 피드백 화면으로 이동한다.		
프롬프트 피드백	음성 가이드	음성 가이드를 들을 수 있다.	[재생] 또는 [중지] 아이콘을 눌러 녹음을 재생 및 중지할 수 있다. 재생바를 움직여 원하는 구간을 확인할 수 있다.	음성 가이드를 들을 수 있다.	성공
	연습 음성	연습 음성을 들을 수 있다.	[재생] 또는 [중지] 아이콘을 눌러 녹음을 재생 및 중지할 수 있다. 재생바를 움직여 원하는 구간을 확인할 수 있다.	연습 음성을 들을 수 있다.	성공
	정확도 피드백	완료된 녹음에 대해 정확도 피드백이 나타난다.	몇초 후 '정확도:' 부분에 0-100 사이의 숫자로 정확도 피드백이 나타난다.	연습 음성에 대한 정확도 피드백이 나타난다.	성공
	다음 버튼	다음 버튼을 클릭해 홈화면으로 이동한다.	하단의 [다음] 버튼을 누르면, 자동으로 홈화면으로 이동한다. 홈화면의 마지막으로 연습한 대본이 해당 대본으로 변경된다.	홈 화면으로 이동한다.	성공

5.5 Loro에 대한 기술 문서

다음은 Loro의 tts를 담당하는 vits 모델, 대본 생성을 위한 RAG 시스템등 프로젝트를 위한 여러 논문들에 대해 저희 팀이 논문 스터디를 하며 정리한 내용입니다.

E3 TTS: Easy end-to-end Diffusion-based Text to Speech

Abstract

본 논문은 Diffusion 기반의 간단하고 효율적인 엔드투엔드 텍스트 음성 변환 모델인 Easy End-to-End Diffusion 기반 Text to Speech를 제안한다. 일반 텍스트를 직접 입력으로 받아 반복적인 정제 과정을 통해 오디오 파형을 생성한다.

이전 연구들과 달리 E3 TTS는 스펙트로그램 피처나 정렬 정보와 같은 중간 표현에 의존하지 않는다. 대신, E3 TTS는 Diffusion 과정을 통해 파형의 시간 구조를 모델링한다. 추가적인 조건 정보에 의존하지 않고 주어진 오디오 내에서 유연한 잠재 구조를 지원할 수 있다. 이를 통해 추가적인 훈련 없이 편집과 같은 zero shot 작업에 쉽게 적응시킬 수 있게 한다.

1. Introduction

Diffusion 모델은 데이터의 잠재적 표현에서 노이즈를 점진적으로 제거하여 실제 데이터와 구별이 어려울 정도로 만든다. Diffusion 모델을 사용하는 TTS 시스템은 최첨단 시스템과 유사한 고품질 음성을 생성할 수 있었다.

이 분야의 대부분의 이전 연구는 두 단계 생성 과정을 기반으로 했다.

1. 생성자 모델은 중간 표현을 생성하며 일반적으로 오디오 토큰 또는 스펙트로그램 기반 특성을 생성한다. 이 중간 표현은 웨이브폼과 정렬되지만 낮은 해상도로 제공된다.
2. vocoder가 중간 특성에서 오디오를 예측한다.

두 단계 TTS 파이프라인은 높은 품질의 오디오를 생성할 수 있지만 중간 특성의 품질에 의존하는 등 다른 문제점을 가질 수 있다. 또한 다양한 상황에서 배포하고 설정하기가 더 복잡할 수 있다.

두 단계 프로세스 외에도 대부분의 모델은 텍스트를 다른 입력 단위로 변환하기 위해 추가적인 신경 모델이나 통계적 방법을 사용한다.

파형에서 강한 시간적 종속성을 효율적으로 모델링하기 어렵기 때문에 텍스트에서 오디오를 엔드투엔드로 생성하는 것은 어렵다.

- 샘플 수준의 자기 회귀 vocoder는 전체 이력에 대한 각 파형 샘플의 생성을 조건화함으로써 이러한 종속성을 처리한다. 그러나 높은 순차적 특성 때문에 현대 병렬 하드웨어에서 샘플링하는 데 비효율적이다.
- 일부 이전 작업은 대신 생성 속도를 높이기 위해 중첩되지 않는 고정 길이 블록의 시퀀스를 자동 회귀적으로 생성한다. 이는 블록 내의 모든 샘플을 병렬로 생성하여 생성 프로세스를 가속화한다.

이전 연구 중 다른 접근 방식은 훈련 중에 정렬 정보를 포함하는 것이다. 정렬 정보는 각 개별 입력 단위 (ex : 음소)와 생성된 오디오의 출력 샘플 간의 매핑을 제공한다. 이는 각 개별 입력 단위의 시작 시간과 종료 시간을 제공하는 외부 정렬 도구를 사용하여 추출된다.

- FastSpeech 2 : 정렬 또는 지속 시간 정보 및 에너지 및 피치와 같은 다른 속성을 활용하여 오디오를 예측한다. 각 속성에 대해 하나의 내부 예측기도 훈련되어 추론 중에 예측 결과를 활용할 수 있다.
- EATS : 미분 가능한 지속 시간 예측기를 사용하고 예측이 대상 오디오와 정렬되도록 하기 위해 Dynamic Time Wrapping (DTW)을 사용한다. 이는 외부 정렬 도구를 사용하지 않게 만들지만 훈련을 더 복잡하게 만든다.

본 논문에서는 웨이브폼의 시간 구조를 보존하기 위해 Diffusion에만 의존하는 쉬운 엔드 투 엔드 텍스트 투 스피치 프레임워크 (E3 TTS) 를 제안한다. 이는 텍스트를 직접 입력으로 받아 pretrained BERT 모델을 사용하여 정보를 추출한다. 그 후에는 BERT 표현에 주의를 기울이며 오디오를

예측하는 UNet 구조가 이어진다. 전체 모델은 non-autoregressive 이며, 직접 웨이브폼을 출력한다.

논문 구성

섹션 2 : TTS의 이전 작업에서 최적화할 수 있는 다양한 구성 요소에 대한 간략한 개요를 제공한다.
섹션 3 : BERT 표현을 입력으로 사용하는 확산 모델만을 포함하는 제안된 시스템을 소개한다.
섹션 4 : 소유 데이터셋에서의 실험을 시작하며 몇몇 이전 작업과 비교한다.
섹션 5 : 제안된 방법으로 달성할 수 있는 일부 응용 프로그램을 소개한다.
섹션 6 : 시스템을 요약하고 몇 가지 미래 작업에 대해 논의한다.

2. Complexities of TTS

기존 TTS 시스템의 복잡성을 크게 증가시키는 여러 구성 요소를 확인했다.

2.1. Text Normalization

입력 텍스트의 정규화는 텍스트를 쓰여진 형태에서 TTS 시스템에서 쉽게 처리할 수 있는 형태로 변환하는 과정이다. 텍스트는 다양한 방식으로 작성될 수 있기 때문에 어려운 작업이다.

- "color"와 "colour"와 같이 동일한 단어가 다르게 쓰일 수 있다.
- 텍스트에는 약어, 두문자어 및 기타 비표준 형태가 포함될 수 있다.

2.2. Input Unit

텍스트 정규화 이후에도 동일한 단어를 다른 맥락에서 어떻게 발음해야 하는지에 대한 모호성이 남을 수 있다. 예를 들어, "record"는 명사인지 동사인지에 따라 발음이 다를 수 있다. 이것이 많은 TTS

시스템이 텍스트 대신 음운 또는 운율적 특성과 같은 발음 형태에 의존하는 이유다.

- 음운 : 음운은 단어를 구성하는 소리의 단위다. 이는 표준 쓰기 시스템이 없는 언어에서 음성을 생성하는 데 유용할 수 있다.
- 운율적 특성 : 운율적 특성은 음성의 특징으로 기본 주파수, 지속 시간 및 에너지와 같은 것들이 있다. 이는 생성된 음성의 억양과 강조를 제어하는 데 사용될 수 있다.

2.3. Alignment Modeling

정렬 모델링은 단어의 각 음운이 발음되어야 하는 시간을 예측하는 과정이다. 이는 생성된 음성이 자연스럽고 유창하게 들리도록 하는 데 중요하다. 정렬 모델링은 각 음운이 발음되는 시간에 영향을 미칠 수 있는 여러 요소가 있기 때문에 어려운 작업일 수 있다.

- 단어에서 음운의 위치
- 단어의 강세

End-to-End STT 시스템에서 정렬 모델의 전형적인 접근 방식은 정렬 정보를 제공하는 외부 정렬 도구에 의존하는 방법이 일반적이다. 모델 훈련 중에 지속 시간 예측기를 학습하여 추론에 대한 정렬을 추정하는 데 사용될 수 있는 정보를 예측한다.

- Non-Attentive Tacotron 프레임워크 : Variational Auto-Encoder를 사용하여 지속 시간을 암묵적으로 학습하는 데 성공했다.
- Glow-TTS 및 Grad-TTS : Monotonic Alignment Search 알고리즘 (Viterbi 훈련의 채택으로 두 시퀀스 간 가장 가능성 있는 숨겨진 정렬을 찾음)을 사용했다.
- E3 TTS : GradTTS에서 언급한 품질 문제를 엔드 투 엔드 실험으로 해결했다.

3. METHOD

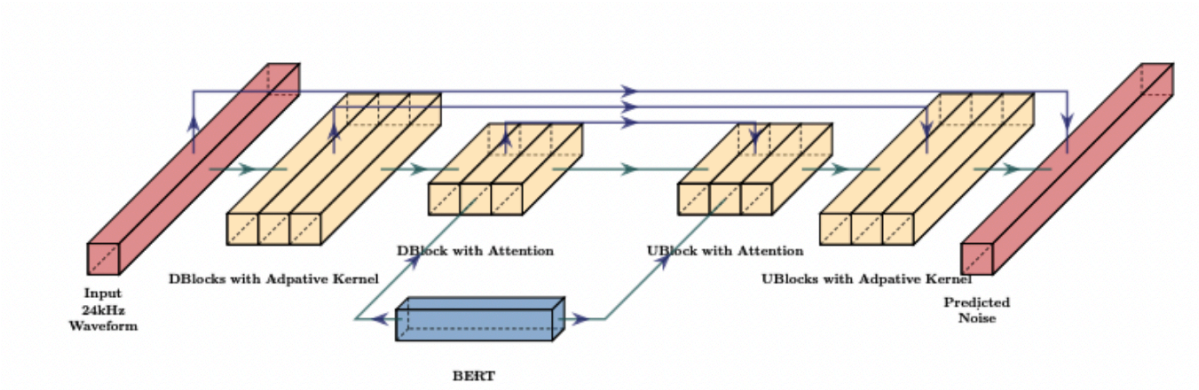


Fig. 1. UNet Structure: DBlock for downsampling block, UBlock for upsampling block

이전 섹션에서 제시된 도전 과제에 대한 해결책으로 TTS 시스템을 보다 폭넓은 커뮤니티에 접근 가능하게 만드는 솔루션을 제안한다.

- Pretrained BERT : 텍스트에서 정보를 추출
- Diffusion UNet : BERT 출력에 주의를 기울이고 노이즈를 포함한 웨이브폼을 반복적으로 정제하여 원시 웨이브폼을 예측

3.1. BERT model

최근의 대규모 언어 모델 개발의 장점을 활용하기 위해 미리 학습된 BERT 모델에 의해 제공되는 텍스트 표현을 기반으로 시스템을 구축했다. BERT 모델은 서브워드를 입력으로 사용하며 음운, 문자와 같은 음성의 다른 표현에 의존하지 않는다. 이는 이전 연구와 달리 사전에 훈련된 텍스트 언어 모델에 의존할 수 있기 때문에 여러 언어에 대한 텍스트 데이터만 사용하여 훈련될 수 있다.

3.2. Diffusion

E3 TTS는 score matching과 Diffusion 확률 모델에 기반한 이전 연구를 토대로 구축되었다.

TTS의 경우, 점수 함수는 조건부 분포 p(y|x)의 로그 도함수로 정의된다:

$$s(y \mid x) = \nabla_y \log p(y \mid x) \qquad (1)$$

x : 조건 신호
y : 웨이브폼

Denoising Diffusion Probabilistic Models 의 특수한 매개변수화를 채택했다:

$$\mathbb{E}_{\bar{\alpha},\epsilon}\left[\left\|\epsilon_\theta\left(\tilde{y}, x, \sqrt{\bar{\alpha}}\right) - \epsilon\right\|_2\right] \qquad (2)$$

ϵ ~ N (0, I) : 잡음 항
α¯ : 노이즈 레벨

점수 네트워크:모델 예측과 실제 값 ϵ 사이의 거리를 최소화하여 스케일링된 도함수를 예측하도록 훈련된다.

$$s(\tilde{y}|x, \bar{\alpha})$$

y¯는 다음과 같이 샘플링된다:

$$\tilde{y} = \sqrt{\bar{\alpha}}\, y_0 + \sqrt{1 - \bar{\alpha}}\, \epsilon \qquad (3)$$

훈련 중에 α¯는 사전에 정의된 β의 선형 스케줄에 따라 간격 [α¯{n}, α¯{n+1}]에서 샘플링된다.

$$\bar{\alpha}_n := \prod_{s=1}^{n}(1-\beta_s) \qquad (4)$$

각 반복에서 업데이트된 웨이브폼은 다음과 같은 확률적 프로세스를 따라 추정된다:

$$y_{n-1} = \frac{1}{\sqrt{\alpha_n}}\left(y_n - \frac{\beta_n}{\sqrt{1-\bar{\alpha}_n}}\epsilon_\theta(y_n, x, \sqrt{\bar{\alpha}_n})\right) + \sigma_n z \quad (5)$$

이 작업에서 수렴을 돕고 ϵ 손실의 크기를 더 잘 조절하기 위해 KL 형태의 손실을 채택했다. 또한 모델은 각 타임스텝에 대한 L2 손실의 분산 e ω(α)를 예측하고 다른 샘플된 타임스텝에서의 손실 가중치를 조절하기 위해 KL 형태의 손실을 사용한다:

$$\mathbb{E}_{\bar{\alpha},\epsilon}\left[\frac{1}{\omega(\bar{\alpha})}\left\|\epsilon_\theta\left(\tilde{y}, x, \sqrt{\bar{\alpha}}\right) - \epsilon\right\|_2 + ln(\omega(\bar{\alpha}))\right] \qquad (6)$$

3.3. U-Net

1차원 U-Net을 도입하였으며, 이는 Photorealistic text-to-image diffusion models with deep language understanding의 구조를 따른다. 일반적인 모델 구조는 Figure 1에 나와 있으며, 일련의 다운샘플링과 업샘플링 블록으로 이루어져 있으며 잔차를 통해 연결된다.

UBlock/DBlock의 자세한 구조 :

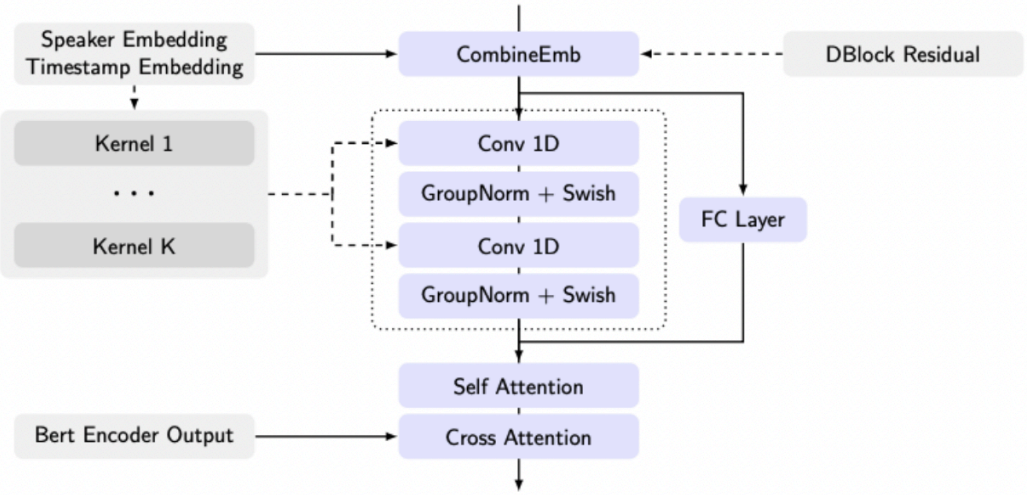


Fig. 2. UBlock/DBlock Struture: Adaptive Kernel and residual optional.

- Autoregressive TTS의 전형적인 방법 과 같이 상위 UBlock/DBlock에서 BERT 출력으로부터 정보를 추출하기 위해 교차 어텐션을 채택한다.
- 하위 UBlock/DBlock에서는 타임스텝과 스피커에 따라 커널이 결정되는 적응형 소프트맥스 CNN 커널을 사용한다.
- 다른 레이어에서는 FiLM을 사용하여 스피커와 타임스텝 임베딩을 결합하며, 이는 채널별 스케일링과 바이어스를 예측하는 병합 레이어로 구성된다.

Feature-wise Linear Modulation (FiLM): 이미지로부터 추출된 각 feature map은 텍스트 입력받는 RNN 네트워크에 의해 독립적으로 조건화된다. 따라서 이미지와 텍스트의 통합된 특징을 활용할 수 있다.

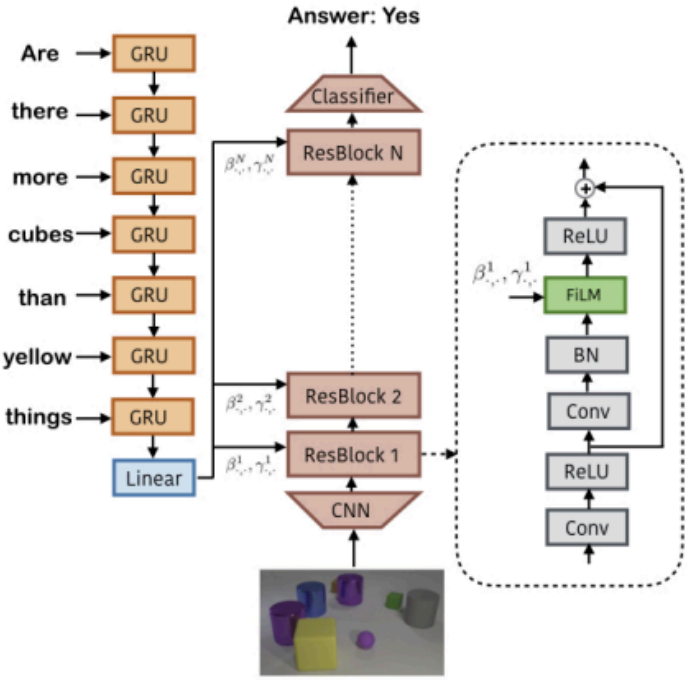


Figure 3: The FiLM generator (left), FiLM-ed network (middle), and residual block architecture (right) of our model.

- 다운샘플러 : 마지막으로 노이즈 정보(24kHz)를 인코딩된 BERT 출력과 유사한 길이의 시퀀스로 정제한다. 이는 실제로 품질을 향상시키는 데 중요한 역할을 한 것으로 입증되었다.
- 업샘플러 : 입력 웨이브폼과 동일한 길이의 노이즈를 예측한다.

훈련 중에는 웨이브폼의 길이를 10.92초로 고정하고 웨이브폼 끝에는 zero padding을 적용했다. 손실을 계산할 때 각 패딩 프레임을 가중치 1/10로 설정했다.

추론 시에는 출력 웨이브폼의 길이를 고정하고 평균 크기를 사용하여 패딩 부분을 구별한다. 실제로 1024 샘플마다 평균 크기를 계산하고 ≤ 0.02 부분을 잘라냈다.

4. Experiement

E3 TTS를 다른 신경망 기반 TTS 시스템과 비교했다. 기준 시스템은 84명의 전문 음성 배우로부터 얻은 385시간의 고품질 미국 영어 음성으로 이루어진 독점 데이터셋에서 훈련되었다. 평가를 위해 훈련 데이터셋에서 여성 스피커를 선택했다.

Block index	0	1	2	3
Base dimension	128	256	512	1024
Kernel Size	[5,5]	[5,5]	[5,5]	[3,3,3,3,3]
Strides	[2,2]	[2,2]	[4]	[4,2,2,2,2]
Adaptive Kernel	[8,8]	[4,4]	[2]	
Blocks	[2,2]	[2,2]	[2]	[1,1,1,1,1]
Self Attention	[×,×]	[×,×]	[×]	[✓,✓,✓,✓,✓]
Cross Attention	[×,×]	[×,×]	[×]	[✓,✓,✓,✓,✓]
Attention Heads				[8,8,8,8,8]

Table 1. Model configuration. Empty cell indicates it is not used in this block.

사전 훈련된 BERT에 대해서는 영어 전용 데이터에서 훈련된 기본 매개변수 크기 모델을 사용했다.

추론에서는 1000 단계 DDPM을 사용하며, 노이즈 스케줄링은 다음과 같이 정의된다:

$$\alpha_n = \exp\left(\ln(1e{-}7) * \left(1 - \cos\left(\frac{n}{1000} * \frac{\pi}{2}\right)\right)^{\frac{3}{2}}\right) \qquad (7)$$

성능은 주관적 청취 테스트를 통해 측정되었으며, 헤드폰을 착용한 일부 원어민 청취자에 의해 수행되었다.

결과는 생성된 샘플의 자연스러움을 1에서 5까지의 십점 척도로 측정하는 평균 의견 점수(Mean Opinion Score, MOS)로 보고되었다. 각 샘플은 적어도 두 명의 다른 원어민 청취자에 의해 최소 두 번 평가되었다.

문자 기반 TTS 모델과 Wave-Tacotron과 본 논문의 모델을 비교하였으며, 결과는 다음과 같다:

Mode	MOS
Ground truth	4.56 ± 0.04
Two-Stage Models	
Tacotron-PN + Griffin-Lim [35] (char)	3.68 ± 0.08
Tacotron + WaveRNN [36] (char)	$\mathbf{4.36 \pm 0.05}$
Tacotron + Flowcoder [37] (char)	3.34 ± 0.07
End-to-End Models	
Wave-Tacotron [21] (char)	4.07 ± 0.06
Our Model	$\mathbf{4.24 \pm 0.06}$

Table 2. TTS performance on the proprietary single speaker dataset, evaluation contains about 600 examples.

결과는 제안된 방법이 다른 end-to-end 시스템보다 더 높은 충실도를 제공한다는 것을 시사한다. 여기서의 작은 차이점은 제안된 시스템이 문자 대신에 하위 단어(sub-word)를 기반으로 하고 있다는 점인데, 본 논문에서는 그것이 TTS 적용에 비교가능해야 한다고 믿는다.

RAG : Retrieval-Augmented Generation

LLM 기반의 시스템 & 제품 구축 패턴들

다음은 기업에서 LLM 기반 서비스를 도입하는 걸 고민할 때, 이 서비스가 “성능 향상 vs. 비용/리스크 감소” 및 “데이터 친화 vs 사용자 친화” 측면에서 어떻게 구현하는지 볼 수 있는 기준점이다.

- Evals: 성능 측정
- RAG(Retrieval-Augmented Generation): 최신, 외부 지식을 추가
- Fine-tuning: 특정 작업을 더 잘 수행하기 위해
- Caching: 레이턴시 및 비용 감소
- Guardrails: 출력 품질 보장
- Defensive UX: 오류를 에측하고 관리하기 위해
- Collect user feedback: 데이터 플라이 휠 구축

RAG(Retrieval-Augmented Generation): 지식 추가

RAG란?

- RAG는 기존 모델 외부에서 관련 데이터를 검색하여 입력을 향상시키는 방법으로, 이를 통해 결과를 개선하며 더 풍부한 맥락을 제공한다.
- 사전 훈련된 대형 언어 모델의 단점(기억을 확장하거나 수정할 수 없음, 생성된 출력에 대한 통찰을 제공하지 않음, 환상 등)을 해결하기 위해 제안되었다.

RAG의 이점:

1. 환상 감소와 사실성 증가: 검색된 맥락에 근거하여 모델을 고정시킴으로써 환상을 줄이고 사실적인 결과를 얻을 수 있다.
2. 비용 효율성: 사전 훈련을 계속하는 것보다 검색 인덱스를 최신 상태로 유지하는 비용이 더 적다. 이는 최근 데이터에 더 쉽게 접근할 수 있게 하여 LLM에 이점을 제공한다.
3. 데이터 업데이트 용이성: 편향된 또는 유해한 데이터를 업데이트하거나 제거해야 할 경우 검색 인덱스를 업데이트하기가 (미세 조정이나 유해한 결과 생성 방지와 비교하여) 더 간단하다.

RAG의 기원:

- RAG는 오픈 도메인 Q&A에서 시작되었으며, 초기 Meta 논문에서는 TF-IDF를 사용하여 관련 문서를 검색하고 이를 언어 모델(BERT)에 맥락으로 제공함으로써 오픈 도메인 Q&A 작업의 성능을 향상시켰다.

Retrieval Augmented Generation (RAG)

Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

추론 과정에서는 입력과 검색된 문서를 연결하고, 언어 모델은 원래 입력, 검색된 문서, 이전 i-1 토큰을 기반으로 토큰 i를 생성한다.

밀집 벡터 검색은 비매개변수 구성 요소로 작용하며, 사전 훈련된 언어 모델은 매개변수 구성 요소로 작용하기 때문에 반 매개변수 모델이라고도 불린다.

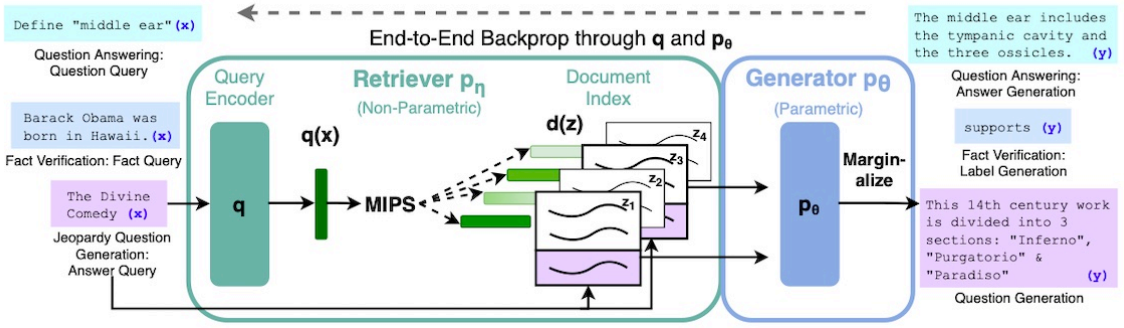


Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x, we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i. For final prediction y, we treat z as a latent variable and marginalize over seq2seq predictions given different documents.

RAG의 두 가지 생성 접근 방식:

1. RAG-Sequence

   완전한 시퀀스를 생성하기 위해 동일한 문서를 사용한다.

   - k개의 검색된 문서에 대해 생성기는 해당 문서에 대한 출력을 생성한다.
   - 각 출력 시퀀스의 확률은 k에서 각 출력 시퀀스의 확률을 합산하고, 각 문서가 검색되는 확률에 따라 가중치를 적용한다.
   - 마지막으로 가장 높은 확률을 가진 출력 시퀀스가 선택된다.

2. RAG-Token

   각 토큰을 다른 문서를 기반으로 생성할 수 있다.

   - k개의 검색된 문서가 주어진 경우, 생성기는 각 문서에 대해 다음 출력 토큰에 대한 분포를 생성한다.
   - 개별 토큰 분포를 모두 집계, 반복한다.

3. 각 토큰 생성마다 원래 입력과 이전에 생성된 토큰을 기반으로 k개의 서로 다른 관련 문서를 검색할 수 있음을 의미한다. 따라서 문서는 다른 검색 확률을 가지며 다음 생성된 토큰에 다르게 기여할 수 있다.

Fusion-in-Decoder (FiD)

Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering

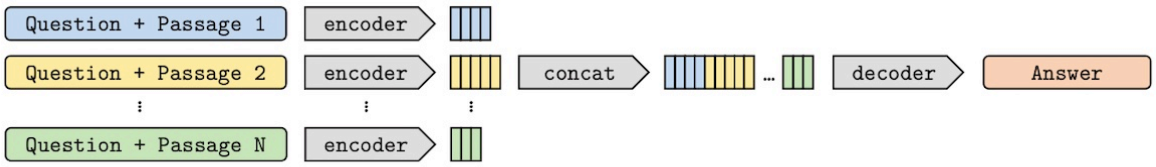


Figure 2: Architecture of the Fusion-in-Decoder method.

오픈 도메인 질문 응답을 위한 모델로, 검색된 정보를 활용하여 답변을 생성하는데 사용된다.

1. 각 검색된 passage에 대해 제목과 passage를 질문과 결합한다.
2. 결합된 쌍은 인코더에서 독립적으로 처리된다.
3. 각 섹션 앞에는 question:, title:, context:와 같은 특별한 토큰이 추가된다.
4. 디코더는 이러한 검색된 passage들의 연결에 attention을 기울인다.

인코더에서는 각 passage를 독립적으로 처리하기 때문에 한 번에 한 문맥에 대한 self-attention만 필요하므로 많은 수의 passage에 대해 확장할 수 있다. 따라서 연산이 검색된 passage의 수와 선형적으로 증가하므로, RAG-Token과 같은 대안에 비해 더 확장 가능하다. 디코더는 인코딩된 passage를 공동으로 처리하여 여러 검색된 passage에 걸쳐 문맥을 더 잘 집계할 수 있다.

Retrieval-Enhanced Transformer (RETRO)

Improving language models by retrieving from trillions of tokens

트랜스포머, 어텐션을 이용한 또다른 접근 방식이다.

Internet-augmented LMs

[Internet-augmented language models through few-shot prompting for open-domain question answering]

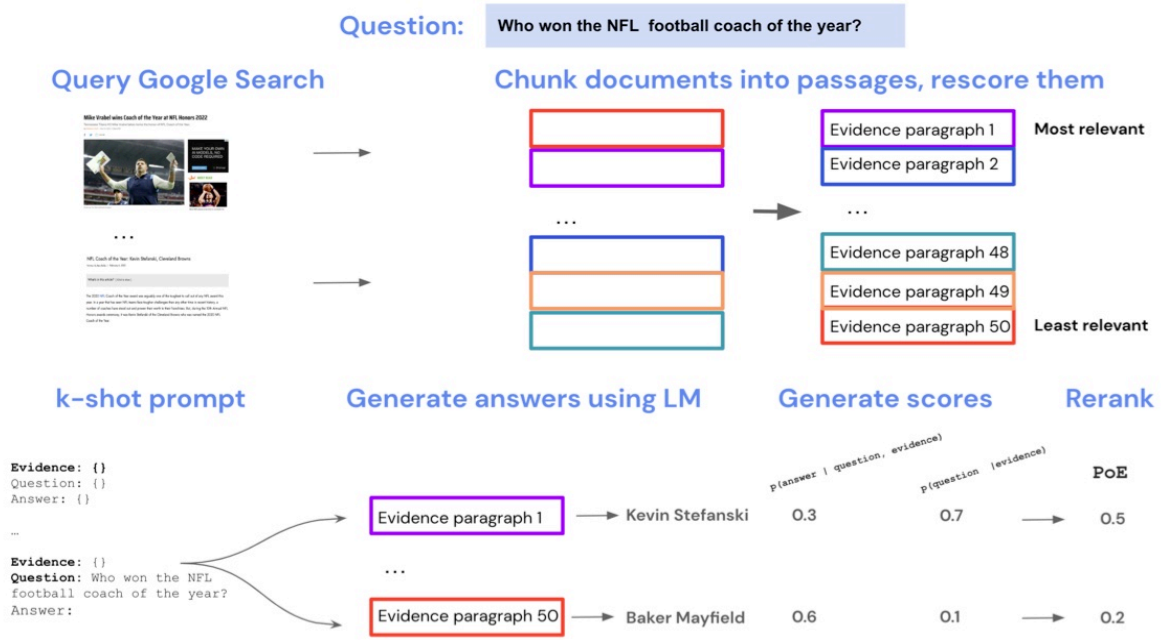


Figure 4: Schematic representation of the method presented in Section 3.

LLMs를 강화하기 위해 일반적인 "off-the-shelf" 검색 엔진을 사용하는 방안이다.

1. Google 검색을 통해 관련 문서 세트를 검색한다.
2. 이러한 검색된 문서들은 보통 길기 때문에(평균 길이 2,056 단어), 각각을 여섯 문장씩으로 나누어 단락으로 만든다.
3. TF-IDF를 사용하여 질문과 단락을 임베딩하고, 각 쿼리에 대해 가장 관련성 높은 단락을 순위로 매긴다.
4. 검색된 단락을 LLM을 few-shot prompting을 통해 조건화하는 데 사용한다.
5. closed-book QA(질문-답변 쌍만 제공)에서 일반적인 k-shot prompting(k=15)을 채택하고,

각 문맥이 증거, 질문, 답변 세트인 방식으로 확장한다.

6. 생성기는 각 질문에 대해 50개의 검색된 단락 중 각각에 기반한 네 개의 후보 답변을 생성한다.
7. 답변 확률을 평가하여 최적의 답변을 선택한다.

Hypothetical document embeddings (HyDE)

Precise Zero-Shot Dense Retrieval without Relevance Labels

질문과 문서 쌍에 대한 관련성 레이블이 없는 경우 쿼리와 문서를 같은 임베딩 공간에 내재시키는 Bi-encoder를 훈련시키기 어렵다는 문제점을 해결하기 위해 제안되었다. 관련성 모델링 문제를 표현 학습 작업에서 생성 작업으로 재구성했다.

Bi-Encoder

Cross-Encoder와 Bi-Encoder (feat. SentenceBERT)

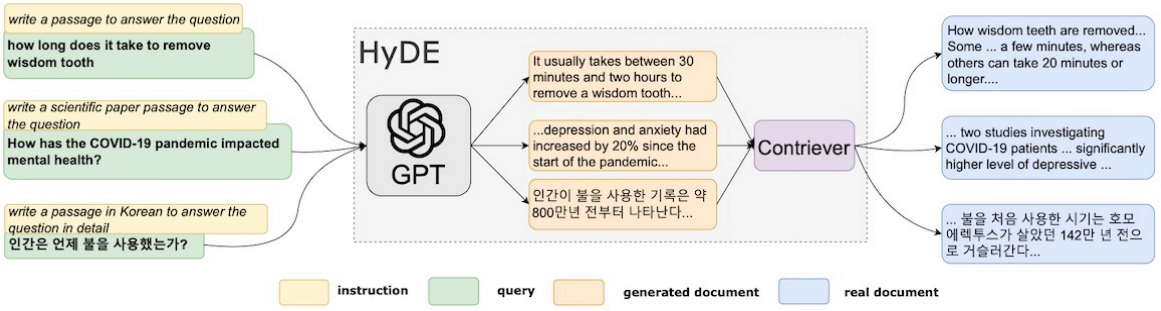


Figure 1: An illustration of the HyDE model. Documents snippets are shown. HyDE serves all types of queries without changing the underlying GPT-3 and Contriever/mContriever models.

1. 주어진 쿼리에 대해, InstructGPT와 같은 LLM에 가상 문서를 생성하도록 유도한다. 이

문서는 관련성 패턴을 포착하지만 실제가 아닐 뿐더러 잘못된 세부사항을 포함할 수 있다.

2. Contriver와 같은 비지도 대조 학습된 인코더가 문서를 임베딩 벡터로 변환한다.
3. 가상 문서와 말뭉치 간에 내적이 계산되고 가장 유사한 실제 문서가 검색된다.

이를 통해 생성된 문서를 실제 코퍼스에 묶어주며, 인코더의 dense bottleneck은 부정확한 세부사항을 걸러낼 수 있다.

RAG 적용 방법과 경험

전통적인 검색 인덱스와 임베딩 기반 검색을 혼합한 하이브리드 검색이 각각 독립적으로 사용하는 것보다 효과적이다. 고전적인 검색(BM25를 통한 OpenSearch)에 의한 검색을 보완하기 위해 의미 기반 검색(e5-small-v2)을 도입했다.

왜 임베딩 기반 검색만 사용하지 않는가?

임베딩 기반 검색이 많은 경우에 효과적이지만, 특정 상황에서는 한계가 있다.

- 사람이나 물체의 이름 검색 (예: Eugene, Kaptir 2.0)
- 약어나 구문 검색 (예: RAG, RLHF)
- ID 검색 (예: gpt-3.5-turbo, titan-xlarge-v1.01)

키워드 검색의 한계:

키워드 검색은 단순한 단어 빈도만을 모델링하며 의미론적이거나 상관 관계 정보를 캡처하지 않는다. 따라서 동의어나 상위어 (즉, 일반화를 나타내는 단어)에 대해 잘 처리하지 못한다. 이는 의미 기반 검색과 조합함으로써 상호 보완적인 효과를 얻을 수 있다.

메타데이터 활용:

전통적인 검색 인덱스를 사용하면 결과를 세분화하는 데 메타데이터를 활용할 수 있다.

- 날짜 필터를 사용하여 최신 문서를 우선적으로 처리하거나 검색을 특정 시간 범위로 제한할 수 있다.
- 전자 상거래와 관련된 검색인 경우, 평균 평점이나 카테고리에 대한 필터가 유용하다.
- 메타데이터는 하향식 순위 지정에 유용하며, 더 많이 인용되는 문서를 우선적으로 처리하거나 판매량에 따라 제품에 가중치를 부여하는 데 도움이 된다.