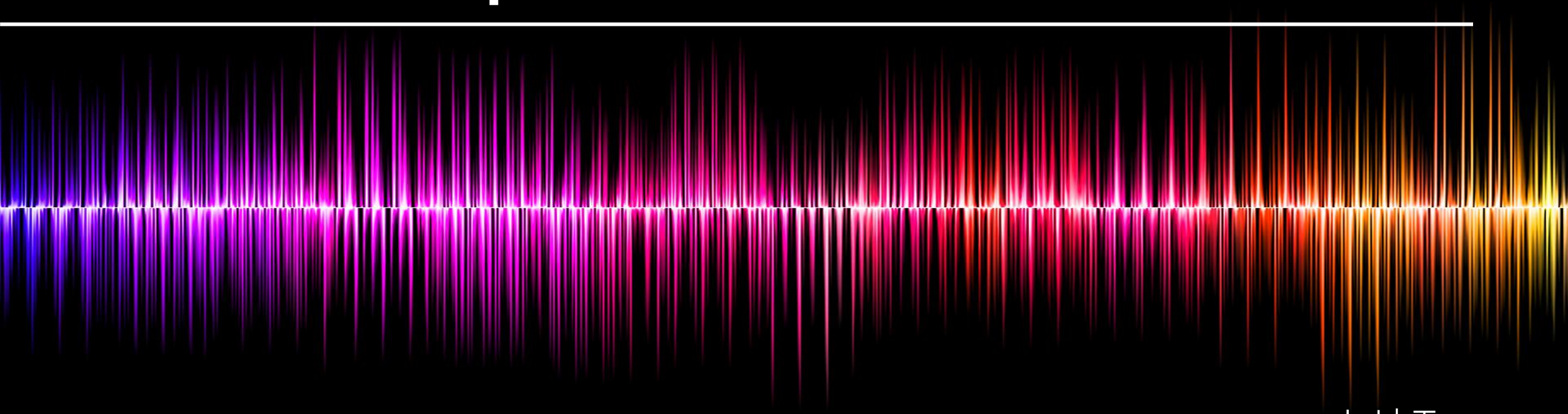
Text-to-Speech



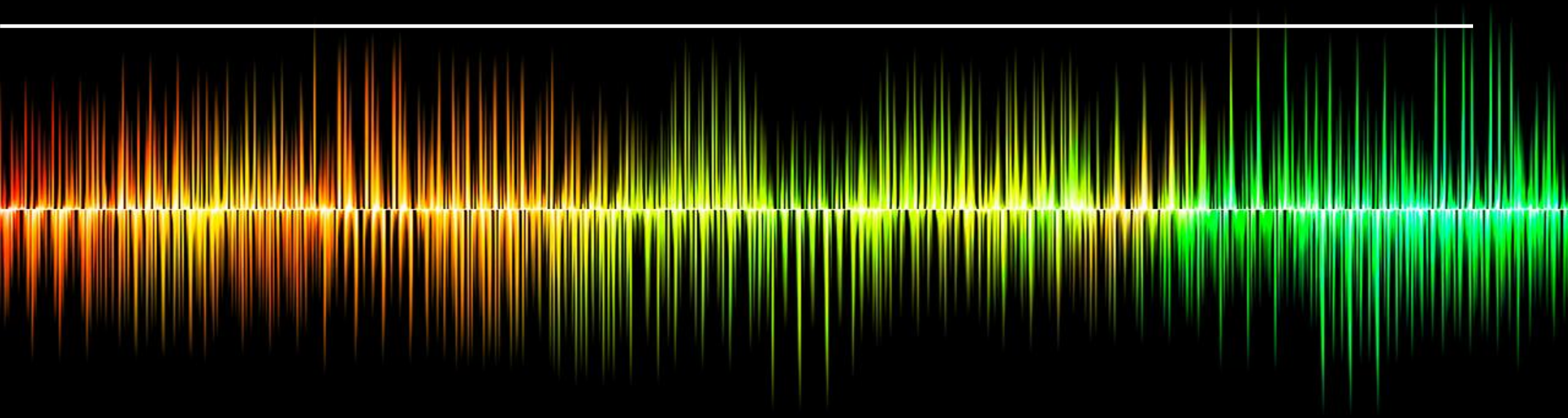
소신 Team

김은식

황태연

목차

Part 1　프로젝트 목표

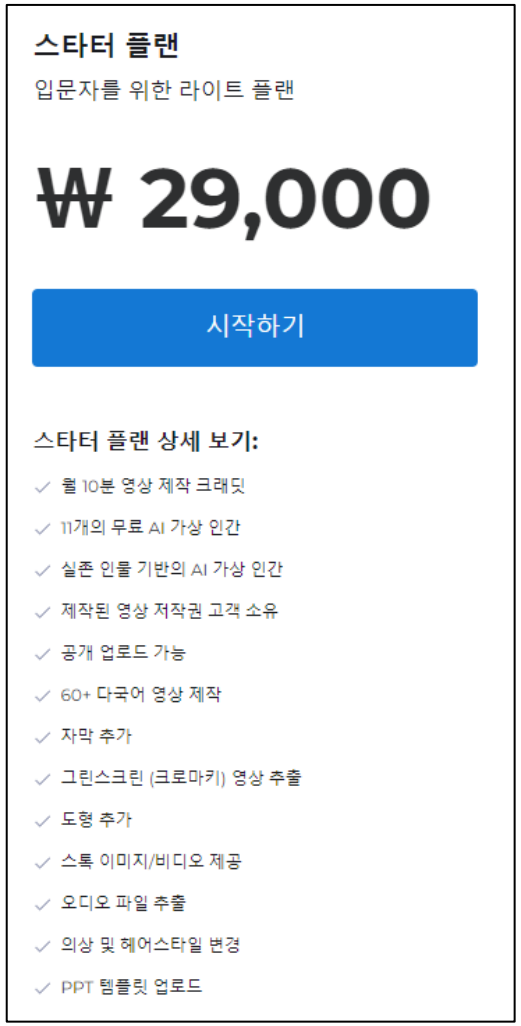


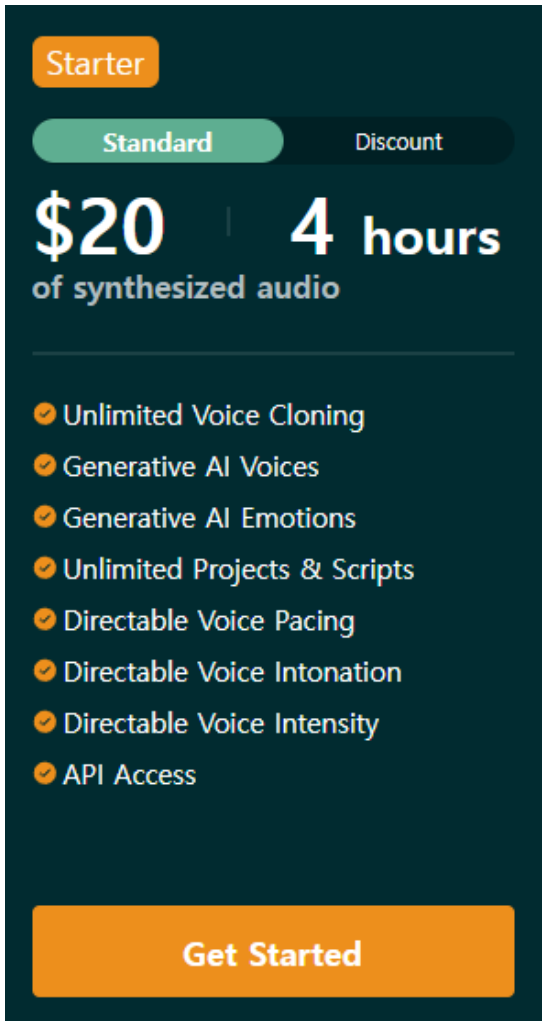
Part 1

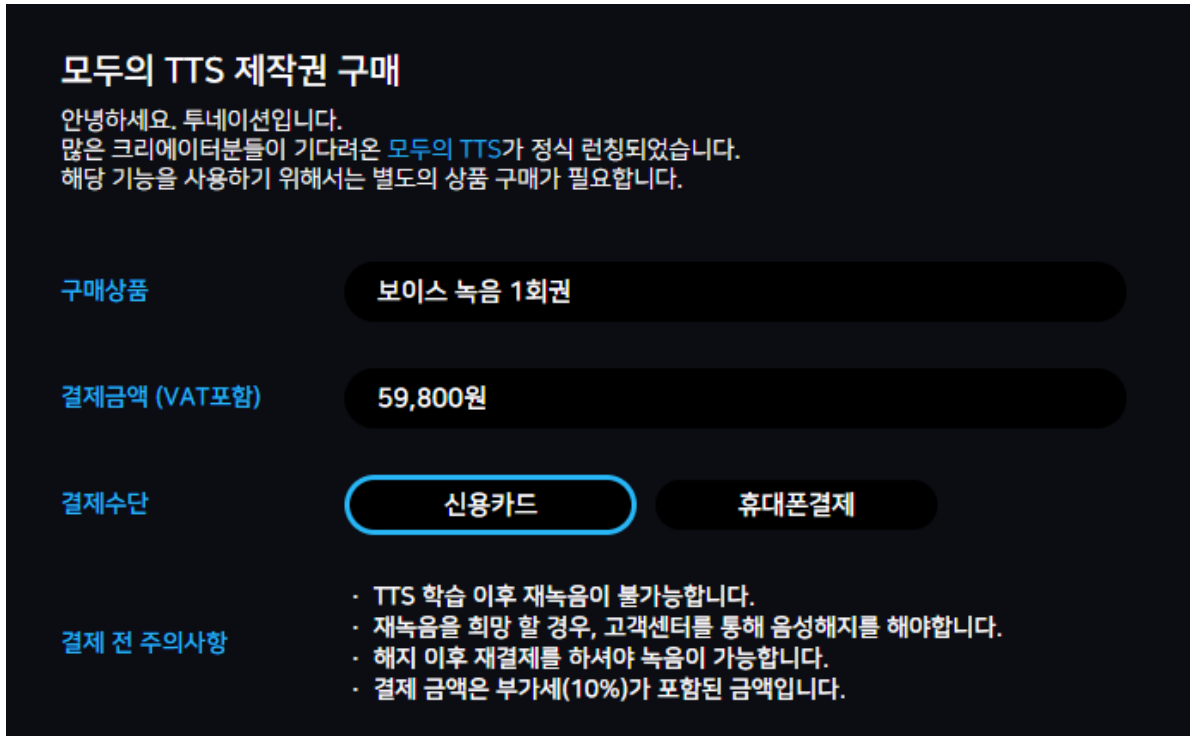
프로젝트 목표

우리의 목소리가 담긴 개인 TTS를 만들자!

Part 1 프로젝트 목표







Part 1 프로젝트 목표

coqui TTS

🐸TTS is a library for advanced Text-to-Speech generation. It's built on the latest research, was designed to achieve the best trade-off among ease-of-training, speed and quality. 🐸 TTS comes with pretrained models, tools for measuring dataset quality and already used in **20+ languages** for products and research projects.

chat on discord 301 online | License MPL 2.0 | pypi package 0.14.3 | Contributor Covenant v2.0 adopted
downloads 286k | DOI 10.5281/zenodo.8009420

aux-tests passing | data-tests passing | Docker build and push passing
inference_tests passing | style-check passing | text-tests passing | tts-tests passing
vocoder-tests passing | zoo-tests-0 passing | zoo-tests-1 failing | zoo-tests-2 passing
docs passing

SCE-TTS: 내 목소리로 TTS 만들기

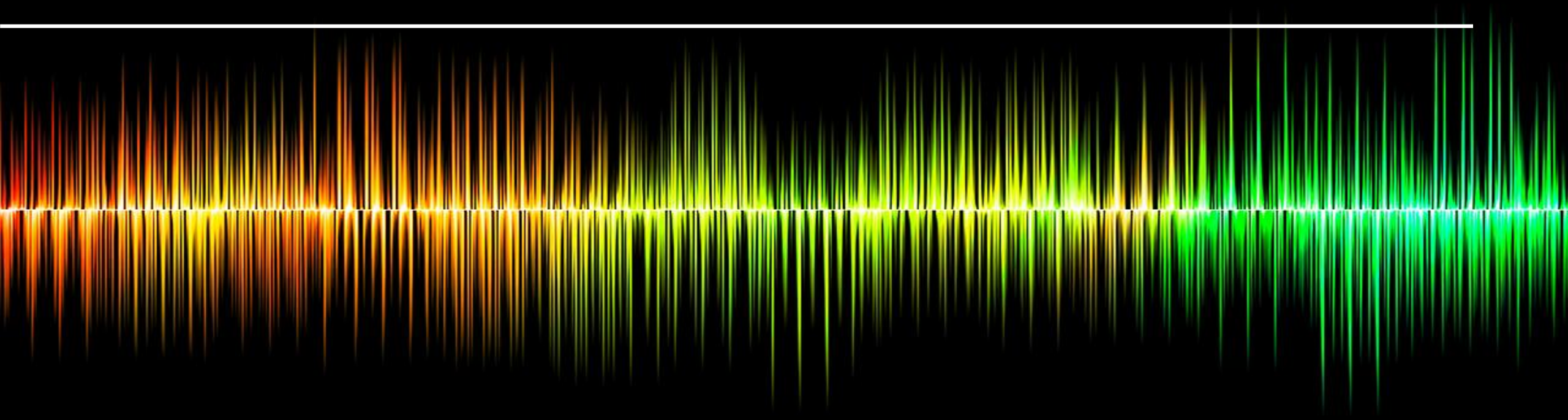
SCE-TTS는 자신의 목소리로 문장을 읽어주는 TTS(Text-To-Speech)를 만드는 프로젝트입니다.
SCE-TTS를 사용하면 머신 러닝을 통해 누구나 자신의 목소리로 TTS를 만들 수 있습니다.

Part 1 프로젝트 목표

우리의 목소리가 담긴 개인 TTS를 만들자!

단, 우리가 직접 코드로 구현해서!

Part 2　모델의 변경 과정



Part 2 모델의 변경 과정

WaveNet	Tacotron1	Tacotron2	HiFi-GAN	Glow-TTS
2016 Google DeepMind	2017 Google Inc.	2018 Google Inc.	2020 Kakao Enterprise	2020 Kakao & SNU AI
End-to-End	End-to-End	End-to-End	Vocoder	Mel-Generator
Autoregressive Model	Autoregressive Model	Autoregressive Model	GAN based Model	Flow based Model

End-to-End || Text → Waveform

Mel-Generator || Text → Mel-spectrogram

Vocoder || Mel-spectrogram → Waveform

Part 2 모델의 변경 과정

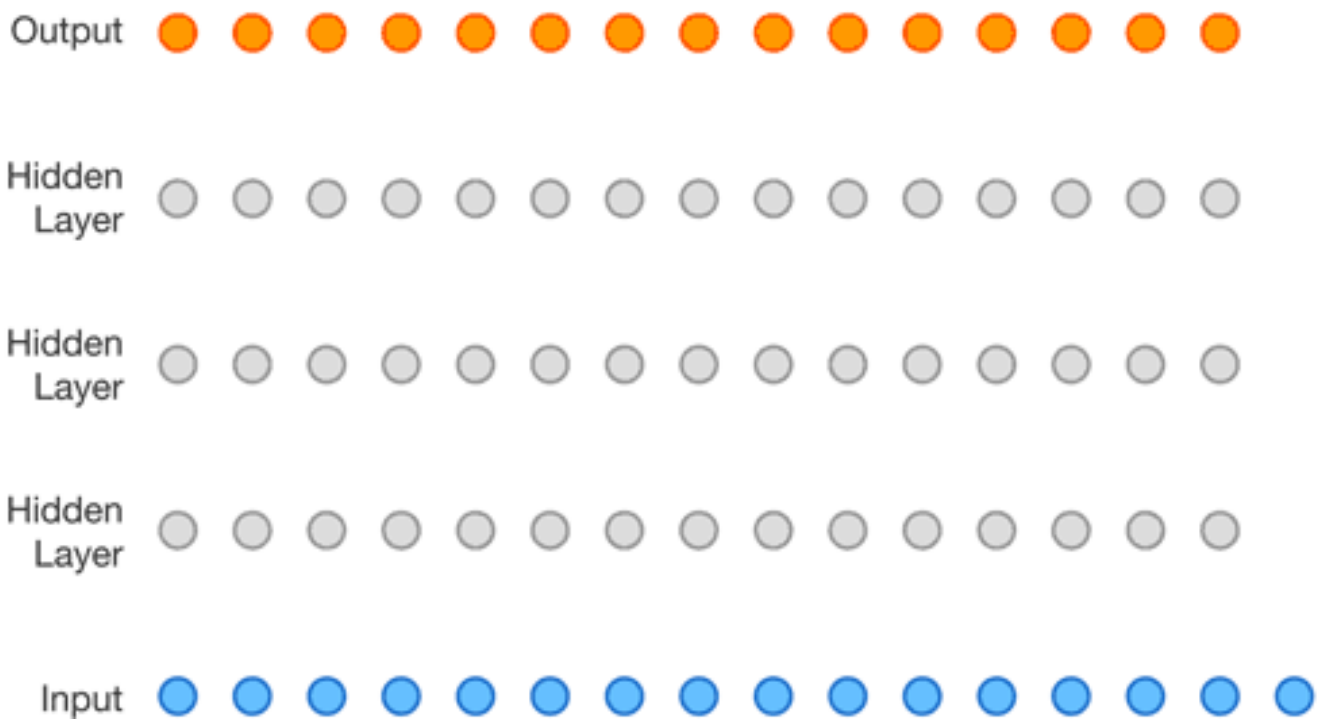
WaveNet

2016
Google DeepMind

End-to-End

Autoregressive
Model

- 초기의 성공적인 딥러닝 TTS 모델



Part 2 모델의 변경 과정

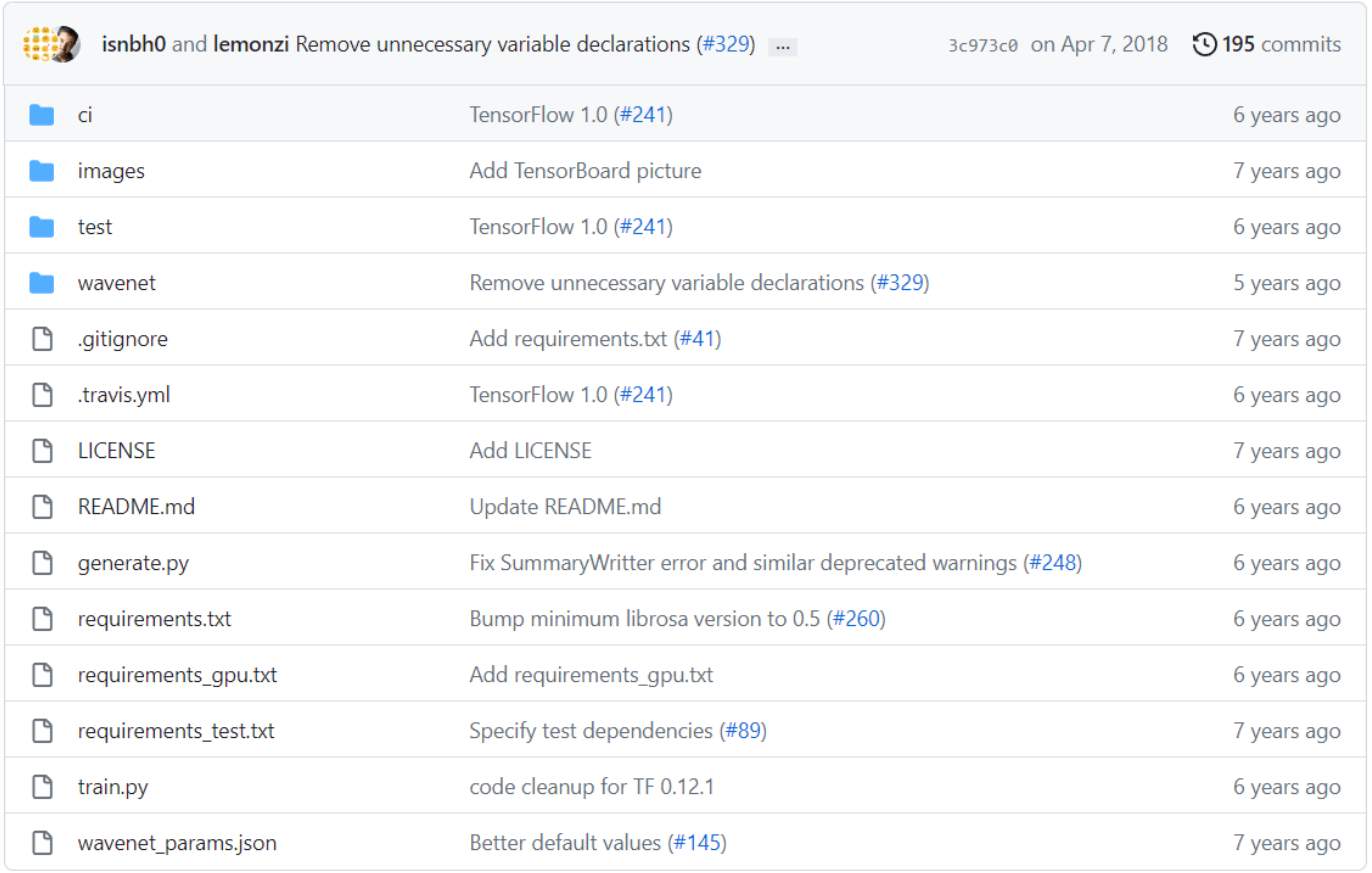
WaveNet

2016
Google DeepMind

End-to-End

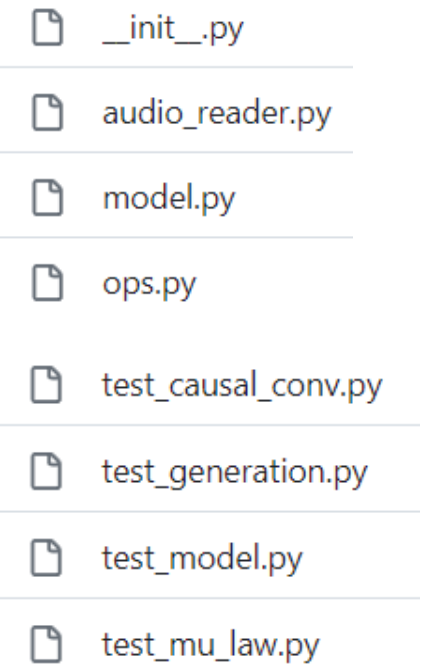
Autoregressive
Model

- WaveNet은 논문 저자의 공식 구현이 없다.



isnbh0 and lemonzi Remove unnecessary variable declarations (#329) ... 3c973c0 on Apr 7, 2018 195 commits

ci	TensorFlow 1.0 (#241)	6 years ago
images	Add TensorBoard picture	7 years ago
test	TensorFlow 1.0 (#241)	6 years ago
wavenet	Remove unnecessary variable declarations (#329)	5 years ago
.gitignore	Add requirements.txt (#41)	7 years ago
.travis.yml	TensorFlow 1.0 (#241)	6 years ago
LICENSE	Add LICENSE	7 years ago
README.md	Update README.md	6 years ago
generate.py	Fix SummaryWritter error and similar deprecated warnings (#248)	6 years ago
requirements.txt	Bump minimum librosa version to 0.5 (#260)	6 years ago
requirements_gpu.txt	Add requirements_gpu.txt	6 years ago
requirements_test.txt	Specify test dependencies (#89)	7 years ago
train.py	code cleanup for TF 0.12.1	6 years ago
wavenet_params.json	Better default values (#145)	7 years ago



__init__.py
audio_reader.py
model.py
ops.py
test_causal_conv.py
test_generation.py
test_model.py
test_mu_law.py

Part 2 모델의 변경 과정

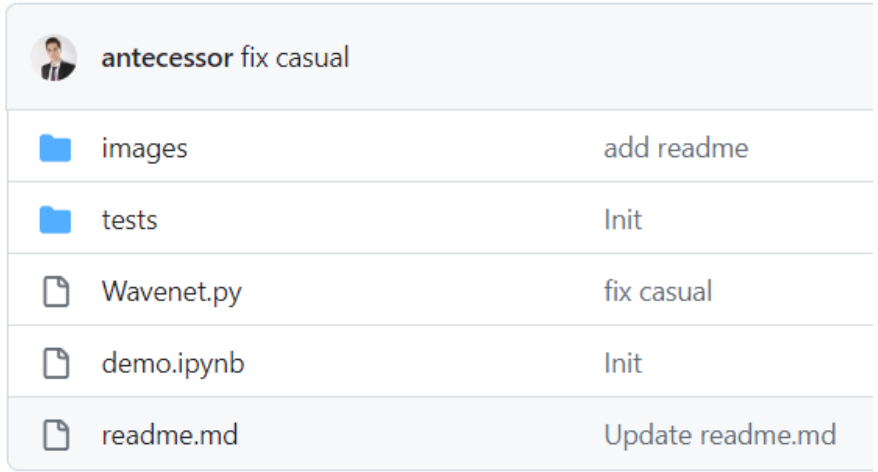
WaveNet

2016
Google DeepMind

End-to-End

Autoregressive
Model

- Jupyter에서 아래의 Github 구현을 따라함.



Part 2 모델의 변경 과정

WaveNet

2016
Google DeepMind

End-to-End

Autoregressive
Model

```
116 class WaveNetClassifier(nn.Module):
117     def __init__(self, seqLen, output_size):
118         super().__init__()
119         self.output_size = output_size
120         self.wavenet = WaveNet(1, 1, 2, 3, 4)
121         self.linear = nn.Linear(seqLen - self.wavenet.calculateReceptiveField(),
122                                 output_size)
123         self.softmax = nn.Softmax(-1) # -1: 입력값의 마지막 차원
124 
125     def forward(self, x):
126         x = self.wavenet(x)
127         x = self.linear(x)
128         return self.softmax(x)
```

- 232 lines
- 구현 성공, WaveNet 동작만 확인

Part 2 모델의 변경 과정

Tacotron1

2017
Google Inc.

End-to-End

Autoregressive
Model

- WaveNet의 느린 속도를 극복한 모델

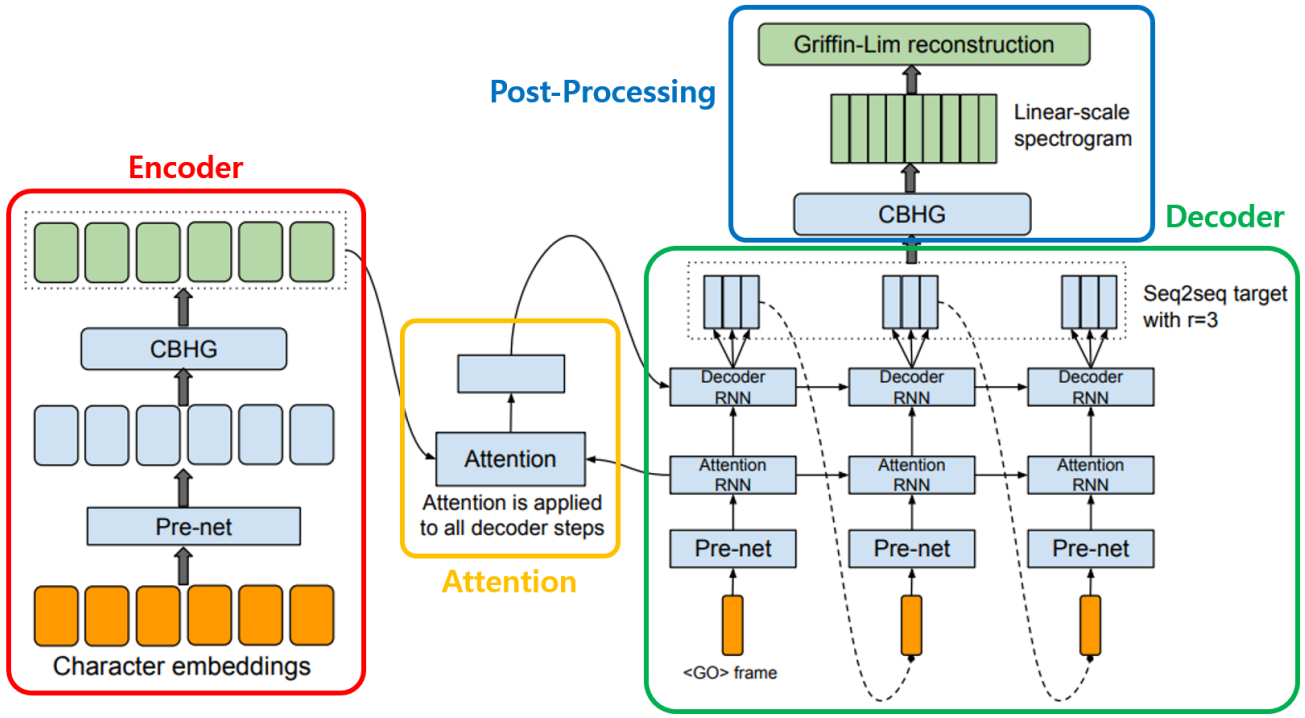


Figure 1: *Model architecture. The model takes characters as input and outputs the corresponding raw spectrogram, which is then fed to the Griffin-Lim reconstruction algorithm to synthesize speech.*

Part 2 모델의 변경 과정

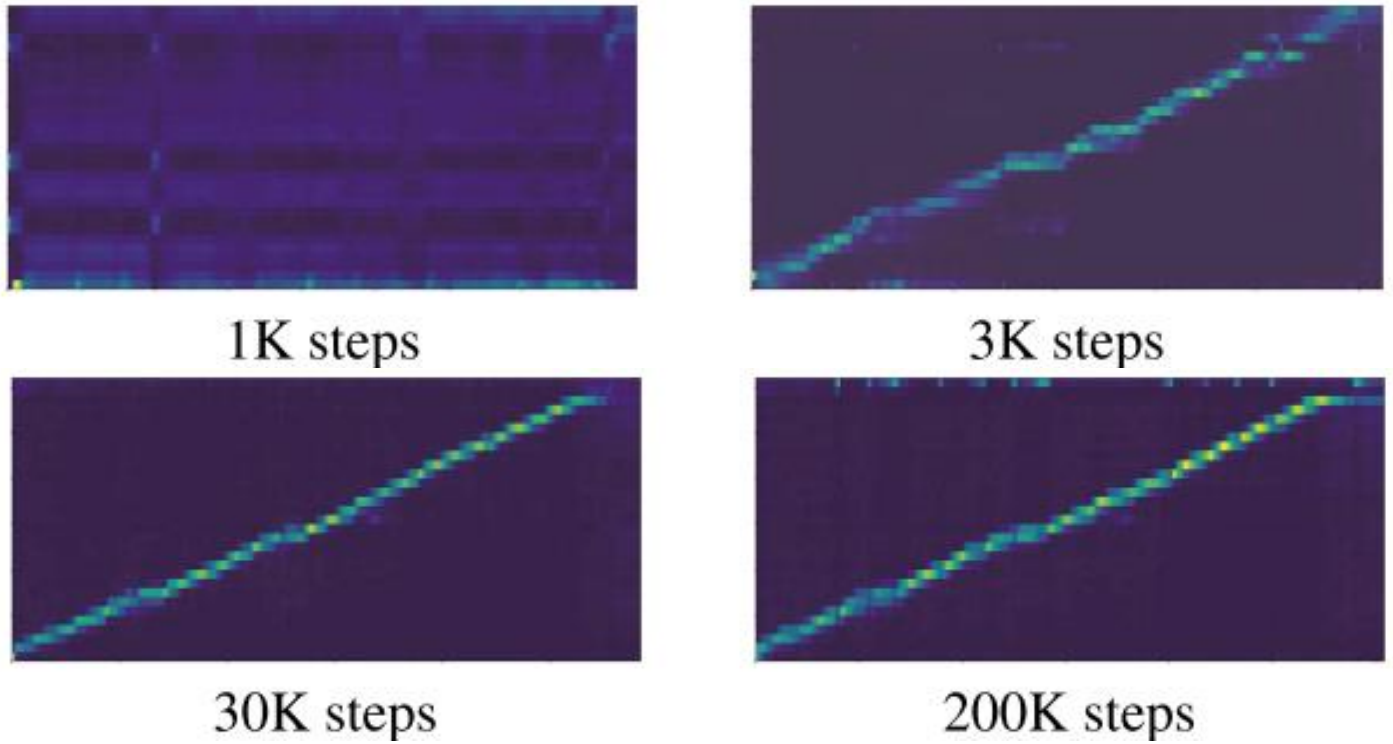
Tacotron1

2017
Google Inc.

End-to-End

Autoregressive
Model

- Attention의 Alignment



Part 2 모델의 변경 과정

Tacotron1

2017
Google Inc.

End-to-End

Autoregressive
Model

- 한국어에 알맞게 구현된 Github 코드를 따라 구현

```python
class BahdanauAttention(Module):
    def __init__(self):
        super(BahdanauAttention, self).__init__()
        self.w1 = Linear(decoder_dim, decoder_dim)
        self.w2 = Linear(decoder_dim, decoder_dim)

    def forward(self, query, value): # (B, 1, 256), (B, T, 256)
        q = torch.unsqueeze(self.w1(query), axis=2) # (B, 1, 1, 256)
        v = torch.unsqueeze(self.w2(value), axis=1) # (B, 1, T, 256)
        score = torch.sum(Tanh()(q + v), dim=-1) # (B, 1, T)
        alignment = Softmax(dim=-1)(score) # (B, 1, T)
        context = torch.matmul(alignment, value) # (B, 1, 256)
        context = torch.cat([context, query], axis=-1) # (B, 1, 512)
        alignment = alignment.transpose(1, 2) # (B, T, 1)
        return context, alignment
```

- 902 lines
- kss 데이터를 이용하여 학습
- 14번의 모델 재학습 시도, 학습이 제대로 이뤄지지 않음.

Part 2 모델의 변경 과정

Tacotron2

2018
Google Inc.

End-to-End

Autoregressive
Model

- Tacotron1보다 더 간단한 구조, 더 좋은 성능을 보인 모델

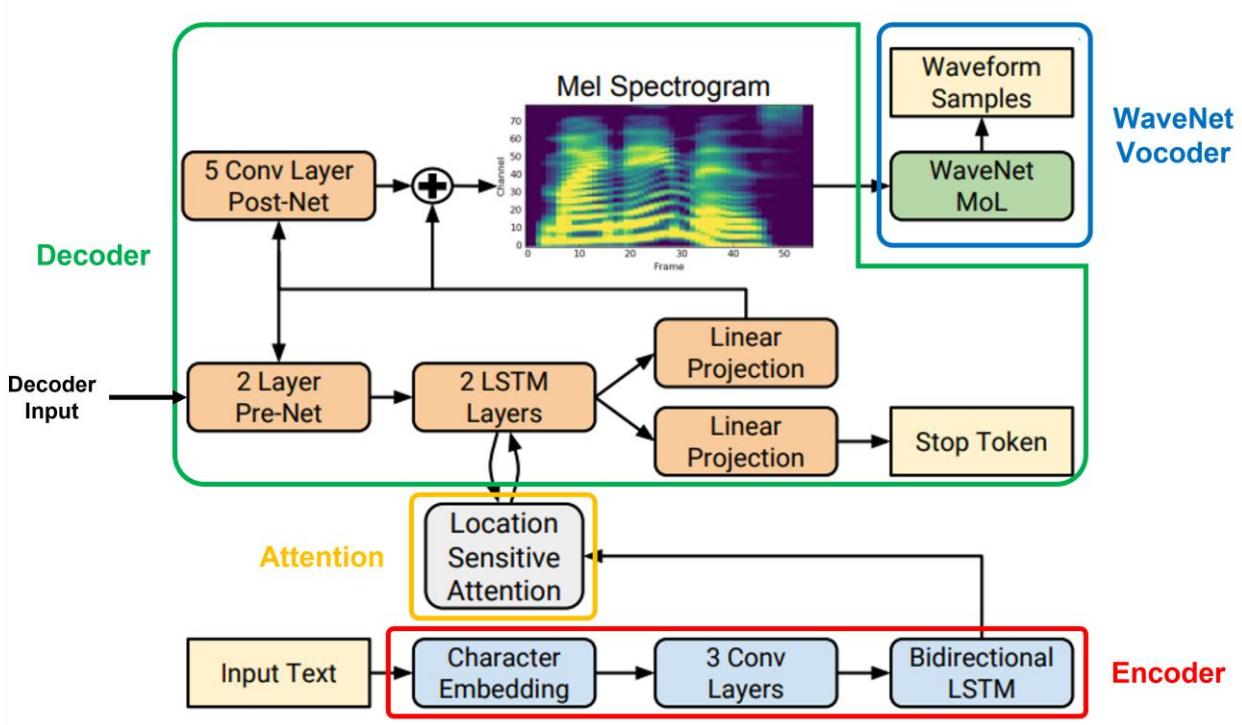


Fig. 1. Block diagram of the Tacotron 2 system architecture.

Part 2 모델의 변경 과정

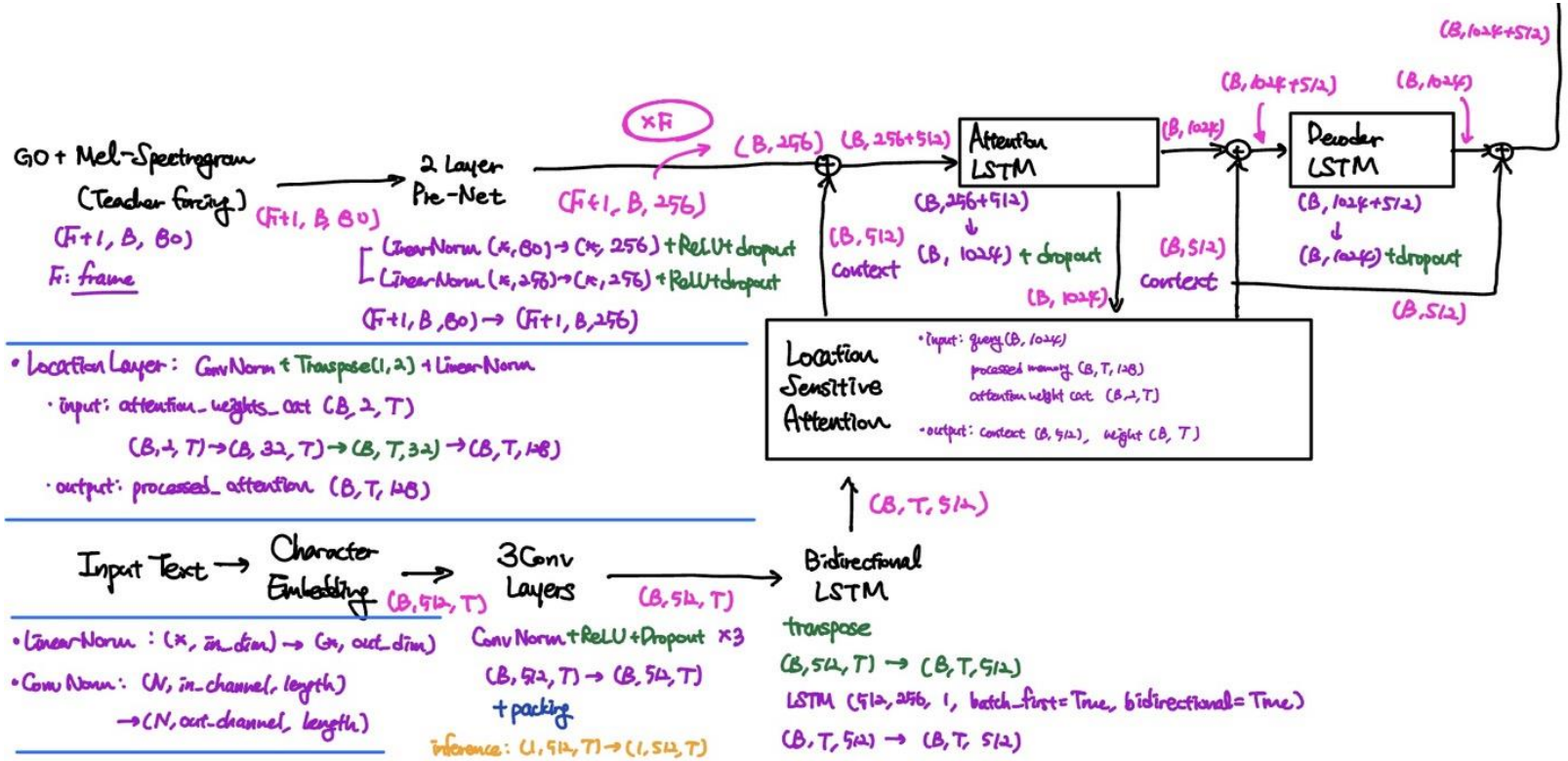
Tacotron2

2018
Google Inc.

End-to-End

Autoregressive
Model

- 세 개의 Github 코드를 참고하며,
  코드를 그대로 따라 쓰지 않고 최대한 스스로 구현



Part 2 모델의 변경 과정

Tacotron2

2018
Google Inc.

End-to-End

Autoregressive
Model

- 세 개의 Github 코드를 참고하며,
  코드를 그대로 따라 쓰지 않고 최대한 스스로 구현

```python
    def forward(self, memory, query, attention_weights_cat, text_len):
        """
        =====inputs=====
        memory: (B, Max_T, 512) # Encoder의 outputs
        query: (B, 1024) # Attention LSTM의 outputs
        attention_weights_cat: (B, 2, Max_T) # 이전 time step의 attention weights과 attention_weights_cum의 concat
        text_len: (B)
        =====outputs=====
        context: (B, 512)
        attention_weights: (B, Max_T) # 현재 time step의 attention weight
        """
        attention_weights = self.get_attention_weights(memory, query, attention_weights_cat, text_len) # (B, Max_T)

        context = torch.bmm(attention_weights.unsqueeze(1), memory) # bmm: batch matrix-matrix product
        # (B, 1, Max_T)@(B, Max_T, 512) = (B, 1, 512)
        context = context.squeeze(1) # (B, 512)

        return context, attention_weights # (B, 512), (B, Max_T)
```

- 1063 lines - kss 데이터를 이용하여 학습
- 8번의 모델 재학습 시도, 학습이 제대로 이뤄지지 않음.

Part 2 모델의 변경 과정

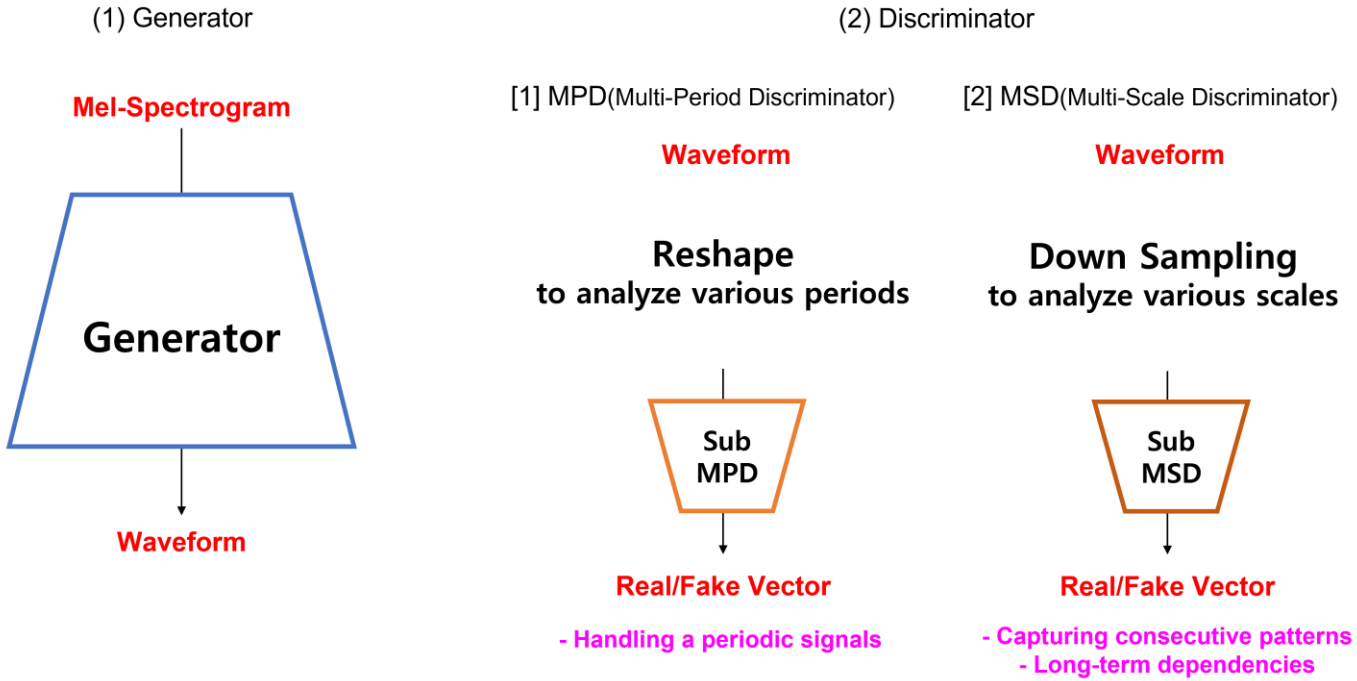
HiFi-GAN

2020
Kakao Enterprise

Vocoder

GAN based
Model

- 매우 빠른 음성 합성 속도, 적은 파라미터 수를 가진 모델



Part 2 모델의 변경 과정

HiFi-GAN

2020
Kakao Enterprise

Vocoder

GAN based
Model

- 논문 저자의 구현을 참고하여 직접 구현

```python
    def forward(self, real_waveform, gen_waveform):
        """
        =====inputs=====
        real_waveform: (B, 1, T) # 실제 음성
        gen_waveform: (B, 1, T) # 생성 음성
        =====outputs=====
        real_outputs: (B, ?) list (len=3) # 실제 음성에 대한 SubSD outputs list
        gen_outputs: (B, ?) list # 생성 음성에 대한 SubSD outputs list
        real_features: features list # 실제 음성에 대한 SubSD features list
        gen_features: features list # 생성 음성에 대한 SubSD features list
        """
        real_outputs, gen_outputs, real_features, gen_features = [], [], [], []
        for idx, sub_sd in enumerate(self.sub_sds):
            if idx != 0:
                real_waveform = self.avgpool(real_waveform)
                gen_waveform = self.avgpool(gen_waveform)
            real_output, real_feature = sub_sd(real_waveform)
            gen_output, gen_feature = sub_sd(gen_waveform)
```

- 897 lines - kss 데이터를 이용하여 학습
- 구현 성공, 좋은 품질의 음성 생성

Part 2 모델의 변경 과정

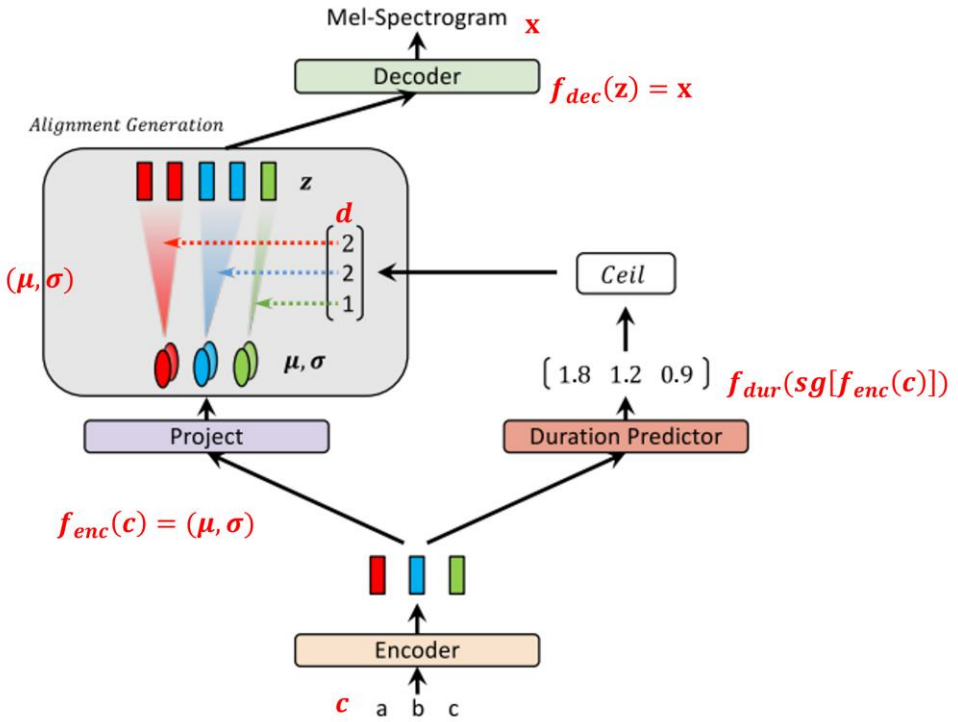
Glow-TTS

2020
Kakao & SNU AI

Mel-Generator

Flow based
Model

- Tacotron2보다 합성 속도가 15.7배 빠른 모델



(b) An abstract diagram of the inference procedure.

Part 2 모델의 변경 과정

Glow-TTS

2020
Kakao & SNU AI

Mel-Generator

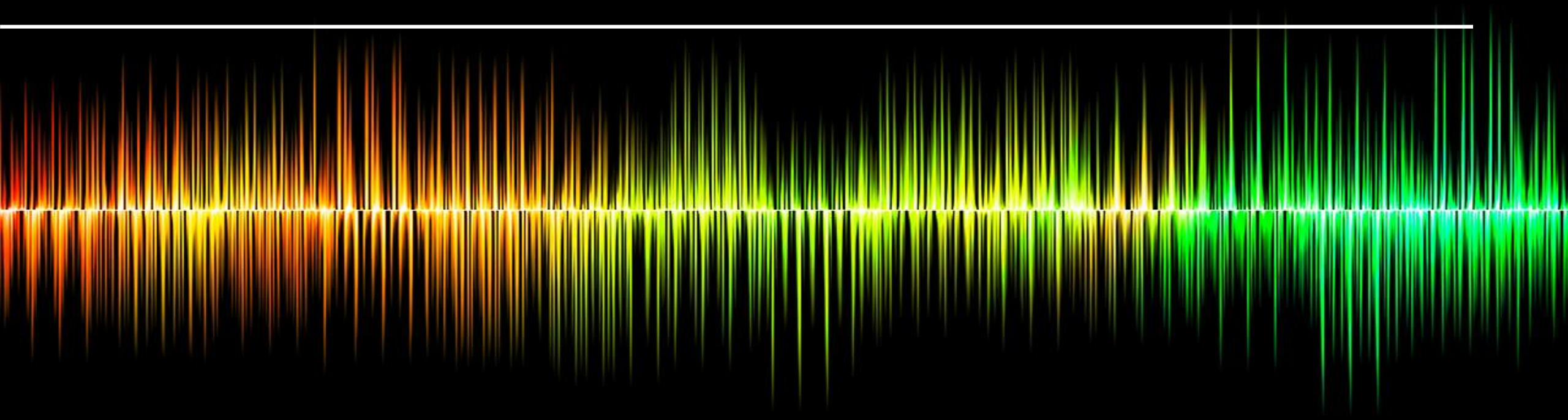
Flow based
Model

- HiFi-GAN과 Glow-TTS 모델을 합쳐 최종 TTS 모델 구현

- 1993 lines

```
def MAS(path, logp, T_max, F_max):
    """
    Glow-TTS의 모듈인 maximum_path의 모듈
    MAS 알고리즘을 수행하는 함수이다.
    =====inputs=====
    path: (T, F)
    logp: (T, F)
    T_max: (1)
    F_max: (1)
    =====outputs=====
    path: (T, F) | 0과 1로 구성된 alignment
    """
    neg_inf = -1e9 # negative infinity
    # forward
    for j in range(F_max):
        for i in range(max(0, T_max + j - F_max), min(T_max, j + 1)): # 평행사변형을 생각하라.
            # Q_i_j-1 (current)
            if i == j:
                Q_cur = neg_inf
            else:
                Q_cur = logp[i, j-1] # j=0이면 i도 0이므로 j-1을 사용해도 된다.
```

Part 3　모델 구현 및 결과



Part 3 Model and Dataset setting

Glow-TTS : the number of parameters 30M

HiFi-GAN : the number of parameters parameter 9.8M V2(light model)

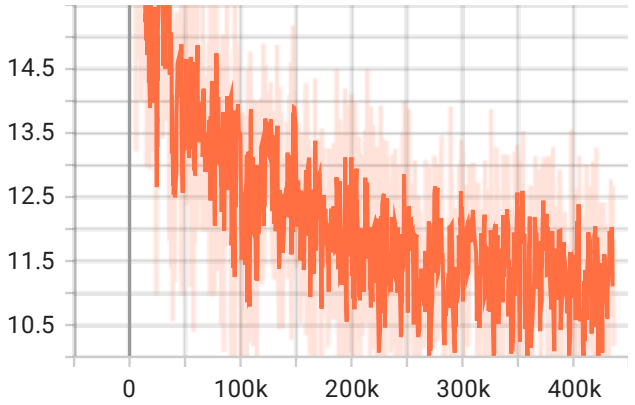
Both Glow-TTS and HiFi-GAN are SOTA model in 2020

KSS(Korean single speaker) : 10H Dataset recoded by professional female voice actoress

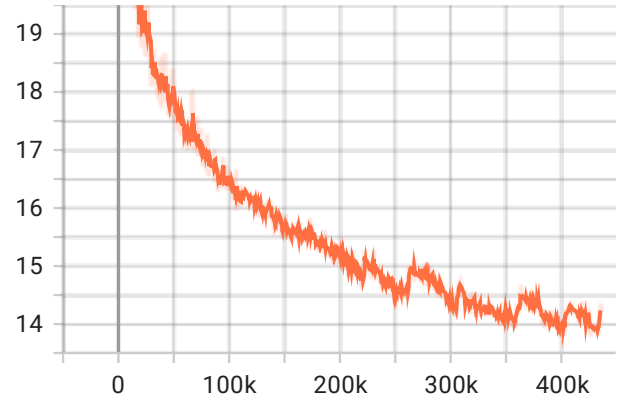
KES(KimEunSik) : 1H Dataset recoded by common person from KSS script

HTY(Teayeon) : 3.5H Dataset 한국어 대화 and 한국어 영어 변역 말뭉치 script from AI Hub

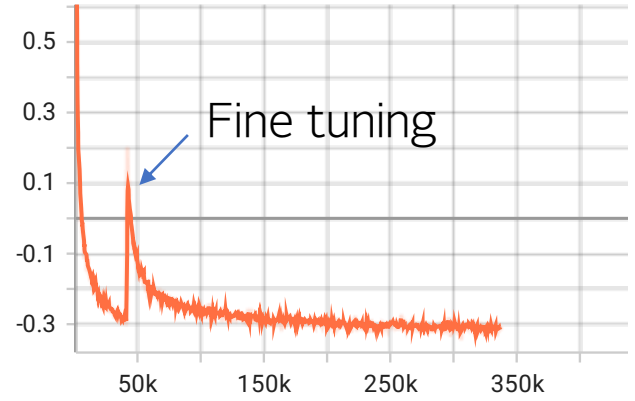
Part 3 Model Loss



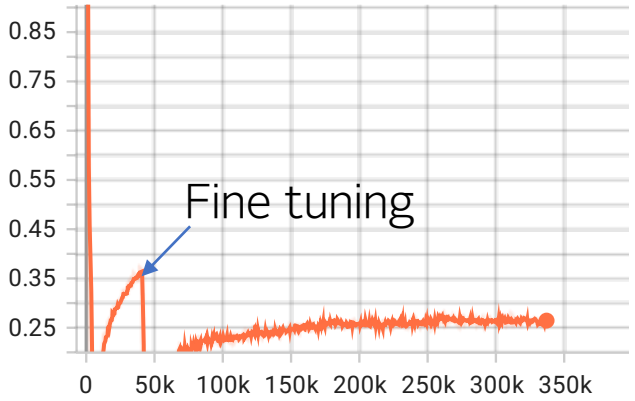
HiFi-GAN Mel-Loss (train)



HiFi-GAN Mel-Loss (validation)

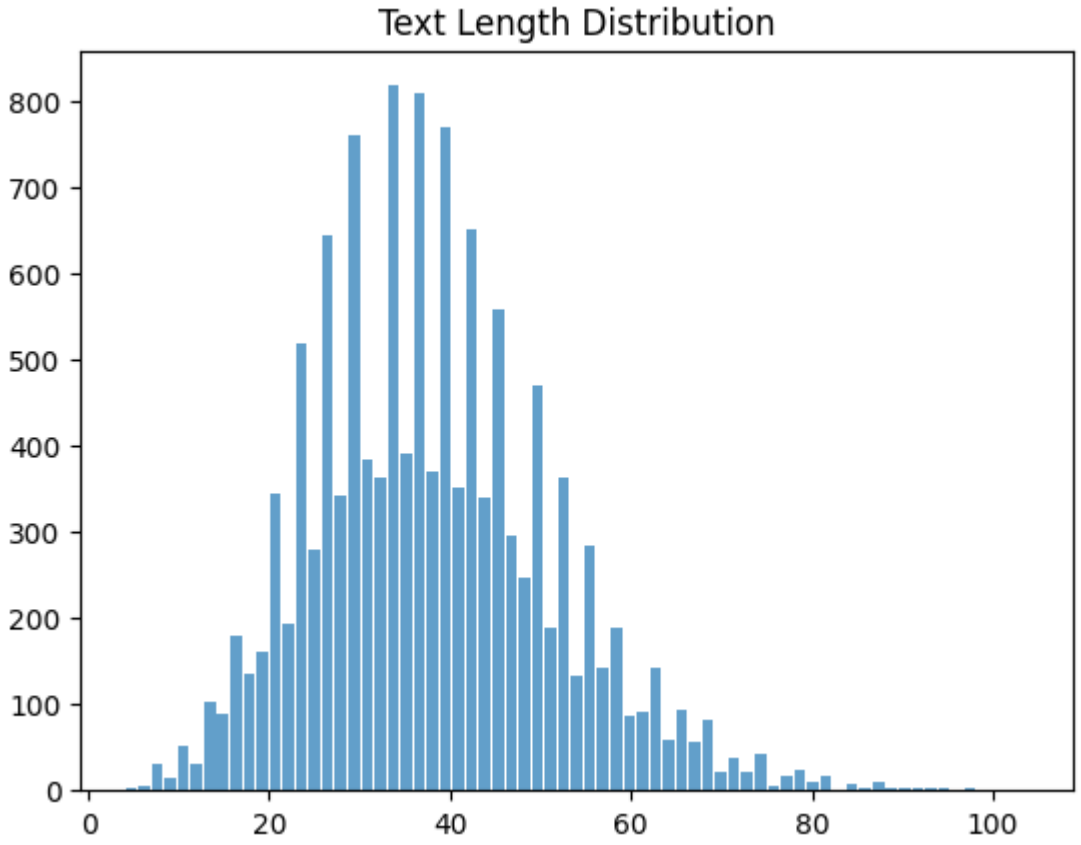


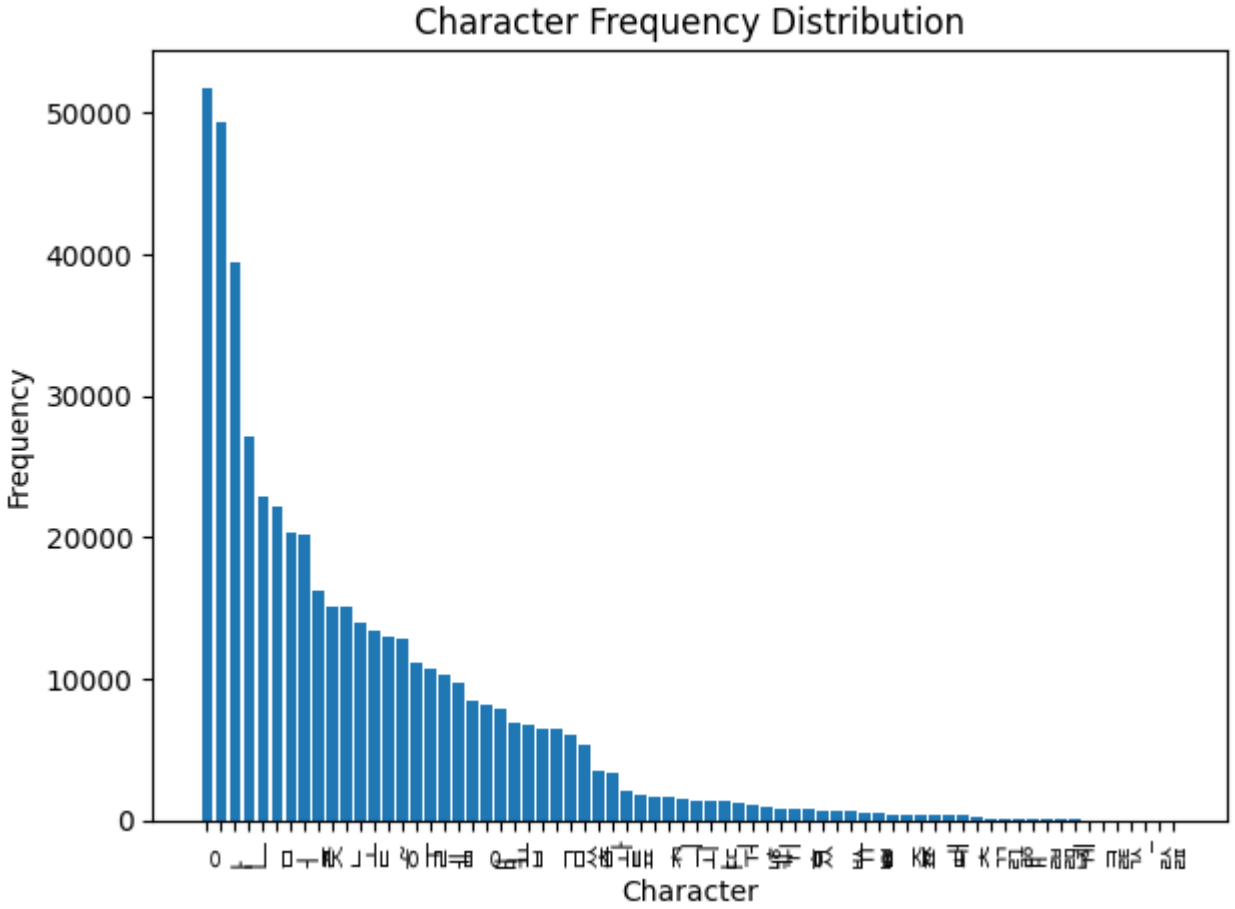
Glow-TTS Loss (train)



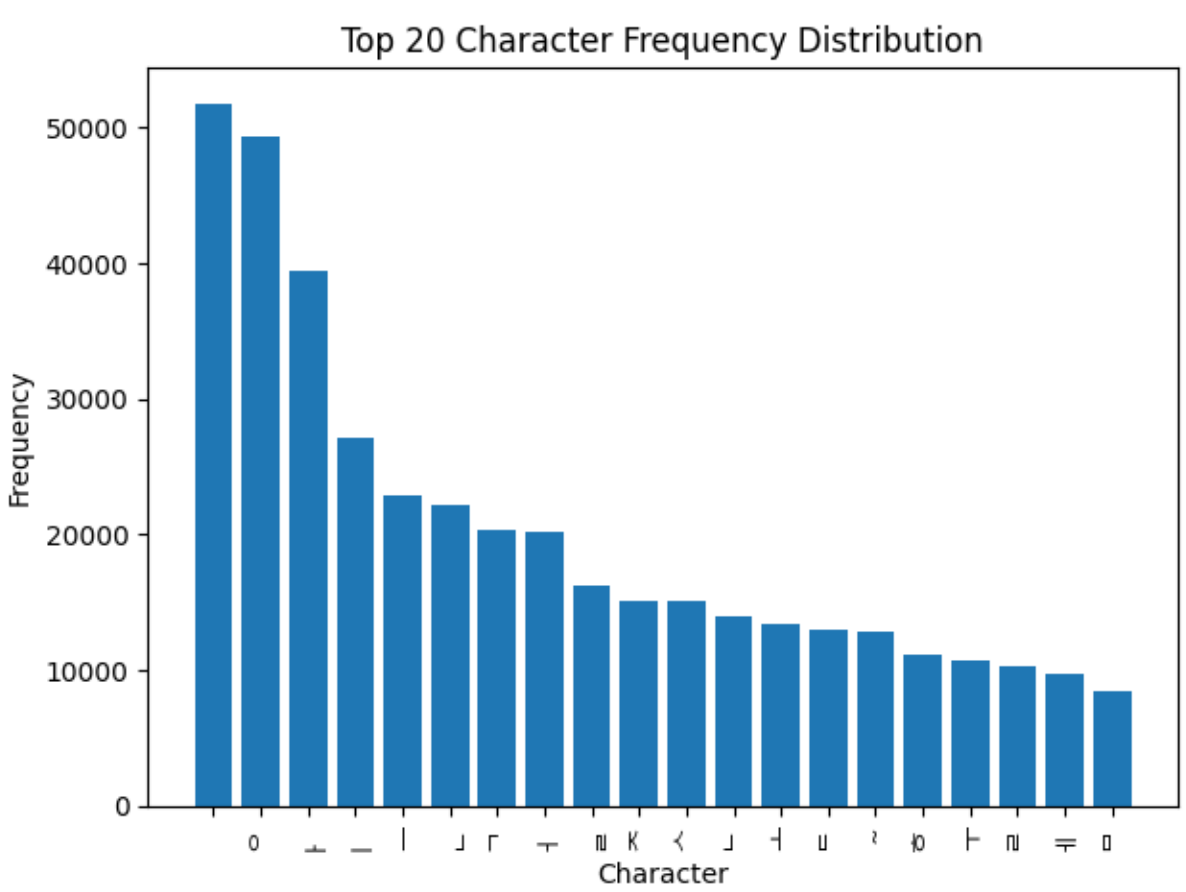
Glow-TTS Loss (validation)

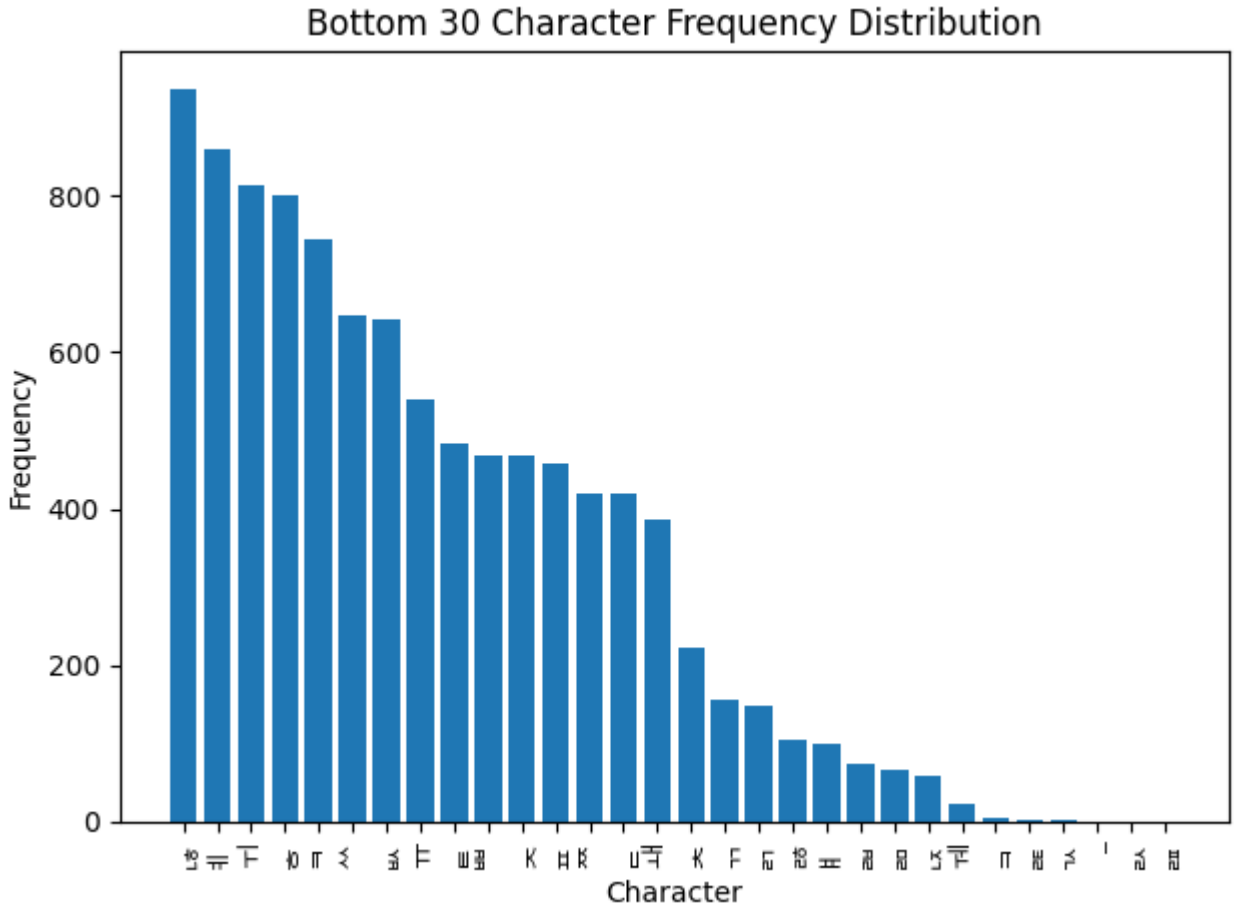
Part 3 Dataset analysis-KSS





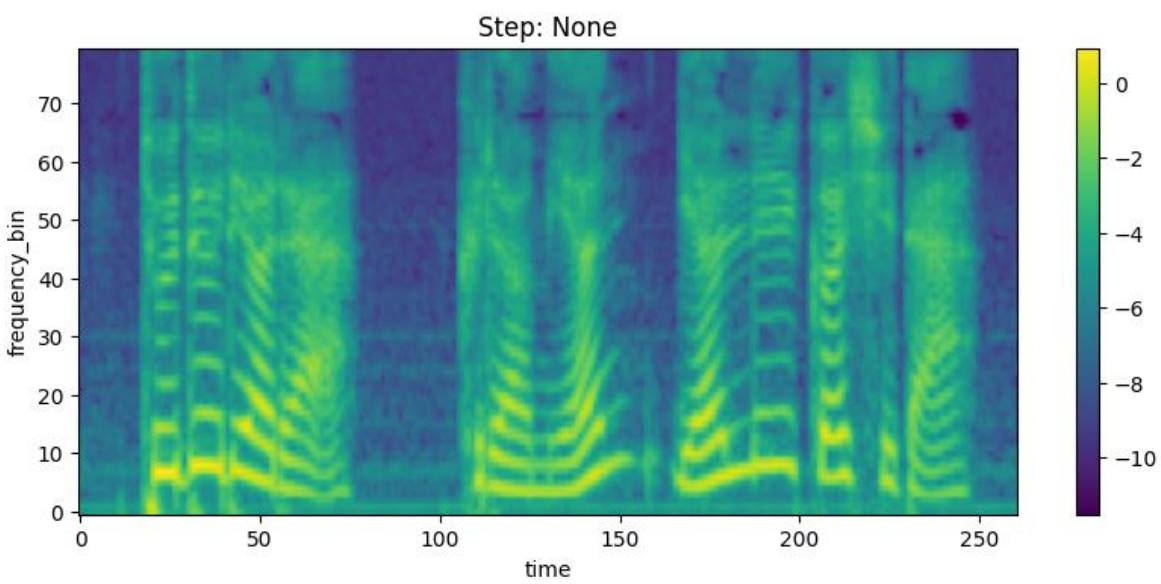
Part 3 Dataset analysis-KSS



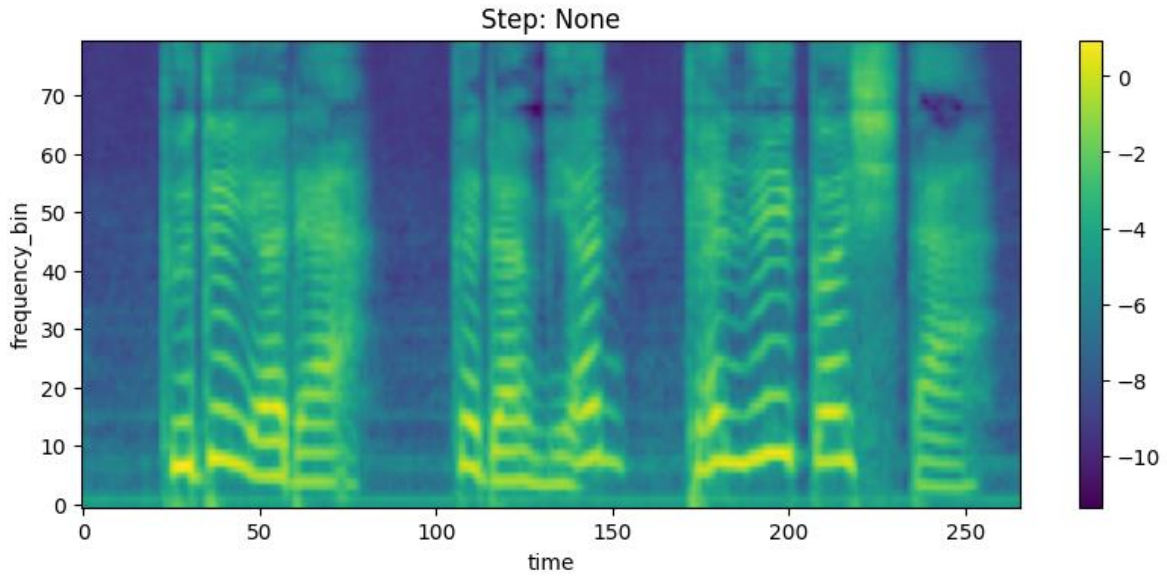


Part 3 Outcome-KSS

origin

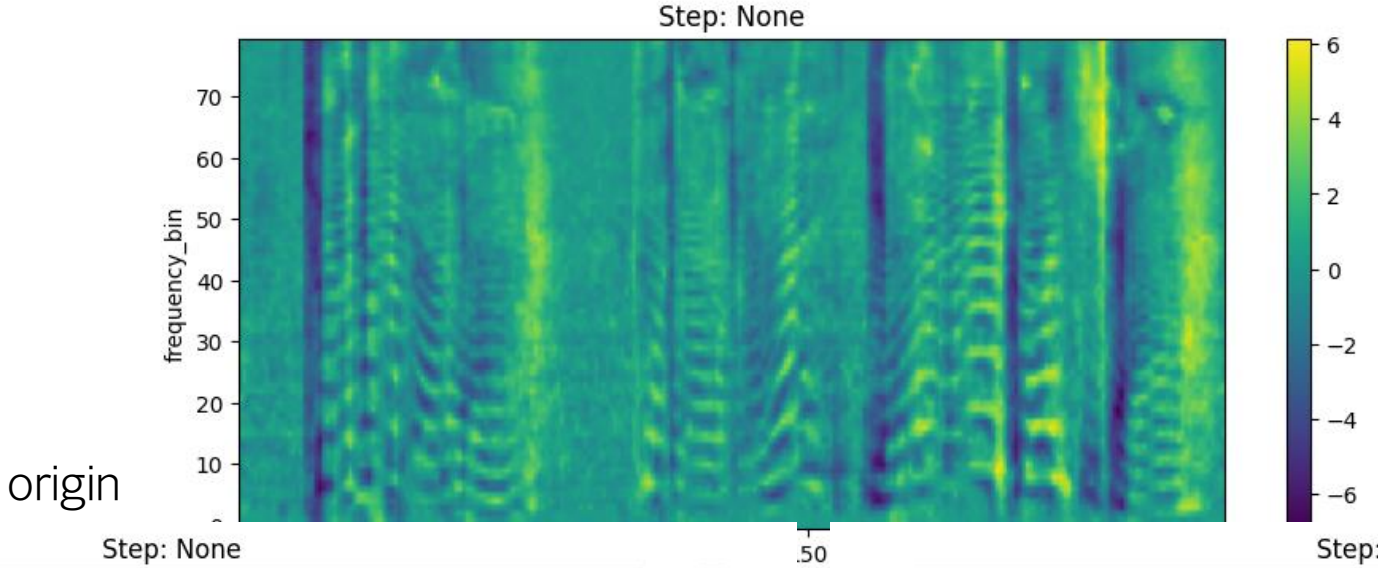


generated



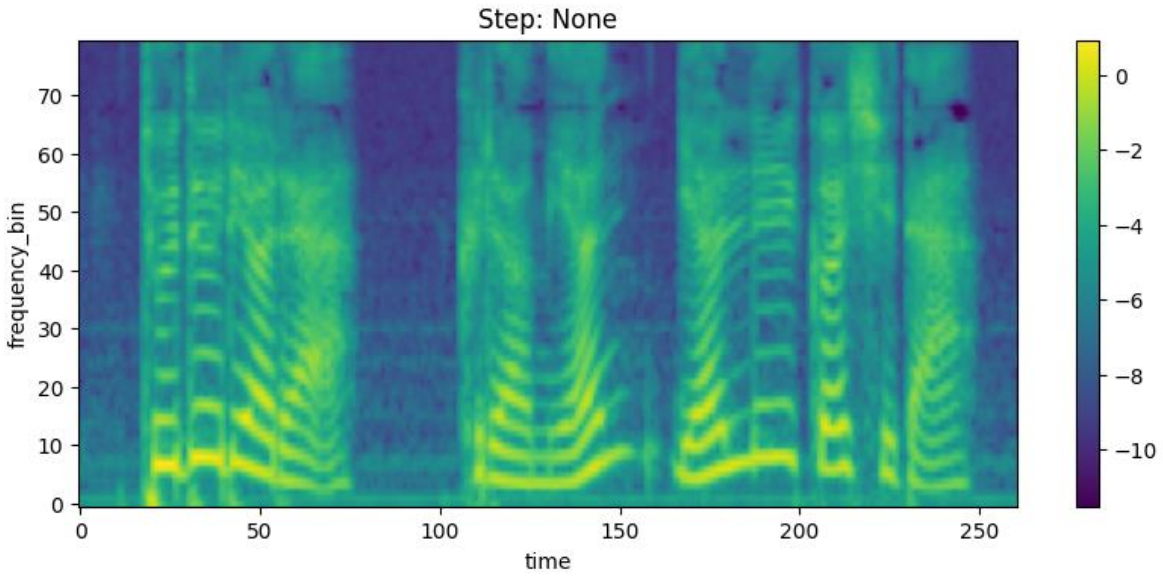
Glow-TTS 280000step
HiFi-GAN 135000step

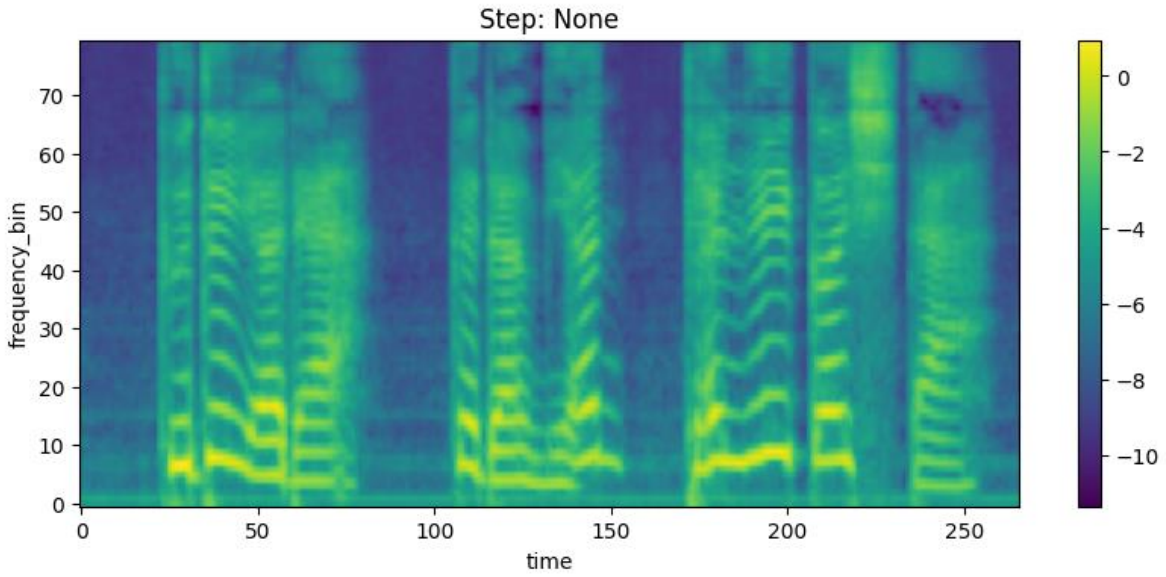
Part 3 Difference



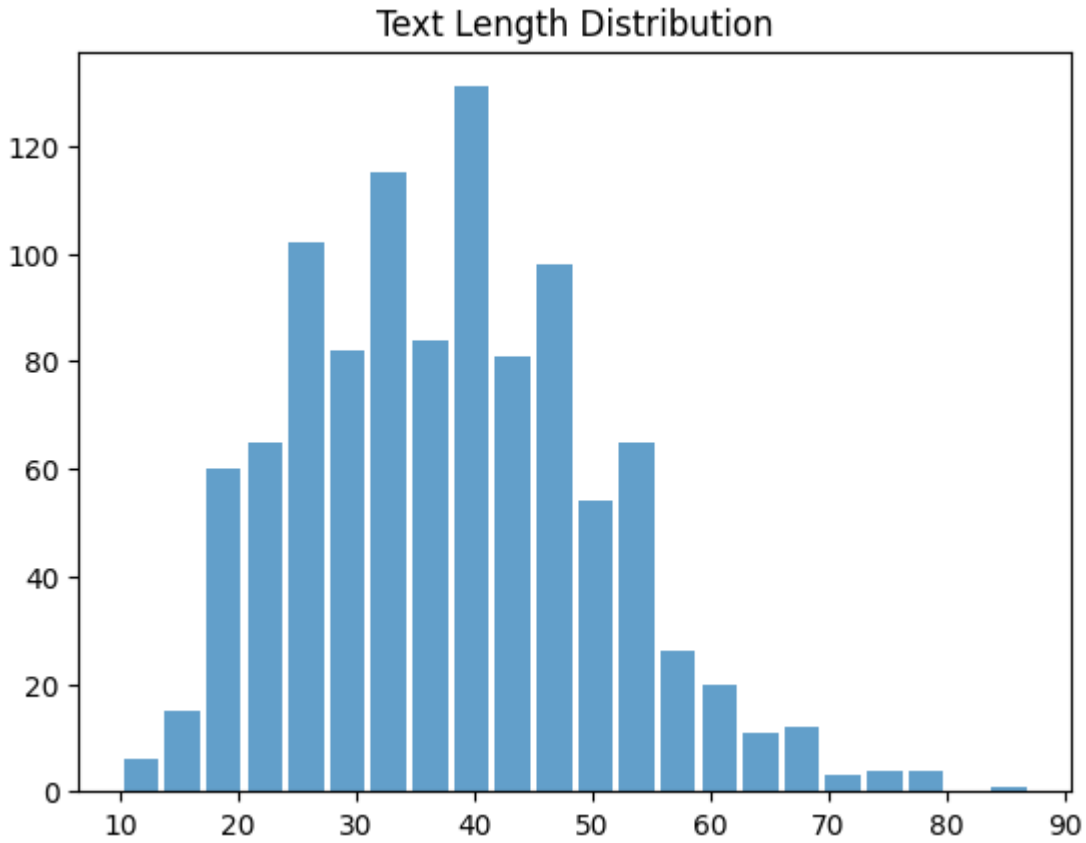
origin

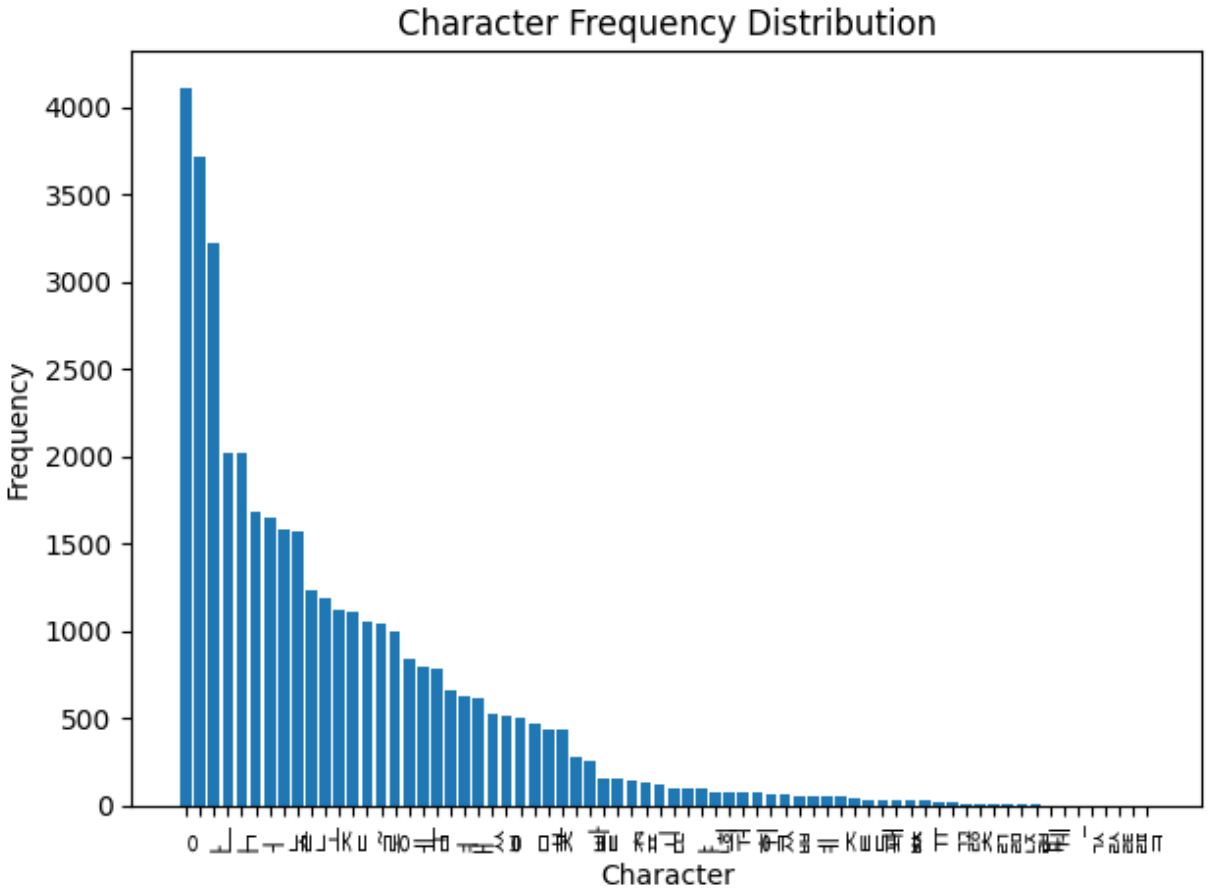
generated



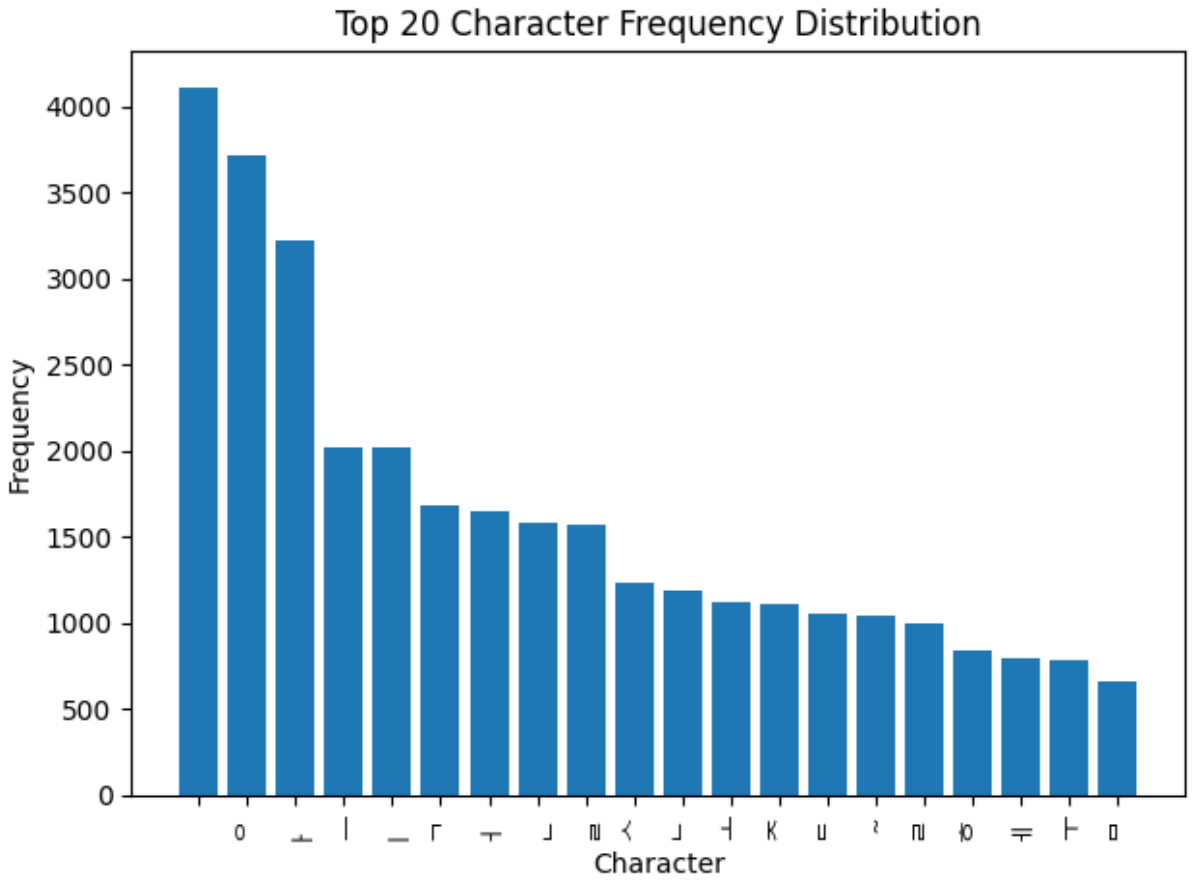


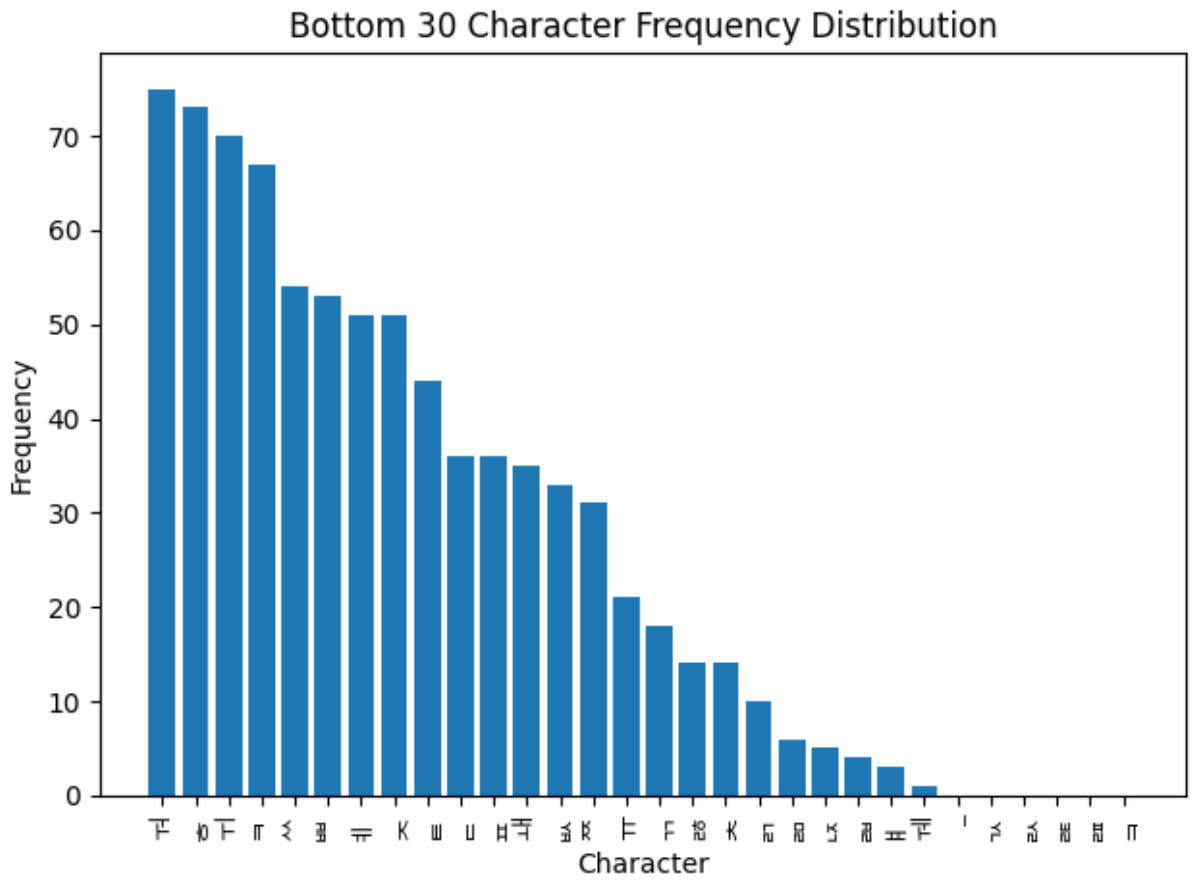
Part 3 Dataset analysis-KES





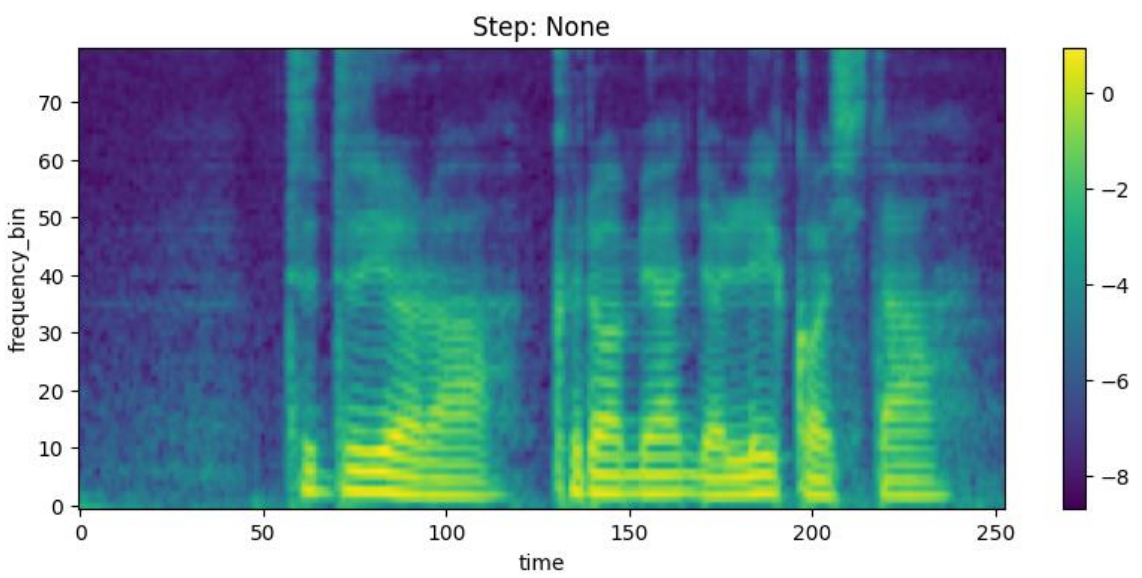
Part 3 Dataset analysis-KES



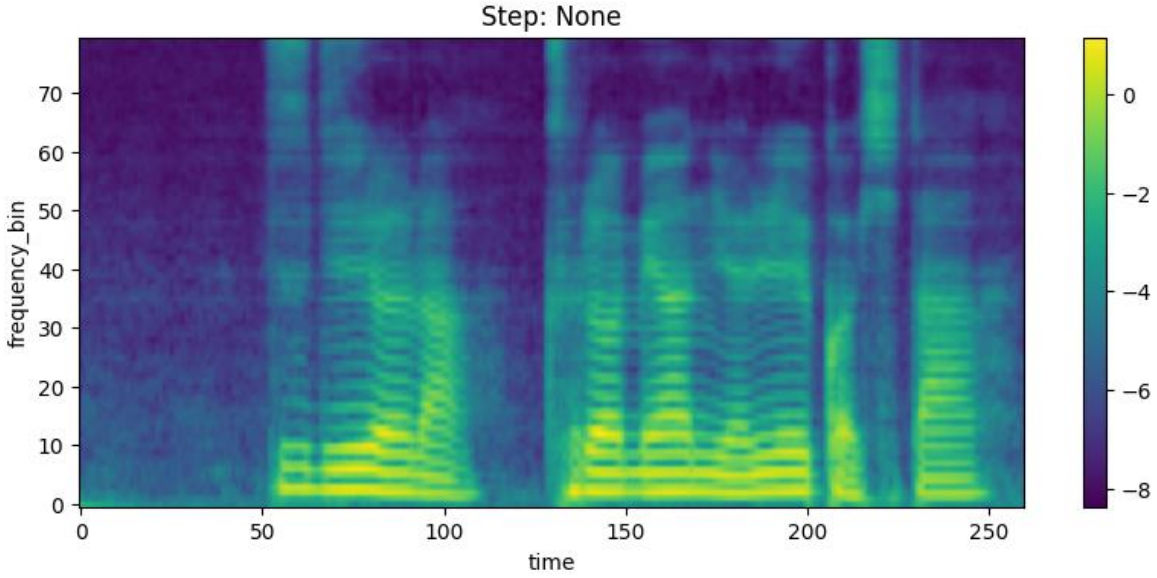


Part 3 Finetuning outcome-KES

origin



generated

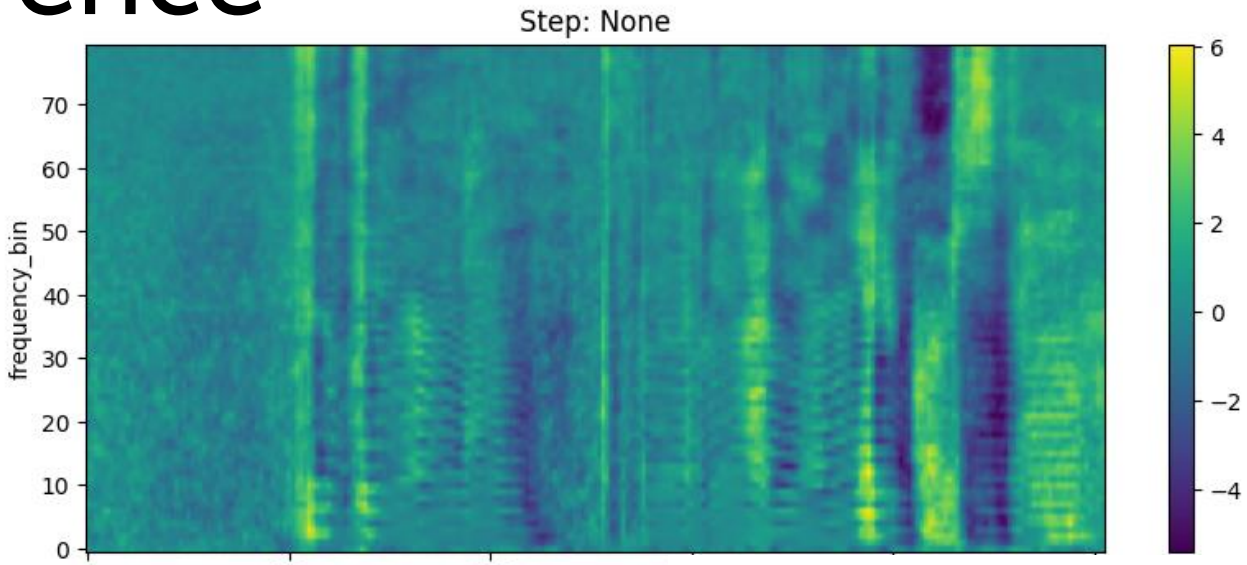


Glow-TTS 280000step(KSS) > 290000 step
HiFi-GAN 400000step(HTY) > 664000 step

Part 3 Difference

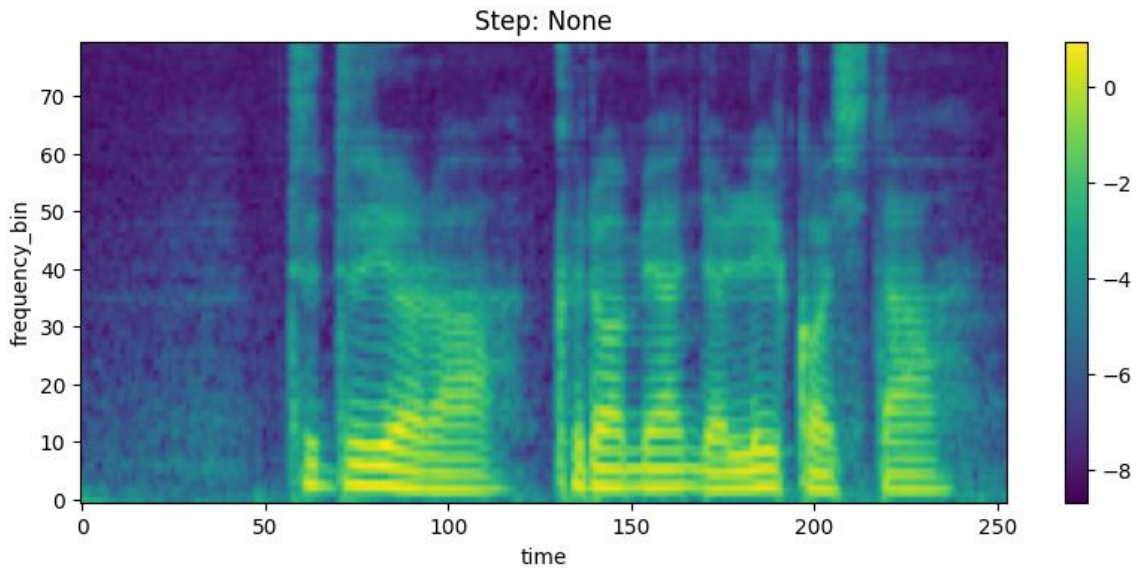
KES > KSS

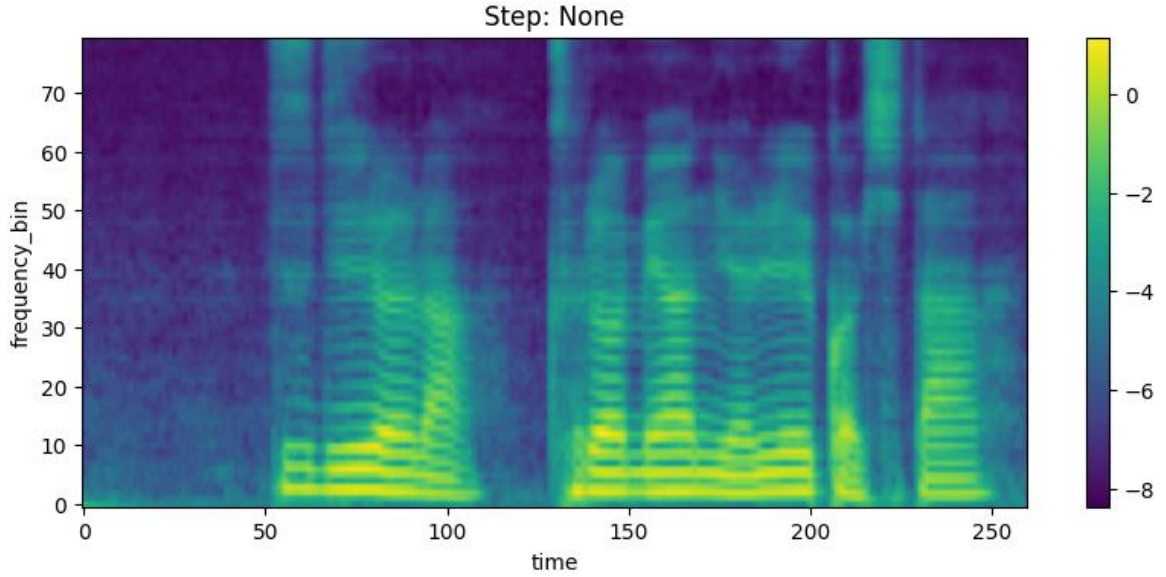
origin



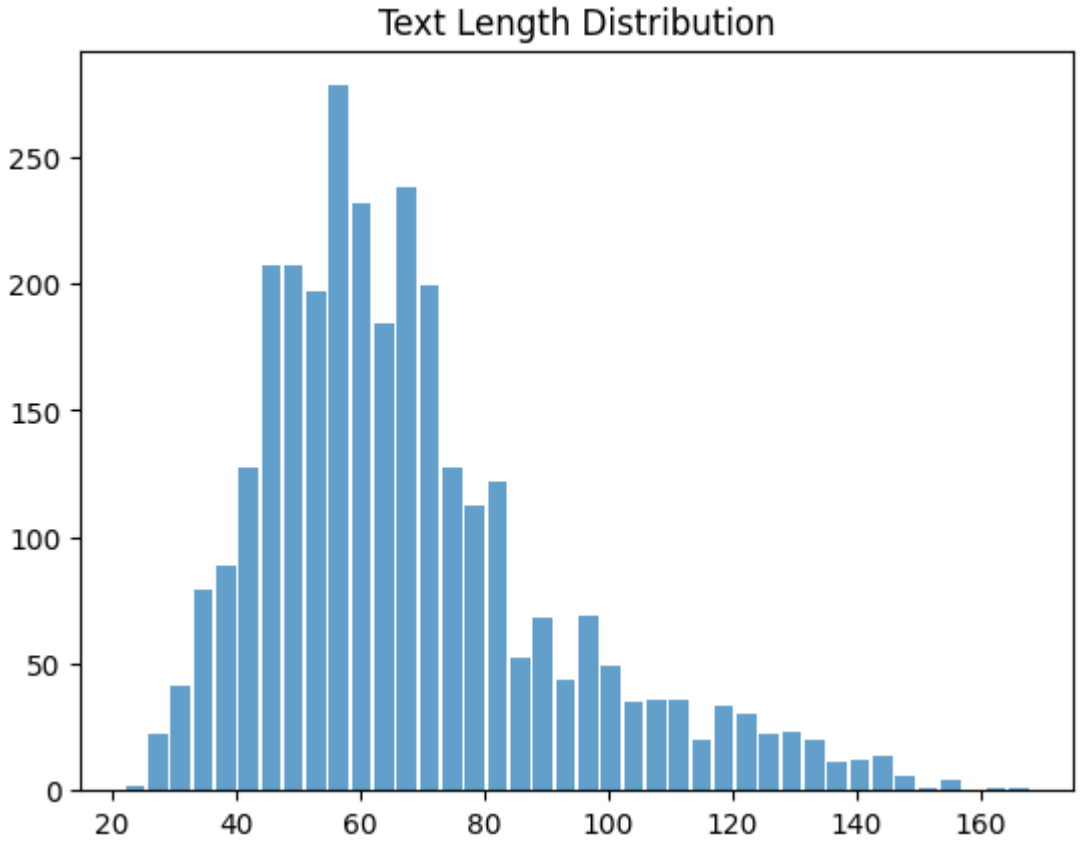
KSS > KES

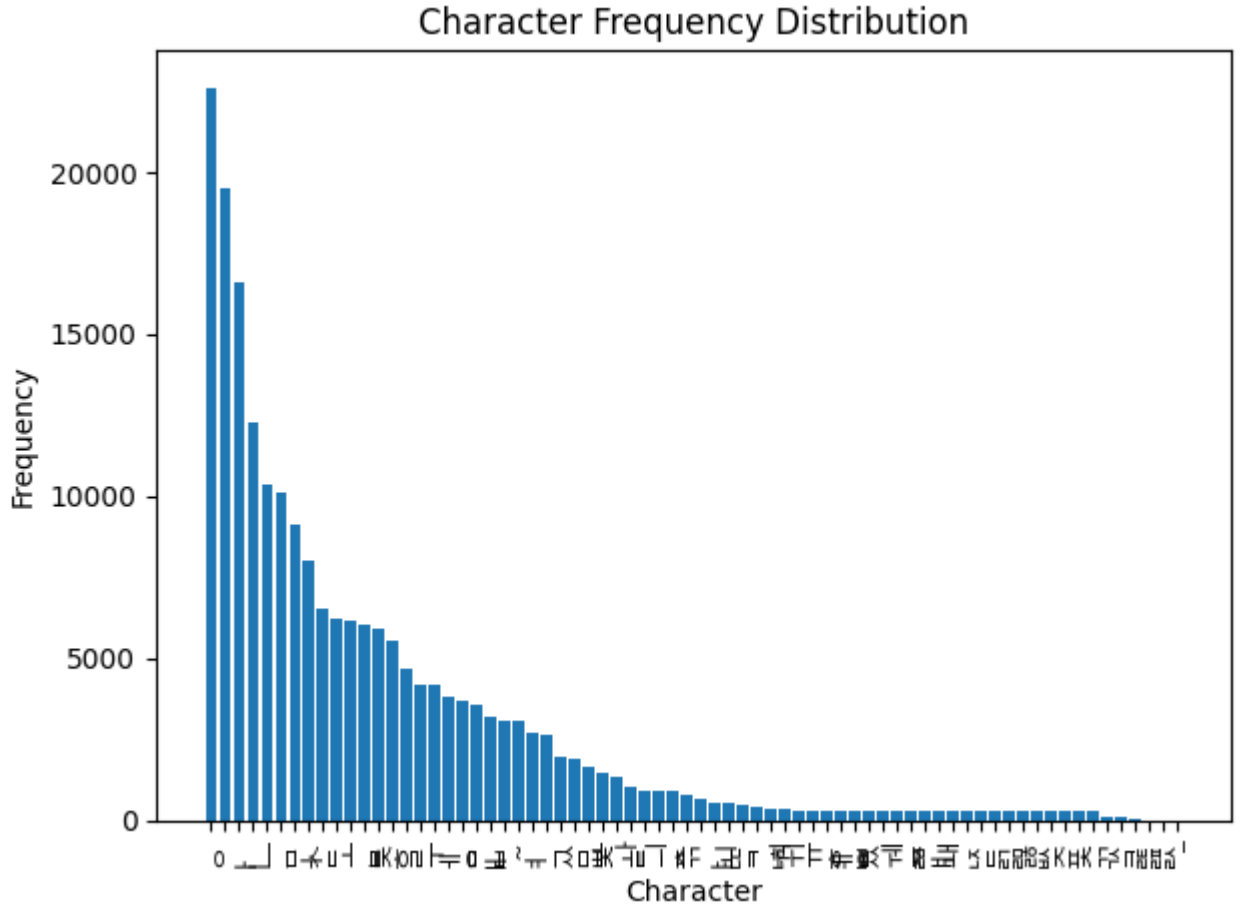
generated



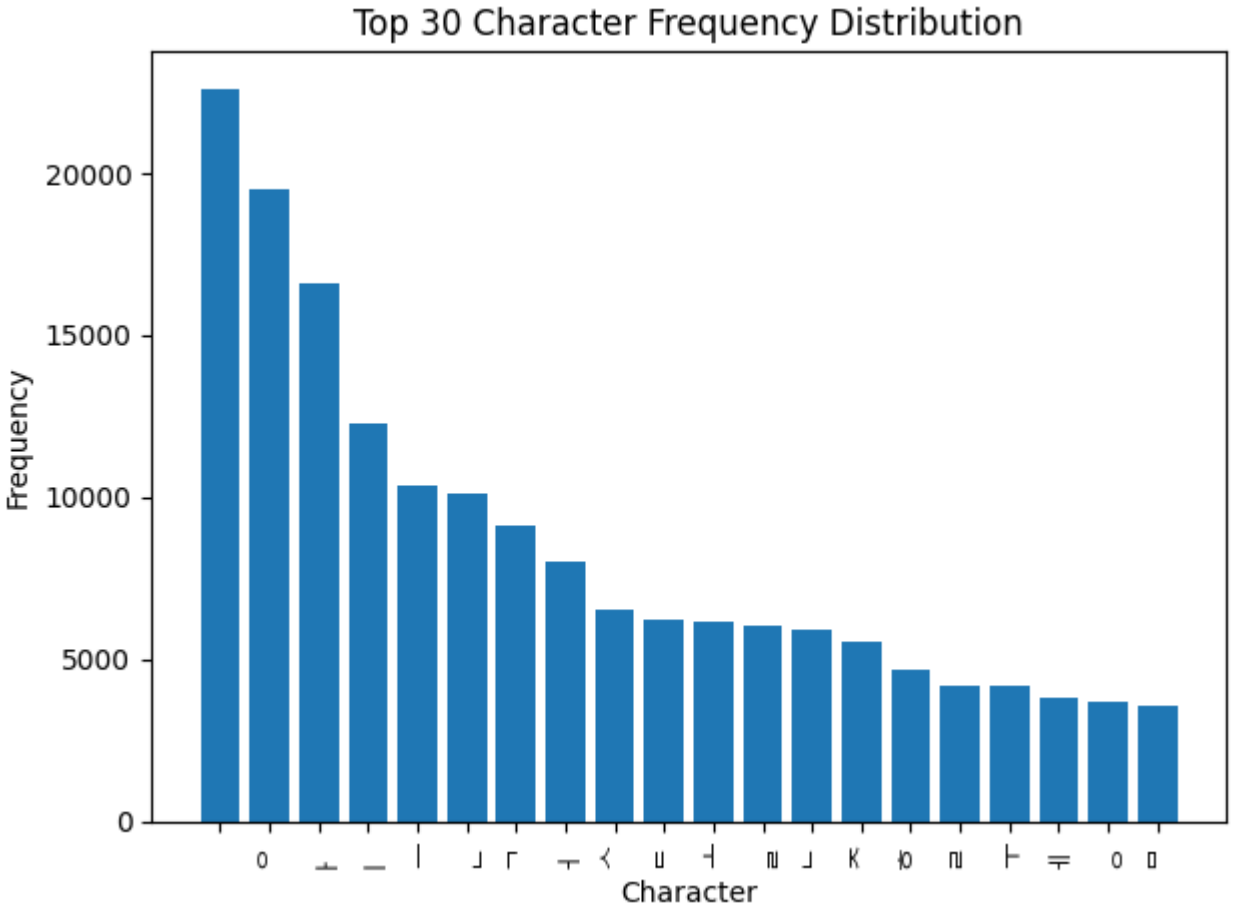


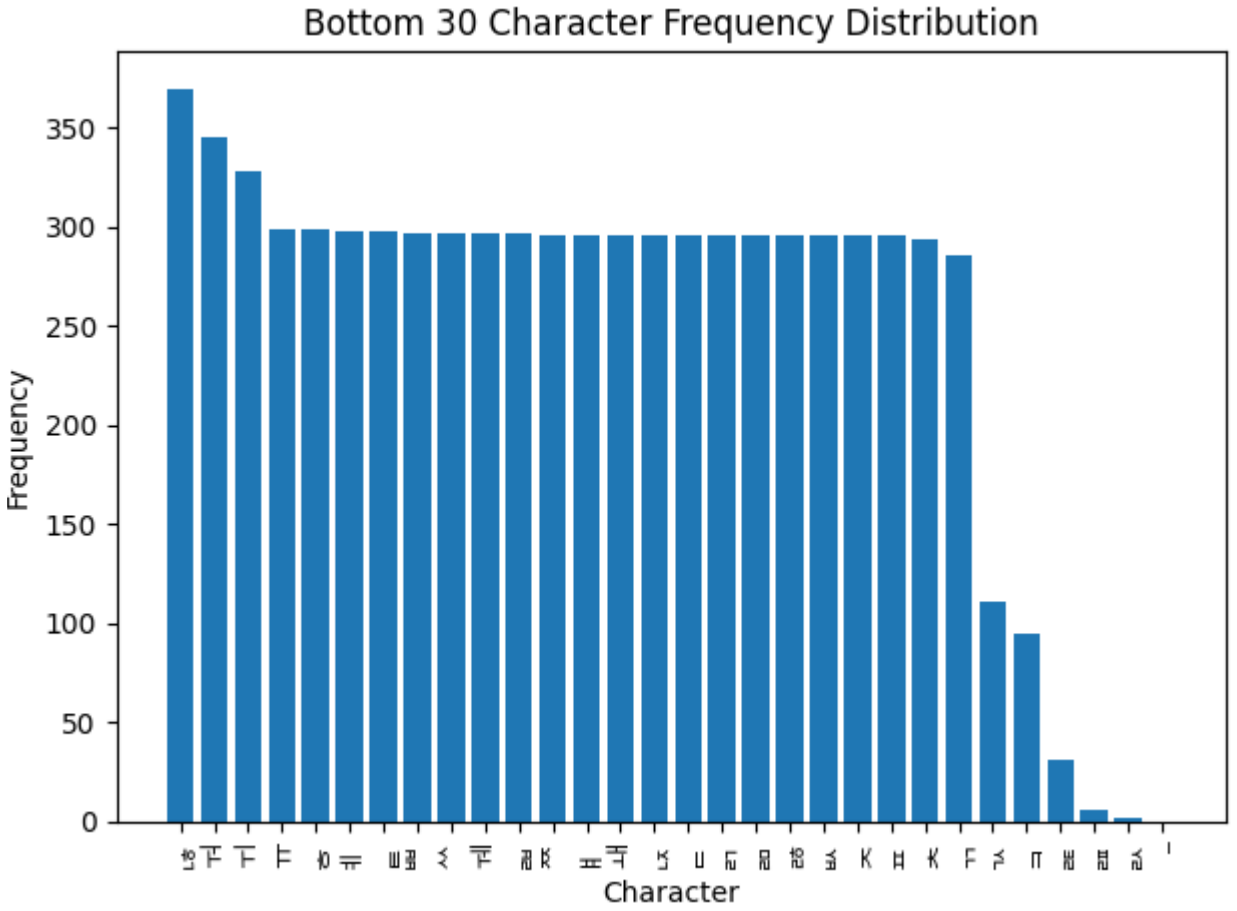
Part 3 Dataset analysis-HTY





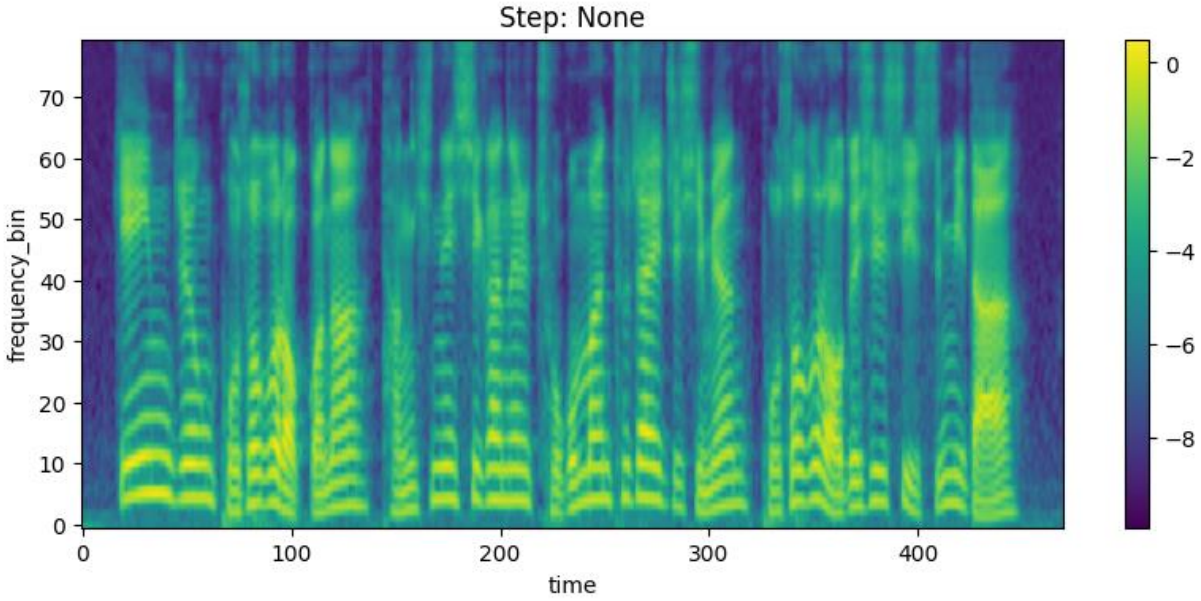
Part 3 Dataset analysis-HTY



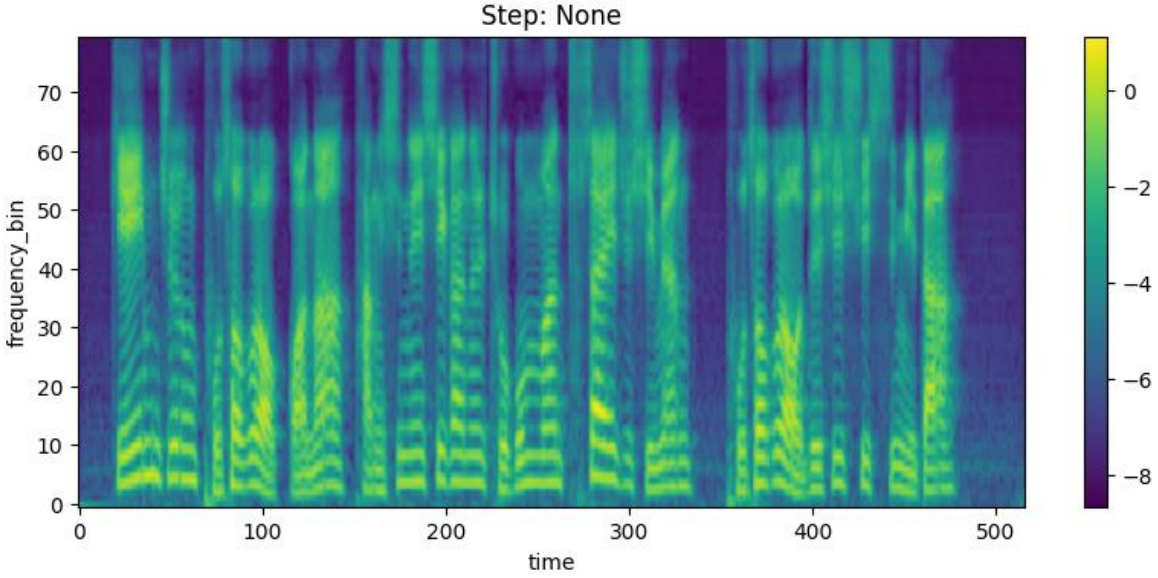


Part 3 Dataset analysis-HTY

origin



generated

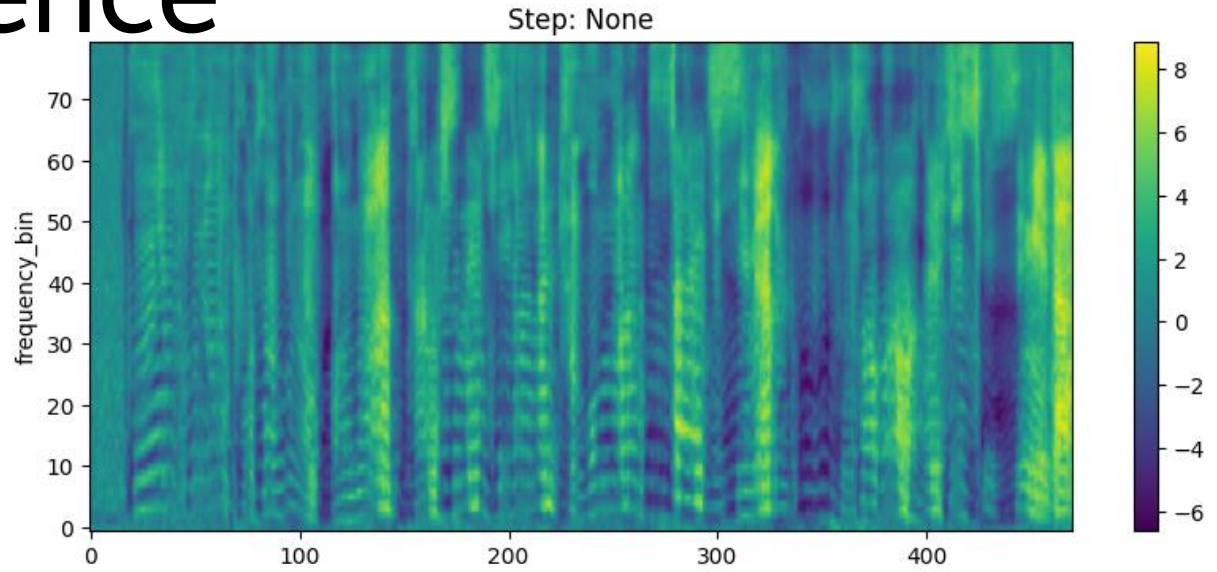


Glow-TTS 40000step(SCE) > 337000 step
HiFi-GAN 400000step(HTY)

Part 3 Difference

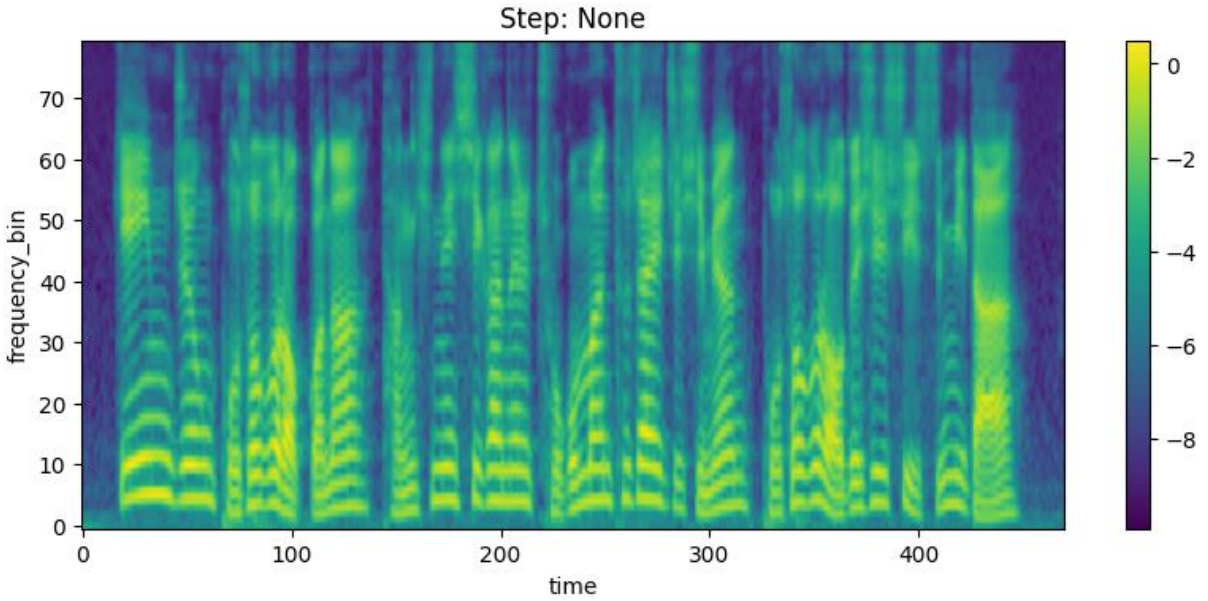
KES > HTY

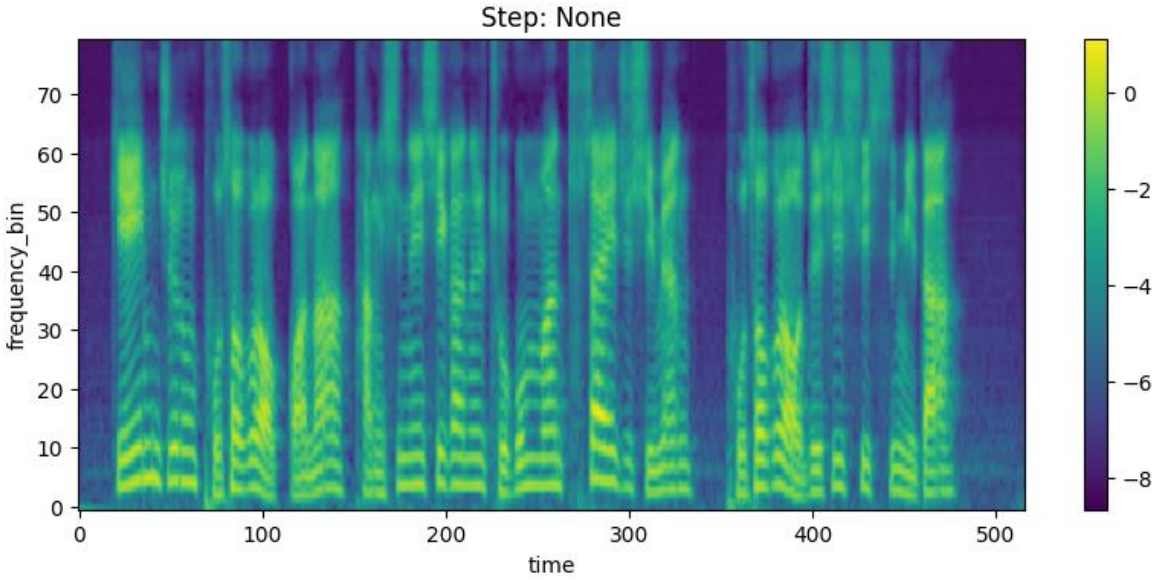
origin



HTY > KES

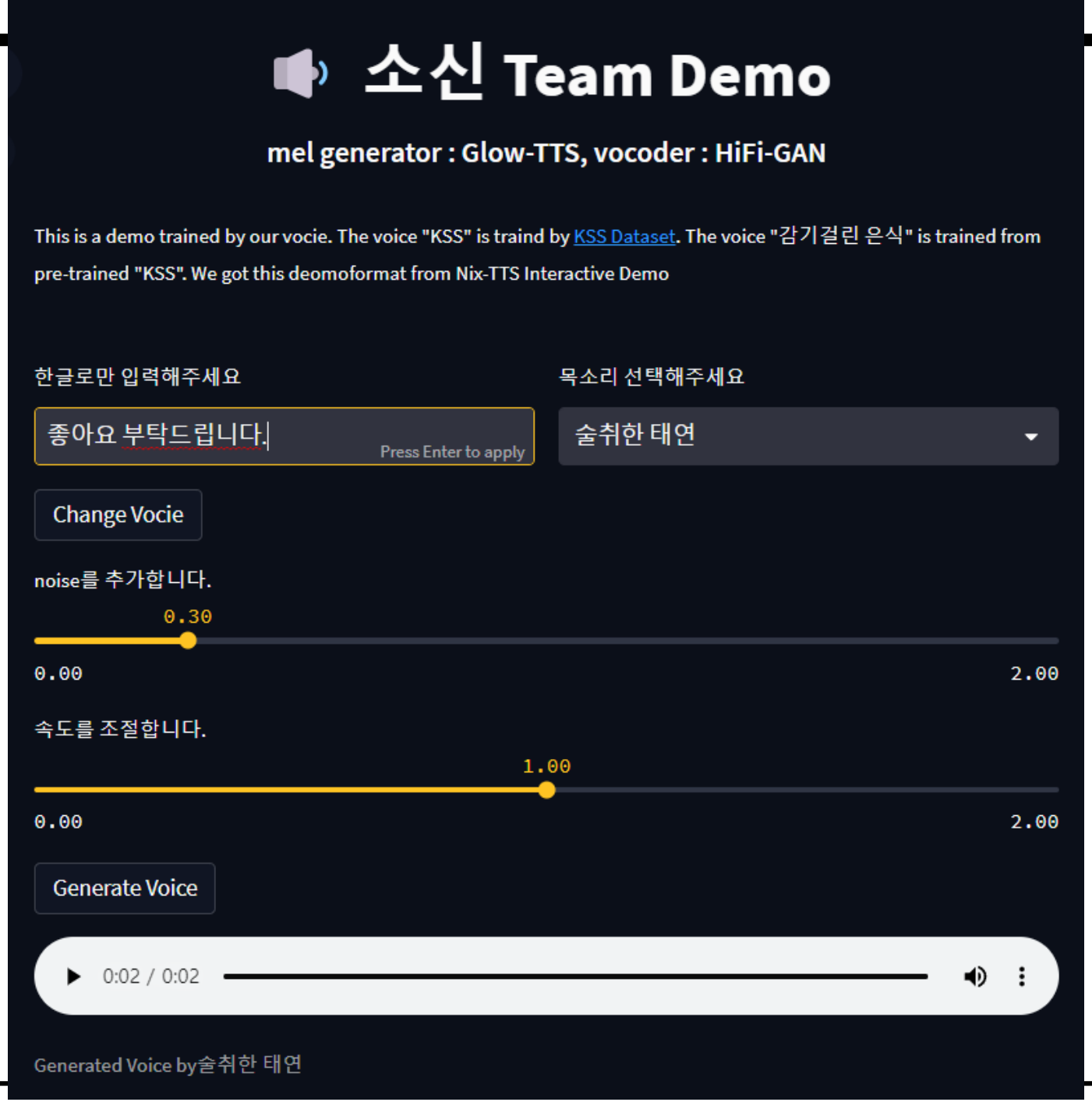
generated



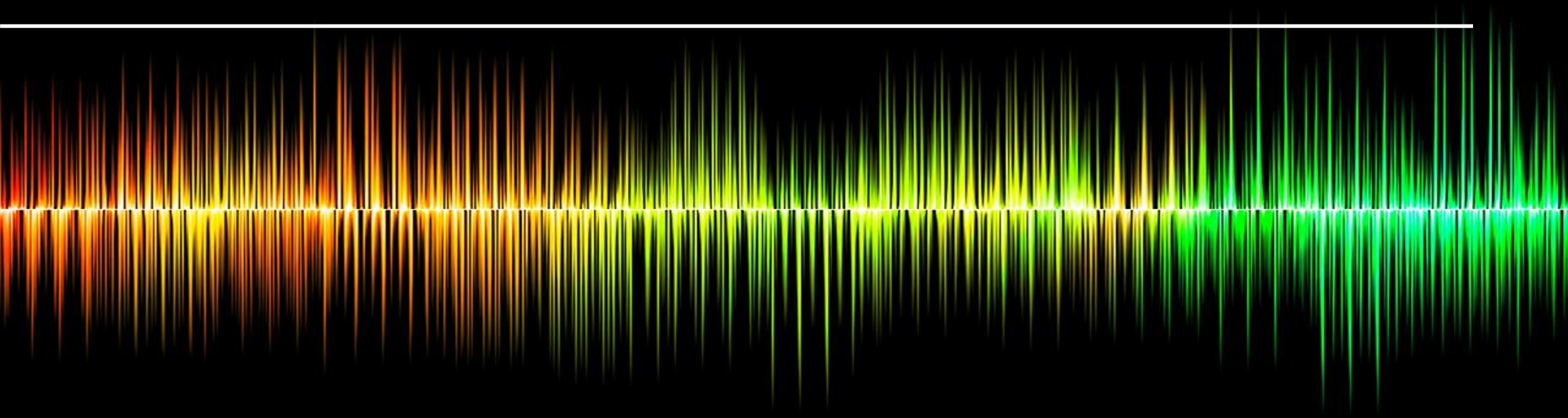


Part 3 Demo

Demo
huggingface SPACE



Part 4　개선점 및 TTS 적용



Part 4 Improvements

더 최신의 SOTA model(VITS, JETS) 구현 혹은 model 구조 변형 시도

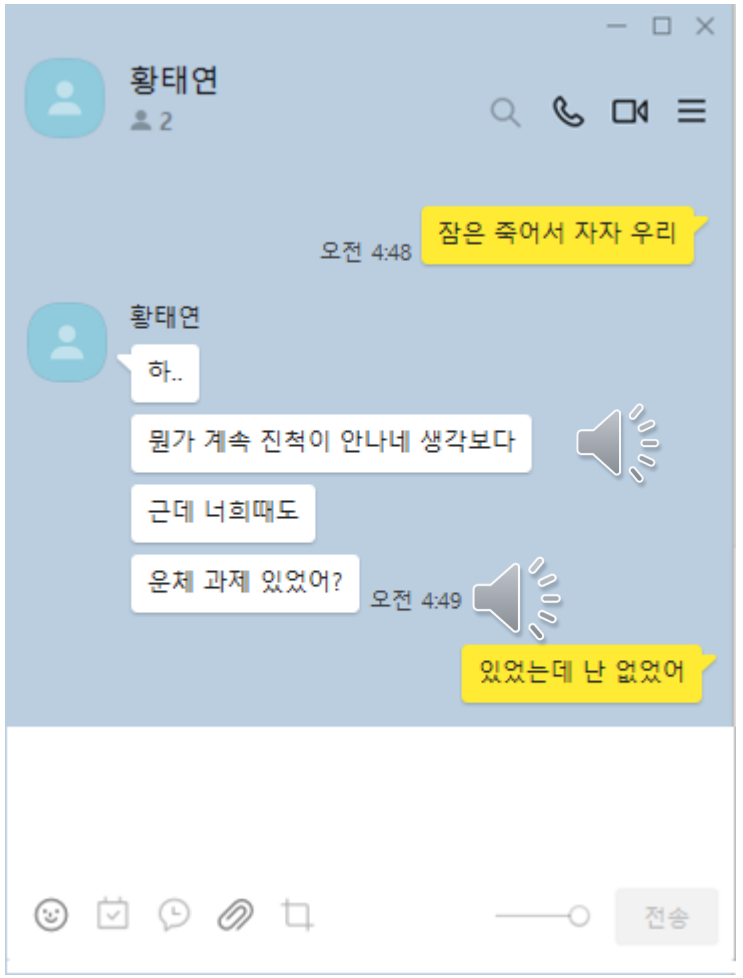
최적의 hyperparameter를 찾기 위한 experiment

Voice adaption을 위한 making dataset에 대한 통계적 분석

TTS 학습 시킬 Dataset 생성위해 STT Model 필요

>>> 더 나은 TTS를 만들 수 있는 여지 존재

Part 4 Chatting with TTS



>> 앱 개발 예정
(to be continued..)

Part 4 Voice Acting with TTS

태연bot 

은식bot   

감 사 합 니 다.