NaturalSpeech 2: Latent Diffusion Models are Natural and Zero-Shot Speech and Singing Synthesizers

- **Arxiv (23.04.18)**
- **Microsoft Research Asia & Microsoft Azure Speech**
- **Paper**, **Demo**

최예린

서강대학교 인공지능학과

Email: lakahaga@u.sogang.ac.kr

2023.4.26

Overview

- 현재 대규모 TTS 모델은 Discrete token을 사용하는데
  - 이는 sequence 길이가 너무 길어서 LM이 생성할 때 unstable하다는 단점

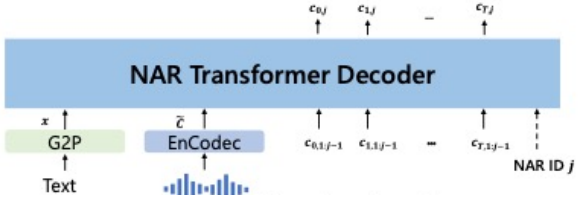
- 기존 discrete token을 continuous한 vector로 바꾸고, 이를 diffusion model을 이용해서 생성하도록 해보자
  - Continuous vector를 사용하면 Each audio frame -> one vector로 표현 => hidden sequence의 길이가 짧아짐

- 데모를 들어봤을 때, Google의 Spear TTS 보다 좋지는 않음
- 하지만, 생성해야하는 sequence의 길이가 길어지는 것을 해결해보려고 했다는 것과
- zero-shot Singing TTS까지 했다는 점

기존 Large-scale TTS의 문제점

- 기존 Large-scale TTS models: VALL-E, SPEAR-TTS, FoundationTTS

- LM이 생성해야 하는 sequence의 길이가 너무 김 -> 생성이 불안정
  - 기존 코덱 모델들은 speech reconstruction 품질을 위해 multiple residual quantizer를 사용하기 때문에 token sequence 길이가 길어짐
    - 예를 들어, quantizer가 8개면, 한 speech frame마다 8개의 token ([Batch, Time, 8])
  - Error propagation 등 unstable한 문제가 생김
  - 시간도 오래걸림



- VALL-E 의 approach
  - Sequence에서 Speech frame 의 8개의 token 중 첫번째 토큰들만 AR로 생성
    - [Batch, Time, C_1]
  - 나머지 7개는 NAR로 생성 -> 같은 NAR transformer로 7번 생성하도록 함
    - Input: text + speaker prompt + 이전 quantizer 생성한 것
    - [Batch, Time, C_{i-1}] -> [Batch, Time, C_i]

FoundationTTS

- *FoundationTTS: Text-to-Speech for ASR Customization with Generative Language Model*
- Phoneme encoder의 output을 prefix/prompt로 하여 speech token을 생성
- Speech token의 길이를 줄이기 위해서 Hierarchical Audio Codec 을 구성
  - Fine-grained audio codec : waveform으로부터 continuous feature를 구성
  - Coarse-grained audio codec: fine-grained audio codec을 Input으로 받아서 보다 적은 quantizer로 discrete representation을 encode
    - Quantizer 개수가 적어짐
    - LM이 생성해야 하는 sequence의 길이도 줄어듦
  - Coarse-grained 를 LM이 생성하게 함
- Inference
  - Coarse-grained -> fine-grained
  - Fine-grained -> waveform

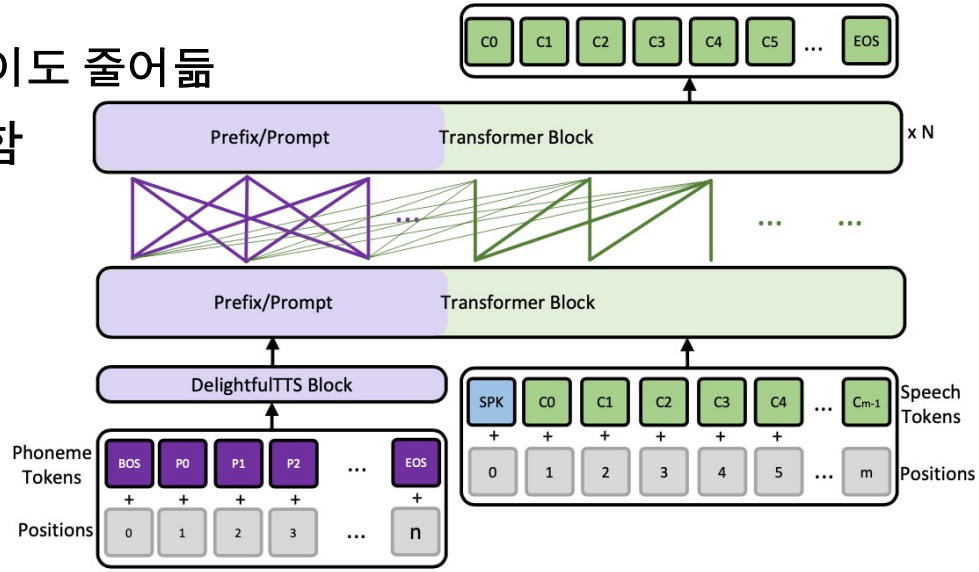


Figure 1: *Prefix language model of FoundationTTS. DelightfulTTS blocks act as the phoneme encoder with phoneme tokens and position embedding as input. A speaker token is also added to the prefix noted as SPK. A set of transformer decoders perform autoregressive decoding of speech tokens with full phoneme tokens and speaker tokens.*

Main idea

- 기존 discrete token을 continuous한 vector로 바꾸고, 이를 diffusion model을 이용해서 생성하도록 해보자

- Continuous vector를 생성하도록 하는 audio codec model을 훈련
- Text로부터 continuous vector를 생성하도록 하는 diffusion 모델을 훈련

- Continous vector를 사용하는 것의 효과
  - Discrete vector보다 bitrate이 높기 때문에 reconstruction quality가 좋음
  - Each audio frame -> one vector로 표현 => hidden sequence의 길이가 짧아짐
  - Continuous vector는 각 quantizer에서 나온 것을 time 축으로 더한 것!

Continuous vector를 사용하는 Neural Audio Codec

- 각 quantizer에서 나온 vector의 Sum
  - [Batch, Time, Q_dim] -> [Batch, Time]
  - 이게 diffusion 모델의 target이 됨

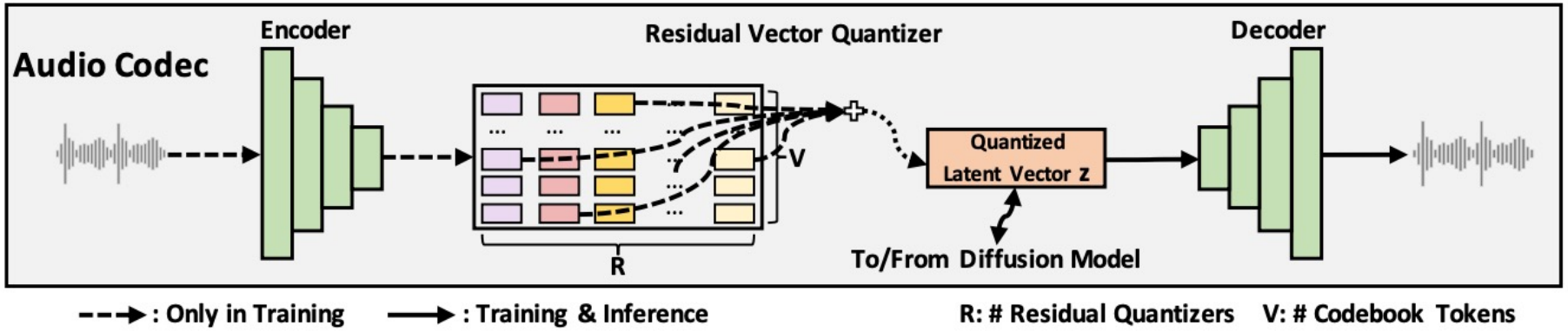


Figure 2: The neural audio codec consists of an encoder, a residual vector-quantizer (RVQ), and a decoder. The encoder extracts the frame-level speech representations from the audio waveform, the RVQ leverages multiple codebooks to quantize the frame-level representations, and the decoder takes the quantized vectors as input and reconstructs the audio waveform. The quantized vectors also serve as the training target of the latent diffusion model.

Diffusion model 구성

- Audio Codec model에서 나온 latent vector z를 생성하도록 학습
- 이때, prior model이 phoneme encoder, duration predictor, pitch predict로 구성

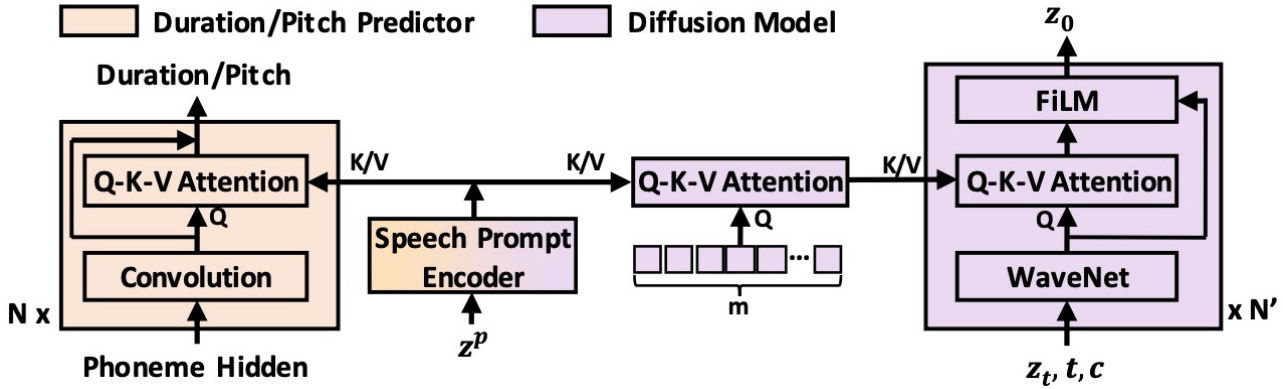
- Input: noise + condition c
  - 이 c를 prior model이 생성
- Output: latent vector z (sum of discrete codes)

- 이때 neural net은 WaveNet의 구조를 사용
  - 40 WaveNet layers
- Forward process, reverse process는 각각 stochastic differential equation(SDE)

Speech Prompt for In-Context learning

- 기존 in-context learning in TTS
  - VALL-E 의 prompt 형식 : text + speech
  - SPEAR-TTS의 prompt 형식 : semantic tokens prompt/target + acoustic tokens
  - 그리고 다음 sequence 를 LM 방식으로 생성하도록 학습

- Pitch, duration prediction와 diffusion 모델에 speech prompt를 활용
  - 이때 attention 을 활용
- Diffusion model 의 경우, 2-stage attention을 사용
  - 1: Random embedding 이 Q, prompt 의 hidden sequence 가 K,V
  - 2: WaveNet layer의 hidden sequence 가 Q, 1st stage의 결과가 K,V



Zero-shot Singing TTS

- 훈련 시 speech + singing data
  - Audio generation에서는 speech와 music, singing data를 모두 섞어서 사용

- Diffusion step 1000으로 설정하고 inference
  - Ground truth pitch, duration 을 다른 Signing voice에서 뽑아서 사용
  - Singing prompt -> different timbre 를 위해 사용

- Prompt
  - Singing
  - 그냥 speech
  - 어떻게 Speech와 singing 을 구분해서 생성하는 지는 나와있지 않음

Experiment

- Training dataset: LibriSpeech (44k hours) / web crawled singing data
  - VALL-E: 60K / SPEAR TTS: audio only LibriLight 600k / parallel 15 min
- Evaluation Dataset: VCTK, LibriSpeech test-clean

- Objective metrics
  - Prosody similarity
    - With prompt / with GT
  - WER
- Subjective Metrics
  - Intelligibility score
  - CMOS, SMOS

VALL-E 와의 비교

Setting	Pitch		Duration	
	Correlation ↑	RMSE ↓	Correlation ↑	RMSE ↓
VALL-E	0.77	47.41	0.61	2.82
NaturalSpeech 2	**0.82**	**39.51**	**0.70**	**2.37**

Table 9: The comparison of VALL-E and NaturalSpeech 2 in prosody similarity between the synthesized and ground-truth speech in terms of the correlation and RMSE on pitch and duration.

Setting	CMOS (v.s. NaturalSpeech 2)
VALL-E	−0.31
NaturalSpeech 2	**0.00**

Table 10: The CMOS results between NaturalSpeech 2 and VALL-E.

결론

- **Continuous vector**를 사용해서 생성해야하는 **sequence** 의 길이를 줄여보고 하는 시도
- 하지만 **diffusion model**의 **Step** 수 등을 봤을 때 **inference speed**가 느릴 것을 예상
- **Zero-shot singing TTS** 까지 할 수 있게 된 것
- **Pitch, duration** 모듈 등을 이용해서 학습

끝

감사합니다.