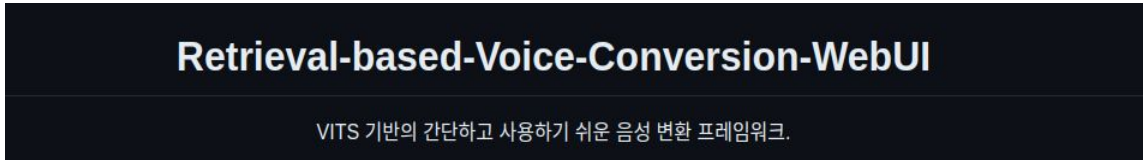
RVC의 기반 모델인
VITS기반 TTS제작과 이해

목차

RVC가 뭐야?

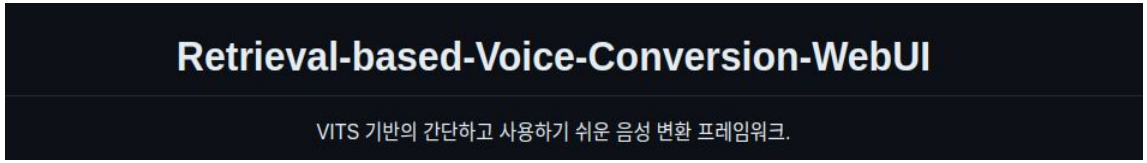
이런 AI 커버 영상에 사용되는 도구!

RVC 너의 정체는.



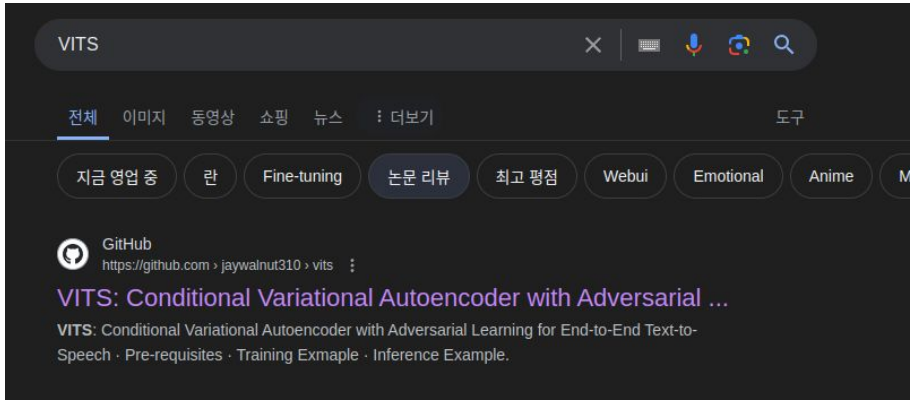
아! RVC는 VITS 기반이다.

RVC 너의 정체는.

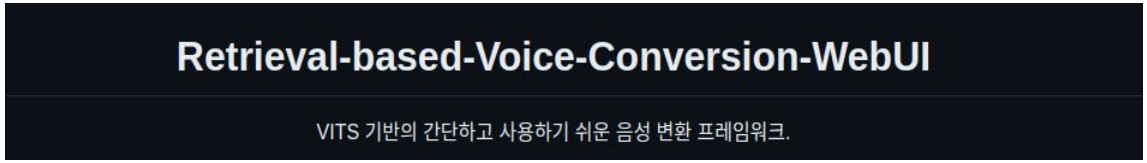


아! RVC는 VITS 기반이다.

VITS를 검색해보자!

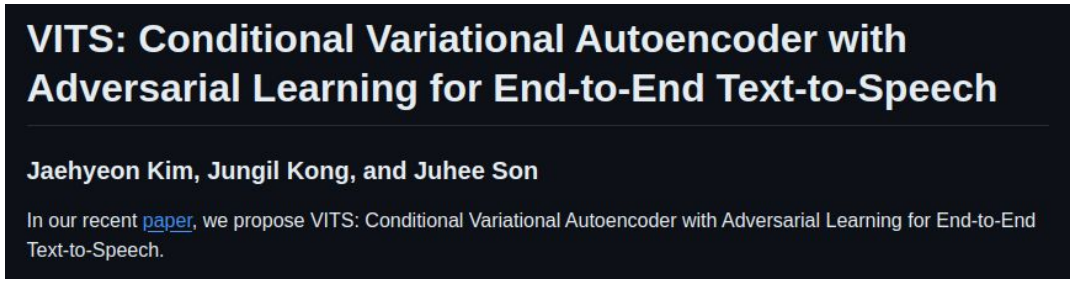


RVC 너의 정체는.



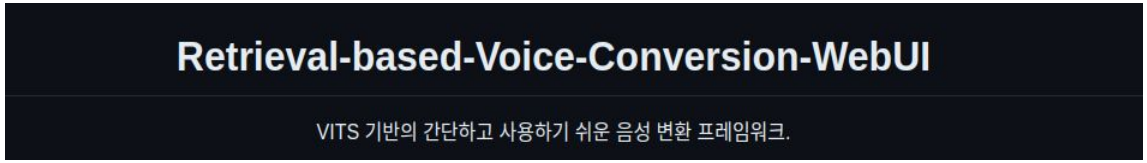
아! RVC는 VITS 기반이다.

VITS란



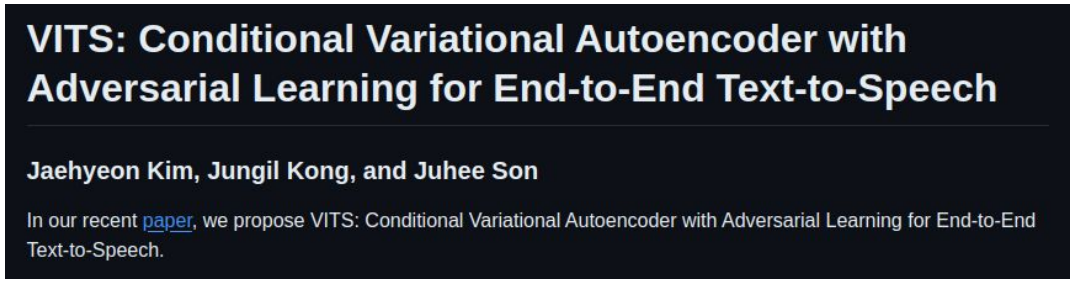
어? 한국인 저자네?

VITS 너의 정체는.



아! RVC는 VITS 기반이다.

VITS란



어? 한국인 저자네? + KAIST, KAKAO 합작 연구네?

논문을 보러 갑시다...?

Conditional Variational Autoencoder with Adversarial Learning for End-to-End Text-to-Speech

Jaehyeon Kim [1] **Jungil Kong** [1] **Juhee Son** [1 2]

Abstract

Several recent end-to-end text-to-speech (TTS) models enabling single-stage training and parallel sampling have been proposed, but their sample quality does not match that of two-stage TTS systems. In this work, we present a parallel end-to-end TTS method that generates more natural sounding audio than current two-stage models. Our method adopts variational inference augmented with normalizing flows and an adversarial training process, which improves the expressive power of generative modeling. We also propose a stochastic duration predictor to synthesize speech with diverse rhythms from input text. With the uncertainty modeling over latent variables and the stochastic duration predictor, our method expresses the natural one-to-many relationship in which a text input can be spoken in multiple ways with different pitches and rhythms. A subjective human evaluation (mean opinion score, or MOS) on the LJ Speech, a single speaker dataset, shows that our method outperforms the best publicly available TTS systems and achieves a MOS comparable to ground truth.

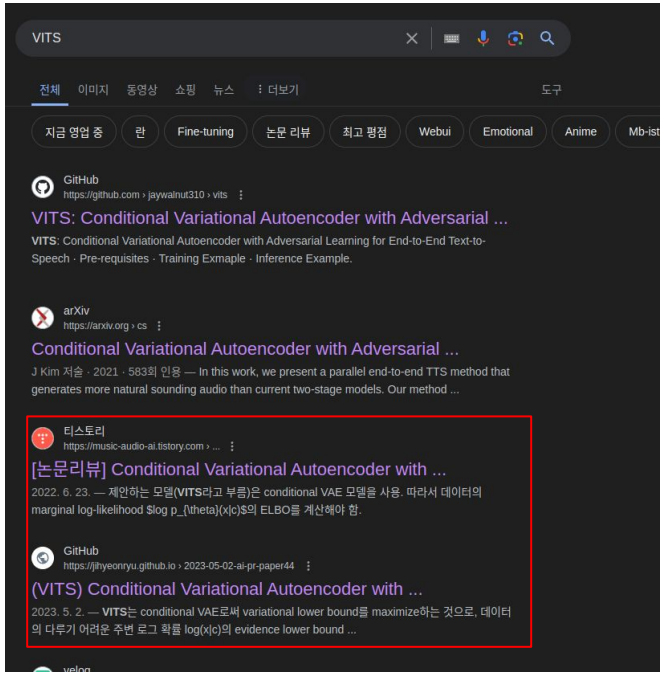
1. Introduction

Text-to-speech (TTS) systems synthesize raw speech waveforms from given text through several components. With the rapid development of deep neural networks, TTS system pipelines have been simplified to two-stage generative modeling apart from text preprocessing such as text normalization and phonemization. The first stage is to produce intermediate speech representations such as mel-et al., 2016) from the preprocessed text,[1] and the second stage is to generate raw waveforms conditioned on the intermediate representations (Oord et al., 2016; Kalchbrenner et al., 2018). Models at each of the two-stage pipelines have been developed independently.

Neural network-based autoregressive TTS systems have shown the capability of synthesizing realistic speech (Shen et al., 2018; Li et al., 2019), but their sequential generative process makes it difficult to fully utilize modern parallel processors. To overcome this limitation and improve synthesis speed, several non-autoregressive methods have been proposed. In the text-to-spectrogram generation step, extracting attention maps from pre-trained autoregressive teacher networks (Ren et al., 2019; Peng et al., 2020) is attempted to decrease the difficulty of learning alignments between text and spectrograms. More recently, likelihood-based methods further eliminate the dependency on external aligners by estimating or learning alignments that maximize the likelihood of target mel-spectrograms (Zeng et al., 2020; Miao et al., 2020; Kim et al., 2020). Meanwhile, generative adversarial networks (GANs) (Goodfellow et al., 2014) have been explored in second stage models. GAN-based feed-forward networks with multiple discriminators, each distinguishing samples at different scales or periods, achieve high-quality raw waveform synthesis (Kumar et al., 2019; Bińkowski et al., 2019; Kong et al., 2020).

Despite the progress of parallel TTS systems, two-stage pipelines remain problematic because they require sequential training or fine-tuning (Shen et al., 2018; Weiss et al., 2020) for high-quality production wherein latter stage models are trained with the generated samples of earlier stage models. In addition, their dependency on predefined intermediate features precludes applying learned hidden representations to obtain further improvements in performance. Recently, several works, i.e., FastSpeech 2s (Ren et al.,

대안


VITS
전체
이미지
동영상
쇼핑
뉴스
더보기
도구
지금 영업 중
란
Fine-tuning
논문 리뷰
최고 평점
Webui
Emotional
Anime
GitHub
https://github.com › jaywalnut310 › vits
VITS: Conditional Variational Autoencoder with Adversarial ...
VITS: Conditional Variational Autoencoder with Adversarial Learning for End-to-End Text-to-Speech · Pre-requisites · Training Exmaple · Inference Example.
arXiv
https://arxiv.org › cs
Conditional Variational Autoencoder with Adversarial ...
J Kim 저술 · 2021 · 583회 인용 — In this work, we present a parallel end-to-end TTS method that generates more natural sounding audio than current two-stage models. Our method ...
티스토리
https://music-audio-ai.tistory.com › ...
[논문리뷰] Conditional Variational Autoencoder with ...
2022. 6. 23. — 제안하는 모델(VITS라고 부름)은 conditional VAE 모델을 사용. 따라서 데이터의 marginal log-likelihood $\log p_{\theta}(x|c)$의 ELBO를 계산해야 함.
GitHub
https://jihyeonryu.github.io › 2023-05-02-ai-pr-paper44
(VITS) Conditional Variational Autoencoder with ...
2023. 5. 2. — VITS는 conditional VAE로써 variational lower bound를 maximize하는 것으로, 데이터의 다루기 어려운 주변 로그 확률 log(x|c)의 evidence lower bound ...

리뷰본을 보자!

2. 수식 증명

AI를 만들 때 필요한 것

손실함수 : 예측한 값과 실제 값이 얼마나 차이나는지 구하는 함수

손실 함수를 보자!

$$L_{vae} = L_{recon} + L_{kl} + L_{dur} + L_{adv}(G) + L_{fm}(G)$$

L_recon

$$L_{vae} = L_{recon} + L_{kl} + L_{dur} + L_{adv}(G) + L_{fm}(G)$$

$$L_{recon} = \|x_{mel} - \hat{x}_{mel}\|_1$$

해석 : 실제 값과 예측 값의 차이의 절댓값으로 이해하면 쉬움

L_kl

$$L_{vae} = L_{recon} + L_{kl} + L_{dur} + L_{adv}(G) + L_{fm}(G)$$

$$L_{kl} = \log q_\phi(z|x_{lin}) - \log p_\theta(z|c_{text}, A),$$
$$z \sim q_\phi(z|x_{lin}) = N(z; \mu_\phi(x_{lin}), \sigma_\phi(x_{lin}))$$

가능도 : x_lin라는 관측값이 주어졌을때 z라는 확률 분포에 들어갈 확률

L_kl

$$L_{vae} = L_{recon} + L_{kl} + L_{dur} + L_{adv}(G) + L_{fm}(G)$$

$$L_{kl} = \log q_\phi(z|x_{lin}) - \log p_\theta(z|c_{text}, A),$$
$$z \sim q_\phi(z|x_{lin}) = N(z; \mu_\phi(x_{lin}), \sigma_\phi(x_{lin}))$$

가능도 : x_lin라는 관측값이 주어졌을때 z라는 확률 분포에 들어갈 확률

L_kl

$$L_{vae} = L_{recon} + L_{kl} + L_{dur} + L_{adv}(G) + L_{fm}(G)$$

$$L_{kl} = \log q_\phi(z|x_{lin}) - \log p_\theta(z|c_{text}, A),$$
$$z \sim q_\phi(z|x_{lin}) = N(z; \mu_\phi(x_{lin}), \sigma_\phi(x_{lin}))$$

가능도 : x_lin라는 관측값이 주어졌을때 z라는 확률 분포에 들어갈 확률

A가 뭔지 설명할 순 있지만 이걸 설명하려면 20분이 더 걸리니 생략!

L_kl

$$L_{vae} = L_{recon} + L_{kl} + L_{dur} + L_{adv}(G) + L_{fm}(G)$$

$$L_{kl} = \log q_\phi(z|x_{lin}) - \log p_\theta(z|c_{text}, A),$$

$$z \sim q_\phi(z|x_{lin}) = N(z; \mu_\phi(x_{lin}), \sigma_\phi(x_{lin}))$$

가능도 : x_lin라는 관측값이 주어졌을때 z라는 확률 분포에 들어갈 확률

$$p_\theta(z|c) = N(f_\theta(z); \mu_\theta(c), \sigma_\theta(c)) \left| \det \frac{\partial f_\theta(z)}{\partial z} \right|$$

$$c = [c_{text}, A]$$

L_kl

$$L_{vae} = L_{recon} + L_{kl} + L_{dur} + L_{adv}(G) + L_{fm}(G)$$

$$L_{kl} = \log q_\phi(z|x_{lin}) - \log p_\theta(z|c_{text}, A),$$

$$z \sim q_\phi(z|x_{lin}) = N(z; \mu_\phi(x_{lin}), \sigma_\phi(x_{lin}))$$

가능도 : x_lin라는 관측값이 주어졌을때 z라는 확률 분포에 들어갈 확률

$$\begin{aligned} A &= \arg\max_{\hat{A}} \log p(x|c_{text}, \hat{A}) \\ &= \arg\max_{\hat{A}} \log N(f(x); \mu(c_{text}, \hat{A}), \sigma(c_{text}, \hat{A})) \end{aligned}$$

$$\begin{aligned} &\arg\max_{\hat{A}} \log p_\theta(x_{mel}|z) - \log \frac{q_\phi(z|x_{lin})}{p_\theta(z|c_{text}, \hat{A})} \\ &= \arg\max_{\hat{A}} \log p_\theta(z|c_{text}, \hat{A}) \\ &= \log N(f_\theta(z); \mu_\theta(c_{text}, \hat{A}), \sigma_\theta(c_{text}, \hat{A})) \end{aligned}$$

L_kl

$$L_{vae} = L_{recon} + L_{kl} + L_{dur} + L_{adv}(G) + L_{fm}(G)$$

$$L_{kl} = \log q_\phi(z|x_{lin}) - \log p_\theta(z|c_{text}, A),$$

$$z \sim q_\phi(z|x_{lin}) = N(z; \mu_\phi(x_{lin}), \sigma_\phi(x_{lin}))$$

가능도 : x_lin라는 관측값이 주어졌을때 z라는 확률 분포에 들어갈 확률

L_dur

$$L_{vae} = L_{recon} + L_{kl} + L_{dur} + L_{adv}(G) + L_{fm}(G)$$

$$\log p_\theta(d|c_{text}) \geq$$

$$\mathbb{E}_{q_\phi(u,\nu|d,c_{text})}\left[\log \frac{p_\theta(d-u,\nu|c_{text})}{q_\phi(u,\nu|d,c_{text})}\right]$$

L_dur도 생략! (이거 설명하려면 또 10분 추가 해야함)

L_adv

$$L_{vae} = L_{recon} + L_{kl} + L_{dur} + L_{adv}(G) + L_{fm}(G)$$

$$L_{adv}(D) = \mathbb{E}_{(y,z)}\left[(D(y) - 1)^2 + (D(G(z)))^2\right],$$

$$L_{adv}(G) = \mathbb{E}_{z}\left[(D(G(z)) - 1)^2\right],$$

G : 생성자

D : 판별자

판별자는 생성자의 예측값을 이용해 진짜인지 구분, 생성자는 그 반대!

L_fm

$$L_{vae} = L_{recon} + L_{kl} + L_{dur} + L_{adv}(G) + L_{fm}(G)$$

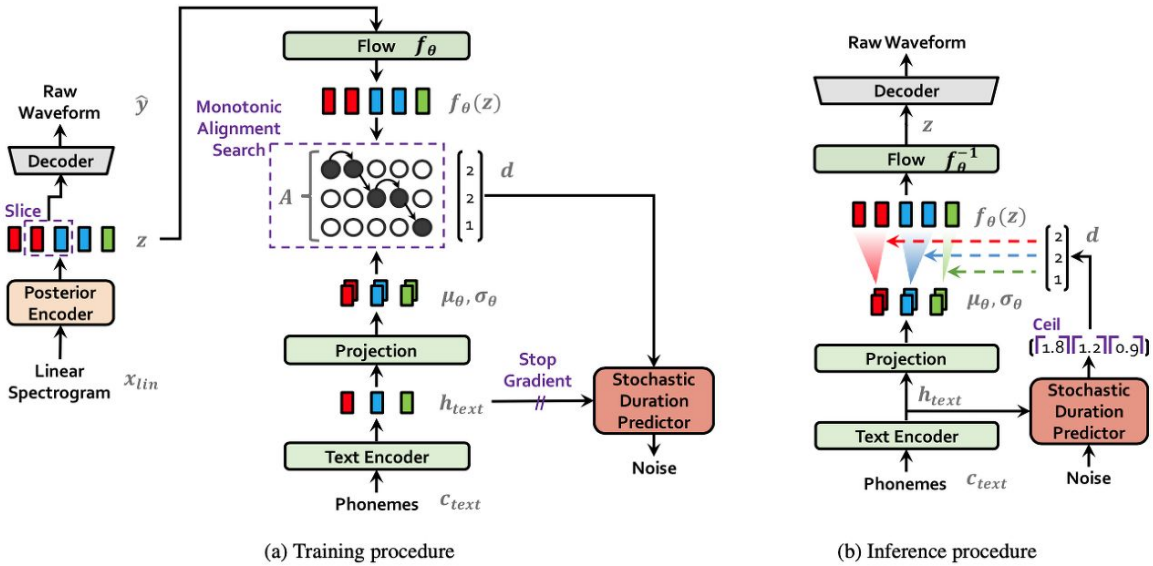
$$L_{fm}(G) = \mathbb{E}_{(y,z)}\Big[\sum_{l=1}^{T} \frac{1}{N_l} \|D^l(y) - D^l(G(z))\|_1\Big]$$

간단히 말해 좀 더 퀄리티를 높이기 위한 수단.

다른 함수와 다르게 연산 중간에 데이터를 꺼내와 구하는 함수

작동 과정 (드디어..)

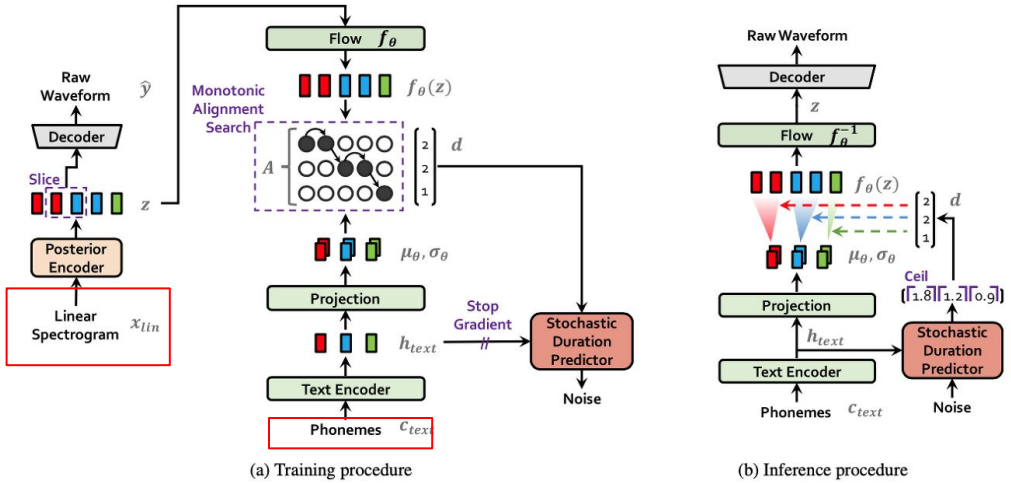
요약


Raw Waveform
$\hat{y}$
Decoder
Slice
z
Posterior Encoder
Linear Spectrogram
x_{lin}
Flow f_θ
Monotonic Alignment Search
$f_\theta(z)$
A
2
2
1
d
$\mu_\theta, \sigma_\theta$
Projection
h_{text}
Stop Gradient
//
Stochastic Duration Predictor
Noise
Text Encoder
Phonemes
c_{text}
(a) Training procedure
Raw Waveform
Decoder
z
Flow f_θ^{-1}
$f_\theta(z)$
2
2
1
d
$\mu_\theta, \sigma_\theta$
Ceil
⌈1.8⌉⌈1.2⌉⌈0.9⌉
Projection
h_{text}
Stochastic Duration Predictor
Text Encoder
Phonemes
c_{text}
Noise
(b) Inference procedure

입력

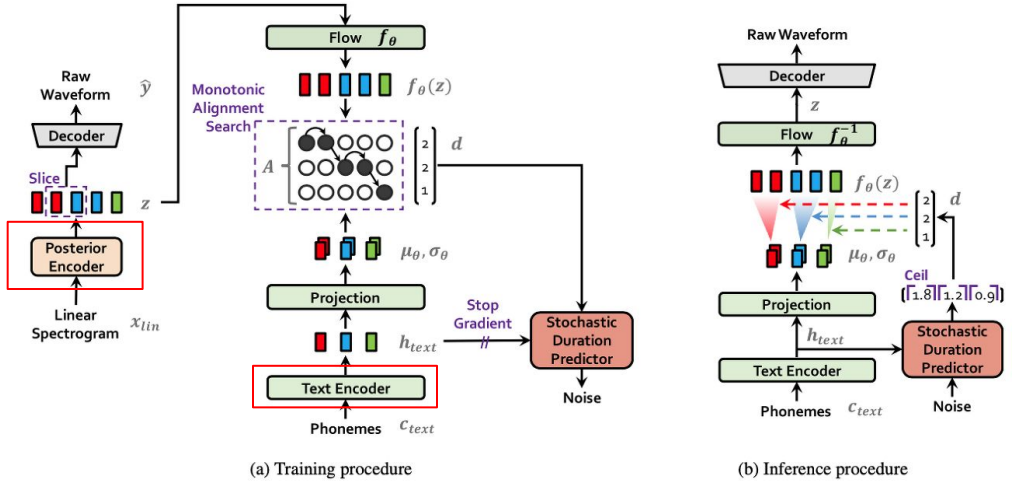
음성을 푸리에 변환한 스펙토그램....과 텍스트

푸리에 변환도 생략! (아마 앞팀이 했거나 뒷팀이 할거임)



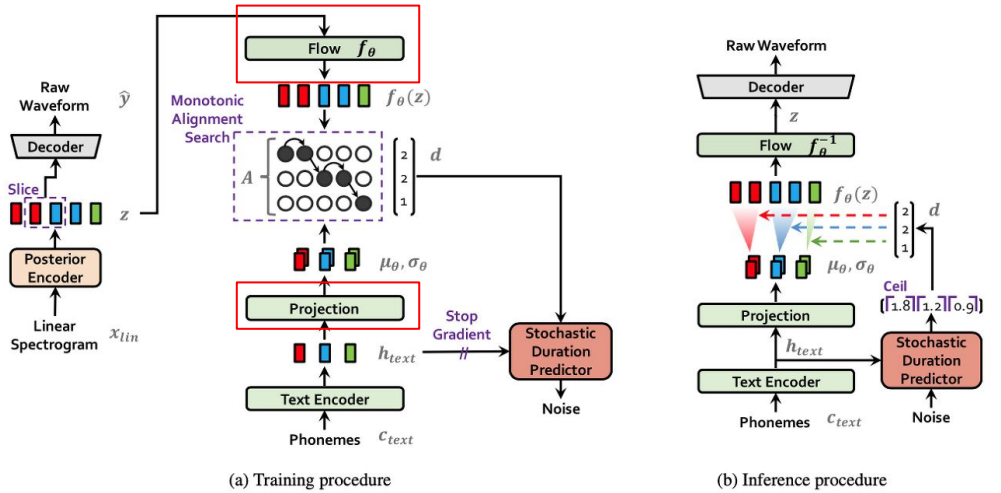
입력을 확률 분포로

텍스트와 스펙토그램 모두 Encoder라는 것을 통해 확률 분포로 변환



확률 분포를 더 복잡하게

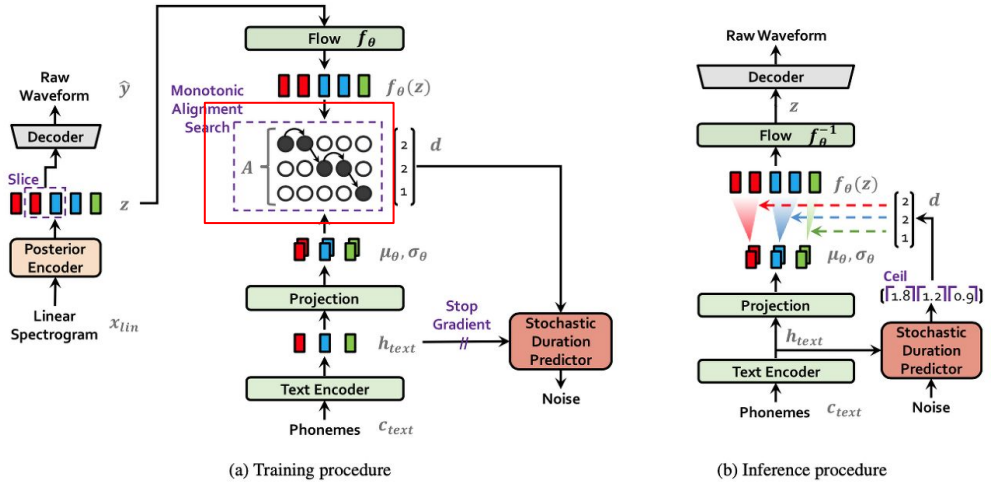
확률 분포를 더 복잡하게 만들고 확률 분포의 분산과 평균을 구함



MAS(음성 정렬)

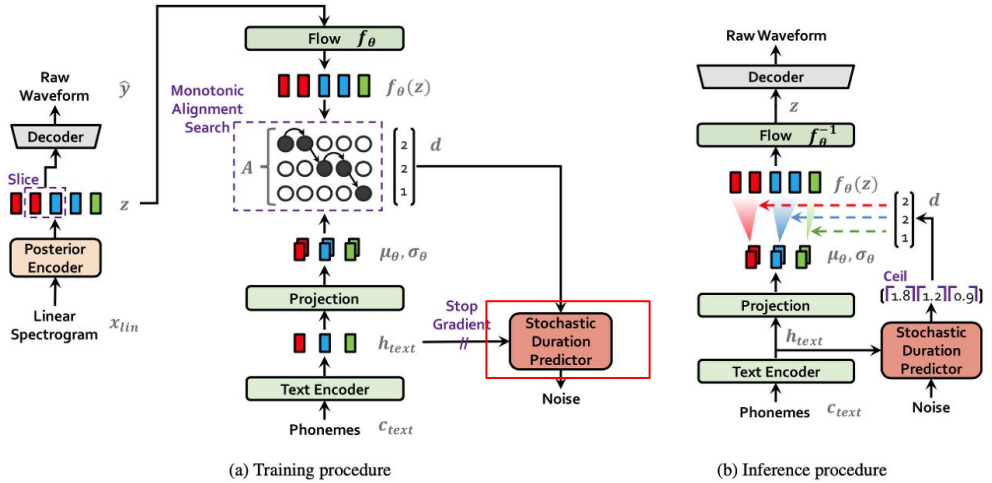
마구잡이로 섞여있는 데이터를 정렬시킴.

알 사람은 알겠지만 dynamic programming 이용



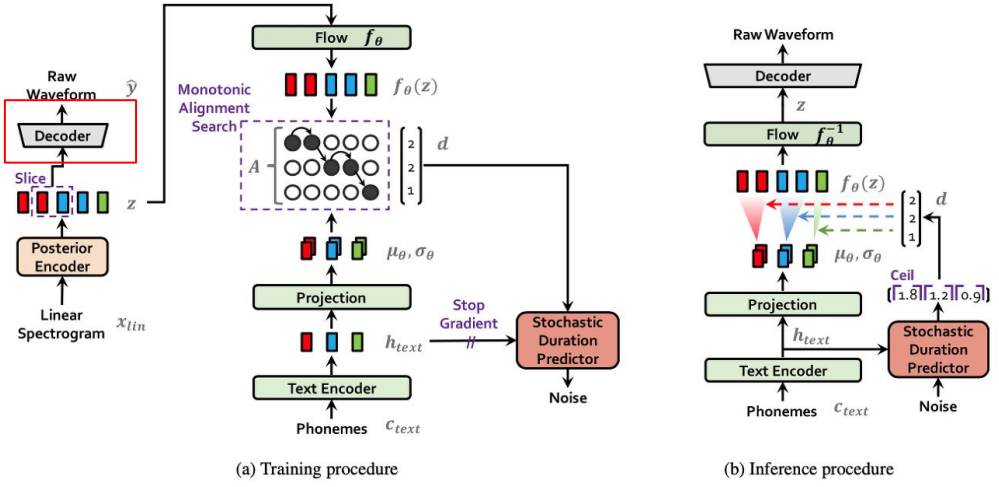
Stochastic duration predictor

음성을 화자의 텍스트를 말하는 길이에 맞추도록 학습



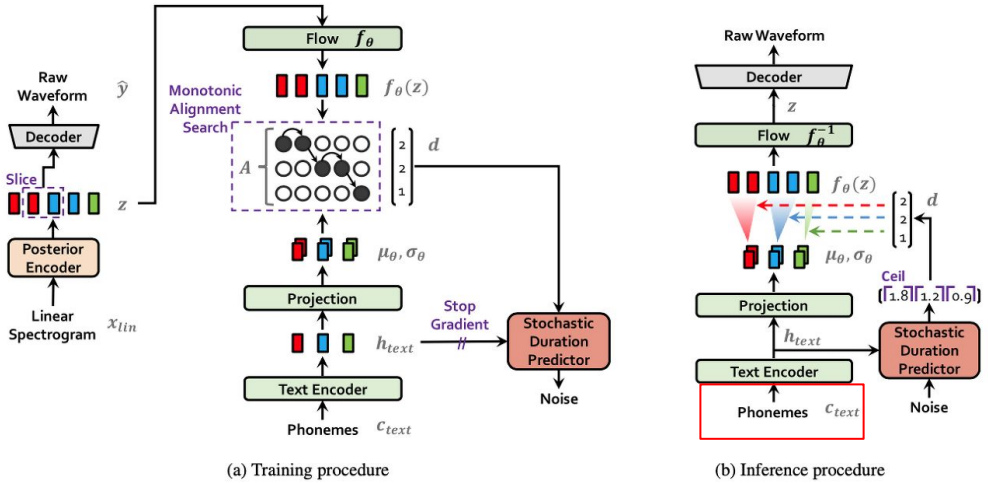
Decoder

원래 음성을 복원하도록 학습



추론!

입력은 텍스트만

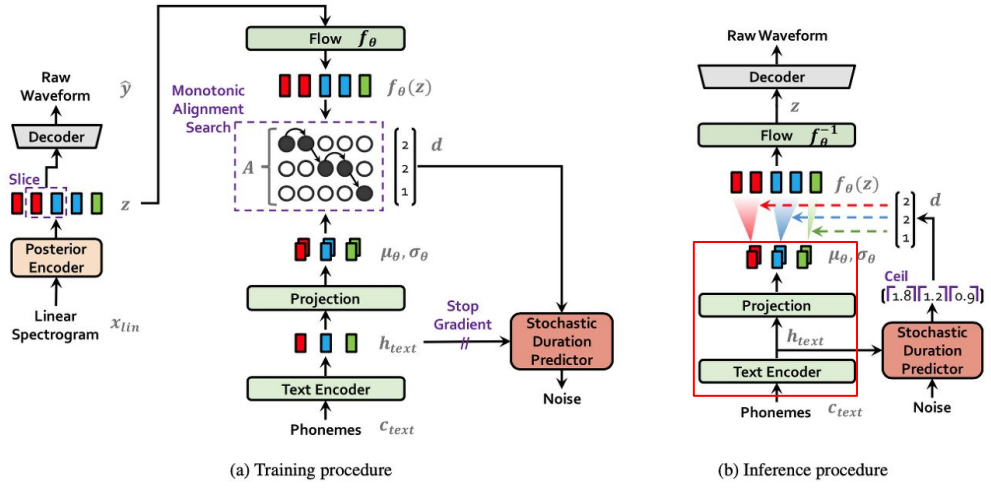


(a) Training procedure

(b) Inference procedure

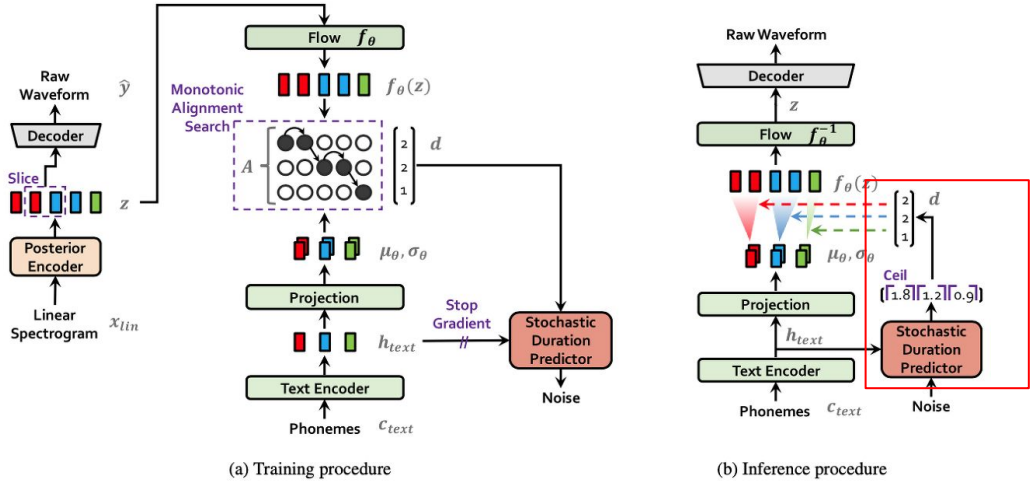
다시 확률 분포로

텍스트를 입력한것을 확률분포로 바꾸고 평균과 분산으로 바꿈



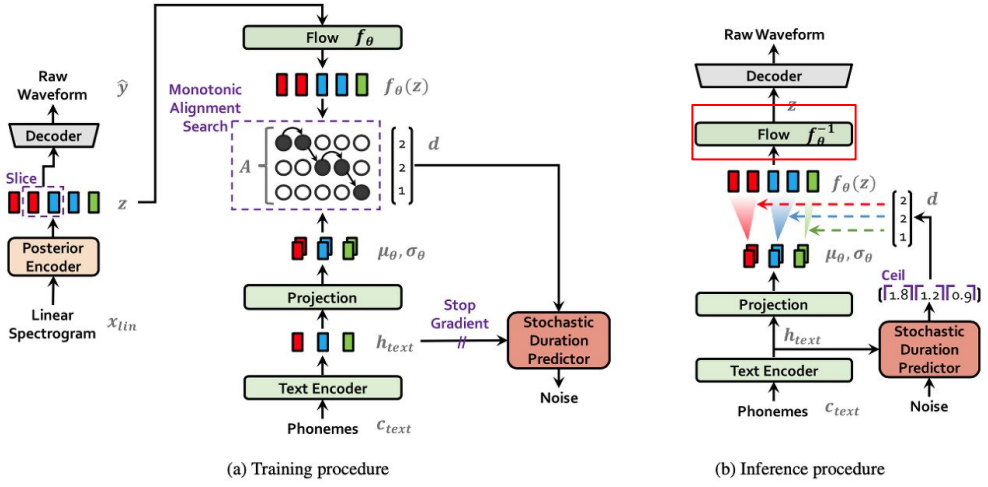
텍스트 발음 길이 예측

아까 학습시켰던 Stochastic duration predictor를 활용해 텍스트의 발음 길이를 예측하고 확률 분포에 대입



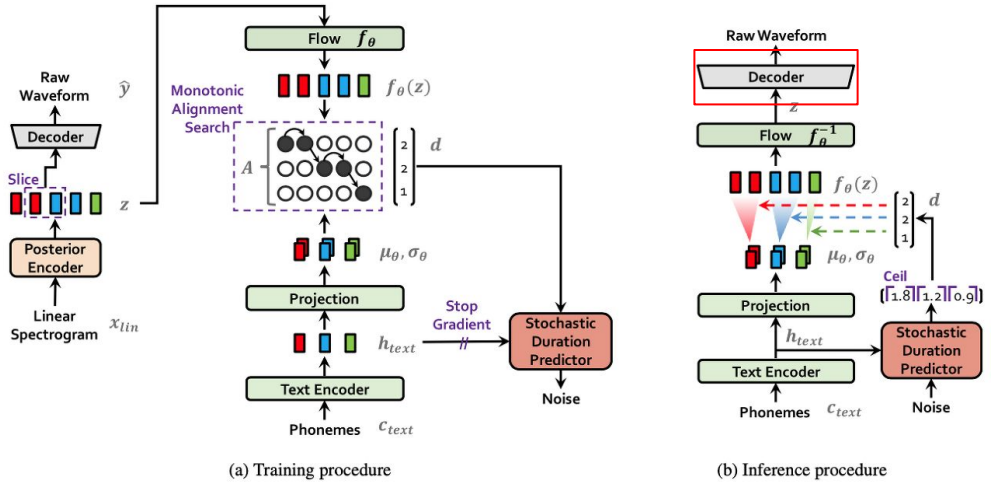
단순한 확률 분포로

확률 분포를 단순화 시킴



드디어 마지막

확률을 음성으로 변환



결과물

http://test.simsim-han.com

힘든점

이 탐구를 하며 어려운 점이 매우 많았음. 대표적으로 작년에 확률과 통계를 완벽하게 하지 않아 가능도를 추정하기 위한 베이즈정리 등을 다시 공부해야 하였음. 또한

가장 큰 문제는 모델이 매우 크고 복잡하여 학습이 오래걸리는 것과 중간중간 메모리 부족으로 끊히는 경우가 많았음. 이 점을 해결하기 위하여 사비를 사용하여 서버를

빌려 학습시킴. (사비 약 10만원 소요)그 외 자잘한 것들은 수식을 고등 수학으로는 이해하지 못하는 것과 코드 상으로 구현할 때 행렬의 차원이 안맞거나 객체의 종류가 안맞는 등의 여러

어려운 점이 있었음. 고등 수학으로 이해하지 못하는 것은 인터넷으로 검색하여 정보를 찾고 선형대수학과 미적분 서적을 구입하여 관련 수학적 내용들을 공부하는

방식으로 해결함. 코드 상 문제는 인터넷 검색하여 해결하였음. 하지만 인터넷으로 검색하였는데 데이터 셋 가공 관련 문제와 코드에 중대한 결함이 있어서 다시 새로

처음부터 작성하는 등으로 해결을 했음.

느낀 점

논문의 수식과 거기에 적힌 내용으로만 코드를 짜려니 마치 눈 가리고 100M 곡선주로를 뛰려고 하는 느낌이였음. 하지만 이 활동을 통해 이 분야에서 주로 쓰이는

논문의 수식을 이해하고 거기에 맞는 코드를 작성하는 귀중한 능력을 얻을 수 있었음. 앞으로는 검색과 논문 리뷰 내용없이 논문의 내용만으로 모델을 이해하고 코드를

짜는 능력을 가지고 싶음. 그러기 위해 앞으로도 노력할 것임.

참조 :

https://github.com/RVC-Project/Retrieval-based-Voice-Conversion-WebUI/blob/main/docs/kr/README.ko.md

https://github.com/jaywalnut310/vits

https://arxiv.org/abs/2106.06103

https://music-audio-ai.tistory.com/22