arXiv:2106.06103v1 [cs.SD] 11 Jun 2021

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-spectrograms (Shen et al., 2018) or linguistic features (Oord 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., 2021) and EATS (Donahue et al., 2021), have proposed efficient end-to-end training methods such as training over short audio clips rather than entire waveforms, leveraging a mel-spectrogram decoder to aid text representation learning,

[1]Kakao Enterprise, Seongnam-si, Gyeonggi-do, Republic of Korea [2]School of Computing, KAIST, Daejeon, Republic of Korea. Correspondence to: Jaehyeon Kim <jay.xyz@kakaoenterprise.com>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

[1]Although there is a text preprocessing step in TTS systems, We herein use preprocessed text interchangeably with the word "text".

and designing a specialized spectrogram loss to relax length-mismatch between target and generated speech. However, despite potentially improving performance by utilizing the learned representations, their synthesis quality lags behind two-stage systems.

In this work, we present a parallel end-to-end TTS method that generates more natural sounding audio than current two-stage models. Using a variational autoencoder (VAE) (Kingma & Welling, 2014), we connect two modules of TTS systems through latent variables to enable efficient end-to-end learning. To improve the expressive power of our method so that high-quality speech waveforms can be synthesized, we apply normalizing flows to our conditional prior distribution and adversarial training on the waveform domain. In addition to generating fine-grained audio, it is important for TTS systems to express the one-to-many relationship in which text input can be spoken in multiple ways with different variations (e.g., pitch and duration). To tackle the one-to-many problem, 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 captures speech variations that cannot be represented by text.

Our method obtains more natural sounding speech and higher sampling efficiency than the best publicly available TTS system, Glow-TTS (Kim et al., 2020) with HiFi-GAN (Kong et al., 2020). We make both our demo page and source-code publicly available.[2]

2. Method

In this section, we explain our proposed method and the architecture of it. The proposed method is mostly described in the first three subsections: a conditional VAE formulation; alignment estimation derived from variational inference; adversarial training for improving synthesis quality. The overall architecture is described at the end of this section. Figures 1a and 1b show the training and inference procedures of our method, respectively. From now on, we will refer to our method as Variational Inference with adversarial learning for end-to-end Text-to-Speech (VITS).

2.1. Variational Inference

2.1.1. Overview

VITS can be expressed as a conditional VAE with the objective of maximizing the variational lower bound, also called the evidence lower bound (ELBO), of the intractable marginal log-likelihood of data $\log p_\theta(x|c)$:

$$\log p_\theta(x|c) \geq \mathbb{E}_{q_\phi(z|x)}\Big[\log p_\theta(x|z) - \log \frac{q_\phi(z|x)}{p_\theta(z|c)}\Big] \quad (1)$$

where $p_\theta(z|c)$ denotes a prior distribution of the latent variables z given condition c, $p_\theta(x|z)$ is the likelihood function of a data point x, and $q_\phi(z|x)$ is an approximate posterior distribution. The training loss is then the negative ELBO, which can be viewed as the sum of reconstruction loss $-\log p_\theta(x|z)$ and KL divergence $\log q_\phi(z|x) - \log p_\theta(z|c)$, where $z \sim q_\phi(z|x)$.

2.1.2. Reconstruction loss

As a target data point in the reconstruction loss, we use a mel-spectrogram instead of a raw waveform, denoted by x_{mel}. We upsample the latent variables z to the waveform domain $\hat{y}$ through a decoder and transform $\hat{y}$ to the mel-spectrogram domain $\hat{x}_{mel}$. Then the L_1 loss between the predicted and target mel-spectrogram is used as the reconstruction loss:

$$L_{recon} = \|x_{mel} - \hat{x}_{mel}\|_1 \quad (2)$$

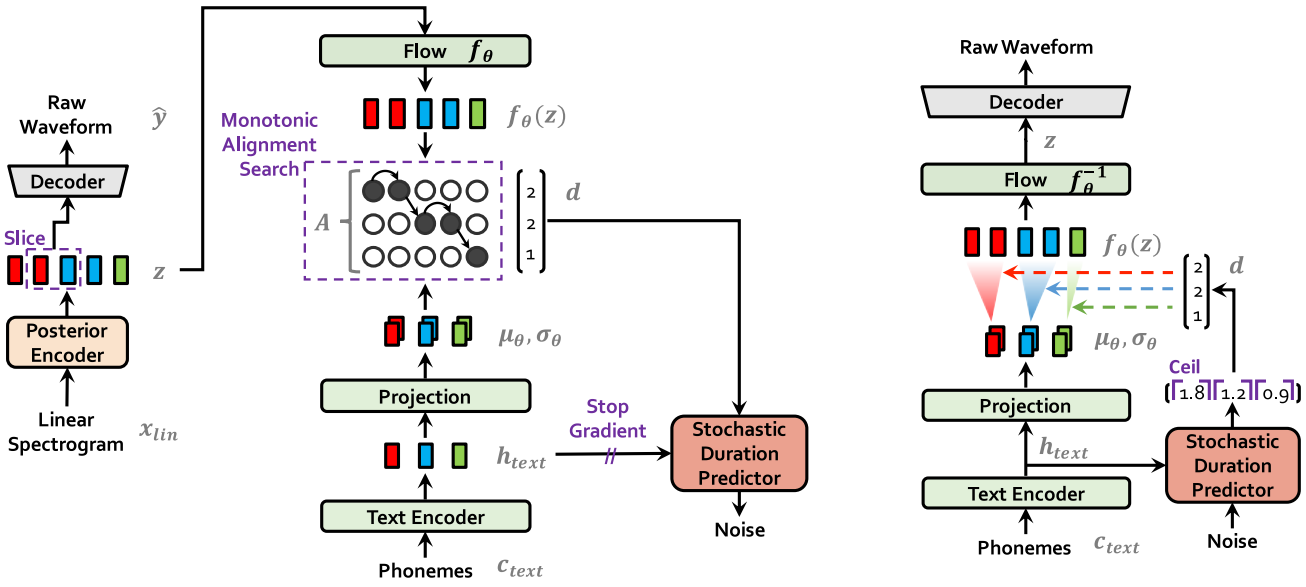
This can be viewed as maximum likelihood estimation assuming a Laplace distribution for the data distribution and ignoring constant terms. We define the reconstruction loss in the mel-spectrogram domain to improve the perceptual quality by using a mel-scale that approximates the response of the human auditory system. Note that the mel-spectrogram estimation from a raw waveform does not require trainable parameters as it only uses STFT and linear projection onto the mel-scale. Furthermore, the estimation is only employed during training, not inference. In practice, we do not upsample the whole latent variables z but use partial sequences as an input for the decoder, which is the windowed generator training used for efficient end-to-end training (Ren et al., 2021; Donahue et al., 2021).

2.1.3. KL-divergence

The input condition of the prior encoder c is composed of phonemes c_{text} extracted from text and an alignment A between phonemes and latent variables. The alignment is a hard monotonic attention matrix with $|c_{text}| \times |z|$ dimensions representing how long each input phoneme expands to be time-aligned with the target speech. Because there are no ground truth labels for the alignment, we must estimate the alignment at each training iteration, which we will discuss in Section 2.2.1. In our problem setting, we aim to provide more high-resolution information for the posterior encoder. We, therefore, use the linear-scale spectrogram of target speech x_{lin} as input rather than the mel-spectrogram. Note that the modified input does not violate the properties of variational inference. The KL divergence is then:

$$L_{kl} = \log q_\phi(z|x_{lin}) - \log p_\theta(z|c_{text}, A), \quad (3)$$
$$z \sim q_\phi(z|x_{lin}) = N(z; \mu_\phi(x_{lin}), \sigma_\phi(x_{lin}))$$

[2]Source-code: https://github.com/jaywalnut310/vits
Demo: https://jaywalnut310.github.io/vits-demo/index.html



(a) Training procedure

(b) Inference procedure

Figure 1. System diagram depicting (a) training procedure and (b) inference procedure. The proposed model can be viewed as a conditional VAE; a posterior encoder, decoder, and conditional prior (green blocks: a normalizing flow, linear projection layer, and text encoder) with a flow-based stochastic duration predictor.

The factorized normal distribution is used to parameterize our prior and posterior encoders. We found that increasing the expressiveness of the prior distribution is important for generating realistic samples. We, therefore, apply a normalizing flow f_θ (Rezende & Mohamed, 2015), which allows an invertible transformation of a simple distribution into a more complex distribution following the rule of change-of-variables, on top of the factorized normal prior distribution:

$$p_\theta(z|c) = N(f_\theta(z); \mu_\theta(c), \sigma_\theta(c)) \left| \det \frac{\partial f_\theta(z)}{\partial z} \right|, \quad (4)$$
$$c = [c_{text}, A]$$

2.2. Alignment Estimation

2.2.1. Monotonic Alignment Search

To estimate an alignment A between input text and target speech, we adopt Monotonic Alignment Search (MAS) (Kim et al., 2020), a method to search an alignment that maximizes the likelihood of data parameterized by a normalizing flow f:

$$A = \arg\max_{\hat{A}} \log p(x|c_{text}, \hat{A})$$
$$= \arg\max_{\hat{A}} \log N(f(x); \mu(c_{text}, \hat{A}), \sigma(c_{text}, \hat{A})) \quad (5)$$

where the candidate alignments are restricted to be monotonic and non-skipping following the fact that humans read text in order without skipping any words. To find the optimum alignment, Kim et al. (2020) use dynamic programming. Applying MAS directly in our setting is difficult because our objective is the ELBO, not the exact log-likelihood. We, therefore, redefine MAS to find an alignment that maximizes the ELBO, which reduces to finding an alignment that maximizes the log-likelihood of the latent variables z:

$$\arg\max_{\hat{A}} \log p_\theta(x_{mel}|z) - \log \frac{q_\phi(z|x_{lin})}{p_\theta(z|c_{text}, \hat{A})}$$
$$= \arg\max_{\hat{A}} \log p_\theta(z|c_{text}, \hat{A})$$
$$= \log N(f_\theta(z); \mu_\theta(c_{text}, \hat{A}), \sigma_\theta(c_{text}, \hat{A})) \quad (6)$$

Due to the resemblance of Equation 5 to Equation 6, we can use the original MAS implementation without modification. Appendix A includes pseudocode for MAS.

2.2.2. Duration Prediction from Text

We can calculate the duration of each input token d_i by summing all the columns in each row of the estimated alignment $\sum_j A_{i,j}$. The duration could be used to train a deterministic duration predictor, as proposed in previous work (Kim et al., 2020), but it cannot express the way a person utters at different speaking rates each time. To generate human-like rhythms of speech, we design a stochastic duration predictor so that its samples follow the duration distribution of given

phonemes. The stochastic duration predictor is a flow-based generative model that is typically trained via maximum likelihood estimation. The direct application of maximum likelihood estimation, however, is difficult because the duration of each input phoneme is 1) a discrete integer, which needs to be dequantized for using continuous normalizing flows, and 2) a scalar, which prevents high-dimensional transformation due to invertibility. We apply variational dequantization (Ho et al., 2019) and variational data augmentation (Chen et al., 2020) to solve these problems. To be specific, we introduce two random variables u and ν, which have the same time resolution and dimension as that of the duration sequence d, for variational dequatization and variational data augmentation, respectively. We restrict the support of u to be $[0, 1)$ so that the difference $d-u$ becomes a sequence of positive real numbers, and we concatenate ν and d channel-wise to make a higher dimensional latent representation. We sample the two variables through an approximate posterior distribution $q_\phi(u, \nu|d, c_{text})$. The resulting objective is a variational lower bound of the log-likelihood of the phoneme duration:

$$\log p_\theta(d|c_{text}) \geq \mathbb{E}_{q_\phi(u,\nu|d,c_{text})}\Big[\log \frac{p_\theta(d-u, \nu|c_{text})}{q_\phi(u, \nu|d, c_{text})}\Big] \quad (7)$$

The training loss L_{dur} is then the negative variational lower bound. We apply the stop gradient operator (van den Oord et al., 2017), which prevents back-propagating the gradient of inputs, to the input conditions so that the training of the duration predictor does not affect that of other modules.

The sampling procedure is relatively simple; the phoneme duration is sampled from random noise through the inverse transformation of the stochastic duration predictor, and then it is converted to integers.

2.3. Adversarial Training

To adopt adversarial training in our learning system, we add a discriminator D that distinguishes between the output generated by the decoder G and the ground truth waveform y. In this work, we use two types of loss successfully applied in speech synthesis; the least-squares loss function (Mao et al., 2017) for adversarial training, and the additional feature-matching loss (Larsen et al., 2016) for training the generator:

$$L_{adv}(D) = \mathbb{E}_{(y,z)}\Big[(D(y)-1)^2 + (D(G(z)))^2\Big], \quad (8)$$

$$L_{adv}(G) = \mathbb{E}_z\Big[(D(G(z))-1)^2\Big], \quad (9)$$

$$L_{fm}(G) = \mathbb{E}_{(y,z)}\Big[\sum_{l=1}^{T}\frac{1}{N_l}\|D^l(y) - D^l(G(z))\|_1\Big] \quad (10)$$

where T denotes the total number of layers in the discriminator and D^l outputs the feature map of the l-th layer of the discriminator with N_l number of features. Notably, the feature matching loss can be seen as reconstruction loss that is measured in the hidden layers of the discriminator suggested as an alternative to the element-wise reconstruction loss of VAEs (Larsen et al., 2016).

2.4. Final Loss

With the combination of VAE and GAN training, the total loss for training our conditional VAE can be expressed as follows:

$$L_{vae} = L_{recon} + L_{kl} + L_{dur} + L_{adv}(G) + L_{fm}(G) \quad (11)$$

2.5. Model Architecture

The overall architecture of the proposed model consists of a posterior encoder, prior encoder, decoder, discriminator, and stochastic duration predictor. The posterior encoder and discriminator are only used for training, not for inference. Architectural details are available in Appendix B.

2.5.1. Posterior Encoder

For the posterior encoder, we use the non-causal WaveNet residual blocks used in WaveGlow (Prenger et al., 2019) and Glow-TTS (Kim et al., 2020). A WaveNet residual block consists of layers of dilated convolutions with a gated activation unit and skip connection. The linear projection layer above the blocks produces the mean and variance of the normal posterior distribution. For the multi-speaker case, we use global conditioning (Oord et al., 2016) in residual blocks to add speaker embedding.

2.5.2. Prior Encoder

The prior encoder consists of a text encoder that processes the input phonemes c_{text} and a normalizing flow f_θ that improves the flexibility of the prior distribution. The text encoder is a transformer encoder (Vaswani et al., 2017) that uses relative positional representation (Shaw et al., 2018) instead of absolute positional encoding. We can obtain the hidden representation h_{text} from c_{text} through the text encoder and a linear projection layer above the text encoder that produces the mean and variance used for constructing the prior distribution. The normalizing flow is a stack of affine coupling layers (Dinh et al., 2017) consisting of a stack of WaveNet residual blocks. For simplicity, we design the normalizing flow to be a volume-preserving transformation with the Jacobian determinant of one. For the multi-speaker setting, we add speaker embedding to the residual blocks in the normalizing flow through global conditioning.

2.5.3. Decoder

The decoder is essentially the HiFi-GAN V1 generator (Kong et al., 2020). It is composed of a stack of trans-

posed convolutions, each of which is followed by a multi-receptive field fusion module (MRF). The output of the MRF is the sum of the output of residual blocks that have different receptive field sizes. For the multi-speaker setting, we add a linear layer that transforms speaker embedding and add it to the input latent variables z.

2.5.4. Discriminator

We follow the discriminator architecture of the multi-period discriminator proposed in HiFi-GAN (Kong et al., 2020). The multi-period discriminator is a mixture of Markovian window-based sub-discriminators (Kumar et al., 2019), each of which operates on different periodic patterns of input waveforms.

2.5.5. Stochastic Duration Predictor

The stochastic duration predictor estimates the distribution of phoneme duration from a conditional input h_{text}. For the efficient parameterization of the stochastic duration predictor, we stack residual blocks with dilated and depth-separable convolutional layers. We also apply neural spline flows (Durkan et al., 2019), which take the form of invertible nonlinear transformations by using monotonic rational-quadratic splines, to coupling layers. Neural spline flows improve transformation expressiveness with a similar number of parameters compared to commonly used affine coupling layers. For the multi-speaker setting, we add a linear layer that transforms speaker embedding and add it to the input h_{text}.

3. Experiments

3.1. Datasets

We conducted experiments on two different datasets. We used the LJ Speech dataset (Ito, 2017) for comparison with other publicly available models and the VCTK dataset (Veaux et al., 2017) to verify whether our model can learn and express diverse speech characteristics. The LJ Speech dataset consists of 13,100 short audio clips of a single speaker with a total length of approximately 24 hours. The audio format is 16-bit PCM with a sample rate of 22 kHz, and we used it without any manipulation. We randomly split the dataset into a training set (12,500 samples), validation set (100 samples), and test set (500 samples). The VCTK dataset consists of approximately 44,000 short audio clips uttered by 109 native English speakers with various accents. The total length of the audio clips is approximately 44 hours. The audio format is 16-bit PCM with a sample rate of 44 kHz. We reduced the sample rate to 22 kHz. We randomly split the dataset into a training set (43,470 samples), validation set (100 samples), and test set (500 samples).

3.2. Preprocessing

We use linear spectrograms which can be obtained from raw waveforms through the Short-time Fourier transform (STFT), as input of the posterior encoder. The FFT size, window size and hop size of the transform are set to 1024, 1024 and 256, respectively. We use 80 bands mel-scale spectrograms for reconstruction loss, which is obtained by applying a mel-filterbank to linear spectrograms.

We use International Phonetic Alphabet (IPA) sequences as input to the prior encoder. We convert text sequences to IPA phoneme sequences using open-source software (Bernard, 2021), and the converted sequences are interspersed with a blank token following the implementation of Glow-TTS.

3.3. Training

The networks are trained using the AdamW optimizer (Loshchilov & Hutter, 2019) with $\beta_1 = 0.8$, $\beta_2 = 0.99$ and weight decay $\lambda = 0.01$. The learning rate decay is scheduled by a $0.999^{1/8}$ factor in every epoch with an initial learning rate of 2×10^{-4}. Following previous work (Ren et al., 2021; Donahue et al., 2021), we adopt the windowed generator training, a method of generating only a part of raw waveforms to reduce the training time and memory usage during training. We randomly extract segments of latent representations with a window size of 32 to feed to the decoder instead of feeding entire latent representations and also extract the corresponding audio segments from the ground truth raw waveforms as training targets. We use mixed precision training on 4 NVIDIA V100 GPUs. The batch size is set to 64 per GPU and the model is trained up to 800k steps.

3.4. Experimental Setup for Comparison

We compared our model with the best publicly available models. We used Tacotron 2, an autoregressive model, and Glow-TTS, a flow-based non-autoregressive model, as first stage models and HiFi-GAN as a second stage model. We used their public implementations and pre-trained weights.[3] Since a two-stage TTS system can theoretically achieve higher synthesis quality through sequential training, we included the fine-tuned HiFi-GAN up to 100k steps with the predicted outputs from the first stage models. We empirically found that fine-tuning HiFi-GAN with the generated mel-spectrograms from Tacotron 2 under teacher-forcing mode, led to better quality for both Tacotron 2 and Glow-TTS than fine-tuning with the generated mel-spectrograms from Glow-TTS, so we appended the better fine-tuned HiFi-

[3]The implementations are as follows:
Tacotron 2 : https://github.com/NVIDIA/tacotron2
Glow-TTS : https://github.com/jaywalnut310/glow-tts
HiFi-GAN : https://github.com/jik876/hifi-gan

GAN to both Tacotron 2 and Glow-TTS.

As each model has a degree of randomness during sampling, we fixed hyper-parameters that controls the randomness of each model throughout our experiments. The probability of dropout in the pre-net of Tactron 2 was set to 0.5. For Glow-TTS, the standard deviation of the prior distribution was set to 0.333. For VITS, the standard deviation of input noise of the stochastic duration predictor was set to 0.8 and we multiplied a scale factor of 0.667 to the standard deviation of the prior distribution.

4. Results

4.1. Speech Synthesis Quality

We conducted crowd-sourced MOS tests to evaluate the quality. Raters listened to randomly selected audio samples, and rated their naturalness on a 5 point scale from 1 to 5. Raters were allowed to evaluate each audio sample once, and we normalized all the audio clips to avoid the effect of amplitude differences on the score. All of the quality assessments in this work were conducted in this manner.

The evaluation results are shown in Table 1. VITS outperforms other TTS systems and achieves a similar MOS to that of ground truth. The VITS (**DDP**), which employs the same deterministic duration predictor architecture used in Glow-TTS rather than the stochastic duration predictor, scores the second-highest among TTS systems in the MOS evaluation. These results imply that 1) the stochastic duration predictor generates more realistic phoneme duration than the deterministic duration predictor and 2) our end-to-end training method is an effective way to make better samples than other TTS models even if maintaining the similar duration predictor architecture.

Table 1. Comparison of evaluated MOS with 95% confidence intervals on the LJ Speech dataset.

Model	MOS (CI)
Ground Truth	4.46 (±0.06)
Tacotron 2 + HiFi-GAN	3.77 (±0.08)
Tacotron 2 + HiFi-GAN (Fine-tuned)	4.25 (±0.07)
Glow-TTS + HiFi-GAN	4.14 (±0.07)
Glow-TTS + HiFi-GAN (Fine-tuned)	4.32 (±0.07)
VITS (DDP)	4.39 (±0.06)
VITS	**4.43 (±0.06)**

We conducted an ablation study to demonstrate the effectiveness of our methods, including the normalized flow in the prior encoder and linear-scale spectrogram posterior input. All models in the ablation study were trained up to 300k steps. The results are shown in Table 2. Removing the normalizing flow in the prior encoder results in a 1.52 MOS decrease from the baseline, demonstrating that the prior distribution's flexibility significantly influences the synthesis quality. Replacing the linear-scale spectrogram for posterior input with the mel-spectrogram results in a quality degradation (-0.19 MOS), indicating that the high-resolution information is effective for VITS in improving the synthesis quality.

Table 2. MOS comparison in the ablation studies.

Model	MOS (CI)
Ground Truth	4.50 (±0.06)
Baseline	4.50 (±0.06)
without Normalizing Flow	2.98 (±0.08)
with Mel-spectrogram	4.31 (±0.08)

4.2. Generalization to Multi-Speaker Text-to-Speech

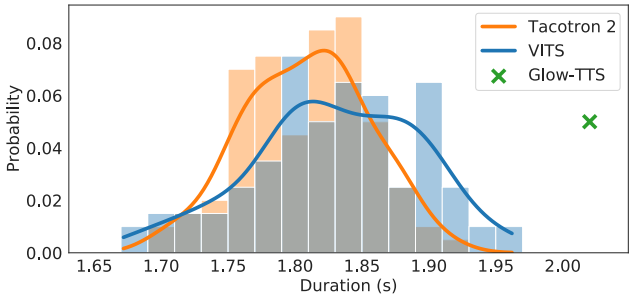
To verify that our model can learn and express diverse speech characteristics, we compared our model to Tacotron 2, Glow-TTS and HiFi-GAN, which showed the ability to extend to multi-speaker speech synthesis (Jia et al., 2018; Kim et al., 2020; Kong et al., 2020). We trained the models on the VCTK dataset. We added speaker embedding to our model as described in Section 2.5. For Tacotron 2, we broadcasted speaker embedding and concatenated it with the encoder output, and for Glow-TTS, we applied the global conditioning following the previous work. The evaluation method is the same as that described in Section 4.1. As shown in Table 3, our model achieves a higher MOS than the other models. This demonstrates that our model learns and expresses various speech characteristics in an end-to-end manner.

Table 3. Comparison of evaluated MOS with 95% confidence intervals on the VCTK dataset.

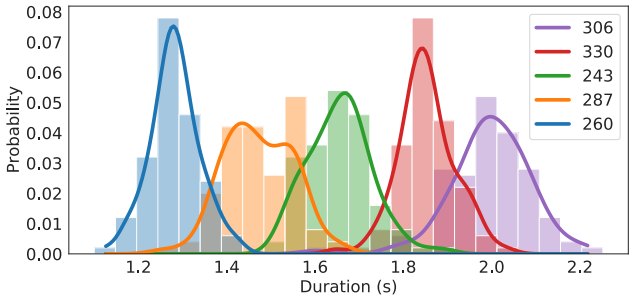
Model	MOS (CI)
Ground Truth	4.38 (±0.07)
Tacotron 2 + HiFi-GAN	3.14 (±0.09)
Tacotron 2 + HiFi-GAN (Fine-tuned)	3.19 (±0.09)
Glow-TTS + HiFi-GAN	3.76 (±0.07)
Glow-TTS + HiFi-GAN (Fine-tuned)	3.82 (±0.07)
VITS	**4.38 (±0.06)**

4.3. Speech Variation

We verified how many different lengths of speech the stochastic duration predictor produces, and how many different speech characteristics the synthesized samples have.

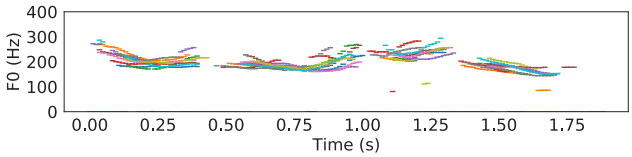


(a) Comparison of sample duration. Glow-TTS only provides a single value due to the deterministic duration predictor.

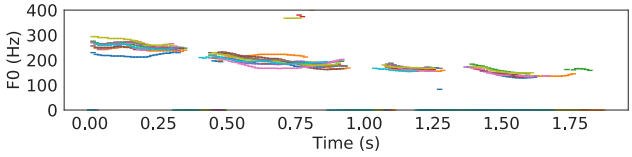


(b) Comparison of sample duration in different speakers.

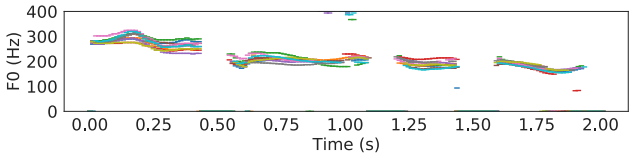
Figure 2. Sample duration in seconds on (a) the LJ Speech dataset and (b) the VCTK dataset.



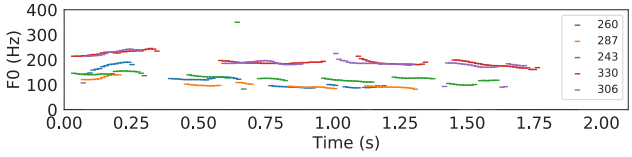
(a) VITS



(b) Tacotron 2



(c) Glow-TTS



(d) VITS (multi-speaker)

Figure 3. Pitch tracks for the utterance "How much variation is there?". Samples are generated from (a) VITS, (b) Tacotron 2, and (c) Glow-TTS in the single speaker setting and from (d) VITS in the multi-speaker setting.

Similar to Valle et al. (2021), all samples here were generated from a sentence "*How much variation is there?*". Figure 2a shows histograms of the lengths of 100 generated utterances from each model. While Glow-TTS generates only fixed-length utterances due to the deterministic duration predictor, samples from our model follow a similar length distribution to that of Tacotron 2. Figure 2b shows the lengths of 100 utterances generated with each of five speaker identities from our model in the multi-speaker setting, implying that the model learns the speaker-dependent phoneme duration. F0 contours of 10 utterances extracted with the YIN algorithm (De Cheveigné & Kawahara, 2002) in Figure 3 shows that our model generates speech with diverse pitches and rhythms, and five samples generated with each of different speaker identities in Figure 3d demonstrates our model expresses very different lengths and pitches of speech for each speaker identity. Note that Glow-TTS could increase the diversity of pitch by increasing the standard deviation of the prior distribution, but on the contrary, it could lower the synthesis quality.

4.4. Synthesis Speed

We compared the synthesis speed of our model with a parallel two-stage TTS system, Glow-TTS and HiFi-GAN. We measured the synchronized elapsed time over the entire process to generate raw waveforms from phoneme sequences with 100 sentences randomly selected from the test set of the LJ Speech dataset. We used a single NVIDIA V100 GPU with a batch size of 1. The results are shown in Table 4. Since our model does not require modules for generating predefined intermediate representations, its sampling efficiency and speed are greatly improved.

5. Related Work

5.1. End-to-End Text-to-Speech

Currently, neural TTS models with a two-stage pipeline can synthesize human-like speech (Oord et al., 2016; Ping et al., 2018; Shen et al., 2018). However, they typically require vocoders trained or fine-tuned with first stage model output, which causes training and deployment inefficiency. They are also unable to reap the potential benefits of an end-to-end approach that can use learned hidden representations rather than predefined intermediate features.

Table 4. Comparison of the synthesis speed. Speed of n kHz means that the model can generate $n \times 1000$ raw audio samples per second. Real-time means the synthesis speed over real-time.

Model	Speed (kHz)	Real-time
Glow-TTS + HiFi-GAN	606.05	$\times$27.48
VITS	1480.15	$\times$67.12
VITS (DDP)	2005.03	$\times$90.93

Recently, single-stage end-to-end TTS models have been proposed to tackle the more challenging task of generating raw waveforms, which contain richer information (e.g., high-frequency response and phase) than mel-spectrograms, directly from text. FastSpeech 2s (Ren et al., 2021) is an extension of FastSpeech 2 that enables end-to-end parallel generation by adopting adversarial training and an auxiliary mel-spectrogram decoder that helps learn text representations. However, to resolve the one-to-many problem, FastSpeech 2s must extract phoneme duration, pitch, and energy from speech used as input conditions in training. EATS (Donahue et al., 2021) employs adversarial training as well and a differentiable alignment scheme. To resolve the length mismatch problem between generated and target speech, EATS adopts soft dynamic time warping loss that is calculated by dynamic programming. Wave Tacotron (Weiss et al., 2020) combines normalizing flows with Tacotron 2 for an end-to-end structure but remains autoregressive. The audio quality of all the aforementioned end-to-end TTS models is less than that of two-stage models.

Unlike the aforementioned end-to-end models, by utilizing a conditional VAE, our model 1) learns to synthesize raw waveforms directly from text without requiring additional input conditions, 2) uses a dynamic programming method, MAS, to search the optimal alignment rather than to calculate loss, 3) generates samples in parallel, and 4) outperforms the best publicly available two-stage models.

5.2. Variational Autoencoders

VAEs (Kingma & Welling, 2014) are one of the most widely used likelihood-based deep generative models. We adopt a conditional VAE to a TTS system. A conditional VAE is a conditional generative model where the observed conditions modulate the prior distribution of latent variables used to generate outputs. In speech synthesis, Hsu et al. (2019) and Zhang et al. (2019) combine Tacotron 2 and VAEs to learn speech style and prosody. BVAE-TTS (Lee et al., 2021) generates mel-spectrograms in parallel based on a bidirectional VAE (Kingma et al., 2016). Unlike the previous works that applied VAEs to first stage models, we adopt a VAE to a parallel end-to-end TTS system.

Rezende & Mohamed (2015), Chen et al. (2017) and Ziegler & Rush (2019) improve VAE performance by enhancing the expressive power of prior and posterior distribution with normalizing flows. To improve the representation power of the prior distribution, we add normalizing flows to our conditional prior network, leading to the generation of more realistic samples.

Similar to our work, Ma et al. (2019) proposed a conditional VAE with normalizing flows in a conditional prior network for non-autoregressive neural machine translation, FlowSeq. However, the fact that our model can explicitly align a latent sequence with the source sequence differs from FlowSeq, which needs to learn implicit alignment through attention mechanisms. Our model removes the burden of transforming the latent sequence into standard normal random variables by matching the latent sequence with the time-aligned source sequence via MAS, which allows for simpler architecture of normalizing flows.

5.3. Duration Prediction in Non-Autoregressive Text-to-Speech

Autoregressive TTS models (Taigman et al., 2018; Shen et al., 2018; Valle et al., 2021) generate diverse speech with different rhythms through their autoregressive structure and several tricks including maintaining dropout probability during inference and priming (Graves, 2013). Parallel TTS models (Ren et al., 2019; Peng et al., 2020; Kim et al., 2020; Ren et al., 2021; Lee et al., 2021), on the other hand, have been relied on deterministic duration prediction. It is because parallel models have to predict target phoneme duration or the total length of target speech in one feed-forward path, which makes it hard to capture the correlated joint distribution of speech rhythms. In this work, we suggest a flow-based stochastic duration predictor that learns the joint distribution of the estimated phoneme duration, resulting in the generation of diverse speech rhythms in parallel.

6. Conclusion

In this work, we proposed a parallel TTS system, VITS, that can learn and generate in an end-to-end manner. We further introduced the stochastic duration predictor to express diverse rhythms of speech. The resulting system synthesizes natural sounding speech waveforms directly from text, without having to go through predefined intermediate speech representations. Our experimental results show that our method outperforms two-stage TTS systems and achieves close to human quality. We hope the proposed method will be used in many speech synthesis tasks, where two-stage TTS systems have been used, to achieve performance improvement and enjoy the simplified training procedure. We also want to point out that even though our method integrates two separated generative pipelines in TTS systems,

there remains a problem of text preprocessing. Investigating self-supervised learning of language representations could be a possible direction for removing the text preprocessing step. We will release our source-code and pre-trained models to facilitate research in plenty of future directions.

Acknowledgements

We would like to thank Sungwon Lyu, Bokyung Son, Sunghyo Chung, and Jonghoon Mo for helpful discussions and advice.

References

Bernard, M. Phonemizer. https://github.com/bootphon/phonemizer, 2021.

Bińkowski, M., Donahue, J., Dieleman, S., Clark, A., Elsen, E., Casagrande, N., Cobo, L. C., and Simonyan, K. High Fidelity Speech Synthesis with Adversarial Networks. In *International Conference on Learning Representations*, 2019.

Chen, J., Lu, C., Chenli, B., Zhu, J., and Tian, T. Vflow: More expressive generative flows with variational data augmentation. In *International Conference on Machine Learning*, pp. 1660–1669. PMLR, 2020.

Chen, X., Kingma, D. P., Salimans, T., Duan, Y., Dhariwal, P., Schulman, J., Sutskever, I., and Abbeel, P. Variational lossy autoencoder. 2017. URL https://openreview.net/forum?id=BysvGP5ee.

De Cheveigné, A. and Kawahara, H. Yin, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using Real NVP. In *International Conference on Learning Representations*, 2017.

Donahue, J., Dieleman, S., Binkowski, M., Elsen, E., and Simonyan, K. End-to-end Adversarial Text-to-Speech. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=rsf1z-JSj87.

Durkan, C., Bekasov, A., Murray, I., and Papamakarios, G. Neural Spline Flows. In *Advances in Neural Information Processing Systems*, pp. 7509–7520, 2019.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative Adversarial Nets. *Advances in Neural Information Processing Systems*, 27:2672–2680, 2014.

Graves, A. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

Ho, J., Chen, X., Srinivas, A., Duan, Y., and Abbeel, P. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, pp. 2722–2730. PMLR, 2019.

Hsu, W.-N., Zhang, Y., Weiss, R., Zen, H., Wu, Y., Cao, Y., and Wang, Y. Hierarchical Generative Modeling for Controllable Speech Synthesis. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rygkk305YQ.

Ito, K. The LJ Speech Dataset. https://keithito.com/LJ-Speech-Dataset/, 2017.

Jia, Y., Zhang, Y., Weiss, R. J., Wang, Q., Shen, J., Ren, F., Chen, Z., Nguyen, P., Pang, R., Lopez-Moreno, I., et al. Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis. In *Advances in Neural Information Processing Systems*, 2018.

Kalchbrenner, N., Elsen, E., Simonyan, K., Noury, S., Casagrande, N., Lockhart, E., Stimberg, F., Oord, A., Dieleman, S., and Kavukcuoglu, K. Efficient neural audio synthesis. In *International Conference on Machine Learning*, pp. 2410–2419. PMLR, 2018.

Kim, J., Kim, S., Kong, J., and Yoon, S. Glow-TTS: A Generative Flow for Text-to-Speech via Monotonic Alignment Search. *Advances in Neural Information Processing Systems*, 33, 2020.

Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*, 2014.

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. Improved variational inference with inverse autoregressive flow. *Advances in Neural Information Processing Systems*, 29:4743–4751, 2016.

Kong, J., Kim, J., and Bae, J. HiFi-GAN: Generative Adversarial networks for Efficient and High Fidelity Speech Synthesis. *Advances in Neural Information Processing Systems*, 33, 2020.

Kumar, K., Kumar, R., de Boissiere, T., Gestin, L., Teoh, W. Z., Sotelo, J., de Brébisson, A., Bengio, Y., and Courville, A. C. MelGAN: Generative Adversarial Networks for Conditional waveform synthesis. volume 32, pp. 14910–14921, 2019.

Larsen, A. B. L., Sønderby, S. K., Larochelle, H., and Winther, O. Autoencoding beyond pixels using a learned

similarity metric. In *International Conference on Machine Learning*, pp. 1558–1566. PMLR, 2016.

Lee, Y., Shin, J., and Jung, K. Bidirectional Variational Inference for Non-Autoregressive Text-to-speech. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=o3iritJHLfO.

Li, N., Liu, S., Liu, Y., Zhao, S., and Liu, M. Neural speech synthesis with transformer network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 6706–6713, 2019.

Loshchilov, I. and Hutter, F. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Ma, X., Zhou, C., Li, X., Neubig, G., and Hovy, E. Flowseq: Non-autoregressive conditional sequence generation with generative flow. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4273–4283, 2019.

Mao, X., Li, Q., Xie, H., Lau, R. Y., Wang, Z., and Paul Smolley, S. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2794–2802, 2017.

Miao, C., Liang, S., Chen, M., Ma, J., Wang, S., and Xiao, J. Flow-TTS: A non-autoregressive network for text to speech based on flow. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7209–7213. IEEE, 2020.

Oord, A. v. d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

Peng, K., Ping, W., Song, Z., and Zhao, K. Non-autoregressive neural text-to-speech. In *International Conference on Machine Learning*, pp. 7586–7598. PMLR, 2020.

Ping, W., Peng, K., Gibiansky, A., Arik, S. O., Kannan, A., Narang, S., Raiman, J., and Miller, J. Deep Voice 3: 2000-Speaker Neural Text-to-Speech. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HJtEm4p6Z.

Prenger, R., Valle, R., and Catanzaro, B. Waveglow: A flow-based generative network for speech synthesis. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3617–3621. IEEE, 2019.

Ren, Y., Ruan, Y., Tan, X., Qin, T., Zhao, S., Zhao, Z., and Liu, T.-Y. FastSpeech: Fast, Robust and Controllable Text to Speech. volume 32, pp. 3171–3180, 2019.

Ren, Y., Hu, C., Tan, X., Qin, T., Zhao, S., Zhao, Z., and Liu, T.-Y. FastSpeech 2: Fast and High-Quality End-to-End Text to Speech. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=piLPYqxtWuA.

Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In *International Conference on Machine Learning*, pp. 1530–1538. PMLR, 2015.

Shaw, P., Uszkoreit, J., and Vaswani, A. Self-Attention with Relative Position Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 464–468, 2018.

Shen, J., Pang, R., Weiss, R. J., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerrv-Ryan, R., et al. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4779–4783. IEEE, 2018.

Taigman, Y., Wolf, L., Polyak, A., and Nachmani, E. Voiceloop: Voice Fitting and Synthesis via a Phonological Loop. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SkFAWax0-.

Valle, R., Shih, K. J., Prenger, R., and Catanzaro, B. Flowtron: an Autoregressive Flow-based Generative Network for Text-to-Speech Synthesis. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Ig53hpHxS4.

van den Oord, A., Vinyals, O., and Kavukcuoglu, K. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pp. 6309–6318, 2017.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is All you Need. *Advances in Neural Information Processing Systems*, 30:5998–6008, 2017.

Veaux, C., Yamagishi, J., MacDonald, K., et al. CSTR VCTK corpus: English multi-speaker corpus for CSTR voice cloning toolkit. *University of Edinburgh. The Centre for Speech Technology Research (CSTR)*, 2017.

Weiss, R. J., Skerry-Ryan, R., Battenberg, E., Mariooryad, S., and Kingma, D. P. Wave-Tacotron: Spectrogram-free end-to-end text-to-speech synthesis. *arXiv preprint arXiv:2011.03568*, 2020.

Zeng, Z., Wang, J., Cheng, N., Xia, T., and Xiao, J. Aligntts: Efficient feed-forward text-to-speech system without explicit alignment. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6714–6718. IEEE, 2020.

Zhang, Y.-J., Pan, S., He, L., and Ling, Z.-H. Learning latent representations for style control and transfer in end-to-end speech synthesis. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6945–6949. IEEE, 2019.

Ziegler, Z. and Rush, A. Latent normalizing flows for discrete sequences. In *International Conference on Machine Learning*, pp. 7673–7682. PMLR, 2019.

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

A. Monotonic Alignment Search

We present pseudocode for MAS in Figure 4. Although we search the alignment which maximizes the ELBO not the exact log-likelihood of data, we can use the MAS implementation of Glow-TTS as described in Section 2.2.1.

```
def monotonic_alignment_search(value):
    """Returns the most likely alignment for the given log-likelihood matrix.
    Args:
        value: the log-likelihood matrix. Its (i, j)-th entry contains
        the log-likelihood of the j-th latent variable
        for the given i-th prior mean and variance:
        .. math::
            value_{i,j} = log N(f(z)_{j}; \mu_{i}, \sigma_{i})
        (dtype=float, shape=[text_length, latent_variable_length])
    Returns:
        path: the most likely alignment.
        (dtype=float, shape=[text_length, latent_variable_length])
    """
    t_x, t_y = value.shape # [text_length, letent_variable_length]
    path = zeros([t_x, t_y])

    # A cache to store the log-likelihood for the most likely alignment so far.
    Q = -INFINITY * ones([t_x, t_y])

    for y in range(t_y):
        for x in range(max(0, t_x + y - t_y), min(t_x, y + 1)):
            if y == 0: # Base case. If y is 0, the possible x value is only 0.
                Q[x, 0] = value[x, 0]
            else:
                if x == 0:
                    v_prev = -INFINITY
                else:
                    v_prev = Q[x-1, y-1]
                v_cur = Q[x, y-1]
                Q[x, y] = value[x, y] + max(v_prev, v_cur)

    # Backtrack from last observation.
    index = t_x - 1
    for y in range(t_y - 1, -1, -1):
        path[index, y] = 1
        if index != 0 and (index == y or Q[index, y-1] < Q[index-1, y-1]):
            index = index - 1

    return path
```

Figure 4. Pseudocode for Monotonic Alignment Search.

B. Model Configurations

In this section, we mainly describe the newly added parts of VITS as we followed configurations of Glow-TTS and HiFi-GAN for several parts of our model: we use the same transformer encoder and WaveNet residual blocks as those of Glow-TTS; our decoder and the multi-period discriminator is the same as the generator and multi-period discriminator of

HiFi-GAN, respectively, except that we use different input dimension for the decoder and append a sub-discriminator.

B.1. Prior Encoder and Posterior Encoder

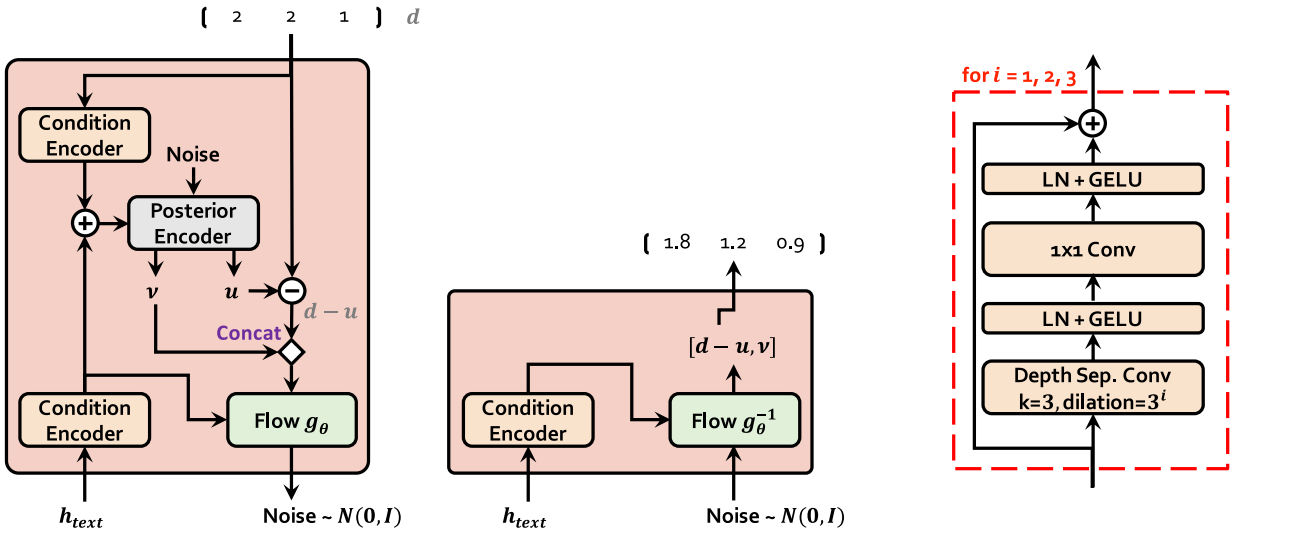
The normalizing flow in the prior encoder is a stack of four affine coupling layers, each coupling layer consisting of four WaveNet residual blocks. As we restrict the affine coupling layers to be volume-preserving transformations, the coupling layers do not produce scale parameters.

The posterior encoder, consisting of 16 WaveNet residual blocks, takes linear-scale log magnitude spectrograms and produce latent variables with 192 channels.

B.2. Decoder and Discriminator

The input of our decoder is latent variables generated from the prior or posterior encoders, so the input channel size of the decoder is 192. For the last convolutional layer of the decoder, we remove a bias parameter, as it causes unstable gradient scales during mixed precision training.

For the discriminator, HiFi-GAN uses the multi-period discriminator containing five sub-discriminators with periods $[2, 3, 5, 7, 11]$ and the multi-scale discriminator containing three sub-discriminators. To improve training efficiency, we leave only the first sub-discriminator of the multi-scale discriminator that operates on raw waveforms and discard two sub-discriminators operating on average-pooled waveforms. The resultant discriminator can be seen as the multi-period discriminator with periods $[1, 2, 3, 5, 7, 11]$.



(a) Training procedure

(b) Inference procedure

(c) Dilated and depth-wise separable convolutional residual block

Figure 5. Block diagram depicting (a) training procedure and (b) inference procedure of the stochastic duration predictor. The main building block of the stochastic duration predictor is (c) the dilated and depth-wise separable convolutional residual block.

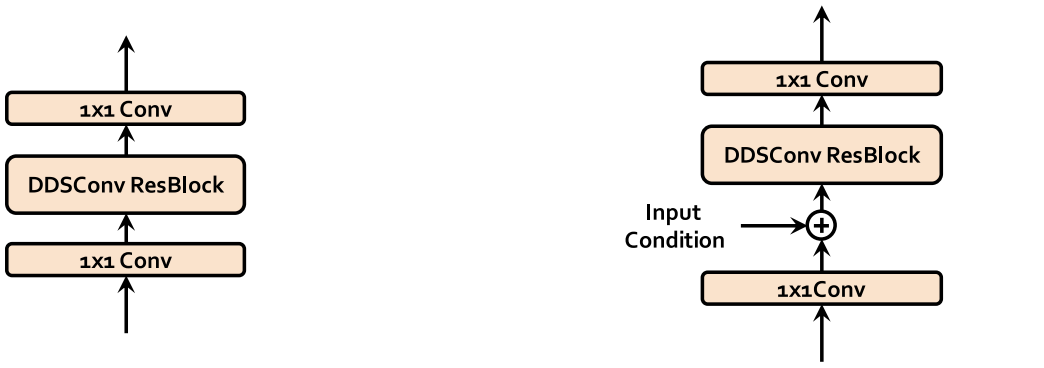
B.3. Stochastic Duration Predictor

Figures 5a and 5b show the training and inference procedures of the stochastic duration predictor, respectively. The main building block of the stochastic duration predictor is the dilated and depth-wise separable convolutional (DDSConv) residual block as in Figure 5c. Each convolutional layer in DDSConv blocks is followed by a layer normalization layer and GELU activation function. We choose to use dilated and depth-wise separable convolutional layers for improving parameter efficiency while maintaining large receptive field size.

The posterior encoder and normalizing flow module in the duration predictor are flow-based neural networks and have the

similar architecture. The difference is that the posterior encoder transforms a Gaussian noise sequence into two random variables ν and u to express the approximate posterior distribution $q_\phi(u, \nu|d, c_{text})$, and the normalizing flow module transforms $d - u$ and ν into a Gaussian noise sequence to express the log-likelihood of the augmented and dequantized data $\log p_\theta(d - u, \nu|c_{text})$ as described in Section 2.2.2.

All input conditions are processed through condition encoders, each consisting of two 1x1 convolutional layers and a DDSConv residual block. The posterior encoder and normalizing flow module have four coupling layers of neural spline flows. Each coupling layer first processes input and input conditions through a DDSConv block and produces 29-channel parameters that are used to construct 10 rational-quadratic functions. We set the hidden dimension of all coupling layers and condition encoders to 192. Figure 6a and 6b show the architecture of a condition encoder and a coupling layer used in the stochastic duration predictor.



(a) Condition encoder in the stochastic duration predictor

(b) Coupling layer in the stochastic duration predictor

Figure 6. The architecture of (a) condition encoder and (b) coupling layer used in the stochastic duration predictor.

C. Side-by-Side Evaluation

We conducted 7-point Comparative Mean Opinion Score (CMOS) evaluation between VITS and the ground truth through 500 ratings on 50 items. Our model achieved -0.106 and -0.270 CMOS on the LJ Speech and the VCTK datasets, respectively, as in Table 5. It indicates that even though our model outperforms the best publicly available TTS system, Glow-TTS and HiFi-GAN, and achieves a comparable score to ground truth in MOS evaluation, there remains a small preference of raters towards the ground truth over our model.

Table 5. Evaluated CMOS of VITS compared to the ground truth.

Dataset	CMOS
LJ Speech	-0.106
VCTK	-0.262

D. Voice Conversion

In the multi-speaker setting, we do not provide speaker identities into the text encoder, which makes the latent variables estimated from the text encoder learn speaker-independent representations. Using the speaker-independent representations, we can transform an audio recording of one speaker into a voice of another speaker. For a given speaker identity s and an utterance of the speaker, we can attain a linear spectrogram x_{lin} from the corresponding utterance audio. We can transform x_{lin} into a speaker-independent representation e through the posterior encoder and the normalizing flow in the prior encoder:

$$z \sim q_\phi(z|x_{lin}, s) \tag{12}$$

$$e = f_\theta(z|s) \tag{13}$$

Then, we can synthesize a voice $\hat{y}$ of a target speaker identity $\hat{s}$ from the representation e through the inverse transformation of the normalizing flow f_θ^{-1} and decoder G:

$$\hat{y} = G(f_\theta^{-1}(e|\hat{s})|\hat{s}) \tag{14}$$

Learning speaker-independent representations and using it for voice conversion can be seen as an extension of the voice conversion method proposed in Glow-TTS. Our voice conversion method provides raw waveforms rather than mel-spectrograms as in Glow-TTS. The voice conversion results are presented in Figure 7. It shows a similar trend of pitch tracks with different pitch levels.

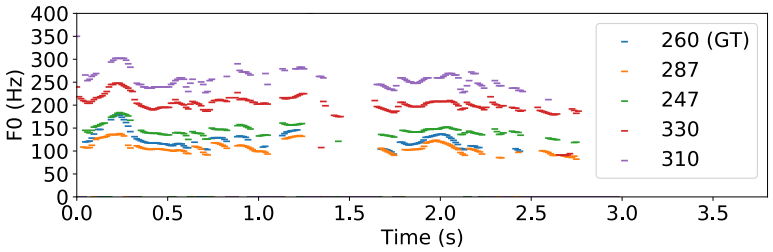


Figure 7. Pitch tracks of a ground truth sample and the corresponding voice conversion samples with different speaker identities.