NaturalSpeech: End-to-End Text to Speech Synthesis with Human-Level Quality

Xu Tan,[*] **Jiawei Chen,**[*] **Haohe Liu,**[*] **Jian Cong, Chen Zhang, Yanqing Liu, Xi Wang**
Yichong Leng, Yuanhao Yi, Lei He, Frank Soong
Tao Qin, Sheng Zhao, Tie-Yan Liu

Microsoft Research Asia & Microsoft Azure Speech

Abstract

Text to speech (TTS) has made rapid progress in both academia and industry in recent years. Some questions naturally arise that whether a TTS system can achieve human-level quality, how to define/judge that quality and how to achieve it. In this paper, we answer these questions by first defining the human-level quality based on the statistical significance of subjective measure and introducing appropriate guidelines to judge it, and then developing a TTS system called NaturalSpeech that achieves human-level quality on a benchmark dataset. Specifically, we leverage a variational autoencoder (VAE) for end-to-end text to waveform generation, with several key modules to enhance the capacity of the prior from text and reduce the complexity of the posterior from speech, including phoneme pre-training, differentiable duration modeling, bidirectional prior/posterior modeling, and a memory mechanism in VAE. Experiment evaluations on popular LJSpeech dataset show that our proposed NaturalSpeech achieves -0.01 CMOS (comparative mean opinion score) to human recordings at the sentence level, with Wilcoxon signed rank test at p-level $p \gg 0.05$, which demonstrates no statistically significant difference from human recordings for the first time on this dataset.

1 Introduction

Text to speech (TTS) aims at synthesizing intelligible and natural speech from text [1], and has made rapid progress in recent years due to the development of deep learning. Neural network based TTS has evolved from CNN/RNN-based models [2, 3, 4, 5, 6, 7, 8] to Transformer-based models [9, 10, 11], from basic generative models (autoregressive) [2, 3, 9] to more powerful models (VAE, GAN, flow, diffusion) [12, 13, 14, 15], from cascaded acoustic models/vocoders [2, 4, 3, 10, 16, 17] to fully end-to-end models [18, 19, 15].

Building TTS systems with human-level quality has always been the dream of the practitioners in speech synthesis. While current TTS systems achieve high voice quality, they still have quality gap compared with human recordings. To pursue this goal, several questions need to be answered: 1) how to define human-level quality in text to speech synthesis? 2) how to judge whether a TTS system has achieved human-level quality or not? 3) how to build a TTS system to achieve human-level quality? In this paper, we conduct a comprehensive study on these problems in TTS. We first give a formal definition on human-level quality in TTS based on a statistical and measurable way (see Definition 1). Then we introduce some guidelines to judge whether a TTS system has achieved human-level quality with a hypothesis test. Using this judge method, we found several previous TTS systems have not achieved it (see Table 1).

---

[*]Equal contribution. Corresponding author: Xu Tan, `xuta@microsoft.com`

In this paper, we further develop a fully end-to-end text to waveform generation system called NaturalSpeech to bridge the quality gap to recordings and achieve human-level quality. Specifically, inspired by image/video/waveform generation [20, 21, 15], we leverage variational autoencoder (VAE) [22] to compress the high-dimensional speech (x) into continuous frame-level representations (denoted as posterior $q(z|x)$), which are used to reconstruct the waveform (denoted as $p(x|z)$). The corresponding prior (denoted as $p(z|y)$) is obtained from the text sequence y. Considering the posterior from speech is more complicated than the prior from text, we design several modules (see Figure 1) to match the posterior and prior as close to each other as possible, to enable text to speech synthesis through $p(z|y) \rightarrow p(x|z)$:

- We leverage large-scale pre-training on the phoneme encoder to extract better representations from phoneme sequence (Section 3.2).
- We leverage a fully differentiable durator[2] that consists of a duration predictor and an upsampling layer to improve the duration modeling (Section 3.3).
- We design a bidirectional prior/posterior module based on flow models [23, 24, 25] to further enhance the prior $p(z|y)$ and reduce the complexity of posterior $q(z|x)$ (Section 3.4).
- We propose a memory based VAE to reduce the complexity of the posterior needed to reconstruct waveform (Section 3.5).

Compared to previous TTS systems, NaturalSpeech has several advantages: 1) Reduce training-inference mismatch. In previous cascaded acoustic model/vocoder pipeline [13, 18, 14] and explicit duration prediction [13, 15, 18], both mel-spectrogram and duration suffer from training-inference mismatch since ground-truth values are used in training the vocoder and mel-spectrogram decoder while predicted values are used in inference. Our fully end-to-end text to waveform generation and differentiable durator can avoid the training-inference mismatch. 2) Alleviate one-to-many mapping problem. One text sequence can correspond to multiple speech utterances with different variation information (e.g., pitch, duration, speed, pause, prosody, etc). Previous works only using variance adaptor [18, 11] to predict pitch/duration cannot well handle the one-to-many mapping problem. Our memory based VAE and bidirectional prior/posterior can reduce the complexity of posterior and enhance the prior, which helps relieve the one-to-many mapping problem. 3) Improve representation capacity. Previous models are not powerful enough to extract good representations from phoneme sequence [13, 15, 14] and learn complicated data distribution in speech [18]. Our large-scale phoneme pre-training and powerful generative models such as flow and VAE can learn better text representations and speech data distributions.

We conduct experimental evaluations on the widely adopted LJSpeech dataset [26] to measure the voice quality of our NaturalSpeech system. Based on the proposed judgement guidelines, NaturalSpeech achieves similar quality with human recordings in terms of MOS (mean opinion score) and CMOS (comparative MOS). Specifically, the speech generated by NaturalSpeech achieves -0.01 CMOS compared to recordings, with p-level $p \gg 0.05$ under Wilcoxon signed rank test, which demonstrates that NaturalSpeech can generate speech with no statistically significant difference from recordings.

2 Definition and Judgement of Human-Level Quality in TTS

In this section, we introduce the formal definition of human-level quality in text to speech synthesis, and describe how to judge whether a TTS system achieves human-level quality or not.

2.1 Definition of Human-Level Quality

We define human-level quality in a statistical and measurable way.

Definition 1. *If there is no statistically significant difference between the quality scores of the speech generated by a TTS system and the quality scores of the corresponding human recordings on a test set, then this TTS system achieves human-level quality on this test set.*

---

[2]Since duration is very important in TTS, especially in non-autoregressive TTS, we name the module related to duration modeling as *durator*, including but not limited to the functionalities of duration prediction and hidden expansion. It is common to come up with new term to revolutionize the concept in speech community, such as *vocoder*, *cepstrum*.

Note that by claiming a TTS system achieves human-level quality on a test set, we do not mean that a TTS system can surpass or replace human, but the quality of this TTS system is statistically indistinguishable from human recordings on this test set.

2.2 Judgement of Human-Level Quality

Judgement Guideline   While there are some objective metrics to measure the quality gap between the generated speech and human recordings, such as PESQ [27], STOI [28], SI-SDR [29], they are not reliable to measure the perception quality in TTS. Therefore, we use subjective evaluation to measure the voice quality. Previous works usually use mean opinion score (MOS) with 5 points (from 1 to 5) to compare the generated speech with recordings. However, MOS is not sensitive enough to the difference in voice quality since the judge simply rates the quality of each sentence alone from the two systems with no paired comparison. Thus, we choose comparative mean opinion score (CMOS) with 7 points (from -3 to 3) as the evaluation metric, where each judge measures the voice quality by comparing samples from two systems head by head. We further conduct Wilcoxon signed rank test [30] to measure whether the two systems are significantly different or not in terms of CMOS evaluation.

Therefore, we list the judgement guidelines of human-level quality as follows: 1) Each utterance from TTS system and human recordings should be listened and compared side-by-side by more than 20 judges, who should be native language speakers. At least 50 test utterances from each system should be used in the judgement. 2) The speech generated by TTS system has no statistically significant difference from human recordings, if and only if the average CMOS is close to 0 and the p-level of Wilcoxon signed rank test satisfies $p > 0.05$.

Judgement of Previous TTS Systems   Based on these guidelines, we test whether current TTS systems can achieve human-level quality or not on the LJSpeech dataset. The systems we study include: 1) FastSpeech 2 [18] + HiFiGAN [17], 2) Glow-TTS [13] + HiFiGAN [17], 3) Grad-TTS [14] + HiFiGAN [17], 4) VITS [15]. We re-produce the results of all these systems by our own, which can match or even beat the quality in their original papers (note that the HiFiGAN vocoder is fine-tuned on the predicted mel-spectrograms for better synthesis quality). We use 50 test utterances, each with 20 judges for MOS and CMOS evaluation. As shown in Table 1, although the current TTS systems can achieve close MOS with recordings, they have a large CMOS gap to recordings, with Wilcoxon signed rank test at p-level $p \ll 0.05$, which shows statistically significant difference from human recordings. We further study where the quality gap comes from by analyzing each component in one of the above TTS systems in Appendix A.

Table 1: The MOS and CMOS comparisons between previous TTS systems and human recordings. Note that the Wilcoxon p-value in MOS is conducted using Wilcoxon rank sum test [30], instead of the Wilcoxon signed rank test in CMOS, due to no paired comparison in MOS evaluation. For Grad-TTS, we use 1000 steps for inference.

System	MOS	Wilcoxon p-value	CMOS	Wilcoxon p-value
Human Recordings	4.52 ± 0.11	-	0	-
FastSpeech 2 [18] + HiFiGAN [17]	4.32 ± 0.10	1.0e-05	-0.30	5.1e-20
Glow-TTS [13] + HiFiGAN [17]	4.33 ± 0.10	1.3e-06	-0.23	8.7e-17
Grad-TTS [14] + HiFiGAN [17]	4.37 ± 0.10	0.0127	-0.23	1.2e-11
VITS [15]	4.49 ± 0.10	0.2429	-0.19	2.9e-04

3 Description of NaturalSpeech System

To bridge the quality gap to human recordings, we develop NaturalSpeech, a fully end-to-end text to waveform generation model. We first describe the design principle of our system (Section 3.1), and then introduce each module of this system (Section 3.2-3.5) and training/inference pipeline (Section 3.6), and finally explain why our system can bridge the quality gap to human recordings (Section 3.7).
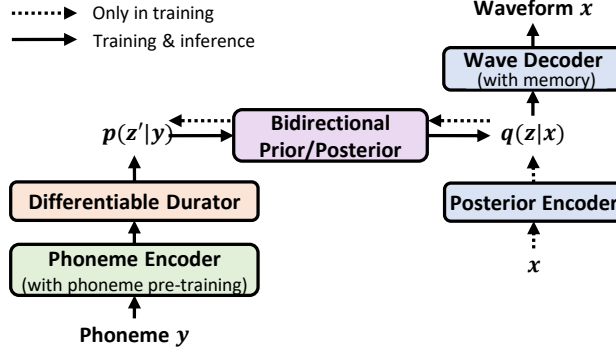
Figure 1: System overview of NaturalSpeech.

3.1 Design Principle

Inspired by image/video generation [21, 31, 32, 33, 34] that uses VQ-VAE [20, 35, 36] to compress high-dimensional image into low-dimensional representations to ease the generation, we leverage VAE [22] to compress high-dimensional speech x into frame-level representations z (i.e., z is sampled from posterior distribution $q(z|x)$), which are used to reconstruct the waveform (denoted as $p(x|z)$). In general formulation of VAE, the prior $p(z)$ is chosen to be standard isotropic multivariate Gaussian. To enable conditional waveform generation from input text in TTS, we predict z from phoneme sequence y, i.e., z is sampled from predicted prior distribution $p(z|y)$. We jointly optimize the VAE and the prior prediction with gradients propogating to both $q(z|x)$ and $p(z|y)$. Derived from the evidence lower bound [22], the loss function consists of a waveform reconstruction loss $-\log p(x|z)$ and a Kullback-Leibler divergence loss between the posterior $q(z|x)$ and the prior $p(z|y)$, i.e., $KL[q(z|x)||p(z|y)]$.

Considering the posterior from speech is more complicated than the prior from text, to match them as close as possible to enable text to waveform generation, we design several modules to simplify the posterior and to enhance the prior, as shown in Figure 1. First, to learn a good representations of phoneme sequence for better prior prediction, we pre-train a phoneme encoder on a large-scale text corpus using masked language modeling on phoneme sequence (Section 3.2). Second, since the posterior is at the frame level while the phoneme prior is at the phoneme level, we need to expand the phoneme prior according to its duration to bridge the length difference. We leverage a differentiable durator to improve duration modeling (Section 3.3). Third, we design a bidirectional prior/posterior module to enhance the prior or simplify the posterior (Section 3.4). Fourth, we propose a memory based VAE that leverages a memory bank through Q-K-V attention [37] to reduce the complexity of posterior needed to reconstruct the waveform (Section 3.5).

3.2 Phoneme Encoder

The phoneme encoder θ_{pho} takes a phoneme sequence y as input and outputs a phoneme hidden sequence. To enhance the representation capability of the phoneme encoder, we conduct large-scale phoneme pre-training. Previous works [38] conduct pre-training in character/word level and apply the pre-trained model to phoneme encoder, which will cause inconsistency, and the works [39] directly using phoneme pre-training will suffer from limited capacity due to too small size of phoneme vocabulary. To avoid these issues, we leverage mixed-phoneme pre-training [40], which uses both phoneme and sup-phoneme (adjacent phonemes merged together) as the input of the model, as shown in Figure 2c. When using masked language modeling [41], we randomly mask some sup-phoneme tokens and their corresponding phoneme tokens and predict the masked phoneme and sup-phoneme at the same time. After mixed phoneme pre-training, we use the pre-trained model to initialize the phoneme encoder of our TTS system.

3.3 Differentiable Durator

The differentiable durator θ_{dur} takes a phoneme hidden sequence as input, and outputs a sequence of prior distribution at the frame level, as shown in Figure 2a. We denote the prior distribution as
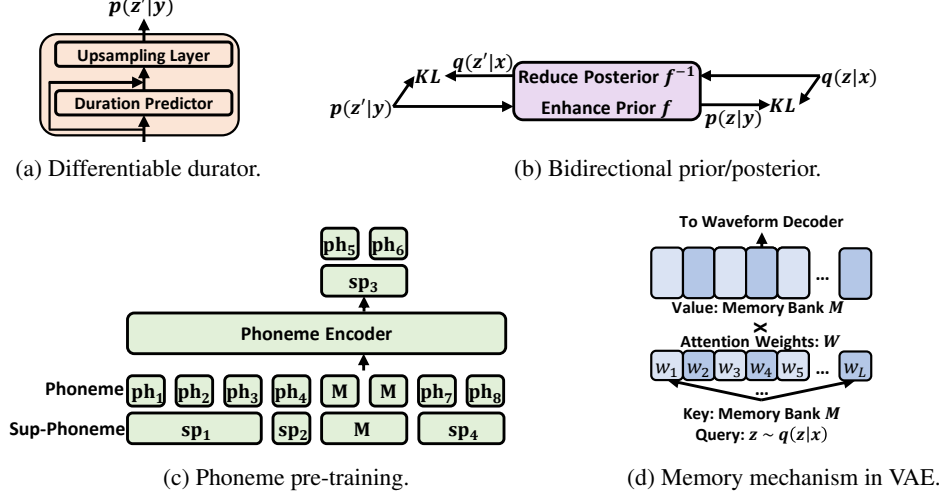
4

Figure 2: The designed modules in NaturalSpeech.

$p(z'|y; \theta_{\text{pho}}, \theta_{\text{dur}}) = p(z'|y; \theta_{\text{pri}})$, where $\theta_{\text{pri}} = [\theta_{\text{pho}}, \theta_{\text{dur}}]$. The differentiable durator θ_{dur} consists of several modules: 1) a duration predictor that builds upon the phoneme encoder to predict the duration for each phoneme, 2) a learnable upsampling layer that leverages the predicted duration to learn a projection matrix to extend the phoneme hidden sequence from phoneme level to frame level in a differentiable way [42], and 3) two additional linear layers on the expanded hidden sequence to calculate the mean and variance of the prior distribution $p(z'|y; \theta_{\text{pri}})$. The detailed formulation of differentiable durator is in Appendix B. We optimize the duration prediction, learnable upsampling layer, and mean/variance linear layers together with the TTS model in a fully differentiable way, which can reduce the training-inference mismatch in previous duration prediction (ground-truth duration is used in training while predicted duration is used in inference) [13, 15, 18] and better use duration in a soft and flexible way instead of a hard expansion, hence the side-effect of inaccurate duration prediction is mitigated.

3.4 Bidirectional Prior/Posterior

As shown in Figure 2b, we design a bidirectional prior/posterior module to enhance the capacity of the prior $p(z'|y; \theta_{\text{pri}})$ or to reduce the complexity of the posterior $q(z|x; \phi)$ where ϕ is the posterior encoder, since there is information gap between the posterior obtained from speech sequence and the prior obtained from phoneme sequence. We choose a flow model [23, 43, 24, 25] as the bidirectional prior/posterior module (denoted as θ_{bpp}) since it is easy to optimize and has a nice property of invertibility.

Reduce Posterior $q(z|x; \phi)$ **with Backward Mapping** f^{-1} The bidirectional prior/posterior module can reduce the complexity of posterior from $q(z|x; \phi)$ to $q(z'|x; \phi, \theta_{\text{bpp}})$ through the backward mapping $f^{-1}(z; \theta_{\text{bpp}})$, i.e., for $z \sim q(z|x; \phi)$, $z' = f^{-1}(z; \theta_{\text{bpp}}) \sim q(z'|x; \phi, \theta_{\text{bpp}})$. The objective is to match the simplified posterior $q(z'|x; \phi, \theta_{\text{bpp}})$ to the prior $p(z'|y; \theta_{\text{pri}})$ by using the KL divergence loss as follows:

$$
\begin{aligned}
\mathcal{L}_{\text{bwd}}(\phi, \theta_{\text{bpp}}, \theta_{\text{pri}}) &= KL[q(z'|x; \phi, \theta_{\text{bpp}}) || p(z'|y; \theta_{\text{pri}})] = \int q(z'|x; \phi, \theta_{\text{bpp}}) \cdot \log \frac{q(z'|x; \phi, \theta_{\text{bpp}})}{p(z'|y; \theta_{\text{pri}})} dz' \\
&= \int q(z|x; \phi) |\det \frac{\partial f^{-1}(z; \theta_{\text{bpp}})}{\partial z}|^{-1} \cdot \log \frac{q(z|x; \phi)|\det \frac{\partial f^{-1}(z; \theta_{\text{bpp}})}{\partial z}|^{-1}}{p(f^{-1}(z; \theta_{\text{bpp}})|y; \theta_{\text{pri}})} \cdot |\det \frac{\partial f^{-1}(z; \theta_{\text{bpp}})}{\partial z}| dz \\
&= \int q(z|x; \phi) \cdot \log \frac{q(z|x; \phi)}{p(f^{-1}(z; \theta_{\text{bpp}})|y; \theta_{\text{pri}})|\det \frac{\partial f^{-1}(z; \theta_{\text{bpp}})}{\partial z}|} dz \\
&= \mathbb{E}_{z \sim q(z|x; \phi)}(\log q(z|x; \phi) - \log(p(f^{-1}(z; \theta_{\text{bpp}})|y; \theta_{\text{pri}})|\det \frac{\partial f^{-1}(z; \theta_{\text{bpp}})}{\partial z}|),
\end{aligned}
\tag{1}
$$

where the third equality (the second line) in Equation 1 is obtained via the change of variables: $dz' = |\det \frac{\partial f^{-1}(z;\theta_{\text{bpp}})}{\partial z}|dz$, and $q(z'|x;\phi,\theta_{\text{bpp}}) = q(z|x;\phi)|\det \frac{\partial f(z';\theta_{\text{bpp}})}{\partial z'}| = q(z|x;\phi)|\det \frac{\partial f^{-1}(z;\theta_{\text{bpp}})}{\partial z}|^{-1}$ according to inverse function theorem.

Enhance Prior $p(z'|y;\theta_{\text{pri}})$ with Forward Mapping f   The bidirectional prior/posterior module can enhance the capacity of prior from $p(z'|y;\theta_{\text{pri}})$ to $p(z|y;\theta_{\text{pri}},\theta_{\text{bpp}})$ through the forward mapping $f(z';\theta_{\text{bpp}})$, i.e., for $z' \sim p(z'|y;\theta_{\text{pri}})$, $z = f(z';\theta_{\text{bpp}}) \sim p(z|y;\theta_{\text{pri}},\theta_{\text{bpp}})$. The objective is to match the enhanced prior $p(z|y;\theta_{\text{pri}},\theta_{\text{bpp}})$ to the posterior $q(z|x;\phi)$ using the KL divergence loss as follows:

$$\mathcal{L}_{\text{fwd}}(\phi,\theta_{\text{bpp}},\theta_{\text{pri}}) = KL[p(z|y;\theta_{\text{pri}},\theta_{\text{bpp}})||q(z|x;\phi)] = \int p(z|y;\theta_{\text{pri}},\theta_{\text{bpp}}) \cdot \log \frac{p(z|y;\theta_{\text{pri}},\theta_{\text{bpp}})}{q(z|x;\phi)}dz$$

$$= \int p(z'|y;\theta_{\text{pri}})|\det \frac{\partial f(z';\theta_{\text{bpp}})}{\partial z'}|^{-1} \cdot \log \frac{p(z'|y;\theta_{\text{pri}})|\det \frac{\partial f(z';\theta_{\text{bpp}})}{\partial z'}|^{-1}}{q(f(z';\theta_{\text{bpp}})|x;\phi)} \cdot |\det \frac{\partial f(z';\theta_{\text{bpp}})}{\partial z'}|dz'$$

$$= \mathbb{E}_{z' \sim p(z'|y;\theta_{\text{pri}})}(\log p(z'|y;\theta_{\text{pri}}) - \log q(f(z';\theta_{\text{bpp}})|x;\phi)|\det \frac{\partial f(z';\theta_{\text{bpp}})}{\partial z'}|),$$
(2)

where the third equality (the second line) in Equation 2 is obtained via the change of variables: $dz = |\det \frac{\partial f(z';\theta_{\text{bpp}})}{\partial z'}|dz'$, and $p(z|y;\theta_{\text{pri}},\theta_{\text{bpp}}) = p(z'|y;\theta_{\text{pri}})|\det \frac{\partial f^{-1}(z;\theta_{\text{bpp}})}{\partial z}| = p(z'|y;\theta_{\text{pri}})|\det \frac{\partial f(z';\theta_{\text{bpp}})}{\partial z'}|^{-1}$ according to inverse function theorem, similar to that in Equation 1.

By using backward and forward loss functions, both directions of the flow model are considered in training, which can reduce the training-inference mismatch in the previous flow models that train in backward direction but infer in forward direction. We also provide another formulation of the bidirectional prior/posterior in Appendix C.

3.5   VAE with Memory

The posterior $q(z|x;\phi)$ in the original VAE model is used to reconstruct the speech waveform, and thus is more complicated than the prior from the phoneme sequence. To further relieve the burden of prior prediction, we simplify the posterior by designing a memory based VAE model. The high-level idea of this design is that instead of directly using $z \sim q(z|x;\phi)$ for waveform reconstruction, we just use z as a query to attend to a memory bank, and use the attention result for waveform reconstruction, as shown in Figure 2d. In this way, the posterior z is only used to determine the attention weights in the memory bank, and thus is largely simplified. The waveform reconstruction loss based on memory VAE can be formulated as

$$\mathcal{L}_{\text{rec}}(\phi,\theta_{\text{dec}}) = -\mathbb{E}_{z \sim q(z|x;\phi)}[\log p(x|\text{Attention}(z,M,M);\theta_{\text{dec}})],$$

$$\text{Attention}(Q,K,V) = [\text{softmax}(\frac{QW_Q(KW_K)^T}{\sqrt{h}})VW_V]W_O,$$
(3)

where θ_{dec} denotes the waveform decoder, which covers not only the original waveform decoder but also the model parameters related to the memory mechanism, including the memory bank M and the attention parameters W_Q, W_K, W_V, and W_O, where $M \in \mathbb{R}^{L \times h}$ and $W_* \in \mathbb{R}^{h \times h}$, L is the size of the memory bank and h is the hidden dimension.

3.6   Training and Inference Pipeline

Besides the waveform reconstruction loss and bidirectional prior/posterior loss, we additionally conduct a fully end-to-end optimization to take the whole inference procedure in training for better voice quality. The loss function is formulated as follows.

$$\mathcal{L}_{\text{e2e}}(\theta_{\text{pri}},\theta_{\text{bpp}},\theta_{\text{dec}}) = -\mathbb{E}_{z' \sim p(z'|y;\theta_{\text{pri}})}[\log p(x|\text{Attention}(f(z';\theta_{\text{bpp}}),M,M);\theta_{\text{dec}})].$$
(4)

Based on Equation 1, 2, 3, and 4, the total loss function is

$$\mathcal{L} = \mathcal{L}_{\text{bwd}}(\phi,\theta_{\text{pri}},\theta_{\text{bpp}}) + \mathcal{L}_{\text{fwd}}(\phi,\theta_{\text{pri}},\theta_{\text{bpp}}) + \mathcal{L}_{\text{rec}}(\phi,\theta_{\text{dec}}) + \mathcal{L}_{\text{e2e}}(\theta_{\text{pri}},\theta_{\text{bpp}},\theta_{\text{dec}}),$$
(5)

where $\theta_{\text{pri}} = [\theta_{\text{pho}}, \theta_{\text{dur}}]$.

Note that there are some special explanations of the above loss functions: 1) Since the frame-level prior distribution $p(z'|y; \theta_{\text{pri}})$ cannot well align with the ground-truth speech frames due to the intrinsically inaccurate duration prediction in durator, we leverage a soft dynamic time warping (DTW) version of KL loss for $\mathcal{L}_{\text{bwd}}$ and $\mathcal{L}_{\text{fwd}}$. See Appendix D for the detailed formulation of the soft-DTW loss. 2) We write the waveform loss in $\mathcal{L}_{\text{rec}}$ and $\mathcal{L}_{\text{e2e}}$ as negative log-likelihood loss for simplicity. Actually following [17], $\mathcal{L}_{\text{rec}}$ consists of GAN loss, feature mapping loss and mel-spectrogram loss, while $\mathcal{L}_{\text{e2e}}$ consists of only GAN loss. We do not use soft-DTW in $\mathcal{L}_{\text{e2e}}$ since we found GAN loss can still perform well with mismatched lengths. See Appendix E for the details of the waveform loss.
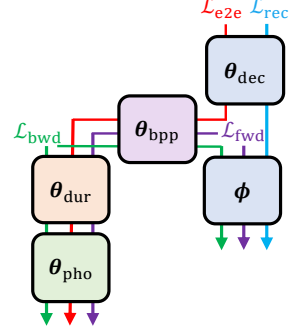

Figure 3: Gradient flows.

There are several different gradient flows in training the model, as shown in Figure 3: 1) $\mathcal{L}_{\text{rec}} \rightarrow \theta_{\text{dec}} \rightarrow \phi$; 2) $\mathcal{L}_{\text{bwd}} \rightarrow \theta_{\text{dur}} \rightarrow \theta_{\text{pho}}$; 3) $\mathcal{L}_{\text{bwd}} \rightarrow \theta_{\text{bpp}} \rightarrow \phi$; 4) $\mathcal{L}_{\text{fwd}} \rightarrow \theta_{\text{bpp}} \rightarrow \theta_{\text{dur}} \rightarrow \theta_{\text{pho}}$; 5) $\mathcal{L}_{\text{fwd}} \rightarrow \phi$; 6) $\mathcal{L}_{\text{e2e}} \rightarrow \theta_{\text{dec}} \rightarrow \theta_{\text{bpp}} \rightarrow \theta_{\text{dur}} \rightarrow \theta_{\text{pho}}$. After training, we discard the posterior encoder ϕ and only use $\theta_{\text{pho}}, \theta_{\text{bpp}}, \theta_{\text{dur}}$ and θ_{dec} for inference. The training and inference pipeline is summarized in Algorithm 1.

Algorithm 1 Training and inference of NaturalSpeech

1: **Training:**
2:     Pre-train the phoneme encoder θ_{pho}.
3:     Train the whole model $[\phi, \theta_{\text{pho}}, \theta_{\text{dur}}, \theta_{\text{bpp}}, \theta_{\text{dec}}]$ using loss $\mathcal{L}$ defined in Equation 5.
4: **Inference:**
5:     Sample prior $z' \sim p(z'|y; \theta_{\text{pho}}, \theta_{\text{dur}})$.
6:     Get enhanced prior $z = f(z'; \theta_{\text{bpp}})$.
7:     Generate waveform sample $x \sim p(x|\text{Attention}(z, M, M); \theta_{\text{dec}})$.

3.7 Advantages of NaturalSpeech

We explain how the designs in our NaturalSpeech system can close the quality gap to recordings.

- *Reduce training-inference mismatch.* We directly generate waveform from text and leverage a differentiable durator to ensure a fully end-to-end optimization, which can reduce the training-inference mismatch in the cascaded acoustic model/vocoder [13, 18, 14, 42] and explicit duration prediction [15, 13, 18]. Note that although VAE and flow can have training-inference mismatch inherently (waveform is reconstructed from the posterior in training while predicted from the prior in inference for VAE, and flow is trained in backward direction and infered in forward direction), we design the backward/forward loss in Equation 1 and 2 and the end-to-end loss in Equation 4 to alleviate this problem.

- *Alleviate one-to-many mapping problem.* Compared to previous methods using reference encoder [44, 45, 46, 11] or pitch/energy extraction [18] for variation information modeling, our posterior encoder ϕ in VAE acts like a reference encoder that can extract all the necessary variance information in posterior distribution $q(z|x; \phi)$. We do not predict pitch explicitly since it can be learned implicitly in the posterior encoder and the memory bank of VAE. To ensure the prior and posterior can match with each other, on the one hand, we simplify the posterior with memory VAE and backward mapping in the bidirectional prior/posterior module, and on the other hand, we enhance the prior with phoneme pre-training, differentiable durator, and forward mapping in the bidirectional prior/posterior module. Thus, we can alleviate the one-to-mapping problem to a large extent.

- *Increase representation capacity.* We leverage large-scale phoneme pre-training to extract better representation from the phoneme sequence, and leverage the advanced generative models (flow, VAE, GAN) to capture the speech data distributions better, which can enhance the representation capacity of the TTS models for better voice quality.

We further list the difference between our NaturalSpeech and previous TTS systems as follows: 1) Compared to previous autoregressive TTS models such as Tacotron 1/2 [4, 3], WaveNet [2], TransformerTTS [9], and Wave-Tacotron [47], our NaturalSpeech is non-autoregressive in nature

with a fast inference speed. 2) Compared to the previous systems with cascaded acoustic model and vocoder, such as Tacotron 1/2 [4, 3], FastSpeech 1/2 [10, 18], ParallelTacotron 2 [42], Glow-TTS [13], and Grad-TTS [14], we are fully end-to-end with no cascaded errors. 3) Compared to previous systems with various reference encoders and pitch/duration prediction, such as FastSpeech 2 [18], AdaSpeech [45], and DelightfulTTS [11], we unify all the variance information with a posterior encoder and model the duration in a fully differentiable way. 4) Compared to previous fully end-to-end TTS systems such as EATS [19], FastSpeech 2s [18], and VITS [15], we bridge the quality gap to recordings with advanced model designs to closely match the prior and posterior in the VAE framework.

4 Experiments and Results

4.1 Experimental Settings

Datasets We evaluate our proposed NaturalSpeech on the LJSpeech dataset [26], which is widely used for benchmarking TTS. LJSpeech is a single speaker English corpus and consists of $13, 100$ audios and text transcripts, with a total length of nearly 24 hours at a sampling rate of 22.05kHz. We randomly split the dataset into training set with $12, 500$ samples, validation set with 100 samples, and test set with 500 samples. For phoneme pre-training on phoneme encoder, we collect a large-scale text corpus with 200 million sentences from the news-crawl dataset [48]. Note that we do not use any extra paired text and speech data except for LJSpeech dataset. We conduct several preprocessings on the speech and text sequences: 1) We convert the text/character sequence into phoneme sequence [49] using a grapheme-to-phoneme tool [50]. 2) We use linear-spectrograms as the input of the posterior encoder [15], instead of original waveform sequence for simplicity. The linear-spectrograms are obtained by short-time Fourier transform (STFT) with FFT size, window size, and hop size of 1024, 1024, and 256, respectively. 3) For the mel-spectrogram loss on the waveform decoder, we obtain the mel-spectrograms by applying 80-dimension mel-filterbanks on the linear-spectrograms of the speech waveform.

Model Configurations Our phoneme encoder is a stack of 6 Feed-Forward Transformer (FFT) blocks [10], where each block consists of a multi-head attention layer and a 1D convolution feed-forward layer, with hidden size of 192. In the differentiable durator, the duration predictor consists of 3-layer convolution. We use 4 consecutive affine coupling layers [51] in our bidirectional prior/posterior module following [15]. We discard the scaling operation in the affine transform to stabilize the bidirectional training. The shifting in the affine transform is estimated by a 4-layer WaveNet [2] with a dilation rate of 1. The posterior encoder is based on a 16-layer WaveNet with a kernel size of 5 and a dilation rate of 1. The waveform decoder consists of 4 residual convolution blocks following [17], where each block has 3 layers of 1D convolution. We perform transpose convolution for upsampling at every convolution block at a rate of $[8, 8, 2, 2]$. The hyperparameters of NaturalSpeech are listed in Appendix G.

Training Details We train our proposed system on 8 NVIDIA V100 GPUs with 32G memory, with a dynamic batch size of $8, 000$ speech frames (under hop size of 256) per GPU, and a total 15k training epochs. We use AdamW optimizer [52] with $\beta_1 = 0.8$, $\beta_2 = 0.99$. The initial learning rate is 2×10^{-4}, with a learning rate decay factor $\gamma = 0.999875$ in each epoch, i.e., the learning rate is multiplied by γ in every epoch. We find it is helpful to stabilize the training of our system and achieve better results through a warmup stage with 1k epochs at the beginning of the training, and a tuning stage with 2k epochs at the end of the training. More details about these training stages can be found in Appendix F.

4.2 Comparison with Human Recordings

We first compare the speech generated by NaturalSpeech with human recordings in terms of both MOS and CMOS evaluation. As described in Section 2, we use 50 test utterances, each with 20 judges for evaluation. As shown in Table 2 and 3, our system achieves similar quality scores with human recordings in both MOS and CMOS. Importantly, our system achieves -0.01 CMOS compared to recordings, with a Wilcoxon p-value [30] $p \gg 0.05$, which demonstrates the speech generated by our

system has no statistically significant difference from human recordings[3][4]. Thus, our NaturalSpeech achieves human-level quality according to the definition and judgement in Section 2.

Table 2: MOS comparison between NaturalSpeech and human recordings. Wilcoxon rank sum test is used to measure the p-value in MOS evaluation.

Human Recordings	NaturalSpeech	Wilcoxon p-value
4.58 ± 0.13	4.56 ± 0.13	0.7145

Table 3: CMOS comparison between NaturalSpeech and human recordings. Wilcoxon signed rank test is used to measure the p-value in CMOS evaluation.

Human Recordings	NaturalSpeech	Wilcoxon p-value
0	-0.01	0.6902

4.3 Comparison with Previous TTS Systems

We compare our NaturalSpeech with previous TTS systems, including: 1) FastSpeech 2 [18] + HiFiGAN [17], 2) Glow-TTS [13] + HiFiGAN [17], 3) Grad-TTS [14] + HiFiGAN [17], and 4) VITS [15]. We re-produce the results of all these systems by our own, which can match or even beat the quality in their original papers (note that the HiFiGAN vocoder is fine-tuned on the predicted mel-spectrograms for better synthesis quality). Both the MOS and CMOS results are shown in Table 4. It can be seen that our NaturalSpeech achieves better voice quality than these systems in terms of both MOS and CMOS.

Table 4: MOS and CMOS comparisons between NaturalSpeech and previous TTS systems.

System	MOS	CMOS
FastSpeech 2 [18] + HiFiGAN [17]	4.32 ± 0.15	-0.33
Glow-TTS [13] + HiFiGAN [17]	4.34 ± 0.13	-0.26
Grad-TTS [14] + HiFiGAN [17]	4.37 ± 0.13	-0.24
VITS [15]	4.43 ± 0.13	-0.20
NaturalSpeech	4.56 ± 0.13	0

4.4 Ablation Studies and Method Analyses

Ablation Studies   We further conduct ablation studies to verify the effectiveness of each module in our system, as shown in Table 5. We describe the ablation studies as follows: 1) By removing phoneme pre-training, we do not initialize the phoneme encoder from pre-trained weights but just random initialization, which brings -0.09 CMOS drop, demonstrating the effectiveness of phoneme pre-training. 2) By removing differentiable durator, we do not use learnable upsampling layer and end-to-end duration optimization, but just use duration predictor for hard expansion. In this way, we use monotonic alignment search [13] to provide the duration label to train the duration predictor through the whole training process. Removing differentiable durator causes -0.12 CMOS drop, demonstrates the importance of end-to-end optimization in duration modeling. 3) By removing bidirectional prior/posterior module, we only use $\mathcal{L}_{\text{bwd}}$ in training and do not use $\mathcal{L}_{\text{fwd}}$. It brings -0.09 CMOS drop, showing the gain by leveraging bidirectional training to bridge the gap between posterior and prior. 4) By removing memory mechanism in VAE, we use original VAE for waveform

---

[3]Audio samples can be found in `https://speechresearch.github.io/naturalspeech/`

[4]Note that some human recordings in LJSpeech dataset may contain strange rhythm ups and downs that affect the rating score. To ensure the human recordings used for evaluation are of good quality, we let judges to exclude the recordings with strange rhythms from evaluation. Otherwise, our NaturalSpeech will achieve better CMOS than human recordings. In a CMOS test without excluding bad recordings, NaturalSpeech achieves $+0.09$ CMOS better than recordings.

reconstruction, which causes -0.06 CMOS drop, showing the effectiveness of memory in VAE to simplify the posterior.

Table 5: Ablation studies on each design in NaturalSpeech .

Setting	CMOS
NaturalSpeech	0
— Phoneme Pre-training	-0.09
— Differentiable Durator	-0.12
— Bidirectional Prior/Posterior	-0.09
— Memory in VAE	-0.06

Inference Latency We compare the inference speed of our NaturalSpeech with previous TTS systems. We measure the latency by using an NVIDIA V100 GPU with a batch size of 1 sentence and averaging the latency over the sentences in the test set. The results are shown in Table 6. The model components θ_{pho}, θ_{dur}, θ_{bpp}, and θ_{dec} in NaturalSpeech are used in inference, with 28.7M model parameters. Our NaturalSpeech achieves faster or comparable inference speed when compared with the previous systems, and achieves better voice quality.

Table 6: Inference speed comparison. RTF (real-time factor) means the time (in seconds) to synthesize a 1-second waveform. Grad-TTS (1000) and Grad-TTS (10) mean using 1000 and 10 steps in inference respectively.

System	RTF
FastSpeech 2 [18] + HiFiGAN [17]	0.011
Glow-TTS [13] + HiFiGAN [17]	0.021
Grad-TTS [14] (1000) + HiFiGAN [17]	4.120
Grad-TTS [14] (10) + HiFiGAN [17]	0.082
VITS [15]	0.014
NaturalSpeech	0.013

5 Conclusions and Discussions

In this paper, we conduct a systematic study on the problems related to human-level quality in TTS. We first give a formal definition of human-level quality and describe the guidelines to judge it, and further build a TTS system called NaturalSpeech to achieve human-level quality. Specifically, after analyzing the quality gap on several competitive TTS systems, we develop a fully end-to-end text to waveform generation system, with several designs to close the gap to human recordings, including phoneme pre-training, differentiable durator, bidirectional prior/posterior module, and memory mechanism in VAE. Evaluations on the popular LJSpeech dataset demonstrate that our NaturalSpeech achieves human-level quality with CMOS evaluations, with no statistically significant difference from human recordings for the first time on this dataset.

Note that by claiming our NaturalSpeech system achieves human-level quality on LJSpeech dataset, we do not mean that we can surpass or replace human, but the quality of NaturalSpeech is statistically indistinguishable from human recordings on this dataset. Meanwhile, although our evaluations are conducted on LJSpeech dataset, we believe the technologies in NaturalSpeech can be applied to other languages, speakers, and styles to improve the general synthesis quality. We will further try to achieve human-level quality in more challenging datasets or scenarios, such as expressive voices, longform audiobook voices, and singing voices that have more dynamic, diverse, and contextual prosody in our future work.

References

[1] Xu Tan, Tao Qin, Frank Soong, and Tie-Yan Liu. A survey on neural speech synthesis. *arXiv preprint arXiv:2106.15561*, 2021.

[2] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[3] Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, et al. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4779–4783. IEEE, 2018.

[4] Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, et al. Tacotron: Towards end-to-end speech synthesis. *Proc. Interspeech 2017*, pages 4006–4010, 2017.

[5] Sercan Ö Arık, Mike Chrzanowski, Adam Coates, Gregory Diamos, Andrew Gibiansky, Yong-guo Kang, Xian Li, John Miller, Andrew Ng, Jonathan Raiman, et al. Deep voice: Real-time neural text-to-speech. In *International Conference on Machine Learning*, pages 195–204. PMLR, 2017.

[6] Andrew Gibiansky, Sercan Ömer Arik, Gregory Frederick Diamos, John Miller, Kainan Peng, Wei Ping, Jonathan Raiman, and Yanqi Zhou. Deep voice 2: Multi-speaker neural text-to-speech. In *NIPS*, 2017.

[7] Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan O Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. Deep voice 3: 2000-speaker neural text-to-speech. *Proc. ICLR*, pages 214–217, 2018.

[8] Hideyuki Tachibana, Katsuya Uenoyama, and Shunsuke Aihara. Efficiently trainable text-to-speech system based on deep convolutional networks with guided attention. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4784–4788. IEEE, 2018.

[9] Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Ming Liu. Neural speech synthesis with transformer network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6706–6713, 2019.

[10] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech: Fast, robust and controllable text to speech. In *NeurIPS*, 2019.

[11] Yanqing Liu, Zhihang Xu, Gang Wang, Kuan Chen, Bohan Li, Xu Tan, Jinzhu Li, Lei He, and Sheng Zhao. Delightfultts: The microsoft speech synthesis system for blizzard challenge 2021. *arXiv preprint arXiv:2110.12612*, 2021.

[12] Ryan Prenger, Rafael Valle, and Bryan Catanzaro. Waveglow: A flow-based generative network for speech synthesis. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3617–3621. IEEE, 2019.

[13] Jaehyeon Kim, Sungwon Kim, Jungil Kong, and Sungroh Yoon. Glow-tts: A generative flow for text-to-speech via monotonic alignment search. *Advances in Neural Information Processing Systems*, 33, 2020.

[14] Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, and Mikhail Kudinov. Grad-tts: A diffusion probabilistic model for text-to-speech. In *International Conference on Machine Learning*, pages 8599–8608. PMLR, 2021.

[15] Jaehyeon Kim, Jungil Kong, and Juhee Son. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *International Conference on Machine Learning*, pages 5530–5540. PMLR, 2021.

[16] Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aaron Oord, Sander Dieleman, and Koray Kavukcuoglu. Efficient neural audio synthesis. In *International Conference on Machine Learning*, pages 2410–2419. PMLR, 2018.

[17] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in Neural Information Processing Systems*, 33, 2020.

[18] Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech 2: Fast and high-quality end-to-end text to speech. In *International Conference on Learning Representations*, 2021.

[19] Jeff Donahue, Sander Dieleman, Mikołaj Bińkowski, Erich Elsen, and Karen Simonyan. End-to-end adversarial text-to-speech. In *ICLR*, 2021.

[20] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6309–6318, 2017.

[21] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. *arXiv preprint arXiv:2102.12092*, 2021.

[22] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[23] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[24] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. *Advances in Neural Information Processing Systems*, 29:4743–4751, 2016.

[25] Diederik P Kingma and Prafulla Dhariwal. Glow: generative flow with invertible 1×1 convolutions. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 10236–10245, 2018.

[26] Keith Ito. The lj speech dataset. `https://keithito.com/LJ-Speech-Dataset/`, 2017.

[27] Antony W Rix, John G Beerends, Michael P Hollier, and Andries P Hekstra. Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs. In *2001 IEEE international conference on acoustics, speech, and signal processing. Proceedings (Cat. No. 01CH37221)*, volume 2, pages 749–752. IEEE, 2001.

[28] Cees H Taal, Richard C Hendriks, Richard Heusdens, and Jesper Jensen. An algorithm for intelligibility prediction of time–frequency weighted noisy speech. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(7):2125–2136, 2011.

[29] Jonathan Le Roux, Scott Wisdom, Hakan Erdogan, and John R Hershey. Sdr–half-baked or well done? In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 626–630. IEEE, 2019.

[30] Frank Wilcoxon. Individual comparisons by ranking methods. In *Breakthroughs in statistics*, pages 196–202. Springer, 1992.

[31] Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, et al. Cogview: Mastering text-to-image generation via transformers. *arXiv preprint arXiv:2105.13290*, 2021.

[32] Chenfei Wu, Jian Liang, Lei Ji, Fan Yang, Yuejian Fang, Daxin Jiang, and Nan Duan. Nüwa: Visual synthesis pre-training for neural visual world creation, 2021.

[33] Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. Videogpt: Video generation using vq-vae and transformers. *arXiv preprint arXiv:2104.10157*, 2021.

[34] Ruslan Rakhimov, Denis Volkhonskiy, Alexey Artemov, Denis Zorin, and Evgeny Burnaev. Latent video transformer. *arXiv preprint arXiv:2006.10704*, 2020.

[35] Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. In *Advances in neural information processing systems*, pages 14866–14876, 2019.

[36] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883, 2021.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[38] Yujia Xiao, Lei He, Huaiping Ming, and Frank K Soong. Improving prosody with linguistic and bert derived features in multi-speaker based mandarin chinese neural tts. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6704–6708. IEEE, 2020.

[39] Ye Jia, Heiga Zen, Jonathan Shen, Yu Zhang, and Yonghui Wu. Png bert: Augmented bert on phonemes and graphemes for neural tts. *Proc. Interspeech 2021*, pages 151–155, 2021.

[40] Guangyan Zhang, Kaitao Song, Xu Tan, Daxin Tan, Yuzi Yan, Yanqing Liu, Gang Wang, Wei Zhou, Tao Qin, Tan Lee, et al. Mixed-phoneme bert: Improving bert with mixed phoneme and sup-phoneme representations for text to speech. *arXiv preprint arXiv:2203.17190*, 2022.

[41] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[42] Isaac Elias, Heiga Zen, Jonathan Shen, Yu Zhang, Jia Ye, RJ Ryan, and Yonghui Wu. Parallel tacotron 2: A non-autoregressive neural tts model with differentiable duration modeling. *arXiv preprint arXiv:2103.14574*, 2021.

[43] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning*, pages 1530–1538. PMLR, 2015.

[44] Yuxuan Wang, Daisy Stanton, Yu Zhang, RJ-Skerry Ryan, Eric Battenberg, Joel Shor, Ying Xiao, Ye Jia, Fei Ren, and Rif A Saurous. Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis. In *International Conference on Machine Learning*, pages 5180–5189. PMLR, 2018.

[45] Mingjian Chen, Xu Tan, Bohan Li, Yanqing Liu, Tao Qin, sheng zhao, and Tie-Yan Liu. Adaspeech: Adaptive text to speech for custom voice. In *International Conference on Learning Representations*, 2021.

[46] Yihan Wu, Xu Tan, Bohan Li, Lei He, Sheng Zhao, Ruihua Song, Tao Qin, and Tie-Yan Liu. Adaspeech 4: Adaptive text to speech in zero-shot scenarios. *arXiv preprint arXiv:2204.00436*, 2022.

[47] Ron J Weiss, RJ Skerry-Ryan, Eric Battenberg, Soroosh Mariooryad, and Diederik P Kingma. Wave-tacotron: Spectrogram-free end-to-end text-to-speech synthesis. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5679–5683. IEEE, 2021.

[48] Machine Translation Group at UEDIN. The news-crawl dataset. `https://data.statmt.org/news-crawl/en/`, 2022.

[49] Hao Sun, Xu Tan, Jun-Wei Gan, Hongzhi Liu, Sheng Zhao, Tao Qin, and Tie-Yan Liu. Token-level ensemble distillation for grapheme-to-phoneme conversion. In *INTERSPEECH*, 2019.

[50] Mathieu Bernard and Hadrien Titeux. Phonemizer: Text to phones transcription for multiple languages in python. *Journal of Open Source Software*, 6(68):3958, 2021.

[51] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

[52] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

[53] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv:1710.05941*, 2017.

[54] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802, 2017.

[55] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[56] Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Interspeech*, volume 2017, pages 498–502, 2017.

A   Study of the Quality Gap of Previous TTS System

To understand where and how the quality gap to recordings comes from, we conduct a systematic study on the current TTS systems, which can help us to find the problems, and is equally important (if not more) than solving the problems. Specifically, we choose a state-of-the-art TTS system using FastSpeech 2 [18] as the acoustic model and HiFiGAN [17] as the vocoder, which consists of four components: phoneme encoder, variance adaptor, mel-spectrogram decoder, and vocoder. We design a series of comparison experiments to measure the quality gap (in terms of CMOS) of each component to its corresponding upper bound. We conduct analyses from this order (from the closest to waveform to the farest): vocoder, mel-spectrogram decoder, variance adaptor, and phoneme encoder.

Table 7: The CMOS of each component to its upper bound. Negative CMOS means this component setting is worse than its upper bound.

Component	Setting	Upper Bound	CMOS
Vocoder	GT Mel→Vocoder	Human Recordings	−0.04
Mel Decoder	GT Pitch/Duration→Mel Decoder	GT Mel	−0.15
Variance Adaptor	Predicted Pitch/Duration	GT Pitch/Duration	−0.14
Phoneme Encoder	Phoneme Encoder	Phoneme Encoder + Pre-training	−0.12

- *Vocoder.* We study the quality drop on the vocoder by comparing the two settings: 1) waveform generated by vocoder with ground-truth mel-spectrograms as input; 2) ground-truth waveform (human recordings). The CMOS is shown in Table 7. It can be seen that when taking ground-truth mel-spectrograms as input, the waveform generated by vocoder has some but not huge gap to human recordings. However, we need to pay attention to the training-inference mismatch in vocoder: in training, vocoder takes ground-truth mel-spectrograms as input, while in inference, it takes predicted mel-spectrograms as input.

- *Mel-spectrogram Decoder.* We study the quality drop on the mel-spectrogram decoder by comparing the two settings: 1) mel-spectrograms generated by mel-spectrogram decoder with ground-truth pitch and duration as input[5]; 2) ground-truth mel-spectrograms (extracted from human recordings). We use the vocoder to convert the mel-sepctrograms in the two settings into waveform for evaluation. As shown in Table 7, the predicted mel-spectrograms have 0.15 CMOS drop compared to the ground-truth mel-spectrograms.

- *Variance Adaptor.* We study the quality drop on the variance adaptor by comparing the predicted pitch/duration with the ground-truth pitch/duration. We need the mel-spectrogram decoder and vocoder to generate the waveform for evaluation in the two settings. As shown in Table 7, the predicted pitch/duration have 0.14 CMOS drop compared to the ground-truth pitch/duration.

---

[5]Ideally, we should also use ground-truth phoneme hidden sequence as input. However, ground-truth hidden sequence cannot be obtained. Thus, this comparison setting is just a approximation.

- *Phoneme Encoder.* Since it is not straightforward to construct the upper bound of the phoneme encoder, we analyze the approximate quality drop through backward verification, by improving phoneme encoder for better voice quality. We conduct large-scale phoneme pre-training on the phoneme encoder, and fine-tune it with the FastSpeech 2 training pipeline, and achieves a 0.12 CMOS gain, as shown in Table 7, which demonstrates the phoneme encoder has improvement space.

According to the above experimental studies, we analyze several reasons causing the quality drop in each component: 1) Training-inference mismatch. Ground-truth mel-spectrogram, pitch, and duration are used in training, while predicted values are used in inference, which causes mismatch in the input of vocoder and mel-spectrogram decoder. Fully end-to-end text to waveform optimization is helpful to eliminate this mismatch. 2) One-to-many mapping problem. Text to speech mapping is one-to-many, where a text sequence can correspond to multiple speech utterances with different variation information (e.g., pitch, duration, speed, pause, prosody, etc). Current systems usually use a variance adaptor to predict variance information (e.g., pitch, duration) to alleviate this problem, which is not enough to well handle this problem. We should rethink previous methods on variance information and come up with some thorough and elegant solutions. 3) Lack of representation capacity. Current models are not powerful enough to extract good representations from phoneme sequence and learn complicated data distribution in speech. More advanced methods such as large-scale pre-training and powerful generative models are critical to enhance the learning capacity.

B Differentiable Durator

To enable end-to-end duration optimization, we design a durator that can upsample a phoneme hidden sequence $\boldsymbol{H}_{n \times h}$ into a frame-level hidden sequence $\boldsymbol{O}_{m \times h}$ in a differentiable way, where h, n, m is the hidden dimension size, phoneme sequence length and frame sequence length, respectively. The differentiable durator consists of a duration predictor θ_{dp} for phoneme duration prediction and a learnable upsampling layer θ_{lu} for sequence expansion from phoneme level to frame level.

Duration Predictor   The input to the duration predictor θ_{dp} is phoneme hidden sequence $\boldsymbol{H}_{n \times h}$ and the output is the estimated phoneme duration $\hat{\boldsymbol{d}}_{n \times 1}$. The duration predictor θ_{dp} consists of 3 layers of one-dimensional convolution, with ReLU activation, layer normalization, and dropout between each layer.

Learnable Upsampling Layer   The learnable upsampling layer θ_{lu} takes phoneme duration $\boldsymbol{d}$ as input and upsamples phoneme hidden sequence $\boldsymbol{H}$ to frame-level sequence $\boldsymbol{O}$ [42]. First, we calculate the duration start and end matrices $\boldsymbol{S}_{m \times n}$ and $\boldsymbol{E}_{m \times n}$ by

$$\boldsymbol{S}_{i,j} = i - \sum_{k=1}^{j-1} d_k, \quad \boldsymbol{E}_{i,j} = \sum_{k=1}^{j} d_k - i, \tag{6}$$

where $\boldsymbol{S}_{i,j}$ indexes the (i, j)-th element in the matrix. We calculate the primary attention matrix $\boldsymbol{W}_{m \times n \times q}$ and auxiliary context matrix $\boldsymbol{C}_{m \times n \times p}$ following [42]:

$$\boldsymbol{W} = \mathrm{Softmax}(\underset{10 \to q}{\mathrm{MLP}}([\boldsymbol{S}, \boldsymbol{E}, \mathrm{Expand}(\mathrm{Conv1D}(\mathrm{Proj}(\boldsymbol{H})))])), \tag{7}$$

$$\boldsymbol{C} = \underset{10 \to p}{\mathrm{MLP}}([\boldsymbol{S}, \boldsymbol{E}, \mathrm{Expand}(\mathrm{Conv1D}(\mathrm{Proj}(\boldsymbol{H})))]), \tag{8}$$

where $\mathrm{Proj}(\cdot)$ represents one linear layer with input and output dimensions of h. $\mathrm{Conv1D}(\cdot)$ is one-dimensional convolution operation with layer normalization and Swish activation [53]. The input and output dimensions of $\mathrm{Conv1D}(\cdot)$ are h and 8. $\mathrm{Expand}(\cdot)$ means adding an extra dimension by repeating the input matrix by m times. $[\cdot]$ stands for matrix concatenation along the hidden dimension, and gets a hidden dimension of $10 = 1 + 1 + 8$. $\mathrm{MLP}(\cdot)$ is a two-layer full-connected network with Swish activations. The numbers underneath MLP denote the input and output hidden dimensions. We set $p = 2$ and $q = 4$. The $\mathrm{Softmax}(\cdot)$ operation is performed on the sequence time dimension. We calculate the frame-level hidden sequence output $\boldsymbol{O}_{m \times d}$ with the following equation:

$$\boldsymbol{O} = \underset{qh \to h}{\mathrm{Proj}}(\boldsymbol{W}\boldsymbol{H}) + \underset{qp \to h}{\mathrm{Proj}}(\mathrm{Einsum}(\boldsymbol{W}, \boldsymbol{C})), \tag{9}$$

where $\text{Einsum}(\cdot)$ represents the einsum operation ('qmn, mnp $\rightarrow$ qmp', $\boldsymbol{W}, \boldsymbol{C}$). We first permute $\boldsymbol{W}$ from $m \times n \times q$ to $q \times m \times n$ for computation, and after we get $\boldsymbol{WH}$ with shape $q \times m \times h$ and $\text{Einsum}(\boldsymbol{W}, \boldsymbol{C})$ with shape $q \times m \times p$, we reshape them to $m \times qh$ and $m \times qp$ respectively for final projection to dimension $m \times h$. Finally, we map $\boldsymbol{O}$ with a mean and variance linear layer to get the frame-level prior distribution parameter $\mu(y; \theta_{\text{pri}})$ and $\sigma(y; \theta_{\text{pri}})$, and get the prior distribution $p(z'|y; \theta_{\text{pri}}) = \mathcal{N}(z'; \mu(y; \theta_{\text{pri}}), \sigma(y; \theta_{\text{pri}}))$.

Compared to simply repeating each phoneme hidden sequence with the predicted duration in a hard way, the learnable upsampling layer enables more flexible duration adjustment for each phoneme. Also, the learnable upsampling layer makes the phoneme to frame expansion differentiable, and thus can be jointly optimized with other modules in the TTS system.

C   Alternative Formulation of Bidirectional Prior/Posterior

We provide another formulation of the backward loss $\mathcal{L}_{\text{bwd}}$ in Equation 1 and forward loss $\mathcal{L}_{\text{fwd}}$ in Equation 2 by directly using KL loss to match two distributions.

For the backward loss, we directly match the posterior $q(z|x; \phi)$ to the prior $p(z|y; \theta_{\text{pri}})$:

$$\mathcal{L}_{\text{bwd}}(\phi, \theta_{\text{bpp}}, \theta_{\text{pri}}) = KL[q(z|x; \phi)||p(z|y; \theta_{\text{pri}}))] = \mathbb{E}_{z \sim q(z|x; \phi)}(\log q(z|x; \phi) - \log p(z|y; \theta_{\text{pri}}))$$

$$= \mathbb{E}_{z \sim q(z|x; \phi)}(\log q(z|x; \phi) - \log p(f^{-1}(z; \theta_{\text{bpp}})|y; \theta_{\text{pri}}))| \det \frac{\partial f^{-1}(z; \theta_{\text{bpp}})}{\partial z}|), \tag{10}$$

where $f^{-1}(z; \theta_{\text{bpp}}) = z'$, and $p(z|y; \theta_{\text{pri}})) = p(f^{-1}(z; \theta_{\text{bpp}})|y; \theta_{\text{pri}}))| \det \frac{\partial f^{-1}(z; \theta_{\text{bpp}})}{\partial z}|)$ according to the change of variable rule.

For the forward loss, we directly match the prior $p(z'|y; \theta_{\text{pri}})$ to the posterior $q(z'|x; \phi)$:

$$\mathcal{L}_{\text{fwd}}(\phi, \theta_{\text{bpp}}, \theta_{\text{pri}}) = KL[p(z'|y; \theta_{\text{pri}})||q(z'|x; \phi)] = \mathbb{E}_{z' \sim p(z'|y; \theta_{\text{pri}})}(\log p(z'|y; \theta_{\text{pri}}) - \log q(z'|x; \phi))$$

$$= \mathbb{E}_{z' \sim p(z'|y; \theta_{\text{pri}})}(\log p(z'|y; \theta_{\text{pri}}) - \log q(f(z'; \theta_{\text{bpp}})|x; \phi)| \det \frac{\partial f(z'; \theta_{\text{bpp}})}{\partial z'}|), \tag{11}$$

where $f(z'; \theta_{\text{bpp}}) = z$, and $q(z'|x; \phi)) = q(f(z'; \theta_{\text{bpp}})|x; \phi)| \det \frac{\partial f(z'; \theta_{\text{bpp}})}{\partial z'}|)$ according to the change of variable rule.

D   Soft Dynamic Time Warping in KL loss

Since the frame-level prior distribution $p(z'|y; \theta_{\text{pri}})$ usually has different lengths from the ground-truth speech frames, the standard KL loss cannot be applied. Therefore, we use a soft dynamic time warping (Soft-DTW) of KL loss for $\mathcal{L}_{\text{bwd}}$ and $\mathcal{L}_{\text{fwd}}$ to circumvent this mismatch.

The Soft-DTW version of the KL loss for $\mathcal{L}_{\text{bwd}}$ can be obtained by recursive calculation:

$$r_{i,j} = \min^\gamma \begin{cases} r_{i-1,j} + KL[q(z'_{i-1}|x; \phi, \theta_{\text{bpp}})||p(z'_j|y; \theta_{\text{pri}})] + \text{warp} \\ r_{i,j-1} + KL[q(z'_i|x; \phi, \theta_{\text{bpp}})||p(z'_{j-1}|y; \theta_{\text{pri}})] + \text{warp} \\ r_{i-1,j-1} + KL[q(z'_{i-1}|x; \phi, \theta_{\text{bpp}})||p(z'_{j-1}|y; \theta_{\text{pri}})] \end{cases}, \tag{12}$$

where $r_{i,j}$ is the KL divergence loss between the simplified posterior $q(z'|x; \phi, \theta_{\text{bpp}})$ from frame 1 to frame i and the prior $p(z'|y; \theta_{\text{pri}})$ from frame 1 to frame j with the best alignment. $KL[q(z'_*|x; \phi, \theta_{\text{bpp}})||p(z'_*|y; \theta_{\text{pri}})]$ is defined in Equation 1. $\min^\gamma$ a soft-min operator, which is defined as $\min^\gamma(a_1, ..., a_n) = -\gamma \log \Sigma_i e^{-\frac{a_i}{\gamma}}$ and $\gamma = 0.01$. warp is a warp penalty for not choosing the diagonal path and is set as 0.07. $q(z'_i|x; \phi, \theta_{\text{bpp}})$ is the i-th frame of the simplified posterior, and $p(z'_j|y; \theta_{\text{pri}})$ is the j-th frame of the prior.

The Soft-DTW version of KL loss for $\mathcal{L}_{\text{fwd}}$ is similar to that of $\mathcal{L}_{\text{bwd}}$, which can be defined as:

$$r_{i,j} = \min^\gamma \begin{cases} r_{i-1,j} + KL[p(z_{i-1}|y; \theta_{\text{pri}}, \theta_{\text{bpp}})||q(z_j|x; \phi)] + \text{warp} \\ r_{i,j-1} + KL[p(z_i|y; \theta_{\text{pri}}, \theta_{\text{bpp}})||q(z_{j-1}|x; \phi)] + \text{warp} \\ r_{i-1,j-1} + KL[p(z_{i-1}|y; \theta_{\text{pri}}, \theta_{\text{bpp}})||q(z_{j-1}|x; \phi)] \end{cases}, \tag{13}$$

where $r_{i,j}$ is the KL divergence loss between the enhanced prior $p(z|y; \theta_{\text{pri}}, \theta_{\text{bpp}})$ from frame 1 to frame i and the posterior $q(z|x; \phi)$ from frame 1 to frame j with the best alignment. $KL[p(z_*|y; \theta_{\text{pri}}, \theta_{\text{bpp}})||q(z_*|x; \phi)]$ is defined in Equation 2. $p(z_i|y; \theta_{\text{pri}}, \theta_{\text{bpp}})$ is the i-th frame of the enhanced prior, and $q(z_j|x; \phi)$ is the j-th frame of the posterior.

E Waveform Decoder Loss

Instead of using negative log-likelihood loss in waveform reconstruction and prediction in Equation 3 and 4, we use GAN loss, feature mapping loss, and mel-spectrogram loss as used in [17].

GAN Loss   The GAN loss follows LS-GAN [54], which is defined as follows. The generator is trained to minimize the loss function while the discriminator is train to maximize it:

$$\mathbb{E}_x[(D(x) - 1)^2] + \mathbb{E}_z[D(G(z))^2] \tag{14}$$

where is x the ground-truth waveform and z is the input of waveform decoder. We follow [15] for the design of discriminators.

Feature Mapping Loss   The feature mapping loss consists of the L1 distance between real samples and fake samples in terms of the intermediate feature in each layer of the discriminator, which can be formulated as:

$$\mathbb{E}_{(x,z)}[\sum_l \frac{1}{N_l}||D^l(x) - D^l(G(z))||_1] \tag{15}$$

where l is the layer index in discriminator, $D^l(\cdot)$ and N_l are the features and the number of features in the l-th layer of the discriminator, respectively.

Mel-Spectrogram Loss   The mel-spectrogram loss is L1 distance between the mel-spectrogram of ground-truth waveform and that of generated waveform, which can be defined as:

$$\mathbb{E}_{(x,z)} = ||S(x) - S(G(z))||_1 \tag{16}$$

where $S(\cdot)$ is the function that converts the waveform into corresponding mel-spectrogram.

F Training Details of NaturalSpeech

Phoneme Pre-training   We pre-train our phoneme encoder on 200M phoneme sequences, which is converted from text with grapheme-to-phoneme conversion. The size of the phoneme dictionary is 182. We learn the sup-phoneme using Byte-Pair Encoding (BPE) [55] with a sup-phoneme dictionary size of $30,088$. We conduct the pre-training on 8 NVIDIA A100 GPUs with 80G memory (we only use A100 for phoneme pre-training, and use V100 for the remaining training of NaturalSpeech), with a total batch size of $1,024$ sentences for 120k training steps. The mask ratio for sup-phoneme is 15%.

Duration Predictor   In the warmup stage (the first 1k epochs), we obtain the duration label to train the duration predictor to speed up the convergence of differentiable durator. We can choose any tools to provide duration label, such as Montreal forced alignment [56]. Here we choose monotonic alignment search (MAS) [13], which estimates the optimal alignment between the phoneme prior distribution $p(z'|y; \theta_{\text{pho}}) = \mathcal{N}(z'; \mu(y; \theta_{\text{pho}}), \sigma(y; \theta_{\text{pho}}))$ and simplified frame-level posterior $q(z'|x; \phi, \theta_{\text{bpp}})$, where $\mu(y; \theta_{\text{pho}}), \sigma(y; \theta_{\text{pri}})$ are the mean and variance parameters obtained from the phoneme hidden sequence by two linear layers. The monotonic and non-skipping constraints of MAS provide the inductive bias that human read words in orders without skipping. The optimal alignment search result A can be formulated as

$$A = \arg\max_A \Sigma_{i=1}^m \mathcal{N}(z_i'; \mu(y; \theta_{\text{pho}})_{A(i)}, \sigma(y; \theta_{\text{pho}})_{A(i)}), \tag{17}$$

where $A(i)$ denotes the aligned phoneme index of the i-th frame z_i' from $q(z'|x; \phi, \theta_{\text{bpp}})$. We search the alignment result using dynamic programming. Let $Q_{i,j}$ denote the probability of z_i' belongs to the prior distribution of the j-th phoneme, then we can formulate $Q_{i,j}$ recursively with $Q_{i-1,j-1}$ and $Q_{i,j-1}$ with the following equation:

$$Q_{i,j} = \max(Q_{i-1,j-1}, Q_{i-1,j}) + \log \mathcal{N}(z'_i; \mu(y; \theta_{\text{pho}})_j, \sigma(y; \theta_{\text{pho}})_j). \tag{18}$$

We calculate all the $Q_{i,j}$ from $i = 0, j = 0$ to $i = m, j = n$. Since the best alignment path is determined by the highest Q value, we utilize all the cached Q value to backtrack from $Q_{m,n}$ to $Q_{0,0}$ for the most probable alignment A.

Note that in the warmup training stage, the duration $\boldsymbol{d}$ comes from MAS. After the warmup stage, the input duration comes from the duration predictor $\hat{\boldsymbol{d}}$. During the whole training process, we apply gradient stop operation on the input of duration predictor.

Bidirectional Prior/Posterior For the two loss terms $\mathcal{L}_{\text{bwd}}$ and $\mathcal{L}_{\text{fwd}}$ in bidirectional prior/posterior module, we only use $\mathcal{L}_{\text{bwd}}$ during the warmup stage to learn a reasonable prior distribution, and then add $\mathcal{L}_{\text{fwd}}$ to the loss function for bidirectional optimization after the warmup stage.

VAE with Memory In the warmup stage, we do not use the memory bank in VAE training, i.e., $z \sim q(z|x; \phi)$ is directly taken as the input of the waveform decoder. After the warmup stage, we initialize the memory banks M as follows: we first get the posterior $z \sim q(z|x; \phi)$ of each frame of the utterances in the training set, and then conduct K-means clustering on these z to get 1K clusters, and use the cluster center to initialize the memory bank M. After the initialization, we jointly train the memory mechanism with the whole TTS system.

In the tuning stage (the last 2k epochs), we only use $\mathcal{L}_{\text{e2e}}$ to tune the model. We freeze the parameters of posterior encoder, waveform decoder, phoneme encoder, and bidirectional prior/posterior, and only update the durator for fully end-to-end duration optimization.

G Hyper-Parameters of NaturalSpeech

The hyper-parameters of NaturalSpeech are listed in Table 8.

The number of model parameters for θ_{pho}, θ_{dur}, θ_{bpp}, and θ_{dec} is 28.7M, for the posterior encoder ϕ is 7.2M, and for the discriminators is 46.7M. Note that only θ_{pho}, θ_{dur}, θ_{bpp}, and θ_{dec} with 28.7M model parameters are used in inference.

Table 8: Hyper-parameters of NaturalSpeech.

Module	Hyper-Parameter	Value
Phoneme Encoder θ_{pho}	Phoneme Encoder Embedding Dimension	192
	Phoneme Encoder Blocks	6
	Phoneme Encoder Multi-Head Attention Hidden Dimension	192
	Phoneme Encoder Multi-Head Attention Heads	2
	Phoneme Encoder Conv Kernel Size	3
	Phoneme Encoder Conv Filter Size	768
	Phoneme Encoder Dropout	0.1
Durator θ_{dur}	Duration Predictor Kernel Size	3
	Duration Predictor Filter Size	192
	Duration Predictor Dropout	0.5
	Upsampling Layer Kernel Size	3
	Upsampling Layer Filter Size	8
Prior/Posterior θ_{bpp}	Flow Model Affine Coupling Layers	4
	Flow Model Affine Coupling Dilation	1
	Flow Model Affine Coupling Kernel Size	5
	Flow Model Affine Coupling Filter Size	192
	Flow Model Affine Coupling WaveNet Layers	4
Waveform Decoder θ_{dec}	Waveform Decoder ConvBlocks	4
	Waveform Decoder ConvBlock Hidden	$[256, 128, 64, 32]$
	Waveform Decoder ConvBlock Upsampling Ratio	$[8, 8, 2, 2]$
	Waveform Decoder ConvLayers	3
	Waveform Decoder ConvLayer Kernel Size	$[3, 7, 11]$
	Waveform Decoder Conv Dilation	$[1, 3, 5]$
	Memory Banks Size	1000
	Memory Banks Hidden Dimension	192
	Memory Banks Attention Heads	2
Posterior Encoder ϕ	Posterior Encoder WaveNet Layers	16
	Posterior Encoder Dilation	1
	Posterior Encoder Conv Kernel Size	5
	Posterior Encoder Conv Filter Size	192
Discriminator D	Multi-Period Discriminator Periods	$[1, 2, 3, 5, 7, 11]$