small 1

StyleTTS: A Style-Based Generative Model for Natural and Diverse Text-to-Speech Synthesis

Yinghao Aaron Li, Cong Han, and Nima Mesgarani

***Abstract*—Text-to-Speech (TTS) has recently seen great progress in synthesizing high-quality speech owing to the rapid development of parallel TTS systems. Yet producing speech with naturalistic prosodic variations, speaking styles, and emotional tones remains challenging. In addition, many existing parallel TTS models often struggle with identifying optimal monotonic alignments since speech and duration generation typically occur independently. Here, we propose StyleTTS, a style-based generative model for parallel TTS that can synthesize diverse speech with natural prosody from a reference speech utterance. Using our novel Transferable Monotonic Aligner (TMA) and duration-invariant data augmentation, StyleTTS significantly outperforms other baseline models on both single and multi-speaker datasets in subjective tests of speech naturalness and synthesized speaker similarity. Through self-supervised learning, StyleTTS can generate speech with the same emotional and prosodic tone as the reference speech without needing explicit labels for these categories. In addition, when trained with a large number of speakers, our model can perform zero-shot speaker adaption. The source code and audio samples can be found on our demo page at https://styletts.github.io/.**

I. Introduction

Text-to-speech (TTS), also known as speech synthesis, has made significant strides with the advent of deep learning, producing increasingly human-like synthetic speech [1], [2], [3]. However, synthesizing expressive speech that captures the full range of prosodic, temporal, and spectral features, also known as paralinguistic information, remains a challenge [4]. A single piece of text can be spoken in various ways, influenced by context, emotional tone, and a speaker's unique linguistic habits. Hence, TTS is fundamentally a one-to-many mapping problem requiring innovative approaches.

While several strategies have been proposed to address this, including methods based on variational inference [1], [5], [6], [7], flow-based modeling [1], [8], [9], controlling pitch, duration and energy [10], [11], and using external prosody encoder [12], [13], [14], the production of synthesized speech still falls short of real human speech. In particular, accurately modeling and incorporating different speakers' speaking styles and emotional tones poses a significant challenge.

Many attempts have been made to integrate style information into TTS models [12], [13], [15], [16]. These approaches are predominantly based on autoregressive models such as Tacotron. Non-autoregressive parallel TTS models, such as Fastspeech [17] and Glow-TTS [9], however, have several advantages over autoregressive models. These models generate speech in parallel, enabling fast speech synthesis, and they are also more robust to longer and out-of-distribution (OOD) utterances. Moreover, because phoneme duration, pitch, and energy are predicted independently from speech, models such as FastSpeech2 [11] and FastPitch [18] allow fully controllable speech synthesis. At the same time, these models have limitations. They predominantly concentrate on speech synthesis from a single target speaker and often achieve multi-speaker extensions by concatenating speaker embeddings with the encoder output. Models that explore speech styles also incorporate styles by concatenating style vectors and phoneme embeddings as input to the decoder. [12], [13], [15], [16]. This approach may not capture the temporal variation of acoustic features in the target speech effectively. In contrast, the domain of style transfer introduces styles through conditional normalization methods like adaptive instance normalization (AdaIN) [19]. This technique has proven effective in neural style transfer [20], [21], [22], generative modeling [23], [24], [25], and neural image editing [26], [27]. Application of these methods in speech synthesis has not been extensively explored yet, restricted primarily to voice conversion and speaker adaptation [28], [29], [30], [31], [32].

The structure of parallel TTS models allows for the entire speech to be synthesized, presenting an opportunity to leverage the powerful AdaIN module for integrating generalized styles in diverse speech synthesis. Recent state-of-the-art models mostly employ the non-autoregressive parallel framework for TTS, but because they do not directly align the input text and speech like autoregressive models do, an external aligner such as the Montreal Forced Aligner [33] that is pre-trained on a large dataset is often required. Since the external aligner is not trained on the TTS data and objectives, the alignments are not optimally suited for the TTS task. Although training internal aligners alleviates some generalization problems [1], [9], [34], [35], overfitting can occur as the aligners are trained on a smaller TTS dataset with only a mel-reconstruction loss.

Here, we introduce StyleTTS, a model that addresses the aforementioned challenges of incorporating diverse speaking styles and learning a reliable monotonic aligner. StyleTTS incorporates style-based generative modeling into a parallel TTS framework to enable natural and expressive speech synthesis. It leverages AdaIN to integrate style vectors derived from reference audio, capturing the full spectrum of a speaker's stylistic features. This allows our model to synthesize speech that emulates the prosodic patterns and emotional tones in the reference audio. With various reference audios, we can synthesize the same text in different speaking styles, effectively realizing the one-to-many mapping that many TTS systems find challenging. In addition, our model employs a novel Transferable Monotonic Aligner (TMA) to find the optimal text alignment, aided by a novel duration-invariant data augmentation scheme to produce naturalistic prosody robust to potentially suboptimal duration predictions. Our model's design is robust against the generalization problems of external aligners and

overfitting problems that can be caused by internal aligners.

This paper presents the following novel contributions: (i) the introduction of the Transferable Monotonic Aligner (TMA), a new transfer learning scheme that refines pre-trained text aligners for TTS tasks, (ii) a duration-invariant data augmentation method for improving prosody prediction, and (iii) a parallel TTS model that incorporates generalized speech styles for natural and expressive speech synthesis. Together, these contributions pave the way for advanced TTS technologies enhancing human-computer interactions.

II. Methods

A. Proposed Framework

Given $\boldsymbol{t} \in \mathcal{T}$ the input phonemes and $\boldsymbol{x} \in \mathcal{X}$ an arbitrary reference mel-spectrogram, our goal is to train a system that generates the mel-spectrogram $\tilde{\boldsymbol{x}} \in \mathcal{X}$ that corresponds to the speech of $\boldsymbol{t}$ and reflects the generalized speech styles of $\boldsymbol{x}$. Generalized speech styles are defined as any characteristics in the reference audio $\boldsymbol{x}$ except the phonetic content [16], including but not limited to prosodic pattern, lexical stress, formants transition, speaking rate, and speaker identity. Our framework consists of eight modules that can be divided into three major categories: (i) speech generation modules that include the text encoder, style encoder, and decoder, (ii) TTS prediction modules that include the duration and prosody predictor, and (iii) utility modules only used during training that include the discriminator, text aligner, and pitch extractor. An overview of our framework is provided in Figure 1. We detail each of the modules below.

Text Encoder. The text encoder T transforms the phonemes $\boldsymbol{t}$ into a hidden representation $\boldsymbol{h}_{\text{text}} = T(\boldsymbol{t})$. It consists of a 3-layer CNN followed by a bidirectional LSTM [36].

Text Aligner. The text aligner A generates an alignment $\boldsymbol{d}_{\text{align}}$ between mel-spectrograms and phonemes. We train a text aligner A alongside the decoder G during the reconstruction phase. Modeled after the decoder of Tacorton 2 with attention, A is initially trained on an automatic speech recognition (ASR) task using the LibriSpeech corpus [37] and then fine-tuned concurrently with our decoder. We call an aligner with this setup (pre-trained on large corpora and fine-tuned for TTS tasks) a **Transferable Monotonic Aligner** (TMA).

Style Encoder. Given an input mel-spectrogram $\boldsymbol{x}$, our encoder derives a style vector $s = E(\boldsymbol{x})$. With various reference audios, E can generate diverse style representations, allowing the decoder G to create speech that mirrors the style s of a reference audio $\boldsymbol{x}$. E consists of four residual blocks [38] followed by an averaging pooling layer along the time axis.

Pitch Extractor. As in FastPitch [11], we extract pitch F0 directly in Hertz without further processing, providing a more straightforward representation and allowing enhanced control of speech pitch. Instead of using the acoustic periodicity detection method [39] employed in FastPitch to estimate the ground truth pitch, we train a pitch extractor F end-to-end with our decoder G for a more accurate estimation. Our pitch extractor F is a JDC network [40], pre-trained on LibriSpeech with ground truth F0 estimated using YIN [41]. This extractor is fine-tuned with the decoder to predict pitch $p_{\boldsymbol{x}} = F(\boldsymbol{x})$ for the reconstruction of $\boldsymbol{x}$.

Decoder. Our decoder G is trained to reconstruct the input mel-spectrogram $\boldsymbol{x}$, represented by $\hat{\boldsymbol{x}} = G\left(\boldsymbol{h}_{\text{text}} \cdot \boldsymbol{d}_{\text{align}}, \boldsymbol{s}, p_{\boldsymbol{x}}, n_{\boldsymbol{x}}\right)$. Here, $\boldsymbol{h}_{\text{text}} \cdot \boldsymbol{d}_{\text{align}}$ is aligned hidden representation of phonemes, $\boldsymbol{s} = E(\boldsymbol{s})$ is the style vector of $\boldsymbol{x}$, $p_{\boldsymbol{x}}$ is pitch contour of $\boldsymbol{x}$, and $n_{\boldsymbol{x}}$ is energy (represented by the log norm) of $\boldsymbol{x}$ per frame. The decoder is comprised of seven residual blocks with AdaIN [19], defined as follows:

$$\text{AdaIN}(c, s) = L_\sigma(s)\frac{c - \mu(c)}{\sigma(c)} + L_\mu(s), \tag{1}$$

where c is a single channel of the feature maps, s is the style vector, $\mu(\cdot)$ and $\sigma(\cdot)$ denote the channel mean and standard deviation and L_σ and L_μ are learned linear projections for computing the adaptive gain and bias using the style vector s. The use of AdaIN is one of the major differences between our model and other TTS models with style encoders such as [13] and [16]. The advantages of AdaIN for diverse speech synthesis are further discussed in Appendix A-B.

To prevent dilution of import features, we concatenate the pitch $p_{\boldsymbol{x}}$, energy $n_{\boldsymbol{x}}$, and residual phoneme features $\boldsymbol{h}_{\text{res}}$ and deliver them to subsequent residual blocks after AdaIN. This process is further detailed in Table , and its effectiveness is discussed in Section IV-D.

Discriminator. We include a discriminator D to facilitate training of our decoder for better sound quality [1]. The discriminator shares the same architecture as our style encoder.

Duration Predictor. Our duration predictor consists of a 3-layer bidirectional LSTM R with adaptive layer normalization (AdaLN) module followed by a linear projection L, where instance normalization is replaced by layer normalization in equation 1. We use AdaLN because R takes discrete tokens similar to those in NLP applications, where layer normalization [42] is preferred. R is shared with the prosody predictor P through $\boldsymbol{h}_{\text{prosody}} = R\left(\boldsymbol{h}_{\text{text}}\right)$ as input to P.

Prosody Predictor. Our prosody predictor P predicts both the pitch $\hat{p}_{\boldsymbol{x}}$ and energy $\hat{n}_{\boldsymbol{x}}$ with given text and style vector. The aligned shared representation $\boldsymbol{h}_{\text{prosody}} \cdot \boldsymbol{a}$ is processed through a shared bidirectional LSTM layer to generate $\boldsymbol{h}_{\text{prosody}}$, which is then fed into two sets of three residual blocks with AdaIN and a linear projection layer, one for the pitch output and another for the energy output (see Appendix D for details).

B. Training Objectives

Our model training process is divided into two stages to allow the integration of duration-invariant prosody data augmentation, a key contribution of our work. During the first stage, the model learns to reconstruct the mel-spectrogram from the text, pitch, energy, and style. The second stage fixes all modules except the duration and prosody predictors, which are trained to predict the duration, pitch, and energy from the given text.

1) First Stage Objectives:

Mel reconstruction. Given a mel-spectrogram $\boldsymbol{x} \in \mathcal{X}$ and its corresponding text $\boldsymbol{t} \in \mathcal{T}$, we train our decoder under the L_1 reconstruction loss

$$\mathcal{L}_{\text{mel}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{t}}\left[\left\|\boldsymbol{x} - G\left(\boldsymbol{h}_{\text{text}} \cdot \boldsymbol{a}_{\text{align}}, \boldsymbol{s}, p_{\boldsymbol{x}}, n_{\boldsymbol{x}}\right)\right\|_1\right] \tag{2}$$

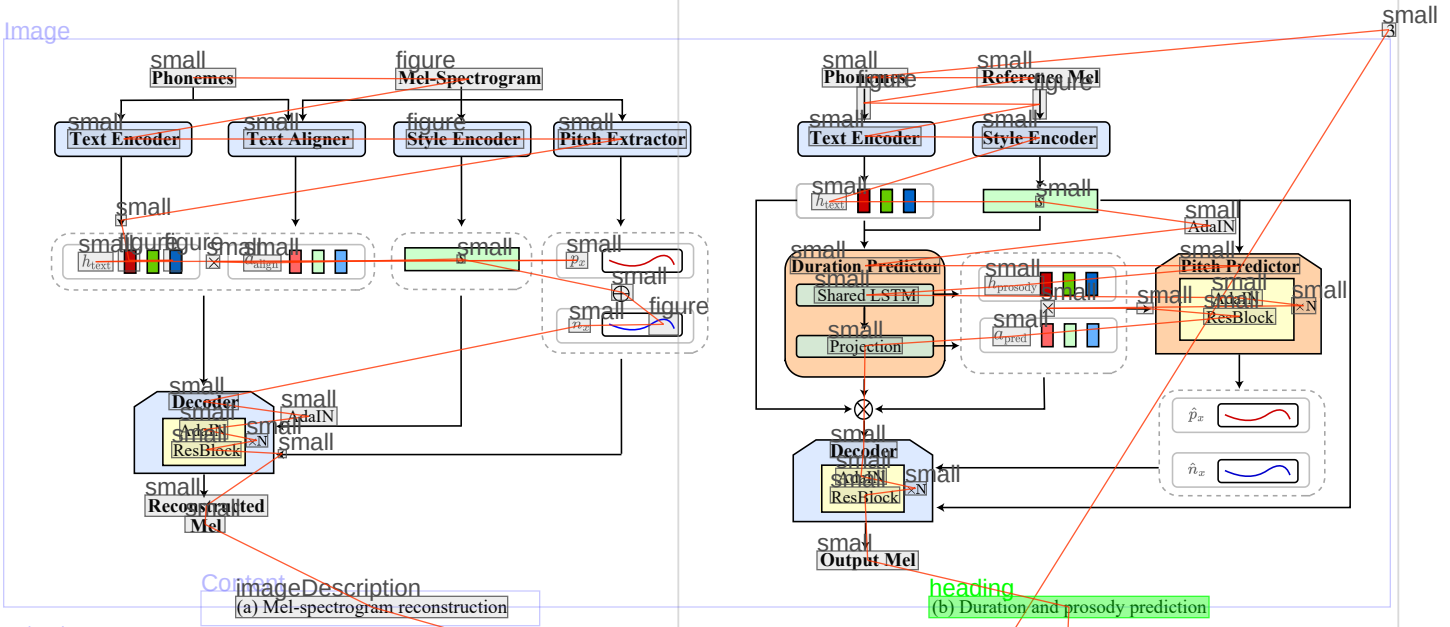


Fig. 1. Training and inference schemes of StyleTTS. (a) Stage 1 of our training procedures where the decoder is trained to reconstruct input mel-spectrogram using pitch, energy, phonemes, alignment, and style vectors. (b) Stage 2 of training and inference procedures where pitch, energy, and alignment are predicted based on input text, and a style vector is extracted from a reference mel-spectrogram for synthesis. Parameters of modules in blue are fixed during this stage of training while those in orange are tuned.

where $\boldsymbol{h}_{\text{text}} = T(\boldsymbol{t})$ is the encoded phoneme representation, $\boldsymbol{a}_{\text{align}} = A(\boldsymbol{x}, \boldsymbol{t})$ is the attention alignment from the text aligner, $s = E(\boldsymbol{x})$ is the style vector of $\boldsymbol{x}$, $p_{\boldsymbol{x}} = F(\boldsymbol{x})$ is the pitch F0 of $\boldsymbol{x}$ and $n_{\boldsymbol{x}}$ is the energy of $\boldsymbol{x}$. For end-to-end (E2E) training with the decoder and the text aligner, we apply a novel 50%-50% strategy: half the time, we use the raw attention output as the alignment, which allows gradient backpropagation through the text aligner; for the other half, we use the non-differentiable monotonic version of alignment through dynamic programming algorithms [9] to train the decoder for generating intelligible speech from monotonic hard alignment during inference. This innovative approach effectively fine-tunes the pre-trained text aligner to produce the optimal alignments for speech reconstruction, thus enhancing the sample quality of generated speech. The effectiveness of this strategy is analyzed in section IV-D.

TMA objectives. We fine-tune our text aligner with the original sequence-to-sequence ASR objectives $\mathcal{L}_{\text{s2s}}$ to ensure that correct attention alignment is kept during the E2E training:

$$\mathcal{L}_{\text{s2s}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{t}}\left[\sum_{i=1}^{N} \mathbf{CE}(\boldsymbol{t}_i, \hat{\boldsymbol{t}}_i)\right], \tag{3}$$

where N is the number of phonemes in $\boldsymbol{t}$, $\boldsymbol{t}_i$ is the i-th phoneme token of $\boldsymbol{t}$, $\hat{\boldsymbol{t}}_i$ is the i-th predicted phoneme token, and $\mathbf{CE}(\cdot)$ is the cross-entropy loss function.

Since this alignment is not necessarily monotonic, we use a simple L-1 loss $\mathcal{L}_{\text{mono}}$ that forces the soft attention alignment to be close to its non-differentiable monotonic version:

$$\mathcal{L}_{\text{mono}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{t}}\left[\left\|\boldsymbol{a}_{\text{align}} - \boldsymbol{a}_{\text{hard}}\right\|_1\right], \tag{4}$$

where $\boldsymbol{a}_{\text{align}} = A(\boldsymbol{x}, \boldsymbol{t})$ is the attention alignment and $\boldsymbol{a}_{\text{hard}}$ is the monotonic hard alignment obtained through dynamic programming algorithms (see Appendix A-A for details).

Adversarial objectives. We employ two adversarial loss functions, the original cross-entropy loss function $\mathcal{L}_{\text{adv}}$ for adversarial training and the additional feature-matching loss [43] $\mathcal{L}_{\text{fm}}$, to improve the sound quality of the reconstructed mel-spectrogram:

$$\mathcal{L}_{\text{adv}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{t}}\left[\log D(\boldsymbol{x}) + \log\left(1 - D(\hat{\boldsymbol{x}})\right)\right], \tag{5}$$

$$\mathcal{L}_{\text{fm}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{t}}\left[\sum_{l=1}^{T} \frac{1}{N_l}\left\|D^l(\boldsymbol{x}) - D^l(\hat{\boldsymbol{x}})\right\|_1\right], \tag{6}$$

where $\hat{\boldsymbol{x}}$ is the reconstructed mel-spectrogram by G, T is the total number of layers in D and D^l denotes the output feature map of l-th layer with N_l number of features. The feature matching loss can be seen as a reconstruction loss of hidden layer features of real and generated speech as judged by the discriminator.

First stage full objectives. Our full objective functions in the first stage can be summarized as follows with hyperparameters λ_{s2s}, λ_{mono}, λ_{adv} and λ_{fm}:

$$\min_{G,A,E,F,T} \max_{D} \mathcal{L}_{\text{mel}} + \lambda_{\text{s2s}}\mathcal{L}_{\text{s2s}} + \lambda_{\text{mono}}\mathcal{L}_{\text{mono}} + \lambda_{\text{adv}}\mathcal{L}_{\text{adv}} + \lambda_{\text{fm}}\mathcal{L}_{\text{fm}} \tag{7}$$

2) Second Stage Objectives:

Duration prediction. We employ the L-1 loss to train our duration predictor

$$\mathcal{L}_{\text{dur}} = \mathbb{E}_{\boldsymbol{d}}\left[\left\|\boldsymbol{d} - \boldsymbol{d}_{\text{pred}}\right\|_1\right] \tag{8}$$

where $\boldsymbol{d}$ is the ground truth duration obtained by summing $\boldsymbol{a}_{\text{align}}$ along the mel frame axis. $\boldsymbol{d}_{\text{pred}} = L(R(\boldsymbol{h}_{\text{text}}, \boldsymbol{s}))$ is the predicted duration under the style vector $\boldsymbol{s}$.

Prosody prediction. We train our prosody predictor via a unique data augmentation scheme. Since the duration predictor is trained separately from other modules (using only $\mathcal{L}_{\text{dur}}$), the duration predictions it produces may not always be optimal or compatible with the prosody predictor. To make the prosody predictor more robust to these potentially suboptimal duration predictions, we augment the data to introduce prosody invariance over the duration.

More specifically, instead of using the ground truth alignment, pitch, and energy of the original mel-spectrogram, we first apply a 1-D bilinear interpolation to stretch or compress the mel-spectrogram in time to obtain the augmented sample $\tilde{\boldsymbol{x}}$. As a result, the speech speed changes, yet the pitch and energy curves remain consistent. Accordingly, the prosody predictor learns to maintain pitch and energy prediction invariance, regardless of the duration of speech. This approach helps mitigate issues with unnatural prosody when the predicted duration is incorrect.

We use $\mathcal{L}_{f_0}$ and $\mathcal{L}_n$, which are F0 and energy reconstruction loss, respectively:

$$\mathcal{L}_{\text{f0}} = \mathbb{E}_{p_{\tilde{\boldsymbol{x}}}}\left[\left\|p_{\tilde{\boldsymbol{x}}} - P_p\left(S\left(\boldsymbol{h}_{\text{text}}, s\right) \cdot \tilde{\boldsymbol{a}}_{\text{align}}\right)\right\|_1\right] \tag{9}$$

$$\mathcal{L}_n = \mathbb{E}_{\tilde{\boldsymbol{x}}}\left[\left\|n_{\tilde{\boldsymbol{x}}} - P_n\left(S\left(\boldsymbol{h}_{\text{text}}, s\right) \cdot \tilde{\boldsymbol{a}}_{\text{align}}\right)\right\|_1\right] \tag{10}$$

where $p_{\tilde{\boldsymbol{x}}}$, $n_{\tilde{\boldsymbol{x}}}$ and $\tilde{\boldsymbol{a}}_{\text{align}}$ are the pitch, energy and alignment of $\tilde{\boldsymbol{x}} \in \tilde{\mathcal{X}}$ the augmented dataset. P_p denotes the pitch output from the prosody predictor, and P_n denotes the energy output.

Decoder reconstruction. Lastly, we aim to ensure that the predicted pitch and energy can be effectively used by the decoder. Given that the mel-spectrogram is stretched or compressed during data augmentation, using them as the ground truth may lead to unwanted artifacts in the predicted prosody. Instead, we train the prosody predictor to produce pitch and energy predictions that can effectively reconstruct the decoder's outputs

$$\mathcal{L}_{\text{de}} = \mathbb{E}_{\tilde{\boldsymbol{x}}, \boldsymbol{t}}\left[\left\|\hat{\boldsymbol{x}} - G\left(\boldsymbol{h}_{\text{text}} \cdot \tilde{\boldsymbol{a}}_{\text{align}}, s, \hat{p}, \hat{n}\right)\right\|_1\right], \tag{11}$$

where $\hat{\boldsymbol{x}} = G\left(\boldsymbol{h}_{\text{text}} \cdot \tilde{\boldsymbol{a}}_{\text{align}}, s, \hat{p}, \|\tilde{\boldsymbol{x}}\|\right)$ is the reconstruction of $\tilde{\boldsymbol{x}} \in \tilde{\mathcal{X}}$, $\hat{p} = P_p\left(S\left(\boldsymbol{h}_{\text{text}}, s\right) \cdot \tilde{\boldsymbol{a}}_{\text{align}}\right)$ the predicted pitch and $\hat{n} = P_n\left(S\left(\boldsymbol{h}_{\text{text}}, s\right) \cdot \tilde{\boldsymbol{a}}_{\text{align}}\right)$ the predicted energy.

Second stage full objectives. Our full objective functions in the second stage can be summarized as follows with hyperparameters $\lambda_{\text{dur}}, \lambda_{\text{f0}}$, and λ_n:

$$\min_{S,L,P} \mathcal{L}_{\text{de}} + \lambda_{\text{dur}}\mathcal{L}_{\text{dur}} + \lambda_{\text{f0}}\mathcal{L}_{\text{f0}} + \lambda_n\mathcal{L}_n \tag{12}$$

III. Experiments

A. Datasets

We conducted experiments on three datasets. We trained a single-speaker model on the LJSpeech dataset [44]. The LJSpeech dataset consists of 13,100 short audio clips with a total duration of approximately 24 hours. We used the same split as VITS where the training set contains 12,500 samples, the validation set 100 samples and the test set 500 samples. We also trained a multi-speaker model on the LibriTTS dataset [45]. The LibriTTS train-clean-460 subset consists of approximately 245 hours of audio from 1,151 speakers. We removed utterances with a duration longer than 30 seconds and shorter than one second. We randomly split the train-clean-460 subset into a training (98%), a validation (1%), and a test (1%) set and use the test set for evaluation following [32]. We also used the VCTK [46] dataset to show that our model is capable of zero-shot speaker adaptation. We used the same training and test speaker split as in [47] for VCTK, where 88 speakers were used for training and the rest 20 were used for testing.

In addition, we trained a multi-speaker model on the emotional speech dataset (ESD) [48] to demonstrate the capacity to synthesize speech with diverse prosodic patterns. ESD consists of 10 Chinese and 10 English speakers reading the same 400 short sentences in five different emotions. We trained our model on 10 English speakers with all five emotions. We

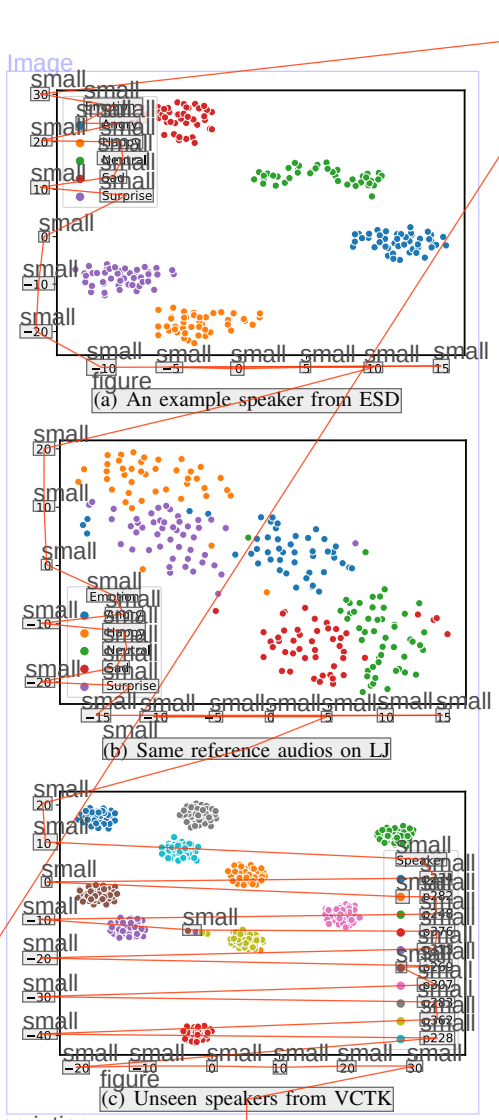


Fig. 2. t-SNE visualization of style vectors. All styles are learned without explicit emotion or speaker labels. (a) Style vectors of reference audios in five different emotions of the speaker 0017 in ESD, computed by the multi-speaker model trained on ESD. (b) Style vectors of the same reference audios as in Fig. 2a, computed by the single-speaker model trained on the LJSpeech dataset. (c) Style vectors from the model trained on the LibriTTS data of 10 unseen speakers in the VCTK dataset.

TABLE I. COMPARISON OF EVALUATED MOS WITH 95% CONFIDENCE INTERVALS (CI) ON THE LJ SPEECH DATASET.

Model	MOS-N (CI)
Ground Truth	4.32 ($\pm$ 0.04)
Tacotron 2 + HiFi-GAN	3.01 ($\pm$ 0.06)
FastSpeech 2 + HiFi-GAN	2.97 ($\pm$ 0.06)
VITS	3.78 ($\pm$ 0.06)
StyleTTS + HiFi-GAN	**4.01** ($\pm$ **0.05**)

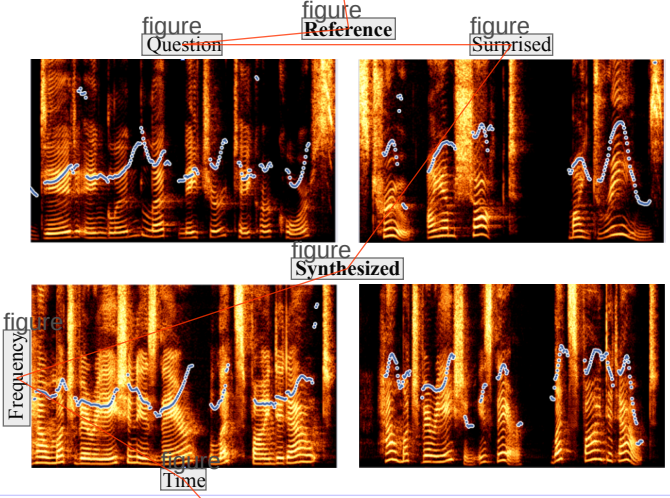


Fig. 3. Spectrograms of example reference audios and their corresponding generated speech reading “How much variation is there? Let’s find it out." from the single-speaker model trained on LJSpeech. The estimated pitch contour is shown as white dots. **Left top**: Reference audio of a question, “Did England let nature take her course?". Note the pitch is mostly going up at the end of each word. **Left bottom**: Synthesized speech. The same pattern of pitch rising at the end of the words is present. **Right top**: Reference audio of surprised speech saying “It’s true! I am shocked! My dreams!". Note the pitch goes up first and then down for each word. **Right bottom**: Synthesized speech with the same pattern of the pitch going up and down for most of the words.

upsampled training audios to 24 kHz to match the LibriTTS dataset. We converted text sequences into phoneme sequences using an open-source tool [1]. We extracted mel-spectrograms with a FFT size of 2048, hop size of 300, and window length of 1200 in 80 mel bins using TorchAudio [49].

B. Training

For both stages, we trained all models for 200 epochs using the AdamW optimizer [50] with $\beta_1 = 0, \beta_2 = 0.99$, weight decay $\lambda = 10^{-4}$, learning rate $\alpha = 10^{-4}$ and batch size of 64 samples. We set $\lambda_{s2s} = 0.2, \lambda_{adv} = 1, \lambda_{mono} = 5, \lambda_{fm} = 0.2, \lambda_{dur} = 1, \lambda_{f0} = 0.1$, and $\lambda_n = 1$. This setting of hyperparameters makes sure that all loss values are on the same scale and that the training is not sensitive to these hyperparameters. The scale factor ranges from 0.75 to 1.25 for data augmentation. We randomly divided the mel-spectrograms into segments of the shortest length in the batch. The training was conducted on a single NVIDIA A40 GPU.

[1] https://github.com/Kyubyong/g2p

TABLE II. COMPARISON OF EVALUATED MOS WITH 95% CONFIDENCE INTERVALS (CI) ON THE LIBRITTS DATASET.

Model	MOS-N (CI)	MOS-S (CI)
Ground Truth	4.35 ($\pm$ 0.04)	3.90 ($\pm$ 0.07)
FastSpeech 2 + HiFi-GAN	3.00 ($\pm$ 0.06)	3.51 ($\pm$ 0.07)
VITS	3.62 ($\pm$ 0.06)	3.70 ($\pm$ 0.07)
StyleTTS + HiFi-GAN	**4.03** ($\pm$ **0.05**)	**3.79** ($\pm$ **0.07**)

C. Evaluations

We performed two subjective evaluations: mean opinion score of naturalness (MOS-N) to measure the naturalness of synthesized speech, and mean opinion score of similarity (MOS-S) to evaluate the similarity between synthesized speech and reference for the multi-speaker model. We recruited native English speakers located in the U.S. to participate in the evaluations on Amazon Mechanical Turk[2]. In every experiment, we randomly selected 100 text samples from the test set. For each text, we synthesized speech using our model and the baseline models and included the ground truth for comparison. The baseline models include Tacorton 2 [51], FastSpeech 2 [11], and VITS [1]. For zero-shot speaker adapation experiments, we compared our model with StyleSpeech [32] and YourTTS [47]. All baseline models are pre-trained and publicly available (see Appendix B for details). The generated mel-spectrograms were converted into waveforms using the Hifi-GAN vocoder [52] for all models. Each set of speech was rated by 10 raters on a scale from 1 to 5 with 0.5 point increments. For a fair comparison, we downsampled our synthesized audio into 22 kHz to match those from baseline models. We used random references when synthesizing speech for the single-speaker and zero-shot speaker adaption experiments. For multi-speaker models, because our training did not require speaker labels, for a fair comparison with other models that use explicit speaker embeddings during training, we averaged the style vectors computed using all samples in the training set from the same speaker as the reference style.

When evaluating each set, we randomly permuted the order of the models and instructed the subjects to listen and rate them without telling them the model labels. It is similar to multiple stimuli with hidden reference and anchor (MUSHRA), allowing the subjects to compare subtle differences among models. We used the ground truth as hidden attention checkers: raters were dropped from analysis if the MOS of the ground truth was not ranked top two among all the models.

We also conducted objective evaluations using ASR metrics. We evaluated the robustness of the models to different lengths of text input. We created four test sets with text length $L < 10$, $10 \leq L < 50$, $50 \leq L < 100$, and $100 \leq L$, respectively. Each set contains 100 texts sampled from the WSJ0 dataset [53]. We calculated the word error rate of the synthesized speech from both single-speaker and multi-speaker models using a pre-trained ASR model from ESPnet [54]. To measure the inference speed, we computed the real-time factor (RFT),

[2] We obtained approval for our protocol (number IRB-AAAR8655) from the Institutional Review Board (IRB) .

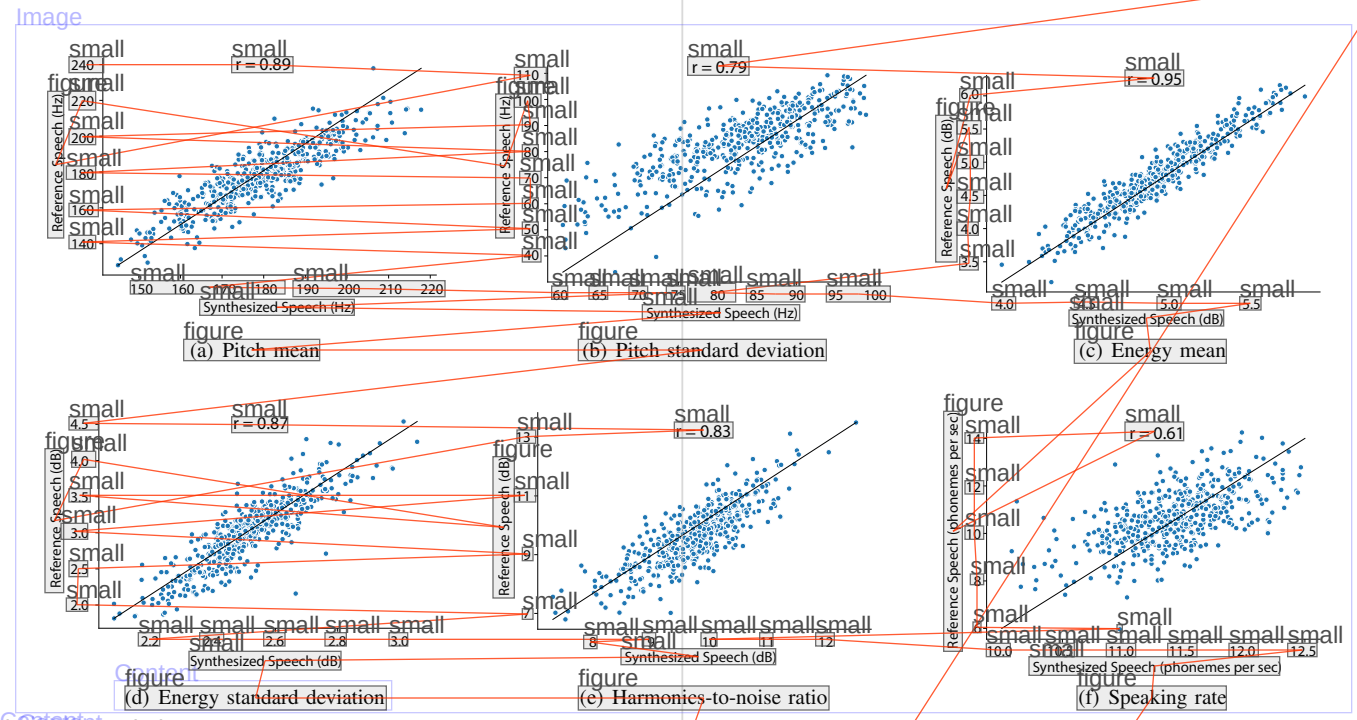


Fig. 4. Pearson correlation coefficients of six acoustic features associated with emotions between reference and synthesized speech on LJ Speech dataset.

TABLE III. COMPARISON OF EVALUATED MOS WITH 95% CONFIDENCE INTERVALS (CI) ON THE VCTK DATASET FOR UNSEEN SPEAKER ADAPTATION.

Model	MOS-N (CI)	MOS-S (CI)
Ground Truth	4.25 (±0.05)	4.28 (±0.06)
StyleTTS + HiFi-GAN	**3.58 (±0.06)**	**3.46 (±0.07)**
YourTTS	3.11 (±0.06)	3.39 (±0.07)
StyleSpeech + HiFi-GAN	2.16 (±0.05)	2.43 (±0.06)

which denotes the time (in seconds) needed for the model to synthesize a one-second waveform. RFT was measured on a server with one NVIDIA 2080Ti GPU and a batch size of 1. In addition, we conducted the same analysis on the correlations of acoustic features associated with emotions between reference and synthesized speech using four multi-speaker models. Since there is no style in FastSpeech 2 and VITS, we used a pre-trained X-vector model [55] from Kaldi [56] to extract the speaker embedding as the reference vector.

IV. RESULTS

A. Model Performance

Tables I, II, and III showcase the results of human subjective evaluations on the LJSpeech and LibriTTS datasets. When assessed for naturalness (MOS-N) and similarity (MOS-S), StyleTTS clearly outperforms other models, demonstrating its superior performance under both single-speaker, multi-speaker, and zero-shot settings. Our models are more robust compared to other models (Table IV), especially for long input texts. Since we do not use generative flows that require inverse Jacobian computation, our model is faster than VITS [1], even though it was not trained end-to-end like VITS (Table VI).

We do note that our evaluation results differ from those reported in the baseline models, particularly for VITS. The VITS model has been reported to yield results very close to the ground truth [1]. However, in our evaluation, VITS was found not to reach ground truth levels of performance. The primary factor leading to this discrepancy is the difference in evaluation methods. In VITS experiments, the traditional Mean Opinion Score (MOS) evaluation was used, where raters evaluated each module individually without any reference. The use of a reference point in our MUSHRA-like evaluation provides an anchor for rating, particularly the ground truth as the reference, which potentially lowers the scores of other models. A similar discrepancy has been reported in a very recent study that examines the effects of evaluation methods on the MOS results [57], and our evaluation of VITS is comparable to other studies that have tried to reproduce it on both LJSpeech and LibriTTS datasets [58], [59], [60], [61].

B. Visualization of Style Vectors

To verify that our model can learn meaningful style representations, we projected the style vectors extracted from reference audios into a 2-D plane for visualization using t-SNE [62]. We selected 50 samples of each emotion from a single speaker in ESD and projected the style vectors of each audio into the 2-D space. It can be seen in Fig. 2(a) that our style vector distinctively encodes the emotional tones of reference sentences even though the training does not use emotion labels. We also computed the style vectors using speech samples from the same speaker with our single-speaker model. This model is only

TABLE IV. ROBUSTNESS EVALUATION ON THE LJSPEECH AND LIBRITTS DATASET. WORD ERROR RATES (%) ARE REPORTED FOR DIFFERENT LENGTHS OF TEXT (L).

Model	WER (↓)			
	$L < 10$	$10 < L < 50$	$50 < L < 100$	$L > 100$
Single-speaker models (on LJSpeech)				
Tacotron 2 + + HiFi-GAN	17.22	12.61	12.95	46.33
FastSpeech 2 + HiFi-GAN	15.37	11.02	14.42	23.04
VITS	14.35	10.69	12.59	32.39
StyleTTS + HiFi-GAN	**9.42**	**7.44**	**11.97**	**22.24**
Multi-speaker models (on LibriTTS)				
FastSpeech 2 + HiFi-GAN	**12.73**	8.90	17.20	17.48
VITS	20.97	15.67	20.95	21.05
StyleTTS + HiFi-GAN	17.35	**8.26**	**14.58**	**15.83**

TABLE V. COMPARISON OF PEARSON CORRELATION COEFFICIENTS OF ACOUSTIC FEATURES ASSOCIATED WITH EMOTIONS BETWEEN REFERENCE AND SYNTHESIZED SPEECH IN MULTI-SPEAKER EXPERIMENTS. FASTSPEECH 2 AND VITS EMPLOY THE X-VECTOR AS THE REFERENCE.

Model	Pitch mean	Pitch standard deviation	Energy mean	Energy standard deviation	Harmonics-to-noise ratio	Shimmer	Jitter
FastSpeech 2	0.95	0.43	0.23	0.51	0.81	0.81	0.58
VITS	0.97	0.32	0.14	0.5	0.84	0.81	0.54
StyleTTS	**0.99**	**0.51**	**0.91**	**0.52**	**0.9**	**0.87**	**0.65**

TABLE VI. REAL TIME FACTOR (RTF) IN SECOND.

Model	RTF (s)
Tacotron 2 + HiFi-GAN	0.0868
VITS	0.0428
StyleTTS + HiFi-GAN	**0.0388**

trained on the LJSpeech dataset and therefore has never seen the selected speaker from ESD during training. Nevertheless, in Fig. 2(b), we see that our model can still clearly capture the emotional tones of the sentences, indicating that even when the reference audio is from a speaker different from the single speaker seen during training, it still can synthesize speech with the correct emotional tones. This shows that our model can implicitly extract emotions from an unlabeled dataset in a self-supervised manner. Lastly, we show projected style vectors from 10 unseen VCTK speakers each with 50 samples in Fig 2(c). Different speakers are perfectly separated from each other in the 2-D projection. This indicates that our model can learn speaker identities without explicit speaker labels and hence perform zero-shot speaker adaptation.

C. Style-Enabled Diverse Speech Synthesis

To show that the learned style vectors indeed enable diverse speech synthesis, we provide an example of synthesized speech with two different reference audios using our single-speaker model trained on the LJSpeech dataset in Figure 3. It can be seen clearly that the synthesized speech captures various aspects of the reference speech, including the pitch, prosody, pauses, and formant transitions. To systematically quantify this effect, we drew six scatter plots showing the correlations between synthesized and reference speech in acoustic features traditionally used for speech emotion recognition (Figure 4). The six features are pitch mean, pitch standard deviation, energy mean, energy standard deviation, harmonics-to-noise ratio, and speaking rate [63]. All six features demonstrate a significant correlation between the synthesized and reference speech ($p < 0.001$) with the correlation coefficients all above 0.6. Our model also outperforms other models on multi-speaker datasets in acoustic feature correlations (Table V). The results indicate that multiple aspects of the synthesized speech are matched to the reference, allowing flexible control over synthesized speech simply by selecting appropriate reference audios. Since our models also allow fully controllable pitch, energy, and duration, our approach is among the most flexible models in terms of controllability for speech synthesis.

D. Ablation Study

We further conduct an ablation study to verify the effectiveness of each component in our model with experiments of subjective human evaluation. We instructed the subjects to compare our single-speaker model to those with one component ablated. We converted the ratings into comparative mean opinion scores (CMOS) by taking the difference between the MOS of the baseline and ablated models. The results are shown in table VII, and more details are in Appendix A.

The leftmost table shows the results related to the proposed Transferable Monotonic Aligner (TMA) training. When training consists of 100% hard alignments so that no gradient is back-propagated to the parameters of the text aligner (equivalent to using an external aligner such as in FastSpeech 2), the rated MOS is decreased by -0.26. This is due to the covariate shift between the pre-training data (LibriSpeech) and TTS

TABLE VII. ABLATION STUDY FOR VERIFYING THE EFFECTIVENESS OF EACH PROPOSED COMPONENT.

Model	CMOS
StyleTTS	0
w/ 100% hard	− 0.26
w/ 0% hard	− 2.98
w/o $\mathcal{L}_{\text{mono}}$	− 0.10
w/o $\mathcal{L}_{\text{s2s}}$	− 2.48

Model	CMOS
StyleTTS	0
w/o pitch extractor	− 0.11
w/o pre-trained aligner	− 0.39
w/o augmentation	− 0.39
w/o discriminator	− 1.79

Model	CMOS
StyleTTS	0
w/o residual	− 0.30
AdaIN → AdaLN	− 0.21
AdaIN → Concat.	− 0.17
AdaIN → IN	− 0.03

TABLE VIII. COMPARISON OF PEARSON CORRELATION COEFFICIENTS OF ACOUSTIC FEATURES ASSOCIATED WITH EMOTIONS BETWEEN REFERENCE AND SYNTHESIZED SPEECH IN ABLATION STUDY.

Model	Pitch mean	Pitch standard deviation	Energy mean	Energy standard deviation	Harmonics-to-noise ratio	Shimmer	Jitter
Baseline	**0.90**	**0.53**	**0.77**	0.15	**0.79**	**0.66**	0.64
AdaIN → AadLN	0.89	0.52	0.67	**0.19**	0.76	0.53	**0.66**
AdaIN → Concat.	0.36	0.16	0.19	0.07	0.58	0.36	0.40
w/o residual	0.88	0.51	0.68	0.11	**0.79**	0.64	0.60

data (LJ Speech). An example of bad alignment of the pre-trained external aligner is shown in Figure 5. This shows that fine-tuning the aligner is effective in improving the quality of synthesized speech. However, when using 0% hard alignment (100% soft attention alignment), the model gets overfitted to reconstruct speech with soft alignment and is unable to produce audible speech using hard alignment during inference (-2.98 CMOS). We also see that both TMA objectives (equations 3 and 4) are important for high-quality speech synthesis.

The table in the middle shows the effects of removing various training techniques and components. Using an external pitch extractor (such as acoustic-based methods) decreases MOS by -0.11. This is likely caused by the acoustic-based pitch extraction method sometimes failing to extract the correct F0 curve, and fine-tuning the pitch extractor along with the decoder makes the model learn better pitch representation (see Appendix A-C). Without a pre-trained text aligner (such as VITS), the rated MOS is decreased by -0.39. This indicates that our transfer learning is helpful for mitigating overfitting problems when training internal aligners with a relatively small dataset. Removing our novel duration-invariant data augmentation also lowers the performance. Lastly, training without discriminators significantly affects the perceived sound quality.

The rightmost table shows architecture changes by removing the residual features and replacing the AdaIN components in the decoder and predictor with instance normalization (IN), AdaLN, and simple feature concatenation (Concat). Their effects on style reflection are also shown in Table VIII. Removing the residual features in the decoder decreases both naturalness and correlations between synthesized and reference speech. Layer normalization is also worse than IN for both metrics. Concatenating styles in place of AdaIN dramatically decreases the correlations and lowers rated naturalness, confirming our observation that all previous methods that use concatenation to incorporate style information ([1], [9], [64], [12], [13], [15], [16]) are not as effective as AdaIN due to the lack of temporal modulations (see Appendix A-B). Lastly, we see that replacing AdaIN with IN does not significantly affect the rated naturalness, suggesting that the improved naturalness is not due to the introduction of styles but our novel technical improvements including TMA, data augmentation, use of IN, pitch extractor, and residual features. Nevertheless, styles enable diverse speech synthesis which models without styles cannot do.

V. CONCLUSIONS

We introduced StyleTTS, a novel natural and diverse text-to-speech (TTS) synthesis approach. Our research takes a distinctive step forward in leveraging the strengths of parallel TTS systems with several novel constitutions that include a unique transferable monotonic aligner (TMA) training while integrating style information via AdaIN. We demonstrated that this method can effectively reflect stylistic features from reference audio. Moreover, the style vectors from our model encode a rich set of information present in the reference audio, including pitch, energy, speaking rates, formant transitions, and speaker identities. This allows easy control of the synthesized speech's prosodic patterns and emotional tones by choosing an appropriate reference style while benefiting from robust and fast speech synthesis of parallel TTS systems. Collectively, they enable natural speech synthesis with diverse speech styles that go beyond what was achieved in previous TTS systems.

Our contribution lies not only in the theoretical underpinnings but also in its practical applicability. Our approach empowers various new applications, including movie dubbing, book narration, unsupervised speech emotion recognition, personalized speech generation, and any-to-any voice conversion (see Appendix C and our follow-up work [65] for more details). Our source code and pre-trained models are publicly available [3] to assist research in this area further.

VI. ACKNOWLEDGMENTS

We thank Gavin Mischler and Vinay Raghavan for their feedback. Funding was from the national institute of health (NIHNIDCD) and Marie-Josee and Henry R. Kravis.

[3] https://github.com/yl4579/StyleTTS

References

[1] J. Kim, J. Kong, and J. Son, "Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139, 18–24 Jul 2021, pp. 5530–5540.

[2] Y. Jia, H. Zen, J. Shen, Y. Zhang, and Y. Wu, "Png bert: augmented bert on phonemes and graphemes for neural tts," *arXiv preprint arXiv:2103.15060*, 2021.

[3] X. Tan, J. Chen, H. Liu, J. Cong, C. Zhang, Y. Liu, X. Wang, Y. Leng, Y. Yi, L. He *et al.*, "Naturalspeech: End-to-end text to speech synthesis with human-level quality," *arXiv preprint arXiv:2205.04421*, 2022.

[4] X. Tan, T. Qin, F. Soong, and T.-Y. Liu, "A survey on neural speech synthesis," *arXiv preprint arXiv:2106.15561*, 2021.

[5] W.-N. Hsu, Y. Zhang, R. Weiss, H. Zen, Y. Wu, Y. Cao, and Y. Wang, "Hierarchical generative modeling for controllable speech synthesis," in *International Conference on Learning Representations*, 2019.

[6] Y.-J. Zhang, S. Pan, L. He, and Z.-H. Ling, "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)*. IEEE, 2019, pp. 6945–6949.

[7] Y. Lee, J. Shin, and K. Jung, "Bidirectional variational inference for non-autoregressive text-to-speech," in *International Conference on Learning Representations*, 2020.

[8] R. Valle, K. J. Shih, R. Prenger, and B. Catanzaro, "Flowtron: an autoregressive flow-based generative network for text-to-speech synthesis," in *International Conference on Learning Representations*, 2020.

[9] J. Kim, S. Kim, J. Kong, and S. Yoon, "Glow-tts: A generative flow for text-to-speech via monotonic alignment search," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[10] R. Valle, J. Li, R. Prenger, and B. Catanzaro, "Mellotron: Multispeaker expressive voice synthesis by conditioning on rhythm, pitch and global style tokens," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6189–6193.

[11] Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech 2: Fast and high-quality end-to-end text to speech," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=piLPYqxtWuA

[12] R. Skerry-Ryan, E. Battenberg, Y. Xiao, Y. Wang, D. Stanton, J. Shor, R. Weiss, R. Clark, and R. A. Saurous, "Towards end-to-end prosody transfer for expressive speech synthesis with tacotron," in *international conference on machine learning*. PMLR, 2018, pp. 4693–4702.

[13] Y. Wang, D. Stanton, Y. Zhang, R.-S. Ryan, E. Battenberg, J. Shor, Y. Xiao, Y. Jia, F. Ren, and R. A. Saurous, "Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5180–5189.

[14] L. Chen, Y. Deng, X. Wang, F. K. Soong, and L. He, "Speech bert embedding for improving prosody in neural tts," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6563–6567.

[15] G. Sun, Y. Zhang, R. J. Weiss, Y. Cao, H. Zen, and Y. Wu, "Fully-hierarchical fine-grained prosody modeling for interpretable speech synthesis," in *ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2020, pp. 6264–6268.

[16] R. Liu, B. Sisman, G. lai Gao, and H. Li, "Expressive tts training with frame and style reconstruction loss," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2021.

[17] Y. Ren, Y. Ruan, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech: fast, robust and controllable text to speech," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019, pp. 3171–3180.

[18] A. Łańcucki, "Fastpitch: Parallel text-to-speech with pitch prediction," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6588–6592.

[19] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1501–1510.

[20] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 172–189.

[21] M.-Y. Liu, X. Huang, A. Mallya, T. Karras, T. Aila, J. Lehtinen, and J. Kautz, "Few-shot unsupervised image-to-image translation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 10 551–10 560.

[22] Y. Choi, Y. Uh, J. Yoo, and J.-W. Ha, "Stargan v2: Diverse image synthesis for multiple domains," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8188–8197.

[23] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4401–4410.

[24] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8110–8119.

[25] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen, and T. Aila, "Alias-free generative adversarial networks," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[26] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, "Maskgan: Towards diverse and interactive facial image manipulation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5549–5558.

[27] P. Zhu, R. Abdal, Y. Qin, and P. Wonka, "Sean: Image synthesis with semantic region-adaptive normalization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5104–5113.

[28] J. C. Chou and H.-Y. Lee, "One-Shot Voice Conversion by Separating Speaker and Content Representations with Instance Normalization," in *Proc. Interspeech 2019*, 2019, pp. 664–668.

[29] Y.-H. Chen, D.-Y. Wu, T.-H. Wu, and H.-y. Lee, "Again-vc: A one-shot voice conversion using activation guidance and adaptive instance normalization," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5954–5958.

[30] Y. A. Li, A. Zare, and N. Mesgarani, "StarGANv2-VC: A Diverse, Unsupervised, Non-Parallel Framework for Natural-Sounding Voice Conversion," in *Proc. Interspeech 2021*, 2021, pp. 1349–1353.

[31] M. Chen, X. Tan, B. Li, Y. Liu, T. Qin, S. Zhao, and T.-Y. Liu, "Adaspeech: Adaptive text to speech for custom voice," *arXiv preprint arXiv:2103.00993*, 2021.

[32] D. Min, D. B. Lee, E. Yang, and S. J. Hwang, "Meta-stylespeech: Multi-speaker adaptive text-to-speech generation," *arXiv preprint arXiv:2106.03153*, 2021.

[33] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, "Montreal forced aligner: Trainable text-speech alignment using kaldi." in *Interspeech*, vol. 2017, 2017, pp. 498–502.

[34] C. Miao, L. Shuang, Z. Liu, C. Minchuan, J. Ma, S. Wang, and J. Xiao, "Efficienttts: An efficient and high-quality text-to-speech architecture," in *International Conference on Machine Learning*. PMLR, 2021, pp. 7700–7709.

[35] I. Elias, H. Zen, J. Shen, Y. Zhang, Y. Jia, R. Skerry-Ryan, and Y. Wu, "Parallel tacotron 2: A non-autoregressive neural tts model with differentiable duration modeling," *arXiv preprint arXiv:2103.14574*, 2021.

[36] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks,"

IEEE transactions on Signal Processing, vol. 45, no. 11, pp. 2673–2681, 1997.

[37] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[39] P. Boersma *et al.*, "Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound," in *Proceedings of the institute of phonetic sciences*, vol. 17, no. 1193. Citeseer, 1993, pp. 97–110.

[40] S. Kum and J. Nam, "Joint detection and classification of singing voice melody using convolutional recurrent neural networks," *Applied Sciences*, vol. 9, no. 7, p. 1324, 2019.

[41] A. De Cheveigné and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–1930, 2002.

[42] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[43] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *Advances in neural information processing systems*, vol. 29, pp. 2234–2242, 2016.

[44] K. Ito and L. Johnson, "The lj speech dataset," https://keithito.com/LJ-Speech-Dataset/, 2017.

[45] H. Zen, V. Dang, R. Clark, Y. Zhang, R. J. Weiss, Y. Jia, Z. Chen, and Y. Wu, "Libritts: A corpus derived from librispeech for text-to-speech," *arXiv preprint arXiv:1904.02882*, 2019.

[46] J. Yamagishi, C. Veaux, K. MacDonald *et al.*, "Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit (version 0.92)," 2019.

[47] E. Casanova, J. Weber, C. D. Shulby, A. C. Junior, E. Gölge, and M. A. Ponti, "Yourtts: Towards zero-shot multi-speaker tts and zero-shot voice conversion for everyone," in *International Conference on Machine Learning*. PMLR, 2022, pp. 2709–2720.

[48] K. Zhou, B. Sisman, R. Liu, and H. Li, "Seen and unseen emotional style transfer for voice conversion with a new emotional speech dataset," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 920–924.

[49] Y.-Y. Yang, M. Hira, Z. Ni, A. Chourdia, A. Astafurov, C. Chen, C.-F. Yeh, C. Puhrsch, D. Pollack, D. Genzel, D. Greenberg, E. Z. Yang, J. Lian, J. Mahadeokar, J. Hwang, J. Chen, P. Goldsborough, P. Roy, S. Narenthiran, S. Watanabe, S. Chintala, V. Quenneville-Bélair, and Y. Shi, "Torchaudio: Building blocks for audio and speech processing," *arXiv preprint arXiv:2110.15018*, 2021.

[50] I. Loshchilov and F. Hutter, "Fixing weight decay regularization in adam," 2018. [Online]. Available: https://openreview.net/forum?id=rk6qdGgCZ

[51] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. 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)*. IEEE, 2018, pp. 4779–4783.

[52] J. Kong, J. Kim, and J. Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[53] J. Garofolo, D. Graff, D. Paul, and D. Pallett, "Csr-i (wsj0) complete ldc93s6a," *Web Download. Philadelphia: Linguistic Data Consortium*, vol. 83, 1993.

[54] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "ESPnet: End-to-end speech processing toolkit," in *Proceedings of Interspeech*, 2018, pp. 2207–2211. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2018-1456

[55] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.

[56] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011, iEEE Catalog No.: CFP11SRW-USB.

[57] C.-H. Chiang, W.-P. Huang, and H.-y. Lee, "Why we should report the details in subjective evaluation of tts more rigorously," *arXiv preprint arXiv:2306.02044*, 2023.

[58] T. Hayashi, R. Yamamoto, T. Yoshimura, P. Wu, J. Shi, T. Saeki, Y. Ju, Y. Yasuda, S. Takamichi, and S. Watanabe, "Espnet2-tts: Extending the edge of tts research," *arXiv preprint arXiv:2110.07840*, 2021.

[59] Y. Lei, S. Yang, J. Cong, L. Xie, and D. Su, "Glow-wavegan 2: high-quality zero-shot text-to-speech synthesis and any-to-any voice conversion," *arXiv preprint arXiv:2207.01832*, 2022.

[60] D. Lim, S. Jung, and E. Kim, "Jets: Jointly training fastspeech2 and hifi-gan for end to end text to speech," *arXiv preprint arXiv:2203.16852*, 2022.

[61] Z. Liu, Y. Guo, and K. Yu, "Diffvoice: Text-to-speech with latent diffusion," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[62] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[63] C. Busso, M. Bulut, S. Narayanan, J. Gratch, and S. Marsella, "Toward effective automatic recognition systems of emotion in speech," *Social emotions in nature and artifact: emotions in human and human-computer interaction, J. Gratch and S. Marsella, Eds*, pp. 110–127, 2013.

[64] M. Chen, X. Tan, Y. Ren, J. Xu, H. Sun, S. Zhao, T. Qin, and T.-Y. Liu, "Multispeech: Multi-speaker text to speech with transformer," in *Proc. Interspeech*, 2020, p. pages 4024–4028.

[65] Y. A. Li, C. Han, and N. Mesgarani, "Styletts-vc: One-shot voice conversion by knowledge transfer from style-based tts models," in *2022 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2023, pp. 920–927.

[66] S.-w. Park, D.-y. Kim, and M.-c. Joe, "Cotatron: Transcription-guided speech encoder for any-to-many voice conversion without parallel data," *arXiv preprint arXiv:2005.03295*, 2020.

[67] J. An, S. Huang, Y. Song, D. Dou, W. Liu, and J. Luo, "Artflow: Unbiased image style transfer via reversible neural flows," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 862–871.

[68] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *arXiv preprint arXiv:1802.05957*, 2018.

[69] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," *Advances in neural information processing systems*, vol. 29, pp. 901–909, 2016.

APPENDIX A
ABLATION STUDY DETAILS

In this section, we describe the detailed settings of each condition in Table VII and provide more discussions of the results in Table VII and Table VIII.

A. TMA-related

There are three Transferable Monotonic Aligner (TMA) related innovations in this work: the decoder is trained with hard monotonic alignment and soft attention in a 50%-50% manner and two TMA objectives functions. The 50%-50% training is motivated by the fact that the monotonic alignment search proposed in [9] is not differentiable, and the soft attention alignment does not necessarily provide correct alignments for duration prediction in the second stage of training. This 50%-50% split is arbitrary and can be changed to anything from 10%-90% to 90%-10%, depending on the dataset and the application. When the ratio is 100%-0%, it becomes the case where the external aligners are not fine-tuned like in most parallel TTS systems such as FastSpeech [17], while when the ratio is 0%-100%, it becomes the case we fine-tune the aligner with only soft attention such as in Cotatron [66] for voice conversion applications. We find that training with external aligners (100% hard, no fine-tuning) decreases the naturalness of the synthesized speech because bad alignments can happen due to covariate shifts between the training dataset (LibriSpeech) and testing dataset (LJSpeech) as in the case of Montreal Forced Aligner [33]. One example is given in the leftmost figure in Figure 5. On the other hand, if we only fine-tune the decoder with soft alignment, the decoder will overfit on the soft alignment and be unable to synthesize audible speech from hard alignment because the soft alignments are not either 0 or 1 and the precise numerical values of alignments are used by the decoder to generate speech.

Another notable case is when we do not use a pre-trained text aligner such as in the case of VITS. This case makes MOS even lower than the case of no fine-tuning, suggesting that overfitting on a smaller dataset can be more detrimental than failure in generalization on the TTS dataset for some samples. The figure in the middle in Fig. 5 shows an alignment with gaps and no background noises. This indicates overfitting of the text aligner to the smaller dataset for the mel-spectrogram reconstruction objective. However, since our goal is to synthesize the speech from predicted alignment, overfitting to speech reconstruction can be harmful to natural speech synthesis during inference.

In addition to the 50%-50% training, we also introduced two TMA objectives $\mathcal{L}_{s2s}$ and $\mathcal{L}_{mono}$. This is motivated by the fact that $\mathcal{L}_{s2s}$ learns correct alignments for S2S-ASR but not necessarily monotonic while non-differentiable monotonic alignments obtained through dynamic programming algorithms proposed in [9] do not necessarily produce correct alignments. By combining $\mathcal{L}_{s2s}$ and $\mathcal{L}_{mono}$, we can learn an aligner that produces both correct and monotonic alignments.

B. AdaIN, AdaLN, and Concatenation

As shown in Table VII and Table VIII, AdaIN outperforms AdaLN and simple concatenation for both naturalness and style reflection. Here we describe our intuitions behind these results.

Concatenation vs. AdaIN. When we concatenate the style vector to each frame of the encoded phonetic representations, we create a representation $\boldsymbol{h}_{\text{style}} = \begin{bmatrix} \boldsymbol{h}_{\text{text}} \\ \hline \boldsymbol{s} \end{bmatrix}$. When the $\boldsymbol{h}_{\text{style}}$ is passed to the next convolution layer whose parameter is W, we get

$$\begin{aligned} \boldsymbol{h}_{\text{style}} \cdot W &= \begin{bmatrix} \boldsymbol{h}_{\text{text}} \\ \hline \boldsymbol{s} \end{bmatrix} \cdot [W_{\text{text}} | W_{\text{style}}] \\ &= \boldsymbol{h}_{\text{text}} \cdot W_{\text{text}} + \boldsymbol{s} \cdot W_{\text{style}} \\ &= \boldsymbol{h}_{\text{text}} \cdot W_{\text{text}} + \text{Concat}\ (\boldsymbol{h}_{\text{text}}, \boldsymbol{s}) \end{aligned} \tag{13}$$

where W_{text} and W_{style} are block matrix notation of the corresponding weights for $\boldsymbol{h}_{\text{style}}$ and $\boldsymbol{s}$ and Concat $(\boldsymbol{h}_{\text{text}}, \boldsymbol{s}) = \boldsymbol{s} \cdot W_{\text{style}}$ denotes the concatenation operation as a function of input $\boldsymbol{h}_{\text{text}}$ and style vector $\boldsymbol{s}$. This Concat (x, s) function is almost like AdaIN in equation 1 where $L_\mu(s) = W_{\text{style}}$ except we do not have the temporal modulation term $L_\sigma(s)$. The modulation term is very important in style transfer, and some works argue that modulation alone is enough for diverse style representations [24], [67]. In contrast, concatenation only provides the addition term (L_μ) but no modulation term (L_σ). Intuitively, the modulation term can determine the variance of the pitch and energy, for example, and therefore without such a term, correlations for pitch and energy standard deviation are much lower than AdaIN and AdaLN as shown in Table VIII.

AdaLN vs. AdaIN. Generative models for speech synthesis learn to generate mel-spectrograms, which is essentially a 1-D feature map with 80 channels. Each channel in the mel-spectrogram represents a single frequency range. When we apply AdaIN, we learn a distribution with a style-specific mean and variance for *each channel*, compared to AdaLN, where a single mean and variance are learned for the *entire feature map*. This inherent difference between feature distributions makes AdaIN more expressive in terms of style reflection than AdaLN.

C. Pitch Extractor

Acoustic methods for pitch estimation sometimes fail because of the presence of non-stationary speech intervals and sensitivity of hyper-parameters as discussed in the original papers that propose these methods [39], [41]. A neural network trained with ground truth from these methods, however, can leverage the problems of failed pitch estimation because the failed pitch estimation can be regarded as noises in the training set, so it does not affect the generalization of the pitch extractor network. Moreover, since the pitch extractor is fine-tuned along with the decoder, there is no ground truth for the pitch beside the sole objective that the decoder needs to use extracted pitch information to reconstruct the speech. This fine-tuning allows better pitch representations beyond the original F0 in Hertz, but it also allows flexible pitch control as we can still recognize the pitch curves and edit them later when needed during inference.

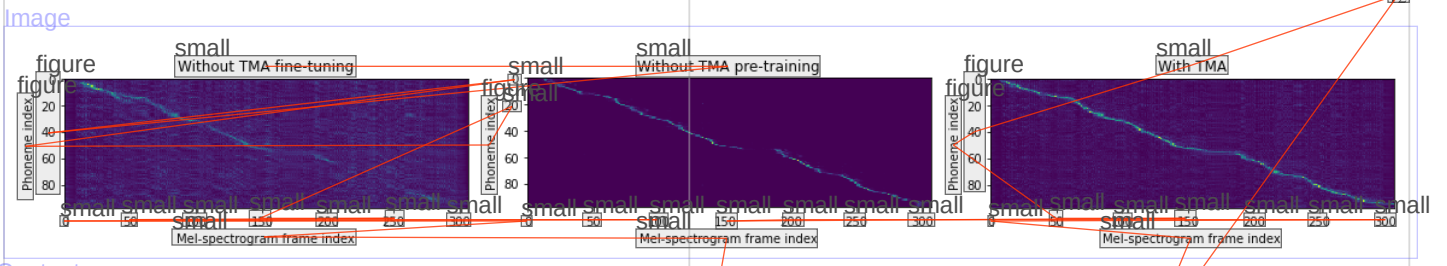


Fig. 5. An example showing text alignments under different conditions. **Left**: No TMA fine-tuning (100% hard alignment such as FastSpeech). This is an example of a failed alignment. **Middle**: No pre-trained text aligner (such as VITS). Note the gaps between alignments and clean attention (with no background noise), indicating some degrees of overfitting to the TTS speech dataset. **Right**: Full TMA fine-tuning. Note that TMA learns an alignment that is both continuous and monotonic compared to without fine-tuning and pre-training.

APPENDIX B
SUBJECTIVE EVALUATION DETAILS

We used the publicly available pre-trained models as baselines for comparison. For the single-speaker experiment on the LJSpeech dataset, we used pre-trained Tacotron2[4], Fastspeech2 [5], HiFiGAN [6] from ESPnet. We used VITS [7] and YourTTS [8] from the official implementation. We randomly selected 100 text samples from the test set to synthesize the speech. Since audios from our model were synthesized using Hifi-GAN trained with audios sampled at 24 kHz, for a fair comparison, we resampled all the audios into 22 kHz and then normalized their amplitude. We used the pre-trained model for StyleSpeech [32] [9] from a public repository in GitHub for comparison of zero-shot speaker adaptation in Appendix C. We did not use the official implementation because the vocoder used was MelGAN sampled at 16 kHz while the implementation we employed uses Hifi-GAN sampled at 22 kHz, which is comparable to other models.

To reduce the listening fatigue, we randomly divided these 100 sets of audios into 5 batches [10] with each batch containing 20 sets of audios for comparison. We launched the 5 batches sequentially on Amazon Mechanical Turk (AMT) [11]. We required participating subjects to be native English speakers located in the United States. For each batch, we made sure that we had collected completed responses from at least 10 self-reported native speakers whose IP addresses were within the United States and residential (i.e., not VPN or proxies). We used the average score that a subject rated on ground truth audios to check whether this subject carefully finished the survey as the subjects did not know which audio was the ground truth. We then disqualified and dropped all ratings from the subjects whose average ground truth score was not ranked top two among all the models. Finally, 46 out of 50 subjects were qualified for this experiment.

In the multi-speaker experiments, we used pre-trained Fastspeech2[12] , VITS [13] , and HiFiGAN [14] from ESPnet. We used pre-trained VITS from ESPnet instead of the official repository because we need the model to be trained on the LibriTTS dataset; however, the official models were trained on the LJSpeech or VCTK dataset.

Similar to the single-talker experiment, we launched 5 batches [15] on AMT when we tested the multi-talker models on the LibriTTS dataset. 48 out of 58 subjects were qualified. We launched 3 batches [16] with batch sizes 33, 33, 34, respectively, when we tested the multi-talker models on the VCTK dataset. 28 out of 30 subjects were qualified.

APPENDIX C
ZERO-SHOT VOICE CONVERSION

Since our text encoder, text aligner, pitch extractor, and decoder are trained in a speaker-agnostic manner, our decoder can reconstruct speech from any aligned phonemes, pitch, energy, and reference speakers. Therefore, our model can perform any-to-any voice conversion by extracting the alignment, pitch, and energy from an input mel-spectrogram and generating speech using a style vector of reference audio from an arbitrary target speaker. Our voice conversion scheme is transcription-guided, similar to Mellotron [10] and Cotatron [66]. We provide one example in Figure 6 with both source and target speaker unseen from the LJSpeech and VCTK datasets. We refer our readers to our demo page for more examples.

APPENDIX D
DETAILED MODEL ARCHITECTURES

In this section, we provide detailed model architectures of StyleTTS, which consists of eight modules. Since we use the same text encoder as in Tacorton 2 [51], very similar architecture to the decoder of Tacotron 2 for text aligner and the

[4]The model was `kan-bayashi/ljspeech_tacotron2` from ESPNet

[5]The model was `kan-bayashi/ljspeech_fastspeech2` from ESPNet

[6]The model was `parallel_wavegan/ljspeech_hifigan.v1` from ESPNet

[7]The implementation can be found at https://github.com/jaywalnut310/vits

[8]The implementation can be found at https://github.com/Edresson/YourTTS

[9]The implementation can be found at https://github.com/jaywalnut310/vits

[10]The survey (batch 1) can be found at https://survey.alchemer.com/s3/6696332/LJ100-B1

[11]https://www.mturk.com/

[12]The model was `kan-bayashi/libritts_xvector_conformer_fastspeech2` from ESPNet

[13]The model was `kan-bayashi/libritts_xvector_vits` from ESPNet

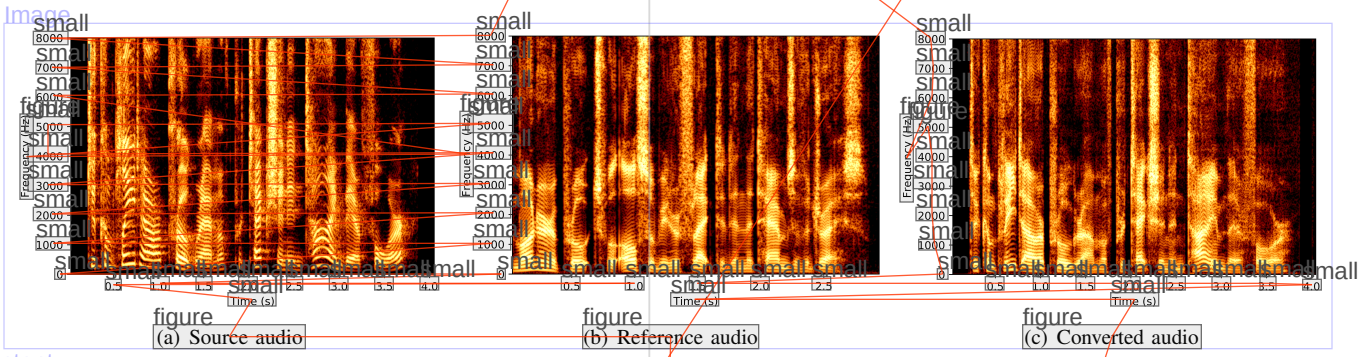
[14]The model was `parallel_wavegan/libritts_hifigan.v1` from ESPNet

[15]The survey (batch 1) can be found at https://survey.alchemer.com/s3/6705095/LibriTTS-seen100-B1

[16]The survey (batch 1) can be found at https://survey.alchemer.com/s3/6706053/zero-shot-B1

TABLE IX. DECODER ARCHITECTURE. T REPRESENTS THE INPUT LENGTH OF THE MEL-SPECTROGRAM, p IS THE INPUT F0, n IS THE INPUT ENERGY, AND s IS THE STYLE CODE. $\tilde{n}$ AND $\tilde{p}$ ARE THE PROCESSED PITCH AND ENERGY, AND h_{RES} IS THE OUTPUT OF THE PHONEME RESIDUAL SUB-MODULE.

Submodule	External Input	Layer	Norm	Output Shape
F0 processing	p	Input F0 p		$1\times T$
		ResBlk	IN	$64\times T$
		Conv 1×1		$1\times T$
Energy processing	n	Input energy n		$1\times T$
		ResBlk	IN	$64\times T$
		Conv 1×1		$1\times T$
Phoneme residual	h_{text}	Input h_{text}		$512\times T$
		Conv 1×1	IN	$64\times T$
IN ResBlks	$\tilde{p}$, $\tilde{n}$, h_{res}	Concat		$(512+2)\times T$
		ResBlk	IN	$1024\times T$
		ResBlk	IN	$1024\times T$
AdaIN ResBlks	$\tilde{p}$, $\tilde{n}$, h_{res}	Concat		$(1024+2+64)\times T$
		ResBlk	AdaIN	$1024\times T$
	$\tilde{p}$, $\tilde{n}$, h_{res}	Concat		$(1024+2+64)\times T$
		ResBlk	AdaIN	$1024\times T$
	$\tilde{p}$, $\tilde{n}$, h_{res}	Concat		$(1024+2+64)\times T$
		ResBlk	AdaIN	$512\times T$
		ResBlk	AdaIN	$512\times T$
		ResBlk	AdaIN	$512\times T$
		Conv 1×1		$80\times T$



(a) Source audio (b) Reference audio (c) Converted audio

Fig. 6. An example of any-to-any voice conversion. The source audio is from the LJSpeech dataset and the reference audio is from the VCTK dataset, both unseen during training.

same architecture as the JDC network [40] for pitch extractor, we leave the readers to the above references for detailed descriptions of these modules. Here, we only provide detailed architectures for the other five modules. All activation functions used are leaky ReLU (LReLU) with a negative slope of 0.2. We apply spectral normalization [68] to all trainable parameters in style encoder and discriminator and weight normalization [69] to those in decoder because they are shown to be beneficial for adversarial training.

Decoder (Table IX). Our decoder takes four inputs: the aligned phoneme representation, the pitch F0, the energy, and the style code. It consists of seven 1-D residual blocks (ResBlk) along with three sub-modules for processing the input F0, energy, and residual of the phoneme representation. The normalization consists of both instance normalization (IN) and adaptive instance normalization (AdaIN). We concatenate (Concat) the processed F0, energy, and residual of phonemes with the output from each residual block as the input to the next block for the first three blocks.

Style Encoder and Discriminator (Table X). Our style encoder and discriminator share the same architecture, which consists of four 2-D residual blocks (ResBlk). The dimension of the style vector is set to 128. We use learned weights for pooling through a dilated convolution (Dilated Conv) layer with a kernel size of 3×3. We apply an adaptive average pooling (AdaAvg) along the time axis of the feature map to make the output independent of the size of the input mel-spectrogram.

Duration and Prosody Predictors (Table XI). The duration predictor and prosody predictors are trained together in the second stage of training. There is a shared 3-layer bidirectional LSTM (bi-LSTM) s between the duration predictor and prosody predictor named text feature encoder, each followed by an adaptive layer normalization (AdaLN). AdaLN is similar to AdaIN where the gain and bias are predicted from the style vector s. However, unlike AdaIN which normalizes each channel independently, AdaLN normalizes the entire feature map. The style vector s is also concatenated (Concat) with the output to every token from each LSTM layer as the input

TABLE X. STYLE ENCODER AND DISCRIMINATOR ARCHITECTURES. T REPRESENTS THE INPUT LENGTH OF THE MEL-SPECTROGRAM, AND D IS THE OUTPUT DIMENSION. FOR STYLE ENCODER, $D = 128$. FOR DISCRIMINATOR, $D = 1$.

Layer	Pooling	Norm	Output Shape
Mel $\boldsymbol{x}$			$1\times80\times T$
Conv 1×1			$64\times80\times T$
ResBlk	Dilated Conv		$128\times40\times T/2$
ResBlk	Dilated Conv	-	$256\times20\times T/4$
ResBlk	Dilated Conv	-	$512\times10\times T/8$
ResBlk	Dilated Conv		$512\times5\times T/16$
LReLU			$512\times5\times T/16$
Conv 5×5			$512\times1\times T/80$
LReLU			$512\times1\times T/80$
AdaAvg			512×1
Linear			$D\times1$

TABLE XI. DURATION AND PROSODY PREDICTOR ARCHITECTURES. N REPRESENTS THE NUMBER OF INPUT PHONEMES AND T REPRESENTS THE LENGTH OF THE ALIGNMENT. h_{TEXT} IS THE HIDDEN PHONEME REPRESENTATION FROM THE TEXT ENCODER, d_{ALIGN} IS THE MONOTONIC ALIGNMENT WITH SHAPE $N \times T$, $\boldsymbol{s}$ IS THE STYLE CODE, a_{PRED} IS THE PREDICTED DURATION, p_{PRED} IS THE PREDICTED PITCH AND $\|x\|_{\text{PRED}}$ IS THE PREDICTED ENERGY. h_{PROSODY} AND h_{APROSODY} ARE INTERMEDIATE OUTPUTS FROM SUBMODULES.

Submodule	External Input	Layer	Norm	Output Shape	Submodule Output
Prosody Encoder	h_{text}, $\boldsymbol{s}$	Concat		$(512+128)\times N$	
	$\boldsymbol{s}$	bi-LSTM	AdaLN	$512\times N$	
	$\boldsymbol{s}$	Concat		$(512+128)\times N$	h_{prosody}
	$\boldsymbol{s}$	bi-LSTM	AdaLN	$512\times N$	
	$\boldsymbol{s}$	Concat	-	$(512+128)\times N$	
	$\boldsymbol{s}$	bi-LSTM	AdaLN	$512\times N$	
Duration Projection	h_{prosody}	bi-LSTM		$512\times N$	
		Linear	-	$1\times N$	a_{pred}
Shared LSTM	h_{prosody}, d_{align}	Dot		$512\times T$	
	$\boldsymbol{s}$	Concat		$(512+128)\times T$	h_{aprosody}
	$\boldsymbol{s}$	bi-LSTM		$512\times T$	
Pitch Predictor	h_{aprosody}, $\boldsymbol{s}$	ResBlk	AdaIN	$512\times T$	
	$\boldsymbol{s}$	ResBlk	AdaIN	$256\times T$	
	$\boldsymbol{s}$	ResBlk	AdaIN	$256\times T$	p_{pred}
	$\boldsymbol{s}$	Linear	-	$1\times T$	
Energy Predictor	h_{aprosody}, $\boldsymbol{s}$	ResBlk	AdaIN	$512\times T$	
	$\boldsymbol{s}$	ResBlk	AdaIN	$256\times T$	
	$\boldsymbol{s}$	ResBlk	AdaIN	$256\times T$	$\|x\|_{\text{pred}}$
	$\boldsymbol{s}$	Linear	-	$1\times T$	

to the next LSTM layer. Lastly, we have a final bidirectional LSTM and a linear projection L that maps h_{prosody} into the predicted duration.

The hidden representation h_{prosody} is dotted with the alignment d_{align} and sent to the prosody decoder. The prosody encoder consists of one bidirectional LSTM and two sets of three residual blocks (ResBlk) with AdaIN followed by a linear projection, one for predicting the F0 and another for predicting the energy, respectively.