HierSpeech: Bridging the Gap between Text and Speech by Hierarchical Variational Inference using Self-supervised Representations for Speech Synthesis

Sang-Hoon Lee[1] **Seung-Bin Kim**[2] **Ji-Hyun Lee**[2]

Eunwoo Song[3] **Min-Jae Hwang**[3] **Seong-Whan Lee**[2*]

[1]Department of Brain and Cognitive Engineering, Korea University, Seoul, Korea
[2]Department of Artificial Intelligence, Korea University, Seoul, Korea
[3]NAVER Corp., Seongnam, Korea
[1,2]{sh_lee, sb_kim, jihyun-lee, sw.lee}@korea.ac.kr
[3]{eunwoo.song, min-jae.hwang}@navercorp.com

Abstract

This paper presents HierSpeech, a high-quality end-to-end text-to-speech (TTS) system based on a hierarchical conditional variational autoencoder (VAE) utilizing self-supervised speech representations. Recently, single-stage TTS systems, which directly generate raw speech waveform from text, have been getting interest thanks to their ability in generating high-quality audio within a fully end-to-end training pipeline. However, there is still a room for improvement in the conventional TTS systems. Since it is challenging to infer both the linguistic and acoustic attributes from the text directly, missing the details of attributes, specifically linguistic information, is inevitable, which results in mispronunciation and over-smoothing problem in their synthetic speech. To address the aforementioned problem, we leverage self-supervised speech representations as additional linguistic representations to bridge an information gap between text and speech. Then, the hierarchical conditional VAE is adopted to connect these representations and to learn each attribute hierarchically by improving the linguistic capability in latent representations. Compared with the state-of-the-art TTS system, HierSpeech achieves +0.303 comparative mean opinion score, and reduces the phoneme error rate of synthesized speech from 9.16% to 5.78% on the VCTK dataset. Furthermore, we extend our model to HierSpeech-U, an untranscribed text-to-speech system. Specifically, HierSpeech-U can adapt to a novel speaker by utilizing self-supervised speech representations without text transcripts. The experimental results reveal that our method outperforms publicly available TTS models, and show the effectiveness of speaker adaptation with untranscribed speech.

1 Introduction

Text-to-speech (TTS) systems have undergone significant improvements in synthesizing high-quality speech from text sequence. Conventional TTS systems generally consist of two parts; an acoustic model and a vocoder. First, acoustic models (Wang et al., 2017; Shen et al., 2018) have shown the success of synthesizing acoustic features (e.g., Mel-spectrogram) as an intermediate feature from

*Corresponding author

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

text sequence, and the vocoder (Oord et al., 2016) converts the acoustic features into raw waveforms consecutively.

However, previous TTS models are subject to two limitations: 1) although speech consists of various attributes (e.g., pronunciation, rhythm, intonation, and timbre) (Qian et al., 2020; Choi et al., 2021), most previous models synthesize acoustic features from the text sequence at once (Ren et al., 2019), which exacerbates the one-to-many mapping problem; and 2) in the two-stage pipeline, each part of the TTS system should be trained independently, which results in the degradation of the audio quality (Ren et al., 2021a,b; Lee et al., 2021b).

Recently, single-stage end-to-end TTS models, which directly generate a raw waveform from text, successfully reduce these limitations of the two-stage pipeline. For instance, VITS (Kim et al., 2021) adopts variational inference augmented with the normalizing flow (Kim et al., 2020) and adversarial training (Kong et al., 2020) to improve the expressiveness of the model, which can learn rich representations from speech data and synthesize waveforms directly from the text. However, despite efforts to reduce the information gap between text and speech, these models are subject to speech mispronunciation and over-smoothing problems. In the process of synthesizing speech, they still generate all the acoustic attributes from text sequence at the same time. Therefore, missing the details of some attributes between text and speech, specifically linguistic information, is inevitable.

To bridge the information gap between text and speech, we adopt self-supervised speech representations as additional linguistic representations. Trained with large-scale speech dataset, these representations can learn useful information without using labeled data. Previous studies (Shah et al., 2021; Choi et al., 2021) also reveal that the representations from the pre-trained model contain rich information trained from a large-scale speech dataset. In particular, the representations from the middle layer of the pre-trained model contain rich linguistic information which has a characteristic of pronunciation. As a result, it has been successfully utilized for various speech tasks such as speech recognition (Baevski et al., 2020, 2021), voice conversion (Choi et al., 2021; Lee et al., 2021a), and speech resynthesis (Polyak et al., 2021). However, these useful representations have not yet received significant attention in TTS systems due to the difficulty to utilize in generative model.

In this paper, we present HierSpeech, which is a hierarchical conditional variational autoencoder using self-supervised speech representations for end-to-end TTS. We leverage self-supervised speech representations (Baevski et al., 2020) to enrich the linguistic information in latent representations, and to learn each attribute hierarchically from linguistic representations to acoustic representations. Although the self-supervised representations of text were previously studied for TTS (Jia et al., 2021), to the best of our knowledge, this is the first study that involves the investigation of self-supervised speech representations for single-stage end-to-end TTS. Based on the state-of-the-art TTS model (Kim et al., 2021), we demonstrate that self-supervised speech representations can reduce the information gap between text and speech by significantly improving the reconstruction quality. Compared with the state-of-the-art TTS model, HierSpeech achieves +0.303 comparative mean opinion score (CMOS) and reduces the phoneme error rate from 9.16 to 5.78 for the VCTK dataset.

Based on the pre-trained HierSpeech, we also present novel adaptive TTS frameworks. Specifically, we extend HierSpeech to HierSpeech-U, which can adapt the pre-trained model to synthesize the voice of novel speakers with untranscribed speech data. To extract the linguistic representations without text transcripts, HierSpeech-U can utilize the self-supervised speech representations, and learn the acoustic representations from untranscribed speech data. The results show that the synthetic quality of HierSpeech-U trained only with the speech data is comparable to that of HierSpeech trained with the text-speech pairs. The main contributions of this paper are as follows:

- We propose HierSpeech, which is a hierarchical conditional variational autoencoder using self-supervised representations that can improve the linguistic information in the latent representations, and learn attributes hierarchically. This significantly improves the reconstruction quality by bridging the gap between text and speech.
- We investigate self-supervised speech representations for the TTS system by thoroughly conducting more than 30,000 GPU hours of experiments. Audio samples are available at `https://sh-lee-prml.github.io/hierspeech-demo/`
- To utilize untranscribed speech data, we extend the model to HierSpeech-U which can adapt the TTS model without text transcripts. The results also reveal that the adaptation quality without text transcripts is comparable to that of the baseline model using text transcripts.

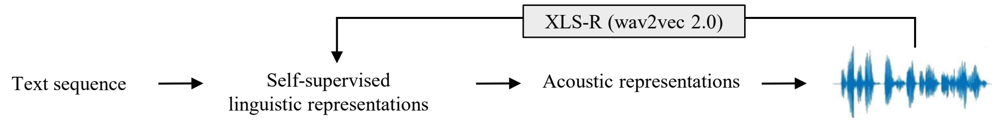


Figure 1: Hierarchical text-to-speech pipeline.

2 HierSpeech

In this paper, we propose the hierarchical conditional variational autoencoder using self-supervised representations for TTS. We use the self-supervised speech representations to improve the linguistic information in the latent representations, and to learn each representation hierarchically. Moreover, we extend HierSpeech to HierSpeech-U, which can adapt the model without text transcripts. The details of the speech representations, architecture, and untranscribed TTS method are described in the following subsections.

2.1 Speech representations

Acoustic representations In TTS systems, Mel-spectrogram is widely used as an intermediate acoustic feature, which is converted from the waveform using the short-time Fourier transform (STFT). However, this acoustic feature consists of various attributes such as linguistic information (e.g., pronunciation) and style information (e.g., rhythm, intonation, and timbre). Hence, synthesizing this rich feature from only the text simultaneously exacerbates the one-to-many mapping problem, and it is difficult to extract the expressive linguistic information from the spectrogram. To mitigate this issue, we adopt additional linguistic features to map the text and acoustic features as follows.

Linguistic representations To bridge the gap between text and speech, we use self-supervised speech representations for additional intermediate linguistic features as shown in Fig. 1. Previous studies reveal that the extracted features from the pre-trained model, such as wav2vec 2.0, contain rich linguistic information, and this can improve various tasks such as automatic speech recognition (ASR) and speech translation. (Shah et al., 2021; Choi et al., 2021) also shows the representations from the middle layer of wav2vec 2.0 contain a large proportion of linguistic information relevant to pronunciation. Specifically, we use the self-supervised speech representations from the 12th layer of the XLS-R (Babu et al., 2021) which is a pre-trained wav2vec 2.0 with a large-scale cross-lingual speech dataset. In addition, we also conduct various experiments to investigate these representations for the TTS system as detailed in Section 3.3 and Appendix C.

2.2 Hierarchical variational inference

To connect the two parts of TTS systems, the previous end-to-end TTS model, VITS (Kim et al., 2021) adopts a conditional variational autoencoder that maximizes the evidence lower bound (ELBO) over the intractable marginal log-likelihood of data $\log p_\theta(x|c)$:

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

where $p_\theta(z|c)$ is a prior distribution over latent variables z given condition c, $p_\theta(x|z)$ is the likelihood function that generates data x given latent variables z as a decoder, and $q_\phi(z|x)$ is the approximate posterior. Subsequently, VITS uses the normalizing flow to improve the expressiveness of the prior distribution and adversarial training in the waveform domain. Based on VITS, HierSpeech uses a hierarchical conditional variational autoencoder to connect multi-level intermediate representations via disentangled latent variables of speech representations, and learns them in an end-to-end manner. Unlike recently proposed hierarchical VAE (Vahdat and Kautz, 2020; Lee et al., 2022a) which uses top-down path networks conditioning each other, we approximate them separately from different representations of speech. As shown in Fig 2, the acoustic posterior distribution and linguistic posterior distribution are encoded separately by acoustic encoder ϕ_a and linguistic encoder ϕ_l respectively. To disentangle each latent variable, we use the linear-scale spectrogram of the target speech x_{spec} for the rich acoustic representations z_a, and the output of the 12th layer of XLS-R x_{w2v} for the rich linguistic representations z_l. The optimization objective of HierSpeech can be expressed

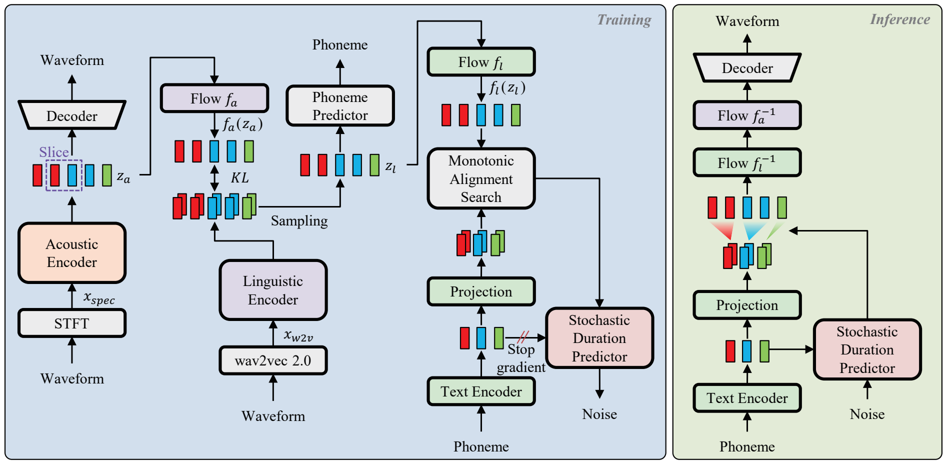


Figure 2: Overall framework of HierSpeech.

as follows:

$$\log p_\theta(x|c) \geq \mathbb{E}_{q_\phi(z|x)}\Big[\log p_{\theta_d}(x|z_a) - \log \frac{q_{\phi_a}(z_a|x_{spec})}{p_{\theta_a}(z_a|z_l)} - \log \frac{q_{\phi_l}(z_l|x_{w2v})}{p_{\theta_l}(z_l|c)}\Big] \tag{2}$$

where $z = [z_a, z_l]$, $\theta = [\theta_d, \theta_a, \theta_l]$, $\phi = [\phi_a, \phi_l]$, $q_{\phi_a}(z_a|x_{spec})$ and $q_{\phi_l}(z_l|x_{w2v})$ are the approximate posterior for acoustic and linguistic representation, and $p_{\theta_l}(z_l|c)$ is a prior distribution over linguistic latent variables z_l given condition c, $p_{\theta_a}(z_a|z_l)$ is a prior distribution over acoustic latent variables z_a, where z_l is sampled from $q_{\phi_l}(z_l|x_{w2v})$, and $p_{\theta_d}(x|z_a)$ is the likelihood function that generates data x given latent variables z_a as a decoder θ_d. For the reconstruction loss, we use Mel-reconstruction loss $\mathcal{L}_{rec}$ which minimizes the $l1$ distance of the Mel-spectrogram between the ground truth and reconstructed waveform using STFT and Mel-scale transformation.

Acoustic encoder and waveform decoder The acoustic encoder ϕ_a is composed of non-casual WaveNet residual blocks which are layers of dilated convolutions with a gated activation unit and skip connection. Thereafter, the output is then fed to the projection layer to sample the acoustic representations z_a from the mean and variance of the posterior distribution using the reparametrization trick. During training, the sliced z_a is fed to a waveform decoder to reconstruct the raw audio x. We use HiFi-GAN generator G (Kong et al., 2020) as waveform decoder θ_d which consists of a stack of transposed convolution and multi-receptive field fusion module. For adversarial feedback, we also use the multi-period discriminator D to capture the different periodic features of the waveform.

$$\mathcal{L}_{adv}(D) = \mathbb{E}_{(x,z_a)}\Big[(D(x)-1)^2 + D(G(z_a))^2\Big], \tag{3}$$

$$\mathcal{L}_{adv}(\phi_a, \theta_d) = \mathbb{E}_{(z_a)}\Big[(D(G(z_a))-1)^2\Big] \tag{4}$$

where x is the ground truth waveform. To ensure stable training, we use the additional feature matching loss $\mathcal{L}_{fm}$ which minimizes the $l1$ distance of each discriminator's features between the ground truth and reconstructed waveform.

Linguistic encoder and phoneme predictor The linguistic encoder ϕ_l has the same structure as the acoustic encoder. However, we use self-supervised speech representations x_{w2v} as the input of the linguistic encoder which is extracted from the pre-trained XLS-R, and the linguistic representations z_l is extracted. To enforce linguistic characteristics, z_l is fed to the auxiliary phoneme predictor. We minimize the connectionist temporal classification (CTC) loss $\mathcal{L}_{ctc}$ to optimize the linguistic encoder and phoneme predictor. However, to remove the additional projection layer, the projected mean and variance from the linguistic encoder are also directly used as the acoustic prior distribution with weight-sharing between θ_a and ϕ_l. To maintain hierarchy, the representation z_a is transformed by normalizing flow. Hence, the KL divergence between acoustic prior and posterior is minimized as:

$$\mathcal{L}_{kl1} = \log q_{\phi_a}(z_a|x_{spec}) - \log p_{\theta_a}(z_a|x_{w2v}) \tag{5}$$

Because the acoustic prior distribution is obtained from the linguistic information, to bridge the gap between distributions, we use the normalizing flow to disentangle the information from the acoustic posterior and increase the expressiveness of the acoustic prior distribution.

$$\begin{aligned} &p_{\theta_a}(z_a|x_{w2v}) = \mathcal{N}(f_a(z_a); \mu_{\theta_a}(x_{w2v}), \sigma_{\theta_a}(x_{w2v}))|\det(f_a(z_a)/z_a)|, \\ &z_a \sim q_{\phi_a}(z_a|x_{spec}) = \mathcal{N}(z_a; \mu_{\phi_a}(x_{spec}), \sigma_{\phi_a}(x_{spec})) \end{aligned} \tag{6}$$

Text encoder The text encoder θ_l consists of a stacked feed-forward Transformer network using a relative positional representation. The phoneme sequence c_{text} is fed to the text encoder, and the projection layer is used to produce the mean and variance for linguistic prior distribution. To align the text with the linguistic representations of speech, we use monotonic alignment search (MAS) of (Kim et al., 2020) which searches the alignment A satisfied with maximizing the likelihood of data:

$$\mathcal{L}_{kl2} = \log q_{\phi_l}(z_l|x_{w2v}) - \log p_\theta(z_l|c_{text}, A) \tag{7}$$

We also use the normalizing flow to increase the expressiveness of the linguistic prior distribution.

$$\begin{aligned} &p_{\theta_l}(z_l|c_{text}, A) = \mathcal{N}(f_l(z_l); \mu_{\theta_l}(c_{text}, A), \sigma_{\theta_l}(c_{text}, A))|\det(f_l(z_l)/z_l)|, \\ &z_l \sim q_{\phi_l}(z_l|x_{w2v}) = \mathcal{N}(z_l; \mu_{\phi_l}(x_{w2v}), \sigma_{\phi_l}(x_{w2v})) \end{aligned} \tag{8}$$

To sample the duration given phonemes, we adopt a flow-based stochastic duration predictor of (Kim et al., 2021) trained with maximum likelihood estimation. We use the negative variational lower bound of it as duration loss $\mathcal{L}_{dur}$. For the multi-speaker settings, we add the global speaker embedding to the residual block of the acoustic/linguistic encoder, residual block in normalizing flow, stochastic duration predictor, and decoder.

Total loss The final objectives for HierSpeech can be expressed as follows:

$$\mathcal{L}_{total} = \mathcal{L}_{kl1} + \lambda_{kl2}\mathcal{L}_{kl2} + \lambda_{rec}\mathcal{L}_{rec} + \lambda_{ctc}\mathcal{L}_{ctc} + \lambda_{dur}\mathcal{L}_{dur} + \lambda_{adv}\mathcal{L}_{adv}(\phi_a, \theta_d) + \lambda_{fm}\mathcal{L}_{fm} \tag{9}$$

2.3 Untranscribed text-to-speech

For the untranscribed TTS model (HierSpeech-U), we train the model using a style encoder (Min et al., 2021) which extracts style embedding from speech as global conditioning (Jia et al., 2018). We use a linear-scale spectrogram as the input of the style encoder. After pre-training with the multi-speaker dataset, we adapt the model to a novel speaker without text transcripts. Through self-supervised speech representations, the pre-trained linguistic encoder is able to extract rich linguistic representations from speech without text transcripts. Hence, HierSpeech-U synthesizes speech with the style of a novel speaker by fine-tuning the acoustic encoder, the normalizing flow blocks of the acoustic prior, and decoder with only speech data.

3 Experiment and Result

3.1 Experimental setup

Datasets We train the models using the VCTK and LibriTTS datasets. The VCTK dataset contains 46 hours of audio for 108 speakers (Veaux et al., 2017). We use the *train-clean* subsets of the LibriTTS dataset (Zen et al., 2019) which contains 110 hours of audio for 1,151 speakers. For both datasets, we downsample the audio at 22,050Hz. Following the (Lee et al., 2021a), we use only non-parallel data for the training dataset which consists of different utterances for each speaker, and we use the parallel data for the test dataset which consists of 25 same utterances in the VCTK dataset. For the LibriTTS dataset, we randomly select two samples from each speaker for the test dataset. For speaker adaptation, we randomly select 98 speakers as the base speakers and 10 speakers (5 males and 5 females) as the novel speakers. To evaluate the speaker adaptation performance for the number of speakers, we also train the model using 1,249 speakers from the VCTK and LibriTTS datasets.

Preprocessing For the input of the acoustic and speaker encoder, we use linear-scale spectrogram with 513 bins which is transformed from the audio. For reconstruction loss, we use a Mel-spectrogram with 80 bins. For the input of the linguistic encoder, we use the output from the middle layer of the XLS-R (0.3B) (Babu et al., 2021) which is pre-trained wav2vec 2.0 with a large-scale cross-lingual speech dataset.[2] To extract the self-supervised speech representations, we use the downsampled audio at 16,000Hz as an input to the XLS-R, and the extracted representations are upsampled to map the spectrogram by interpolation.[3] For the input of the text encoder, we use the text sequences converted to International Phonetic Alphabet (IPA) sequences using the open-source Phonemizer.[4] Following (Kim et al., 2021), we intersperse the phoneme sequences with a blank token. However, we use phoneme sequences without a blank token for the target sequence of the phoneme predictor.[5]

Training We train HierSpeech using the AdamW optimizer (Loshchilov and Hutter, 2019) with $\beta_1 = 0.8$, $\beta_2 = 0.99$, and weight decay $\lambda = 0.01$, and apply the learning rate schedule by the decay of $0.999^{1/8}$ at an initial learning rate of 2×10^{-4} for the generator and discriminator. We train HierSpeech with a batch size of 256 for 600k steps using mixed precision training on four NVIDIA A100 GPUs. For speaker adaptation, we fine-tune HierSpeech/HierSpeech-U with few samples of novel speakers for only 2000 steps. For the ablation study, we train the model with a batch size of 64 for 300k steps.

3.2 Evaluation metrics

Subjective metrics We conduct two mean opinion score (MOS) tests for naturalness and similarity. For the naturalness MOS (nMOS), each sample from the target and synthesized speech are rated by at minimum of 20 listeners on a scale of 1-5. For the similarity MOS (sMOS), the synthesized and target speech are presented to a minimum of 20 listeners on a scale of 1-4. The nMOS and sMOS are reported with 95% confidence intervals.

Objective metrics For the naturalness measurement, we calculate the phoneme error rate (PER) and word error rate (WER) of the synthesized speech. We used the fined-tuned wav2vec 2.0 (Baevski et al., 2020) without a language model for automatic speech recognition. For the similarity measurement, we conduct three metrics: the equal error rate (EER) of the automatic speaker verification (ASV), the Mel-cepstral distortion (MCD), and the F0 root mean square error (RMSE_{f0}). We used a pre-trained ASV model (Chung et al., 2020). The ASV model is trained using a large-scale speech dataset, VoxCeleb2 (Chung et al., 2018) via online data augmentation (Heo et al., 2020). We calculate the EER where both acceptance and rejection rates are equal for the sample pairs from the synthesized and target speech ($2,700 \times 108 = 291,600$). We compute the MCD and RMSE_{f0} by applying dynamic time warping (DTW) between the synthesized speech and target speech. We conducted a duration prediction performance evaluation using the average absolute differences of the utterance duration (DDUR) (Zhang et al., 2019). To compare the inference speeds, we calculate the Speed which is the synthetic waveform frame per second, and the synthesis speed over real-time (Real-time).

3.3 Analysis of self-supervised representations

Previous researches (Fan et al., 2020; Choi et al., 2021) reveal that the representations from the front layer of wav2vec 2.0 are clustered by each speaker. Hence, (Choi et al., 2021) uses the 1st layer of pre-trained wav2vec 2.0 as an input to the speaker encoder. As an input of the speaker encoder, we compared three different features; linear-scale spectrogram (linear spectrogram), the representations from the 1st layer of XLS-R and 12th layer of XLS-R. Table 1 shows that the linear spectrogram has better transfer performance than the representations of XLS-R. Although the speaker embedding extracted from the 1st layer of the XLS-R has better performance than one of the 12th layer, the results show that the linear spectrogram contains a significant amount of high-resolution information for speaker characteristics. In this respect, we use linear spectrogram as the input for the speaker encoder, and we can conclude that there is a little speaker information to transfer speaking style in the representations of the middle layer of XLS-R.

Table 1: Results of different input features for the speaker encoder.

Input feature	PER	WER	EER
Linear spectrogram	5.45	20.13	2.77
1st layer of XLS-R	5.67	20.36	4.62
12th layer of XLS-R	6.29	21.50	10.23

[2] Specifically, we use the 12th layer of the XLS-R. The results are detailed in Section 3.3 and Appendix C.

[3] We simply interpolate with respect to the time-domain. The ASR model uses the audio with a sampling rate of 16,000 Hz. However, the TTS model use higher resolution audio for high-quality audio synthesis.

[4] `https://github.com/bootphon/phonemizer`

[5] We observe that the blank token degraded the classification performance under the experimental conditions.

To evaluate the speech disentanglement performance with respect to the layer of XLS-R, we conduct frame-level speaker classification on the representations from a specific layer of the XLS-R and the linguistic latent variables z_l respectively. Table 2 shows that the representations from each layer contain some speaker information to be classified. However, the speaker information is reduced through the linguistic encoder, and the z_l from the 12th layer of XLS-R has the lowest accuracy. Hence, we use representations from the 12th layer for the linguistic encoder. It should be noted that all representations, with the exception of the 23th layer, improve the TTS performance of HierSpeech when compared with that of VITS, which means that all representations contain rich information trained with large-scale speech dataset.

Table 2: Speaker classification accuracy on linguistic representations from the different layer of XLS-R.

layer	Accuracy (w2v)	Accuracy (z_l)
1st layer	79.12%	15.09%
12th layer	59.46%	8.95%
24th layer	48.73%	11.35%

3.4 Evaluation on TTS

Table 3: The TTS evaluation results on the VCTK dataset.

Method	nMOS	sMOS	PER	WER	EER	MCD	RMSE$_{f0}$	DDUR	Speed (kHz)	Real-time
GT	4.06±0.02	3.34±0.03	5.64	18.94	4.03	-	-	-	-	-
GT (HiFi-GAN)	4.03±0.02	3.30±0.03	5.94	19.52	5.04	1.25	28.32	-	6,484.09	×294.06
Tacotron2	3.76±0.02	3.16±0.03	11.73	22.48	9.11	4.18	35.30	0.49	263.94	×11.97
Glow-TTS	3.95±0.02	3.09±0.03	11.77	26.40	5.33	4.31	32.98	0.38	1,410.75	×63.97
PortaSpeech	3.97±0.02	3.15±0.03	11.35	25.46	5.48	4.34	32.89	0.43	1,163.21	×52.75
VITS	4.02±0.02	3.19±0.03	9.16	25.54	3.83	4.27	32.93	0.37	**1,610.77**	×**72.83**
HierSpeech (Ours)	**4.04±0.02**	**3.22±0.03**	**5.78**	**19.55**	**3.74**	**4.05**	**32.15**	**0.33**	1,459.95	×66.21

Table 4: The speaker transfer evaluation results on the LibriTTS dataset.

Method	nMOS	sMOS	PER	WER	EER	MCD	RMSE$_{f0}$	DDUR	Speed (kHz)	Real-time
GT	4.04±0.03	3.40±0.03	7.01	18.28	4.45	-	-	-	-	-
VITS	3.96±0.03	3.26±0.03	13.62	29.83	5.00	**4.37**	34.18	1.09	**1,781.40**	×**80.78**
HierSpeech (Ours)	**3.98±0.03**	3.26±0.03	**7.47**	**20.34**	5.00	4.42	**32.95**	**0.72**	1,678.79	×76.13

Table 3 shows that our model outperforms the other models with respect to the nMOS and sMOS for both datasets. In terms of the ASR evaluation, our model shows a lower PER and WER than the other models by synthesizing speech with more accurate pronunciation. In terms of EER, all models have similar performance, which indicates the target speaker embedding (ID) is useful supervision for multi-speaker TTS. For the speaker transfer, we evaluate each model using the same speaker encoder in Table 4 and subsection 3.5. In terms of the MCD and RMSE$_{f0}$, our model has the lowest error distance. Although VITS has better performance in inference speed, our model has a faster inference speed than two-stage end-to-end TTS models. We also compare the models using speaker encoder, which are trained with VCTK and LibriTTS dataset in Table 4. Our model outperforms VITS in terms of nMOS and ASR evaluation. However, our model has a slightly lower transfer performance in MCD. We found that transferring the speaker from the long sentence or noisy audio results in low performance of ASR for VITS and low speaker transfer quality for HierSpeech. We also conduct the comparative mean opinion score (CMOS) tests between the models trained with each dataset; VCTK and LibriTTS. Table 5 also shows that our model has a better performance than VITS in CMOS evaluation with t-test p-values. Also, our model achieves -0.096 CMOS compared to ground truth (GT) audio on the VCTK dataset. However, Table 5b shows that our model has -0.505 CMOS compared to GT, and it means that there is a room for improvement in TTS system by improving the expressiveness and robustness of the model.

Table 5: CMOS comparison. Positive score indicates that HierSpeech is rated better than the baseline.

(a) The CMOS results on the VCTK dataset.

Method	CMOS	p-value
HierSpeech (Ours)	0	-
GT	-0.096	0.003
VITS	+0.303	$<10^{-24}$

(b) The CMOS results on the LibriTTS dataset.

Method	CMOS	p-value
HierSpeech (Ours)	0	-
GT	-0.505	$<10^{-104}$
VITS	+0.297	$<10^{-25}$

Table 6: Results for untranscribed text-to-speech. We compare few-shot speaker adaptation performance of HierSpeech-U with that of HierSpeech. Both models use the pre-trained HierSpeech which is trained using VCTK and LibriTTS datasets. We used 10 unseen speakers of VCTK dataset as novel speakers, and fine-tuned each model with 20 samples from each speaker.

Method	Transcript	nMOS	sMOS	PER	WER	EER	MCD	RMSE$_{f0}$	DDUR
GT	-	4.13±0.10	3.38±0.10	4.26	16.69	4.14	-	-	-
HierSpeech	✓	4.09±0.10	3.18±0.11	4.40	16.95	6.40	3.96	29.56	0.28
HierSpeech-U	✗	4.08±0.09	3.15±0.12	3.71	15.85	6.40	4.09	30.64	0.36

3.5 Untranscribed text-to-speech

To evaluate the untranscribed text-to-speech performance of HierSpeech-U, we compare the few-shot speaker adaptation performance of HierSpeech-U with one of HierSpeech which is fine-tuned by unseen speakers using few audio-text pairs (Arik et al., 2018). Both models use the pre-trained HierSpeech which is trained using VCTK and LibriTTS datasets. For the speaker adaptation of HierSpeech, we fine-tune the entire model. Although HierSpeech-U is fine-tuned without text transcripts, Table 6 shows that HierSpeech-U has comparable performance to HierSpeech in terms of nMOS and sMOS. Moreover, HierSpeech-U achieves a lower PER and WER than that of HierSpeech, which is slightly increased compared to PER 3.58 and WER 15.50 for the pre-trained model. Although there is a limitation in adapting duration, HierSpeech-U is able to adapt the speaker in terms of voice by utilizing the linguistic representations from the self-supervised speech representations without text transcripts. Note that we fail to fine-tune VITS without text transcripts in that the PER for VITS increases from 7.47 to 12.27. Furthermore, we evaluate the adaptation quality with different numbers of adaptation samples (1, 5, 10, and 20) in Figure 3. We also investigate the effectiveness for different numbers of pre-trained speakers (98 from VCTK and 1,151 from LibriTTS) and the evaluation results are described in Appendix C.

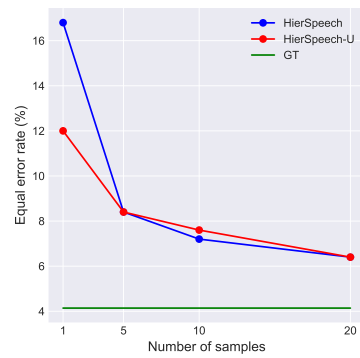


Figure 3: The EER results for different numbers of adaptation samples.

3.6 Ablation study and hyperparameter search

To compare the effectiveness for the number of normalizing flow, we train VITS which has the same number of flow blocks ($4 \rightarrow 8$) with HierSpeech. However, we observe that increasing the number of flow blocks does not always indicate an improvement in model performance, resulting in the degradation of the pronunciation and audio quality as shown in Table 7. Adding a phoneme predictor (PP) to the posterior encoder of VITS improves the PER and WER by guiding the alignment search. Removing the acoustic encoder (AE) and synthesizing waveform directly from the linguistic representations from the 12th layer of XLS-R degrade the performance, and it also has higher Mel-spectrogram reconstruction error.

Table 7: Results of the ablation study.

Method	PER	WER	EER	MCD
VITS	12.24	30.62	3.85	4.36
w flow 8	13.42	32.77	4.00	4.36
w PP	7.60	22.98	3.74	4.17
HierSpeech	**6.25**	**20.89**	**3.48**	**4.15**
w.o AE	8.00	23.84	3.63	4.25
w.o AE and PP	16.70	37.43	3.89	4.39

We conduct a hyperparameter search and ablation study of the phoneme predictor in Appendix C. The proper value of $\lambda_{ctc} = 45$ increases the overall performance in all the objective metrics. In contrast, when using an excessively large λ_{ctc}, the KL divergence between the acoustic and linguistic distribution increases, which decreases the performance of audio quality with noisy sound. For better generalization, we attempted to combine label smoothing with CTC loss (Kim et al., 2018) and conduct experiments on data augmentation for the effective speech disentanglement of linguistic information following (Choi et al., 2021). However, label smoothing and data augmentation lead to the problem for KL divergence optimization, thus resulting in the degradation of the synthesis quality.

3.7 Evaluation on VC

To evaluate speech disentanglement, we compared the voice conversion (VC) task of our model with three models; AutoVC (Qian et al., 2019), VoiceMixer (Lee et al., 2021a), and VITS (Kim et al., 2021). For a fair comparison, we train all models using the same dataset. However, note that AutoVC and VoiceMixer do not use the phoneme information. Following (Lee et al., 2021a), we randomly select 20 speakers. Thereafter, a single audio sample is selected from each speaker, and then all the possible pairs of samples ($20 \times 20 = 400$) are produced for evaluation. Table 8 shows that our model outperforms the other models in nMOS and ASR evaluation metrics. Although VITS has similar performance in the EER evaluation, our model has better performance in ASR evaluation, which means that our model can disentangle content and style information with a small loss of content information. We also observe that using the representations from the middle layer of XLS-R improves the PER performance when compared with that of the previous layer, and the auxiliary phoneme predictor also helps retain the linguistic information.

Table 8: The VC evaluation results on the VCTK.

Model	nMOS	sMOS	PER	WER	EER
GT	4.19±0.03	3.37±0.04	5.20	14.05	0.98
AutoVC	3.74±0.05	3.04±0.05	59.42	89.01	12.50
VoiceMixer	4.00±0.05	3.11±0.05	12.67	31.80	9.35
VITS	4.06±0.04	3.16±0.05	6.46	18.73	**2.75**
HierSpeech	**4.08±0.04**	3.16±0.05	**5.51**	**16.25**	**2.75**

4 Broader Impact

Recently, self-supervised speech representations have been utilized in TTS tasks (Siuzdak et al., 2022; Kim et al., 2022; Du et al., 2022) and we see that our hierarchical speech synthesis structure using self-supervised speech representations can be also utilized for various tasks. First, the application of a low-resource language can utilize self-supervised speech representations to improve the synthesis quality. Second, cross-lingual speaker adaptation for dubbing can be used in the film industry. However, as recent speech synthesis systems such as TTS and VC can generate the audio with realistic sound, there is an increased potential risk of harm, malicious use, and ethical issues. Specifically, these systems could be misused in various manners, such as fake news generation, voice spoofing, and unauthorized use of web crawl speech data. To mitigate these issues, fake audio detection is studied (Singhal et al., 2019; Tak et al., 2021). Moreover, we provide a mitigation strategy for the proposed system by releasing the fake audio detection model with an ensemble of discriminators that can distinguish real and fake audio based on results of multiple discriminators. Although it is difficult to differentiate in human evaluation score, the model achieves a 16.59% EER for fake audio detection on 1,852 test samples of the ground-truth and synthesized speech from our model.

5 Conclusion

We presented an end-to-end TTS model, HierSpeech, which can learn and synthesize speech from text through hierarchical intermediate representations in an end-to-end manner. By bridging the gap between text and speech through self-supervised speech representations, the proposed model significantly improved the reconstruction quality. We successfully demonstrated that our hierarchical conditional variational autoencoder can improve linguistic capability in latent representations, and learn each attribute hierarchically using self-supervised speech representations. We thoroughly conducted more than 30,000 GPU hours of experiments on self-supervised representations for the TTS system, and we hope that these results can serve as a basis for future speech research. Furthermore, we also demonstrate the effectiveness of a novel speaker adaptation framework without text transcripts. We see our hierarchical structure extending to cross-lingual TTS systems or other low-resource TTS systems. Also, we will try to improve the expressive and robustness of model by modeling prosody (Im et al., 2022) and noise from speech (Saeki et al., 2022). However, the single-stage end-to-end TTS model is limited in terms of computational complexity in that the training process requires 20 days using four A100 GPUs. Hence, in future works, an attempt will be made to decrease the computational cost without quality degradation by adopting iDWT (Lee et al., 2022b) in the decoder and simplifying the discriminator to facilitate more rapid training (Andreev et al., 2022), and replace the decoder with an diffusion-based neural vocoder (Koizumi et al., 2022b,a). In addition, we discussed the potential positive and negative impact of our model in Section 4. To mitigate the negative impact, we will release fake audio detector that can distinguish between real and fake audio.

Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2019-0-00079, Artificial Intelligence Graduate School Program(Korea University), No. 2019-0-01371, Development of Brain-inspired AI with Human-like Intelligence, No. 2021-0-02068, Artificial Intelligence Innovation Hub, and No. 2022-0-00984, Development of Artificial Intelligence Technology for Personalized Plug-and-Play Explanation and Verification of Explanation) and Clova Voice, NAVER Corp., Seongnam, Korea.

References

Pavel Andreev, Aibek Alanov, Oleg Ivanov, and Dmitry Vetrov. Hifi++: a unified framework for neural vocoding, bandwidth extension and speech enhancement. *arXiv preprint arXiv:2203.13086*, 2022.

Sercan Arik, Jitong Chen, Kainan Peng, Wei Ping, and Yanqi Zhou. Neural voice cloning with a few samples. *Advances in Neural Information Processing Systems*, 31, 2018.

Arun Babu, Changhan Wang, Andros Tjandra, Kushal Lakhotia, Qiantong Xu, Naman Goyal, Kritika Singh, Patrick von Platen, Yatharth Saraf, Juan Pino, et al. Xls-r: Self-supervised cross-lingual speech representation learning at scale. *arXiv preprint arXiv:2111.09296*, 2021.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33, 2020.

Alexei Baevski, Wei-Ning Hsu, Alexis Conneau, and Michael Auli. Unsupervised speech recognition. *Advances in Neural Information Processing Systems*, 34, 2021.

Hyeong-Seok Choi, Juheon Lee, Wansoo Kim, Jie Lee, Hoon Heo, and Kyogu Lee. Neural analysis and synthesis: Reconstructing speech from self-supervised representations. *Advances in Neural Information Processing Systems*, 34, 2021.

Joon Son Chung, Arsha Nagrani, and Andrew Zisserman. Voxceleb2: Deep speaker recognition. In *Proc. Interspeech 2018*, pages 1086–1090, 2018.

Joon Son Chung, Jaesung Huh, Seongkyu Mun, Minjae Lee, Hee Soo Heo, Soyeon Choe, Chiheon Ham, Sunghwan Jung, Bong-Jin Lee, and Icksang Han. In defence of metric learning for speaker recognition. In *Proc. Interspeech 2020*, pages 2977–2981, 2020.

Chenpeng Du, Yiwei Guo, Xie Chen, and Kai Yu. Vqtts: High-fidelity text-to-speech synthesis with self-supervised vq acoustic feature. *arXiv preprint arXiv:2204.00768*, 2022.

Zhiyun Fan, Meng Li, Shiyu Zhou, and Bo Xu. Exploring wav2vec 2.0 on speaker verification and language identification. *arXiv preprint arXiv:2012.06185*, 2020.

Hee Soo Heo, Bong-Jin Lee, Jaesung Huh, and Joon Son Chung. Clova baseline system for the voxceleb speaker recognition challenge 2020. *arXiv preprint arXiv:2009.14153*, 2020.

Chae-Bin Im, Sang-Hoon Lee, Seung-Bin Kim, and Seong-Whan Lee. Emoq-tts: Emotion intensity quantization for fine-grained controllable emotional text-to-speech. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6317–6321. IEEE, 2022.

Ye Jia, Yu Zhang, Ron Weiss, Quan Wang, Jonathan Shen, Fei Ren, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, Yonghui Wu, et al. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. *Advances in Neural Information Processing Systems*, 31, 2018.

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

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*, 33:8067–8077, 2020.

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.

Minchan Kim, Myeonghun Jeong, Byoung Jin Choi, Sunghwan Ahn, Joun Yeop Lee, and Nam Soo Kim. Transfer learning framework for low-resource text-to-speech using a large-scale unlabeled speech corpus. In *Proc. Interspeech 2022*, pages 788–792, 2022. doi: 10.21437/Interspeech.2022-225.

Suyoun Kim, Michael Seltzer, Jinyu Li, and Rui Zhao. Improved training for online end-to-end speech recognition systems. pages 2913–2917, 2018.

Yuma Koizumi, Kohei Yatabe, Heiga Zen, and Michiel Bacchiani. Wavefit: An iterative and non-autoregressive neural vocoder based on fixed-point iteration. *arXiv preprint arXiv:2210.01029*, 2022a.

Yuma Koizumi, Heiga Zen, Kohei Yatabe, Nanxin Chen, and Michiel Bacchiani. Specgrad: Diffusion probabilistic model based neural vocoder with adaptive noise spectral shaping. In *Proc. Interspeech 2022*, pages 803–807, 2022b. doi: 10.21437/Interspeech.2022-301.

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

Ji-Hyun Lee, Sang-Hoon Lee, Ji-Hoon Kim, and Seong-Whan Lee. Pvae-tts: Adaptive text-to-speech via progressive style adaptation. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6312–6316. IEEE, 2022a.

Sang-Hoon Lee, Ji-Hoon Kim, Hyunseung Chung, and Seong-Whan Lee. Voicemixer: Adversarial voice style mixup. *Advances in Neural Information Processing Systems*, 34, 2021a.

Sang-Hoon Lee, Ji-Hoon Kim, Kang-Eun Lee, and Seong-Whan Lee. Fre-gan 2: Fast and efficient frequency-consistent audio synthesis. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022b.

Yoonhyung Lee, Joongbo Shin, and Kyomin Jung. Bidirectional variational inference for non-autoregressive text-to-speech. In *International Conference on Learning Representations*, 2021b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Dongchan Min, Dong Bok Lee, Eunho Yang, and Sung Ju Hwang. Meta-stylespeech: Multi-speaker adaptive text-to-speech generation. In *International Conference on Machine Learning*, pages 7748–7759. PMLR, 2021.

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.

Adam Polyak, Yossi Adi, Jade Copet, Eugene Kharitonov, Kushal Lakhotia, Wei-Ning Hsu, Abdelrahman Mohamed, and Emmanuel Dupoux. Speech Resynthesis from Discrete Disentangled Self-Supervised Representations. In *Proc. Interspeech 2021*, 2021.

Kaizhi Qian, Yang Zhang, Shiyu Chang, Xuesong Yang, and Mark Hasegawa-Johnson. Autovc: Zero-shot voice style transfer with only autoencoder loss. In *International Conference on Machine Learning*, pages 5210–5219. PMLR, 2019.

Kaizhi Qian, Yang Zhang, Shiyu Chang, Mark Hasegawa-Johnson, and David Cox. Unsupervised speech decomposition via triple information bottleneck. In *International Conference on Machine Learning*, pages 7836–7846. PMLR, 2020.

Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech: Fast, robust and controllable text to speech. *Advances in Neural Information Processing Systems*, 32, 2019.

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*, 2021a.

Yi Ren, Jinglin Liu, and Zhou Zhao. Portaspeech: Portable and high-quality generative text-to-speech. *Advances in Neural Information Processing Systems*, 34, 2021b.

Takaaki Saeki, Kentaro Tachibana, and Ryuichi Yamamoto. Drspeech: Degradation-robust text-to-speech synthesis with frame-level and utterance-level acoustic representation learning. In *Proc. Interspeech 2022*, pages 793–797, 2022. doi: 10.21437/Interspeech.2022-294.

Jui Shah, Yaman Kumar Singla, Changyou Chen, and Rajiv Ratn Shah. What all do audio transformer models hear? probing acoustic representations for language delivery and its structure. *arXiv preprint arXiv:2101.00387*, 2021.

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.

Shivangi Singhal, Rajiv Ratn Shah, Tanmoy Chakraborty, Ponnurangam Kumaraguru, and Shin'ichi Satoh. Spotfake: A multi-modal framework for fake news detection. In *2019 IEEE fifth international conference on multimedia big data (BigMM)*, pages 39–47. IEEE, 2019.

Hubert Siuzdak, Piotr Dura, Pol van Rijn, and Nori Jacoby. Wavthruvec: Latent speech representation as intermediate features for neural speech synthesis. In *Proc. Interspeech 2022*, pages 833–837, 2022. doi: 10.21437/Interspeech.2022-10797.

RJ Skerry-Ryan, Eric Battenberg, Ying Xiao, Yuxuan Wang, Daisy Stanton, Joel Shor, Ron Weiss, Rob Clark, and Rif A. Saurous. Towards end-to-end prosody transfer for expressive speech synthesis with tacotron. In *International Conference on Machine Learning*, pages 4700–4709, 2018.

Hemlata Tak, Jose Patino, Massimiliano Todisco, Andreas Nautsch, Nicholas Evans, and Anthony Larcher. End-to-end anti-spoofing with rawnet2. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6369–6373. IEEE, 2021.

Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. *Advances in Neural Information Processing Systems*, 2020.

Christophe Veaux, Junichi Yamagishi, Kirsten MacDonald, et al. Superseded-cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit. 2017.

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. *arXiv preprint arXiv:1703.10135*, 2017.

Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu. Libritts: A corpus derived from librispeech for text-to-speech. pages 1526–1530, 2019.

Jing-Xuan Zhang, Zhen-Hua Ling, Li-Juan Liu, Yuan Jiang, and Li-Rong Dai. Sequence-to-sequence acoustic modeling for voice conversion. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(3):631–644, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 5
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 4
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The details is described at Secotion 3.3 and the supplemental material
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 3 and the supplemental material
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 3
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [Yes] See Appendix E
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [Yes] See Appendix E

A Implementation Details

Table 9: Hyperparameters of HierSpeech.

	Hyperparameter	HierSpeech
Text Encoder	Phoneme Embedding Layers Hidden Size Conv1D Kernel Conv1D Filter Size	192 6 192 3 768
Linguistic/Acoustic Encoder	Input Feature Input Size WaveNet Layers WaveNet Channel Size Conv1D Kernel	12th layer of XLS-R/linear spectrogram 1024/513 16 192 5
Flow (f_a and f_l)	Affine Coupling Layers Affine Coupling Dilation Affine Coupling WaveNet Layers Affine Coupling Kernel Size Affine Coupling Filter Size	4 1 4 5 192
Decoder	Initial Hidden Size MRF Kernel Size MRF Dilation Size Upsampling Rate Upsampling Transposed Conv1D Kernel Size	512 [3,7,11] [[1,3,5],[1,3,5],[1,3,5]] [8,8,2,2] [16,16,4,4]
Loss	λ_{kl2}/ λ_{rec}/λ_{ctc}/λ_{dur}/λ_{adv}/ λ_{fm}	1/45/45/1/1/2

The details of hyperparameter are described in Table 9. Moreover, we describe the objective function with the respect to the trained parameters in details as follows:

$$\mathcal{L}_{total} = \mathcal{L}_{kl1}(\phi_a, \theta_a, \theta_{f_a}) + \lambda_{kl2}\mathcal{L}_{kl2}(\phi_l, \theta_l, \theta_{f_l}) + \lambda_{rec}\mathcal{L}_{rec}(\phi_a, \theta_d) + \lambda_{ctc}\mathcal{L}_{ctc}(\phi_l, \theta_{pp}) + \lambda_{dur}\mathcal{L}_{dur}(\theta_{dp}) + \lambda_{adv}\mathcal{L}_{adv}(\phi_a, \theta_d) + \lambda_{fm}\mathcal{L}_{fm}(\phi_a, \theta_d) \tag{10}$$

where θ_{f_a} is the parameter of normalizing flow network in acoustic prior encoder, θ_{f_l} is the parameter of normalizing flow network in linguistic prior encoder, and θ_{pp} and θ_{dp} is the phoneme predictor and duration predictor.

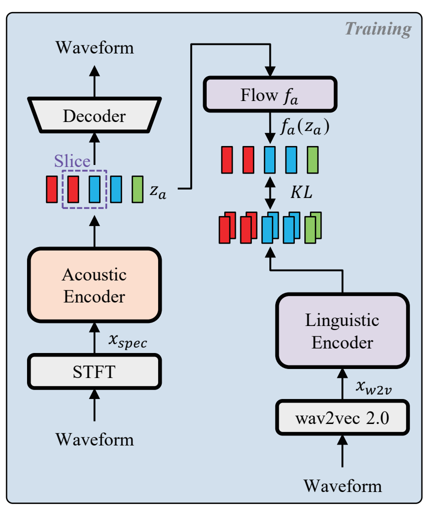


Figure 4: The training procedure of untranscribed text-to-speech.

For untranscribed text-to-speech as shown in Figure 4, we fine-tune the model without linguistic prior encoder, phoneme predictor, and duration predictor as follows objective:

$$\mathcal{L}_{finetuning} = \mathcal{L}_{kl1}(\phi_a, \theta_a, \theta_{f_a}) + \lambda_{rec}\mathcal{L}_{rec}(\phi_a, \theta_d) + \lambda_{adv}\mathcal{L}_{adv}(\phi_a, \theta_d) + \lambda_{fm}\mathcal{L}_{fm}(\phi_a, \theta_d) \tag{11}$$

B Voice Conversion

By disentangling speech representations into linguistic and acoustic representations, we assume that the linguistic representations contain a small amount of speaker information. To evaluate the speech disentanglement, we conduct the voice conversion task using the linguistic representations. In our model, we can extract the linguistic representations by four different methods.

First, we convert the speech x into a linear spectrogram x_{spec}, and we predict the linguistic representations $\hat{z}_l$ through the acoustic encoder q_{ϕ_a} and the normalizing flow in the acoustic prior encoder f_a given the source speaker information s and linear spectrogram x_{spec}. Using the predicted linguistic representations $\hat{z}_l$, we generate a converted speech x_t with a voice of target speaker information s_t through the inverse transformation of the normalizing flow f_a^{-1} and decoder G as follows:

$$\begin{aligned} &z_a \sim q_{\phi_a}(z_a|x_{spec}, s) \\ &\hat{z}_l = f_a(z_a|s) \\ &x_t = G(f_a^{-1}(\hat{z}_l|s_t)|s_t) \end{aligned} \tag{12}$$

Second, we convert the predicted linguistic representations $\hat{z}_l$ into a speaker-independent representations e through the normalizing flow in the linguistic prior encoder f_l given the source speaker information s. Using e and the target speaker information s_t, we generate a converted speech s_t through inverse transformation of the normalizing flow f_l^{-1} and f_a^{-1}, and decoder G as follows:

$$\begin{aligned} &z_a \sim q_{\phi_a}(z_a|x_{spec}, s) \\ &\hat{z}_l = f_a(z_a|s) \\ &e = f_l(\hat{z}_l|s) \\ &x_t = G(f_a^{-1}(f_l^{-1}(e|s_t)|s_t)|s_t) \end{aligned} \tag{13}$$

Third, we extract the linguistic representations z_l directly through the linguistic encoder q_{ϕ_l} given the source speaker information s and the representations from the XLS-R x_{w2v}. Then, we generate a converted speech x_t with a voice of target speaker information s_t through the inverse transformation of the normalizing flow f_a^{-1} and decoder G as follows:

$$\begin{aligned} &z_l \sim q_{\phi_l}(z_l|x_{w2v}, s) \\ &x_t = G(f_a^{-1}(z_l|s_t)|s_t) \end{aligned} \tag{14}$$

Finally, we convert the extracted linguistic representations z_l into a speaker-independent representations e through the normalizing flow in the linguistic prior encoder f_l given the source speaker information s. Using e and the target speaker information s_t, we generate a converted speech s_t through inverse transformation of the normalizing flow f_l^{-1} and f_a^{-1} consecutively, and decoder G. The objectives for fine-tuning are represented as:

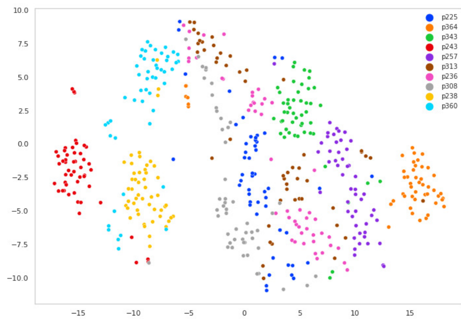
$$\begin{aligned} &z_l \sim q_{\phi_l}(z_l|x_{w2v}, s) \\ &e = f_l(z_l|s) \\ &x_t = G(f_a^{-1}(f_l^{-1}(e|s_t)|s_t)|s_t) \end{aligned} \tag{15}$$

We conduct the ASR evaluation and ASV evaluation to compare the above methods. Table 10 shows that the first method with Eq. 12 has better performance than others, and this means that x_{spec} contains more information to reconstruct the audio than x_{w2v} alongside the ablation study of Table 7. The EER results also show that all method can convert the voice of speech, indicating that speech is disentangled through linguistic and acoustic encoder.

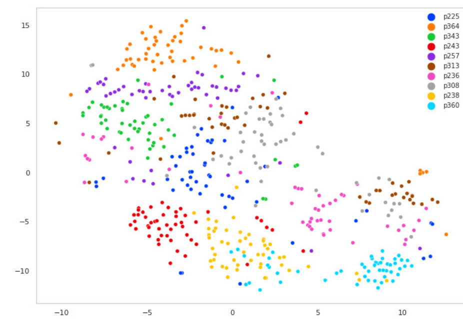
Table 10: Comparison for the different methods of voice conversion.

Model	CER	PER	WER	EER
HierSpeech (Eq. 12)	6.37	5.65	18.74	2.77
HierSpeech (Eq. 13)	8.42	7.96	20.34	2.92
HierSpeech (Eq. 14)	18.38	16.56	36.67	2.92
HierSpeech (Eq. 15)	20.15	18.89	40.09	3.72

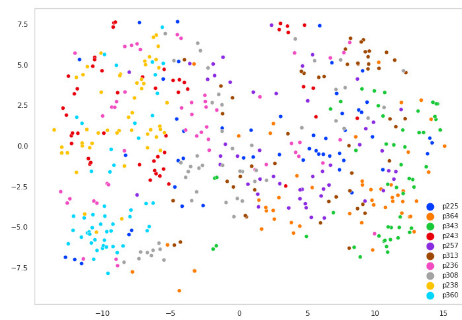
C Experiments



(a) The 1st layer of XLS-R

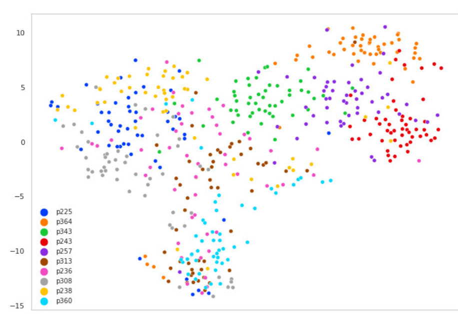


(b) The 12th layer of XLS-R

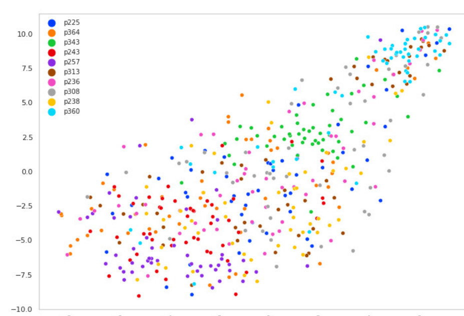


(c) The 24th layer of XLS-R

Figure 5: The t-SNE visualization for the representations from the different layers of XLS-R.



(a) Acoustic representations z_a



(b) Linguistic representations z_l

Figure 6: The t-SNE visualization for the acoustic/linguistic representations.

t-SNE visualization In Figure 5, we present the t-SNE visualization for the representations from the different layers of XLS-R. Following (Choi et al., 2021), we average each representation from the 1st, 12th, and 24th layer of XLS-R on the time-axis for 50 utterances of 10 speakers. Similar to the previous analysis of XLSR-53 (Choi et al., 2021), the representations from the 1st layer of XLS-R are already clustered by each speaker while it is hard to distinguish the representations of the latter layer by each speaker. In Figure 6, we also present the t-SNE visualization for the acoustic representations z_a and linguistic representations z_l. While acoustic representations are clustered by each speaker, it is difficult to differentiate the linguistic representations by speakers. This means that the linguistic encoder is able to extract the linguistic information from speech, and this allow the model to learn each representation hierarchically. It is worth nothing that using different features from the same speech can simply extract the disentangled representations of speech in our model.

Table 11: Results for speaker adaptation. To compare the adaptation performance with respect to the number of speakers, two models trained using VCTK and LibriTTS datasets are used for speaker adaptation. We used 10 speakers of VCTK dataset as novel speakers for fine-tuning.

Method	Pre-training	Fine-tuning	Transcript	PER	WER	EER	MCD	RMSE$_{f0}$
GT	-	-	-	4.26	16.69	4.14	-	-
HierSpeech	VCTK	✗	-	4.01	16.85	14.61	4.27	30.66
HierSpeech	VCTK	✓(1)	✓	3.32	15.04	10.80	4.32	29.66
HierSpeech	VCTK	✓(5)	✓	3.89	17.04	8.84	4.08	29.39
HierSpeech	VCTK	✓(10)	✓	4.74	18.26	8.08	4.00	29.15
HierSpeech	VCTK	✓(20)	✓	5.02	19.17	7.60	3.97	29.28
HierSpeech-U	VCTK	✓(1)	✗	4.76	18.71	13.33	4.29	29.74
HierSpeech-U	VCTK	✓(5)	✗	4.99	18.47	8.40	4.13	28.80
HierSpeech-U	VCTK	✓(10)	✗	4.28	16.94	7.60	4.15	28.83
HierSpeech-U	VCTK	✓(20)	✗	4.36	17.04	6.97	4.05	29.27
HierSpeech	VCTK+LibriTTS	✗	-	3.58	15.50	13.20	4.38	32.90
HierSpeech	VCTK+LibriTTS	✓(1)	✓	3.44	15.91	16.80	4.47	29.98
HierSpeech	VCTK+LibriTTS	✓(5)	✓	3.94	17.08	8.40	4.06	29.66
HierSpeech	VCTK+LibriTTS	✓(10)	✓	4.95	18.32	7.20	4.03	30.02
HierSpeech	VCTK+LibriTTS	✓(20)	✓	4.40	16.95	6.40	3.96	29.56
HierSpeech-U	VCTK+LibriTTS	✓(1)	✗	4.16	16.99	12.00	4.31	29.61
HierSpeech-U	VCTK+LibriTTS	✓(5)	✗	4.28	16.70	8.40	4.12	28.93
HierSpeech-U	VCTK+LibriTTS	✓(10)	✗	4.45	17.15	7.60	4.12	30.02
HierSpeech-U	VCTK+LibriTTS	✓(20)	✗	3.71	15.85	6.40	4.09	30.64

Untranscribed text-to-speech We describe the results of the objective evaluation for speaker adaptation in Table 11. We compare the adaptation quality with different numbers of samples (1, 5, 10, and 20) and different numbers of pre-trained speakers (98 from VCTK and 1,151 from LibriTTS). Table 11 shows that the adaptation quality is improved with an increase in the number of samples.

Phoneme predictor We conduct the ablation study of phoneme predictor. At first, we attempt to extract the linguistic representations without conditioning speaker information to extract the speaker-independent linguistic representations. However, adding speaker condition in linguistic encoder improves the model performance, indicating that a speaking style with the exception of voice is also trained from the speaker condition. Also, although the speaker conditioning is used in the linguistic encoder, the linguistic representations z_l contains a little speaker information as shown in Table 2 and Figure 6. Following (Kim et al., 2021), we remove a bias parameter of phoneme predictor, which causes unstable training during mixed precision training. For better generalization, we attempt to train the phoneme predictor with label smoothing. However, it decreases the model performance, as it disturbs to optimize the KL divergence.

Data augmentation For effective speech disentanglement, we use the data augmentation for the input of the linguistic encoder. We use the information perturbation of (Choi et al., 2021) to the waveform for the input of XLS-R. After data augmentation, the representations from the XLS-R may lose their speaker and pitch information, and then the linguistic representations are extracted from the linguistic-related information of XLS-R. However, this information perturbation decreases the model performance as shown in Table 12b. Also, our model already extracts the linguistic representations guided by phoneme predictor. Hence, the data augmentation for speech disentanglement is not necessary in our method.

Table 12: The results of ablation study and data augmentation.

(a) Ablation study for phoneme predictor.

Model	CER	PER	WER
GT	6.26	5.64	18.94
HierSpeech	6.77	6.25	20.89
w.o speaker condition	7.38	6.77	21.82
w bias	7.51	6.93	22.31
w Label Smoothing	7.99	7.60	23.00

(b) Experiment for data augmentation.

Model	Augmentation	ratio	CER	PER	WER	EER
GT			6.26	5.64	18.94	4.03
HierSpeech	✗	-	6.77	6.25	20.89	3.45
HierSpeech	✓	0.1	6.98	6.54	20.96	3.81
HierSpeech	✓	0.2	7.63	7.14	22.35	3.71
HierSpeech	✓	0.3	7.12	6.59	21.80	3.81
HierSpeech	✓	0.4	7.63	7.16	22.25	4.09
HierSpeech	✓	0.5	7.57	6.98	22.26	3.96
HierSpeech	✓	0.6	7.69	7.26	22.90	3.59
HierSpeech	✓	0.8	7.88	7.31	22.81	3.96
HierSpeech	✓	1.0	7.87	7.37	22.77	3.74

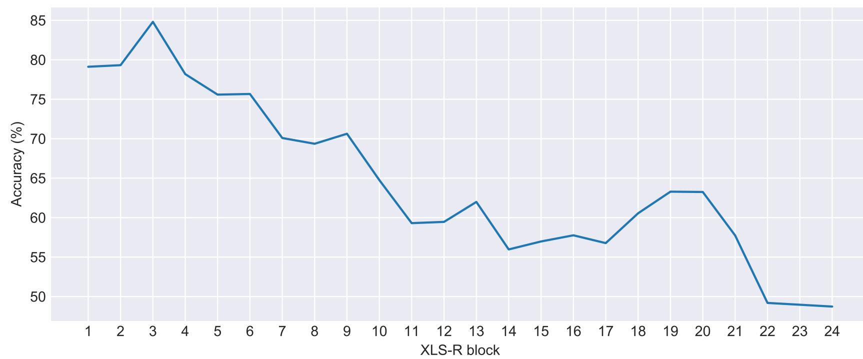


Figure 7: Speaker classification on self-supervised representations from different layers of XLS-R.

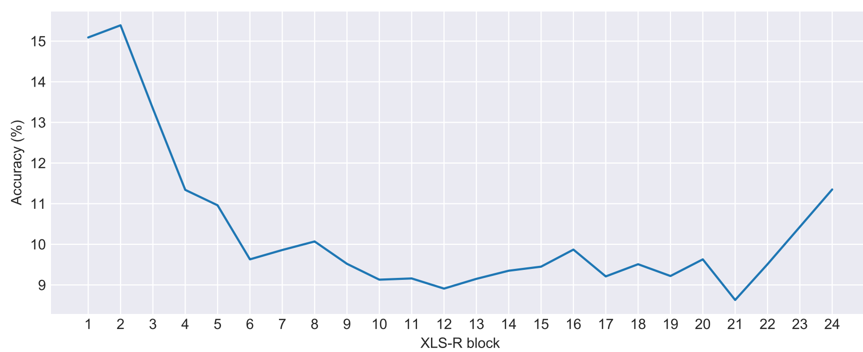


Figure 8: Speaker classification on linguistic representations from different layers of XLS-R.

Analysis of the self-supervised speech representations for TTS Self-supervised speech representations contain rich information which is trained with large-scale speech dataset. Currently, wav2vec 2.0 (Baevski et al., 2020) is the most widely used model where the public can be easy to accessible to it, and it has shown the improved performance on downstream tasks such as ASR and speech translation. In this paper, we investigate these self-supervised representations to distill this rich information for text-to-speech task. Previous works show that each representation from the different layer of these model has different characteristics, and especially the representations from the middle layer contains the pronunciation information of speech (Shah et al., 2021). Our goal is to bridge the gap between text and speech through the additional representations, and the linguistic information such as pronunciation of speech may be appropriate to improve the pronunciation of synthesized speech. To demonstrate it, we train the model with each layer from the 24-layer transformer networks respectively. Table 13 shows that all layer with the exception of the 23th layer improve the model in ASR evaluation, indicating each representation contains rich information which is trained with large-scale speech dataset. Note that we fail to train the model with the representations from the 23th layer of XLS-R. For the hierarchical training of different information and the untranscribed text-to-speech, we should disentangle the linguistic and acoustic representations from speech. Hence, we conduct the speaker classification on each representation from the layer of XLS-R, and on each linguistic representation z_l from each layer of XLS-R to measure the speaker information on the representations. We train the frame-level speaker classifier which consists of 4 Conv1D with kernel size of 7 followed by the projection layer. Figure 8 shows that the speaker classification results have an tendency to be decreased from the first layer to latter layer. However, the linguistic representations from the 12th layer of XLS-R have the lowest speaker classification accuracy in middle layer, indicating these representations may contain much more linguistic information which is trained with phoneme classification. In this regard, we use the representations from the 12th layer of XLS-R to extract the linguistic representations.

Table 13: Hyperparameter search for XLS-R (wav2vec 2.0) layer.

Model	w2v layer	CER	PER	WER	EER
GT		6.26	5.64	18.94	4.03
VITS	-	12.53	12.24	30.62	3.85
HierSpeech	1	6.94	6.49	21.54	3.74
HierSpeech	2	7.01	6.65	21.39	3.74
HierSpeech	3	6.78	6.35	20.87	3.86
HierSpeech	4	6.93	6.44	21.05	3.96
HierSpeech	5	7.14	6.75	21.33	4.00
HierSpeech	6	7.00	6.55	21.16	3.86
HierSpeech	7	6.81	6.30	20.68	3.74
HierSpeech	8	6.87	6.32	20.89	3.56
HierSpeech	9	6.76	6.11	20.86	3.82
HierSpeech	10	6.72	6.12	20.71	3.59
HierSpeech	11	7.04	6.51	21.32	3.66
HierSpeech	12	6.77	6.25	20.89	3.48
HierSpeech	13	6.92	6.40	21.14	3.79
HierSpeech	14	6.89	6.36	20.60	4.01
HierSpeech	15	6.97	6.54	21.28	3.51
HierSpeech	16	6.73	6.35	20.81	3.62
HierSpeech	17	6.88	6.23	20.70	3.71
HierSpeech	18	7.13	6.62	21.41	3.74
HierSpeech	19	6.83	6.38	21.09	3.55
HierSpeech	20	7.53	7.09	22.00	3.96
HierSpeech	21	7.44	7.00	22.30	3.40
HierSpeech	22	7.62	7.18	22.34	4.03
HierSpeech	23	-	-	-	-
HierSpeech	24	7.42	6.85	21.68	3.75

D Baseline Models

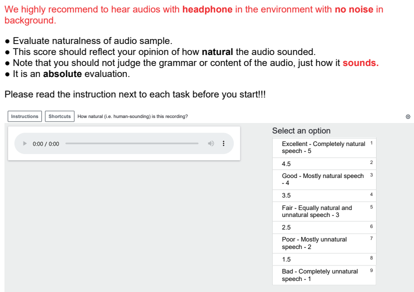
TTS We use an open source implementation of Tacotron 2[6], an official implementation of Glow-TTS[7], PortaSpeech[8], and VITS[9]. Since baseline models synthesize Mel-spectrogram unlike VITS and HierSpeech, we transform audio into Mel-spectrogram following (Kong et al., 2020) with a window size of 1024, hop size of 256, 1024 points of Fourier transform, and 22,050 Hz. For Tacotron 2, we use 32-dimensional speaker embedding which is concatenated with the output of text encoder following Skerry-Ryan et al. (2018). We train Tacotron 2 with batch size of 256 for 100k steps. For Glow-TTS, we condition 256-dimensional speaker embedding into the affine coupling layer in decoder and duration predictor to predict the speaker-specific duration. We train Glow-TTS with batch size of 128 for 960k steps. For PortaSpeech, we add 256-dimensional speaker embedding with the output of encoder and duration predictor. We train PortaSpeech with batch size of 64 with 320k steps. To convert Mel-spectrogram to waveform audio, we use the official implementation of pre-trained HiFi-GAN V1[10]. For VITS, we use the same speaker conditioning of ours and train the model with batch size of 256 with 600k steps. For VITS and HierSpeech, we sample 32 sequences from the whole z_a, and upsample it by 256x which is the same size as hop size for STFT. We use VCTK and *train-clean-360* and *train-clean-100* subset of LibriTTS dataset. Both dataset are licensed under the Creative Commons Attribution 4.0.

VC We use the official implementation of AutoVC[11] and VoiceMixer[12]. Both models are trained with Mel-spectrogram segments of 192 frames during training. We train the AutoVC with the information bottleneck of 32 frames and a batch size of 16 for 100k steps. For VoiceMixer, we train the model with a batch size of 64 for 150k steps. We also use the pre-trained HiFi-GAN V1 to convert the Mel-spectrogram to waveform audio.

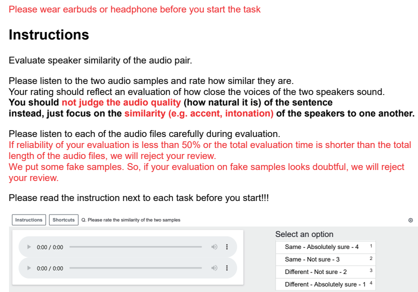
[6] `https://github.com/NVIDIA/tacotron2`
[7] `https://github.com/jaywalnut310/glow-tts`
[8] `https://github.com/NATSpeech/NATSpeech`
[9] `https://github.com/jaywalnut310/vits`
[10] `https://github.com/jik876/hifi-gan`
[11] `https://github.com/auspicious3000/autovc`
[12] `https://github.com/anonymous-speech/voicemixer/tree/main/code`

E Evaluation Details

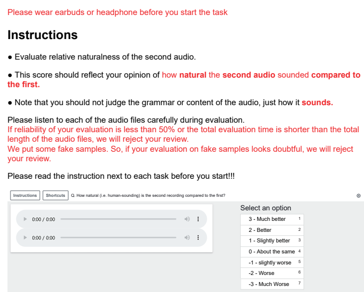
Mean opinion score We include the details of instructions of MOS evaluation in Figure 9.



(a) nMOS



(b) sMOS



(c) CMOS

Figure 9: The screenshots of MOS evaluation. $0.1 per 1 hit is paid to participants for nMOS (7 samples) and $0.01 per 1 hit is paid to participants for sMOS and CMOS.

Automatic speech recognition We use the pre-trained wav2vec 2.0 model (base model[13]) to evaluate the character error rate, phoneme error rate, and word error rate (WER). We calculate the WER based on the character prediction results to evaluate the pronunciation without language model.

Automatic speaker verification We use the pre-trained ASV model[14] to evaluate the speaker similarity. The Fast ResNet-34 model (Chung et al., 2020) is used to extract the features, and the similarities of features are compared for the verification results. The model was trained by VoxCeleb2 (Chung et al., 2018) with online data augmentation (Heo et al., 2020). The online data augmentation improves the model performance from EER of 2.1792 to 1.1771 as described in (Heo et al., 2020).

Mel cepstral distortion We extract the Mel-frequency cepstral coefficients (MFCCs) by discrete cosine transform to raw waveform[15]. Then, we calculate the MCD between ground-truth and synthesized speech from the first 13 MFCCs by using dynamic time warping (DTW) to align the different length of sequences.

F0 root mean square error We extract the fundamental frequency F0 by World vocoder[16]. Then, we compute the $l2$ distance between the ground-truth and synthesized speech for RMSE_{f0}. We use the DTW to align two sequences.

Average differences of the utterance duration We conduct DDUR evaluation presented in (Zhang et al., 2019) to evaluate the duration prediction performance. For fair evaluation, we trim the silence of audio, and then we calculate the average absolute differences of duration between ground-truth and synthesized speech.

[13] `https://huggingface.co/docs/transformers/model_doc/wav2vec2`
[14] `https://github.com/clovaai/voxceleb_trainer`
[15] `https://github.com/MTG/essentia/`
[16] `https://github.com/JeremyCCHsu/Python-Wrapper-for-World-Vocoder`