INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

FACTSpeech: Speaking a Foreign Language Pronunciation Using Only Your Native Characters

Hong-Sun Yang, Ji-Hoon Kim, Yoon-Cheol Ju, Il-Hwan Kim, Byeong-Yeol Kim, Shuk-Jae Choi, Hyung-Yong Kim

42dot Inc., Seoul, Republic of Korea

{hongsun.yang, jh.kim, yooncheol.ju, ilhwan.kim, byeongyeol.kim, shukjae.choi, hyungyong.kim}@42dot.ai

Abstract

Recent text-to-speech models have been requested to synthesize natural speech from language-mixed sentences because they are commonly used in real-world applications. However, most models do not consider transliterated words as input. When generating speech from transliterated text, it is not always natural to pronounce transliterated words as they are written, such as in the case of song titles. To address this issue, we introduce FACTSpeech, a system that can synthesize natural speech from transliterated text while allowing users to control the pronunciation between native and literal languages. Specifically, we propose a new language shift embedding to control the pronunciation of input text between native or literal pronunciation. Moreover, we leverage conditional instance normalization to improve pronunciation while preserving the speaker identity. The experimental results show that FACTSpeech generates native speech even from the sentences of transliterated form.

Index Terms: text-to-speech, cross-lingual, accent control, language-mixed

1. Introduction

Text-to-Speech (TTS) aims to synthesize natural speech from input text. With the introduction of deep neural networks, the TTS model has been significantly improved, making it possible to generate human-like speech [1, 2, 3, 4, 5, 6, 7]. Due to the increasing usage of sentences written in two or more languages (i.e., code-mixed sentences), recent TTS models have been expanded from monolingual TTS to multilingual TTS that generates speeches with multiple languages by multi-speaker (e.g., generating fluent English speech in the voice of Korean speaker) [8, 9, 10, 11, 12].

Owing to the difficulty of getting bilingual and parallel speech corpus [13], a lot of existing works have attempted to construct multilingual TTS utilizing a mixture of monolingual training data from different languages [8, 9, 10, 11]. However, the limitation of the monolingual dataset lies in that the speaker identity is entangled with assigned linguistic information. As a result, preserving the speaker identity can be difficult when replacing the source language representation with the target language representation. To separate speaker identity from phonetic representations, most of the previous works utilize domain adversarial training [9, 11, 14, 15]. Moreover, T. Nekvinda and O. Dusek [11] propose a meta-learning approach to predict the language-specific weights of text encoder concurrently with domain adversarial training. D. Xin et al. [14] adds mutual information minimization objective to strengthen speaker-text disentanglement.

The use of language-mixed sentences can take two forms: code-mixed or transliterated. An example of code-mixed form is

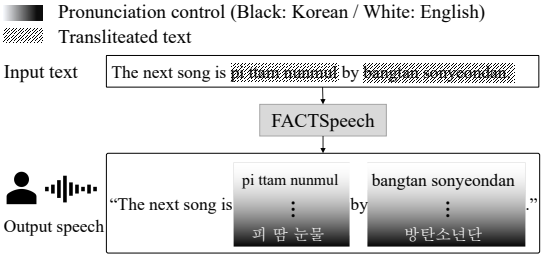

Figure 1: *Overview of FACTSpeech. Our model can control pronunciation between native or literal pronunciation.*

"The next song is 피 땀 눈물," whereas an example of transliterated form is "The next song is pi ttam nunmul." Although many studies aim to generate natural speech from code-mixed sentences, they often neglect transliterated words as input [16, 17]. However, transliterated forms are commonly used in real-world scenarios, such as when referring to song titles or place names.

In this paper, we introduce Foreign-Accent Controllable Transliterated Text-to-Speech (FACTSpeech), which aims to overcome the limitations of existing methods [16, 17] that focus on synthesizing natural speech from code-mixed sentences, without considering transliterated words as input. The main contribution of FACTSpeech is its ability to synthesize speech from transliterated text while controlling the pronunciation between native and literal languages. This means that native pronunciation can be obtained without prior knowledge of the characters of the target language from the user. To this end, we propose transliteration-based data augmentation (TDA) to learn transliterated transcripts, and the proposed language shift embedding (LSE) enables the model to relate between languages in the transliterated text. Moreover, we employ conditional instance normalization (CIN) [18] to improve pronunciation accuracy while preserving the speaker identity.

The experimental results indicate that FACTSpeech can generate natural pronunciation with only source language text inputs and control pronunciation between source and target languages. The effectiveness of FACTSpeech is demonstrated through various metrics, including subjective mean opinion score (MOS) test, and spoken language identification (SLID). The synthesized audio samples are presented on our demo page[1].

2. Model description

Fig. 2 depicts the network architecture of FACTSpeech. Rooted from FastPitch [6], FACTSpeech consists of three modules: text encoder, variance predictors, and decoder. Here, variance predictors consist of pitch, energy, and duration predictor. We uti-

[1] https://atozto9.github.io/demo/FACTSpeech/

10.21437/Interspeech.2023-2303

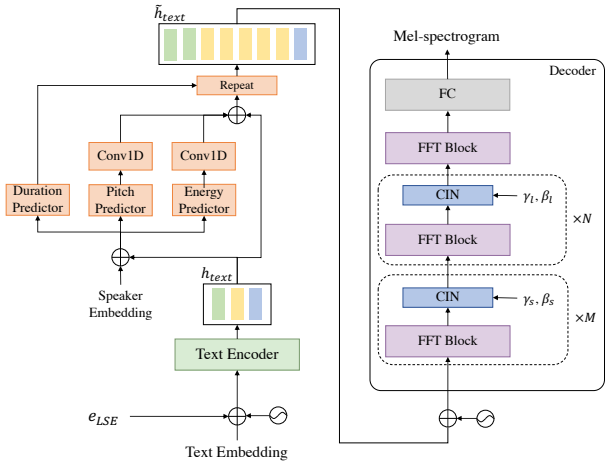

Figure 2: *The model architecture of FACTSpeech. LSE refers to language shift embedding and CIN means conditional instance normalization. "Sinusoidal symbol" refers to the positional encoding. γ and β indicate learnable parameters in CIN. In our experiments, we fixed $N = 2$ and $M = 3$.*

lize a speaker lookup table for multi-speaker conditioning to variance predictors to obtain speaker representation. We also add CIN [18] to the decoder for native pronunciation. In the following subsections, we will describe further details on each module in FACTSpeech.

2.1. Text encoder

FACTSpeech takes text embedding as input. Here, the text embedding input is obtained by language-dependent grapheme sequence and lookup table. The acquired text embedding is added to positional encoding and LSE, then fed into the text encoder. The text encoder generates the hidden linguistic representation h_{text}.

2.2. Pitch and energy predictor

Extending FastPitch [6], we employ an energy predictor [4] in addition to a pitch predictor. Pitch/energy predictors aim to provide pitch/energy information to h_{text}, respectively. Combined with speaker embedding, the predictors take h_{text} as input and predict pitch/energy values. The pitch/energy embeddings are obtained from the predicted values through a single 1D convolutional layer. Finally, the predicted pitch/energy embeddings are added to h_{text}.

The pitch predictor is optimized by mean square error (MSE) loss between the predicted and the ground truth pitch value, which is extracted by `pyin` algorithm [19] as in FastPitch [6]. The ground truth energy value is computed by L2 normalization from the amplitude of each mel-spectrogram frame. Similar to the pitch predictor, the energy predictor is optimized with MSE between the predicted and ground truth energy values. This can be formulated as follows:

$$\mathcal{L}_{\text{pitch}} = \text{MSE}\left(x_{\text{pitch}}, \hat{x}_{\text{pitch}}\right), \tag{1}$$

$$\mathcal{L}_{\text{energy}} = \text{MSE}\left(x_{\text{energy}}, \hat{x}_{\text{energy}}\right), \tag{2}$$

where $\text{MSE}\left(x, \hat{x}\right) = \frac{1}{T}\sum_{t=1}^{T}\left(x(t) - \hat{x}(t)\right)^2$, x and $\hat{x}$ refer to the ground truth and predicted value, respectively, and T represents the length of the sequence.

2.3. Duration predictor with online aligner

To match the length between linguistic and acoustic representations, we utilize the duration predictor [3, 4, 6]. Conditioning with speaker embedding, the duration predictor takes h_{text} as an input and predicts the token duration. The predicted duration is used for upsampling h_{text} combined with pitch and energy embeddings. We optimize the duration predictor with MSE loss between the predicted and the ground truth duration, which can be formulated as:

$$\mathcal{L}_{\text{duration}} = \text{MSE}\left(x_{\text{duration}}, \hat{x}_{\text{duration}}\right), \tag{3}$$

where x_{duration} and $\hat{x}_{\text{duration}}$ refer to the target and predicted duration, respectively.

To extract the target duration value, we utilize the online duration search algorithm [20, 21]. It enables the model to easily extend to other languages since it doesn't depend on the language-specific external aligners.

$\mathcal{L}_{\text{ForwardSum}}$ and $\mathcal{L}_{\text{bin}}$ are used to optimize online aligner [20, 21]. $\mathcal{L}_{\text{ForwardSum}}$ is obtained through connectionist temporal classification loss [22] to maximize the likelihood of input text S given mel-spectrogram X. $\mathcal{L}_{\text{bin}}$ makes closer soft alignments A_{soft} to hard alignments A_{hard}, which represents the KL-divergence of two alignments. We can formulate $\mathcal{L}_{\text{ForwardSum}}$ and $\mathcal{L}_{\text{bin}}$ as follow:

$$\mathcal{L}_{\text{ForwardSum}} = -\log P(S|X), \tag{4}$$

$$\mathcal{L}_{\text{bin}} = D_{KL}(A_{\text{soft}}||A_{\text{hard}}), \tag{5}$$

where A_{soft} is obtained by taking the softmax of the minus L2 distance of the text and mel-spectrogram representations. Two representations are obtained by forwarding text and mel to 1D convolutional layers. By the Viterbi algorithm, A_{soft} can be converted to A_{hard} which represents the target duration. Here, we denote $\mathcal{L}_{\text{align}} = \mathcal{L}_{\text{ForwardSum}} + \mathcal{L}_{\text{bin}}$ for brevity.

2.4. Decoder

The decoder, which is built upon feed-forward transformer (FFT) blocks [24, 25], takes upsampled hidden representation $\tilde{h}_{text}$ as input and predicts mel-spectrogram. It is well-known that the hidden vectors of the top layers tend to learn language information, while the bottom layers are useful for learning speaker information [26, 27]. Motivated by this, we condition the corresponding information to our decoder through CIN, which will be described in Sec. 3.3.

From the predicted mel-spectrogram, the mel-spectrogram reconstruction loss is calculated. This can be formulated as:

$$\mathcal{L}_{\text{mel}} = \text{MSE}\left(y_{\text{mel}}, \hat{y}_{\text{mel}}\right), \tag{6}$$

where y_{mel} denotes the ground truth mel-spectrogram and $\hat{y}_{mel}$ refers to the predicted mel-spectrogram.

The overall training loss is represented as follows,

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{mel}} + \lambda_p\mathcal{L}_{\text{pitch}} + \lambda_e\mathcal{L}_{\text{energy}} + \mathcal{L}_{\text{duration}} + \mathcal{L}_{\text{align}}, \tag{7}$$

and the model is trained to minimize this loss. We set the hyperparameters λ_p and λ_e to 0.1, following the same scaling as in FastPitch [6].

3. Multilingual TTS extension

We leverage transliteration-based data augmentation and propose LSE. Combined with the proposed LSE, transliterated data

Table 1: *MOS, SMOS, and SLID evaluation results. MOS and SMOS are presented with 95% confidence intervals. SLID is calculated using the model of [23]. GT is the ground truth speech.*

Method	Intra-lingual						Cross-lingual					
	EN			T-EN			EN			T-EN		
	MOS	SMOS	SLID	MOS	SMOS	SLID	MOS	SMOS	SLID	MOS	SMOS	SLID
GT	4.31±0.03	3.98±0.07	98.9%	-	-	-	-	-	-	-	-	-
Vocoded	4.27±0.04	3.90±0.07	98.8%	-	-	-	-	-	-	-	-	-
SANE-TTS	4.03±0.06	3.78±0.07	97.7%	4.04±0.06	3.70±0.07	97.8%	3.78±0.05	3.53±0.07	91.3%	3.78±0.05	3.57±0.07	91.9%
Y. Zhang et al.	3.99±0.06	**3.79±0.07**	98.1%	3.98±0.06	**3.78±0.07**	97.9%	3.74±0.05	3.44±0.07	84.6%	3.68±0.05	3.47±0.07	85.4%
FACTSpeech	**4.04±0.06**	3.75±0.07	**98.5%**	4.04±0.06	3.72±0.07	**98.3%**	**3.90±0.04**	**3.54±0.07**	**96.8%**	**3.81±0.05**	**3.60±0.07**	**95.9%**

enables FACTSpeech to synthesize natural pronunciation without knowledge of the target foreign language and control pronunciation between source and target languages. Moreover, we employ CIN [18] to improve pronunciation accuracy while preserving the speaker identity.

3.1. Transliteration-based data augmentation

Transliteration refers to the way of mapping from scripts of the source language to that of the target language based on their phonetic similarity.

Although transliteration-based data augmentation enables the model to learn the relationship between transliterated text and the pronunciation of target languages, the synthesized speech's pronunciation is determined by the training data. Consequently, the model lacks control over the pronunciation style and relies solely on the input text.

To address this limitation and enable control over the pronunciation style in transliterated input, we propose LSE. The details of LSE will be described in the following subsection.

3.2. Language shift embedding

LSE is a learned embedding that enables the model to learn native pronunciation from transliterated data and is trained together with the entire model to minimize the total loss. During training, the model learns to associate the LSE with the corresponding text embedding, allowing it to shift the input text representation from one language domain to another. This integration of the LSE into the training process ensures that it becomes an integral part of the model's learned representations.

Additionally, it is worth noting that when training transliterated scripts from language A to language B, the LSE e_{LSE}^{A2B} is added to the corresponding text embedding. This addition enables the model to generate native pronunciations in language B, leveraging the learned associations between the LSE and text embeddings. Similarly, when training transliterated scripts from language B to language A, the LSE e_{LSE}^{B2A} is computed as the negative of e_{LSE}^{A2B}.

The effectiveness of the LSE in generating natural native pronunciation and controlling pronunciation between two languages is verified in Sec. 4.5.

3.3. Conditional instance normalization

To improve pronunciation accuracy while preserving speaker identity, we leverage CIN [18]. CIN is a popular technique for regulating the hidden state with the desired information [28, 29, 30]. We adopt CIN to guide the training of our decoder.

Let $\mathbf{X} \in \mathbb{R}^{C \times T}$ be the hidden state of the decoder, where C is the number of channels, and T means the length of the sequence. We set learnable affine parameters $\{\gamma_l, \beta_l\}$ and $\{\gamma_s, \beta_s\}$ for language and speaker information, respectively. These are used to regulate $\mathbf{X}$ so that the hidden states can be guided by a specific language and speaker style.

The following equations summarize the CIN:

$$\text{CIN}\,(x_c) = \gamma_c \frac{x_c - \mu_c}{\sigma_c} + \beta_c, \tag{8}$$

where $\mu_c = \frac{1}{T}\sum_{t=1}^{T} x_t^c$, and $\sigma_c = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(x_t^c - \mu_c) + \epsilon}$. $\mathbf{x}_c^t \in \mathbf{X}$ represents the scalar value at the c^{th} channel and the t^{th} time step of $\mathbf{X}$ and T means the length of the hidden sequence.

4. EXPERIMENTS

This section confirmed our model's speech quality, speaker similarity, and pronunciation accuracy. For this, subjective evaluation based on MOS and objective evaluation using SLID were conducted.

4.1. Experimental settings

We trained FACTSpeech based on a mixture of proprietary English and Korean datasets, containing 24 hours of audio. The dataset contains 60 speakers, 30 for each language. We trimmed the silence at the beginning and end of the audio and re-sampled the audio to $22,050$ Hz. The 80-bin log mel-spectrogram is calculated with a fast Fourier transform size of 1024, hop size of 256, and window size of 1024. We employed the LAMB [31] optimizer with $\beta_1 = 0.9, \beta_2 = 0.9, \epsilon = 1e-9$, and a batch size of 16. FACTSpeech was trained on an Nvidia T4 for 300 epochs, which took about 4 days. Additionally, we adopted the learning rate scheduling method used in FastPitch [6]. We utilized an internal transliteration module to convert English transcripts into Korean transliterated form and used open-sourced transliterate module[2] to convert vise versa. The predicted mel-spectrogram was converted into an audible waveform by pre-trained Fre-GAN vocoder [32].

We basically followed Fastpitch's hyperparameters setting. We use 384-dimensional text embedding and LSE. Y. Zhang et al. [9] and SANE-TTS [15] were used as TTS quality comparison models. We replaced the backbone network of these models with FastPitch to ensure a fair comparison.

4.2. Evaluation on TTS

We measured speech naturalness and speaker similarity using MOS. MOS and similarity MOS (SMOS) evaluations were conducted on Amazon Mechanical Turk [33]. For the test, 80 utterances were used. Audio samples were rated by at least 15 participants. Whether the generated speech has the correct pronunciation was measured through SLID [23].

The results in Table 1 show that FACTSpeech can generate natural speech from transliterated scripts and standard text input while maintaining speaker identity. The results show that outperforms the other models with respect to the MOS, SMOS,

[2] https://github.com/osori/korean-romanizer

Table 2: *The additional studies of decoder component change in terms of SLID are shown. Through FACTSpeech and w/o CIN row, the effect of improving the pronunciation of CIN is confirmed. The FACTSpeech and w/ $Decoder_m$ rows confirmed that the speaker condition preceding the language condition improves the performance.*

	[Input Type] to [Target Pronunciation]			
	EN to EN	KO to KO	T-EN to EN	T-KO to KO
FACTSpeech	**97.6%**	**99.9%**	**97.1%**	**99.9%**
w/o CIN	89.2%	99.6%	89.6%	99.6%
w/ $Decoder_m$[3]	87.4%	98.5%	87.8%	98.4%

and SLID for the cross-lingual situation. For intral-lingual situation, we observed that our model's SMOS score is slightly lower than the other models, but the difference is within the margin of error.

4.3. Spoken language identification

We evaluated how well our model generates speech in the pronunciation of the target language in four input types[4]. Table 2 shows the probability of being classified as the target language. The probability is obtained from the SLID model published by [23]. Comparing the SLID results of FACTSpeech and FACTSpeech without CIN, it can be seen that the probability of being classified as the target language is increased in all four cases. From this, it can be seen that adding CIN improves the pronunciation of speech generated from all input types. In particular, it can be seen that the probability of being recognized as the target language is high even in transliterated text. This means we can generate native speech without prior knowledge of the characters of the target language.

When comparing FACTSpeech and FACTSpeech w/ $Decoder_m$, FACTSpeech shows high SLID results for all cases. It can be confirmed that giving speaker information at the bottom layers and language information at the top layers in the decoder is effective in improving pronunciation.

4.4. Speaker-language disentanglement

To investigate the effect of transliterated data on speaker-language disentanglement, we measured discrete probability distribution (DPD) from a text representation. We trained *test-only* speaker classifier, consisting of two linear layers with ReLU activation and dropout.

Figure 3 represents the DPD results of a model (a) without TDA and (b) with TDA. In Figure 3 (a), Korean text inputs are mainly classified as Korean speakers, which demonstrates the unsolved speaker-language entanglement. However, when TDA is applied, it shows an even DPD results; this indicates using TDA helps to remove speaker information from text representations.

4.5. Pronunciation control between languages

We evaluated if our model can control the pronunciation between native and foreign-accent using LSE. We also compare a model using language embedding (LE) [10] instead of LSE. Table 3 compares the effect of LSE with LE. To do so, we used

[3]The language condition comes before the speaker condition in the decoder.

[4]Four input types: "EN", "KO", "T-EN", and "T-KO" represents English, Korean, a transliterated sentence from English to Korean, and a transliterated sentence from Korean to English, respectively.

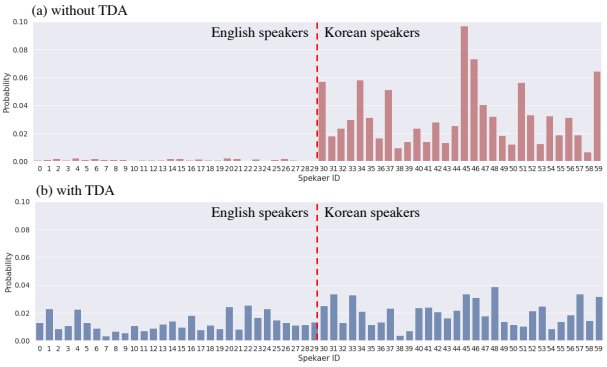

Figure 3: *The speaker identification probability distribution from text embeddings of Korean sentences is shown. Speaker IDs 0 to 29 are for English speakers, while the other IDs are for Korean speakers. We can see that using TDA disentangled the language and speaker information.*

Table 3: *The effect of LSE on language style control. The values represent the probability of SLID identifying the language as the target language. LSE, as demonstrated, can synthesize speech for target language pronunciation. In particular, transliterated text to literal pronunciation cases outperformed the traditional method with LE.*

	[Input Type] to [Target Pronunciation]			
	T-KO to EN	T-KO to KO	T-EN to EN	T-EN to KO
LSE + TDA	**45.9%**	**99.6%**	**89.6%**	**91.5%**
LE + TDA	0.0%	98.6%	86.3%	7.8%

three types of e_{LSE} to specify the language style of the generated speech. In detail, e_{LSE} is set to $\overrightarrow{\mathbf{0}}$, $e_{LSE}^{eng2kor}$, $e_{LSE}^{kor2eng}$, and $\overrightarrow{\mathbf{0}}$ for T-KO to EN, T-KO to KO, T-EN to EN, and T-EN to KO, respectively. As a result, we confirmed using the SLID model that the pronunciation style was uttered in the desired language as the LSEs were changed. However, in the case of the model to which LE was applied, the pronunciation style was not controlled. We verified that the pronunciation control was valid with inference examples. These results are on our demo page[5] (LSE interpolation demo), and we strongly suggest that you check them out.

5. Conclusions

We present FACTSpeech, a TTS system that can model natural pronunciation using only one native character set. To achieve this, we propose the introduction of LSE and TDA. These methods help the model synthesize speech with accurate native pronunciation and control pronunciation between two languages. We apply CIN to improve pronunciation accuracy while preserving speaker identity. Through experiments, we confirmed that pronunciation control was possible using LSE and TDA, and pronunciation performance improved using CIN. Pronunciation control can make speech sound more diverse in terms of accent. The study can also be extended as a data augmentation method as examples of non-natives' speech. Although the FACTSpeech system currently relies on the transliteration module, we aim to investigate methods for training FACTSpeech without the dependency on transliteration in future work.

[5]https://atozto9.github.io/demo/FACTSpeech/#interpolation

6. References

[1] 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 *Proc. ICASSP*, 2018, pp. 4779–4783.

[2] E. Song, R. Yamamoto, M.-J. Hwang, J.-S. Kim, O. Kwon, and J.-M. Kim, "Improved parallel WaveGAN vocoder with perceptually weighted spectrogram loss," in *2021 IEEE Spoken Language Technology Workshop (SLT)*, 2021, pp. 470–476.

[3] 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 *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019.

[4] 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 *Proc. International Conference on Learning Representation (ICLR)*, 2020.

[5] J. Kim, S. Kim, J. Kong, and S. Yoon, "Glow-TTS: A generative flow for text-to-speech via monotonic alignment search," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 8067–8077.

[6] A. Łańcucki, "Fastpitch: Parallel text-to-speech with pitch prediction," in *Proc. ICASSP*, 2021, pp. 6588–6592.

[7] S.-H. Lee, H.-W. Yoon, H.-R. Noh, J.-H. Kim, and S.-W. Lee, "Multi-SpectroGAN: high-diversity and high-fidelity spectrogram generation with adversarial style combination for speech synthesis," in *Proc. AAAI*, vol. 35, no. 14, 2021, pp. 13 198–13 206.

[8] E. Nachmani and L. Wolf, "Unsupervised polyglot text-to-speech," in *Proc. ICASSP*, 2019, pp. 7055–7059.

[9] Y. Zhang, R. J. Weiss, H. Zen, Y. Wu, Z. Chen, R. Skerry-Ryan, Y. Jia, A. Rosenberg, and B. Ramabhadran, "Learning to speak fluently in a foreign language: Multilingual speech synthesis and cross-language voice cloning," in *Proc. Interspeech*, 2019, pp. 2080–2084.

[10] J. Yang and L. He, "Towards universal text-to-speech." in *Proc. Interspeech*, 2020, pp. 3171–3175.

[11] T. Nekvinda and O. Dušek, "One Model, Many Languages: Meta-learning for multilingual text-to-speech," in *Proc. Interspeech*, 2020, pp. 2972–2976. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2020-2679

[12] J.-H. Kim, H.-S. Yang, Y.-C. Ju, I.-H. Kim, and B.-Y. Kim, "CrossSpeech: speaker-independent acoustic representation for cross-lingual speech synthesis," *arXiv preprint arXiv:2302.14370*, 2023.

[13] S. Maiti, E. Marchi, and A. Conkie, "Generating multilingual voices using speaker space translation based on bilingual speaker data," in *Proc. ICASSP*, 2020, pp. 7624–7628.

[14] D. Xin, T. Komatsu, S. Takamichi, and H. Saruwatari, "Disentangled speaker and language representations using mutual information minimization and domain adaptation for cross-lingual TTS," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6608–6612.

[15] H. Cho, W. Jung, J. Lee, and S. H. Woo, "SANE-TTS: Stable and natural end-to-end multilingual text-to-speech," *arXiv preprint arXiv:2206.12132*, 2022.

[16] X. Zhou, X. Tian, G. Lee, R. K. Das, and H. Li, "End-to-end code-switching TTS with cross-lingual language model," in *Proc. ICASSP*, 2020, pp. 7614–7618.

[17] Y. Cao, X. Wu, S. Liu, J. Yu, X. Li, Z. Wu, X. Liu, and H. Meng, "End-to-end code-switched TTS with mix of monolingual recordings," in *Proc. ICASSP*, 2019, pp. 6935–6939.

[18] V. Dumoulin, J. Shlens, and M. Kudlur, "A learned representation for artistic style," in *Proc. International Conference on Learning Representation (ICLR)*, 2017.

[19] M. Mauch and S. Dixon, "PYIN: A fundamental frequency estimator using probabilistic threshold distributions," in *Proc. ICASSP*, 2014, pp. 659–663.

[20] K. J. Shih, R. Valle, R. Badlani, A. Lancucki, W. Ping, and B. Catanzaro, "RAD-TTS: Parallel flow-based TTS with robust alignment learning and diverse synthesis," in *Proc. ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models*, 2021.

[21] R. Badlani, A. Łańcucki, K. J. Shih, R. Valle, W. Ping, and B. Catanzaro, "One TTS alignment to rule them all," in *Proc. ICASSP*, 2022, pp. 6092–6096.

[22] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 369–376.

[23] J. Valk and T. Alumäe, "VoxLingua107: a dataset for spoken language recognition," in *Proc. IEEE SLT Workshop*, 2021.

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[25] N. Li, S. Liu, Y. Liu, S. Zhao, and M. Liu, "Neural speech synthesis with transformer network," in *Proc. AAAI*, vol. 33, no. 01, 2019, pp. 6706–6713.

[26] Z. Fan, M. Li, S. Zhou, and B. Xu, "Exploring wav2vec 2.0 on speaker verification and language identification," *arXiv preprint arXiv:2012.06185*, 2020.

[27] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao *et al.*, "WavLM: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, 2022.

[28] J.-c. Chou, C.-c. Yeh, and H.-y. Lee, "One-shot voice conversion by separating speaker and content representations with instance normalization," in *Proc. Interspeech*, 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 *Proc. ICASSP*. IEEE, 2021, pp. 5954–5958.

[30] S.-H. Lee, J.-H. Kim, H. Chung, and S.-W. Lee, "VoiceMixer: Adversarial voice style mixup," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, 2021, pp. 294–308.

[31] Y. You, J. Li, S. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel, K. Keutzer, and C.-J. Hsieh, "Large batch optimization for deep learning: training BERT in 76 minutes," in *Proc. International Conference on Learning Representation*, 2020.

[32] J.-H. Kim, S.-H. Lee, J.-H. Lee, and S.-W. Lee, "Fre-GAN: Adversarial frequency-consistent audio synthesis," in *Proc. Interspeech*, 2021, pp. 2197–2201.

[33] A. Kittur, E. H. Chi, and B. Suh, "Crowdsourcing user studies with mechanical turk," in *Proc. ACM CHI*, 2008, pp. 453–456.