The SRCB-SL system for Blizzard Challenge 2021

Chunhui Lu[1], *Xue Wen*[1], *Ruolan Liu*[1], *Xiaoyan Lou*[1], *Liming Song*[1], *June Sig Sung*[2], *Gunu Jho*[2], *Hyoungmin Park*[2]

[1]Samsung Research China-Beijing (SRC-B), China
[2]Samsung Electronics, South Korea
{chunhui.lu}@samsung.com

Abstract

This paper presents the SRCB-SL text-to-speech system that participated in Blizzard Challenge 2021. This year's Challenge was in European Spanish and had come with 5 hours of clean speech data from a female native speaker. It included two tasks: a hub task that asked the participant to build a voice from the provided data and synthesize all-Spanish speech, and a spoke task in which the target speech contained a few English words. Our system featured a text analysis - acoustic model - vocoder pipeline. The text analyzer combined several old and new function modules to convert input text to a sequence of Spanish phonemes with prosodic boundary (break) markers. English phonemes were mapped to their Spanish counterparts in spoke task. The acoustic model was built around FastSpeech, and converted the phoneme sequences from text analysis to mel-spectrograms. For vocoder we used HiFi-GAN, which we trained on Challenge data and fine-tuned using predicted mel-spectrogram as input. This same system was used for both tasks. Challenge results showed that our system (identified as K) worked well by most of the criteria, which validated the effectiveness of our method.

Index Terms: Blizzard Challenge 2021, FastSpeech, HiFi-GAN

1. Introduction

The annual Blizzard Challenges are held to help understand and compare research techniques in building corpus-based speech synthesizers on the same data. The Challenges have witnessed the progress of text-to-speech (TTS) technology since 2005: From unit concatenation [1, 2] and hidden Markov model (HMM) based statistical parametric speech synthesis (SPSS) [3, 4] to latest end-to-end systems [5, 6]. In the past two years we have seen acoustic models based on Tacotron [7] and Tacotron2 [8] dominate the Challenge while neural vocoders like WaveNet [9], WaveRNN [10] and LPCNet [11] supercede conventional methods like Griffin-Lim (GL) [12], STRAIGHT [13] and WORLD [14].

This year's Challenge had two tasks:

• Hub task (2021-SH1): Synthesize speech from texts containing only Spanish words.

• Spoke task (2021-SS1): Synthesize speech from Spanish texts containing a small number of English words in each sentence.

About 5 hours of European Spanish speech data from a female native speaker was provided to build the voice from. 10 natural recordings of Spanish-with-English-words sentences were provided to help explain task SS1. These examples showed that the English words were expected to carry Spanish accent.

We participated in both tasks. Our system included three parts: text analyzer, acoustic model, and vocoder. The text analyzer converted input text to a sequence of Spanish phonemes with prosodic boundaries. English words in task SS1 were first converted to English phonemes then mapped to the Spanish phone set. The acoustic model was based on FastSpeech [15], which we augmented with phoneme-level latent features to capture local prosodic variations. This differed from FastSpeech2 [16], which used frame-level, visible prosodic features (pitch and energy). The acoustic model took Spanish phoneme sequence and prosodic boundaries as input to predict frame-level acoustic features. For vocoder we used HiFi-GAN [17] to reconstruct waveform audio, which offered good balance between efficiency and quality. We used the same system in both tasks, with a binary flag to tell the system to bypass English-related submodules in task SH1.

In the following sections we describe our system and workflow in detail, and briefly summarize our results.

2. System description

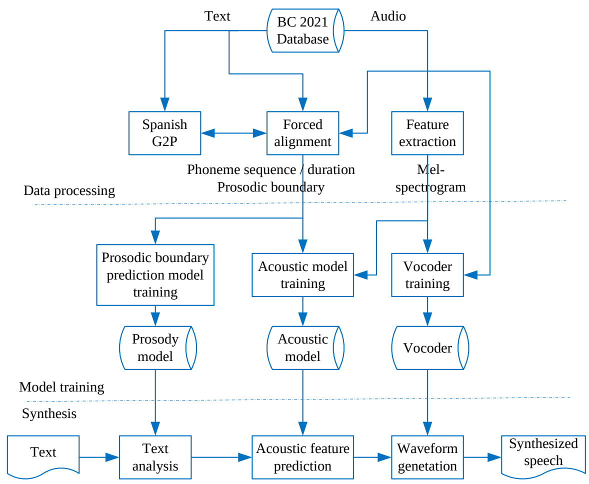

Figure 1: *The overview of our workflow*

Figure 1 gives an overview of our workflow, which included data processing, model training and synthesis stages. In data processing stage we extracted mel-spectrograms and prepared annotations for phoneme sequences, phoneme durations, and prosodic boundaries. In training stage we trained several data-driven models, including prosody (break and phoneme duration) prediction model, main acoustic model, and the vocoder, using only officially provided data and labels derived from it.

In synthesis stage we chained up all submodules into an automated pipeline to produce the final results for submission. Open source resources used in different stages are listed in table 1.

Table 1: *Open source resources used when building our system*

Resource	Stage
SoX [18]	data processing
Librosa [19]	data processing
Montreal Forced Aligner (MFA) [20]	data processing
MFA open source models	data processing
Pretrained Spanish BERT model	model training
Pretrained Multilingual BERT model	model training
HiFi-GAN official implementation	model training
FastText language identification model	synthesis
g2p-en python package	synthesis

2.1. Data processing

2.1.1. *mel spectrogram*

Data provided with the Challenge included 4920 audio files at 48 kHz sample rate, and corresponding text scripts. We first down-sampled the audio files to 24 kHz using SoX, and trimmed leading and trailing silences beyond 0.1s and 0.2s, respectively, using Librosa. We extracted spectrogram using FFT size 2048, hop size 12.5ms, and window size 50ms. This was then converted to mel-spectrogram with 80 frequency bands.

2.1.2. *Phoneme sequence*

We used an internel Mexican Spanish grapheme-to-phoneme (G2P) conversion module to generate phoneme sequences for Challenge data. This was a letter-to-sound model based on the classification and regression tree (CART) [21]. For each letter in a word the model predicts the corresponding phoneme deterministically according to the letter itself and three contextual letters on either side (7 in total).

2.1.3. *Phoneme duration*

The phoneme duration label tells the length of each phoneme at frame (12.5ms) precision. To prepare this label we performed forced alignment using the MFA toolkit with their open source Spanish acoustic model[1]. This forced aligner assigns frames to phonemes or silences, from which we derived phoneme duration labels. MFA used a phone set that slightly differed from ours (MFA 40 phonemes vs. ours 34 phonemes), primarily in that MFA used diphthongs and we did not. Accordingly we split MFA diphthongs into monophthongs, so that we could directly map MFA forced alignment results to our format. Running MFA required specifying a pronunciation dictionary, which we generated using MFA's own G2P model[2]. We ran an automated routine to check for inconsistency between MFA's G2P model and ours on Challenge scripts, and manually corrected found differences according to actual pronunciation. The corrections were written into the pronunciation dictionary before running the forced aligner.

[1] https://raw.githubusercontent.com/MontrealCorpusTools/mfa-models/main/acoustic/spanish.zip

[2] https://raw.githubusercontent.com/MontrealCorpusTools/mfa-models/main/g2p/spanish_g2p.zip

2.1.4. *Prosodic boundary*

TTS systems widely employ a hierarchical prosodic structure to distinguish different levels of breaks in a sentence. The prosodic boundary label tells how much break is expected at each word boundary. We annotated the prosodic boundary at each word at one of three levels: no break, short break and long break, according to the silence duration after that word as reported from forced alignment (2.1.3).

2.2. Model architectures and training

2.2.1. *Break predictor*

The break predictor predicts the prosodic boundary type (one of no break, short break, long break) of every word from text. We used the Bidirectional Encoder Representations from Transformers (BERT) [22] model for this job. More concretely, we took two open source pre-trained BERT models, one for Spanish text [23][3] which we call BERT-ES, the other for multilingual cased text[4] which we call BERT-ML. We fine-tuned both BERT-ES and BERT-ML on Challenge scripts to predict the prepared prosodic boundary labels, supervised with cross-entropy objective.

2.2.2. *Acoustic model*

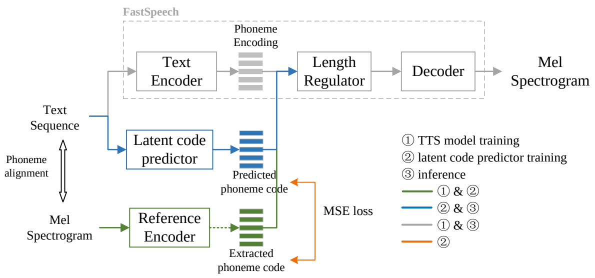

Figure 2: *Acoustic model structure in different stages: ① TTS model training - grey and green parts, ② Code predictor training - green, blue and red parts, ③ Inference - grey and blue pars. Dashed green lines denote sampling via reparameterization [24] in TTS model training.*

Following our previous work [25], we built a TTS model based on a variant of FastSpeech, as shown in Figure 2. Its core part was an encoder-decoder DNN that converts a sequence of phonemes to a sequence of mel spectral frames, using a length regulator to match their lengths by repeating encoder outputs. We followed [15] to predict phoneme durations in log domain, trained supervised using prepared phone duration labels. We augmented the basic FastSpeech model by introducing phoneme-level latent variables that conceptually captured unaccounted-for local prosodic variations. The latent code joined the main FastSpeech network at encoder output to condition the decoder. We learned the latent code with the variational autoencoder (VAE) framework. A reference encoder computed a variational posterior of the latents from phoneme-aligned spectrogram, from which a latent code was drawn and appended to the phoneme encoding before sending to the length regulator. The objective function was formulated as an evidence

[3] https://github.com/dccuchile/beto

[4] https://github.com/google-research/bert

lower bound (ELBO) of expected reconstruction loss:

$$\mathcal{L} = \mathcal{L}_{ELBO} \tag{1}$$

$\mathcal{L}_{ELBO}$ is actually a β-VAE objective [26] under standard Gaussian latent prior:

$$\begin{aligned}&\mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{y},\mathbf{z})]\\&-\lambda_{KL}\sum_{u=1}^{U} D_{KL}(q(\mathbf{z}_u|\mathbf{x}))\|\mathcal{N}(0,I))\end{aligned} \tag{2}$$

where $\mathbf{z}$ represents the sequence of latents and $\mathbf{z}_u$ is the latent code for the u-th phoneme. U is the number of phones. We used $0<\lambda_{KL}<1$, which favors accuracy over latent space exploration.

After this enhanced FastSpeech model was trained, we collected mean latent codes for every phoneme in corpus using the reference encoder, and trained a separate code predictor to predict them from text under the mean square error (MSE) loss. This model were later used to provide the latent code during synthesis.

Details of the FastSpeech network were same as [15]. The reference encoder closely followed that in [27] and extracted a 3-dim latent code for each phoneme. We set λ_{KL} to 0.01. Training the TTS engine took about 300k steps at batch size 16 on one NVIDIA P40 GPU.

2.2.3. Vocoder

We used HiFi-GAN for reconstructing speech waveform from mel-spectrogram. The model was trained using official implementation[5] with modifications to match our 24 kHz sample rate, as listed in Table 2. We further fine-tuned the vocoder using mel-spectrogram predicted from the acoustic model as input, to make up for the mismatch between ground truth and predicted mel-spectrograms.

Table 2: *Configurations modified when training 24 kHz HiFi-GAN vocoder*

Configuration	
upsample rates	[6,5,5,2]
upsample kernel size	[12,10,10,4]
segment size	9600
hop size	300
win size	1200
fmax	12000

2.3. Synthesis

In the synthesis stage we chained up old and new modules into an automated pipeline for text-to-speech generation. In summary, the text analyzer converted text to phoneme sequence and break markers; the acoustic model converted break-marked phoneme sequence to mel-spectrogram; the vocoder converted mel-spectrogram to speech audio. The acoustic model and vocoder were shared by SH1 and SS1 tasks. The text analyzer was task-dependent, due to different input domains, as shown in table 3.

[5]https://github.com/jik876/hifi-gan

Table 3: *Task-dependent text analyzer*

Submodules	**Task SH1**	**Task SS1**
Break prediction	BERT-ES	BERT-ML
Spanish word G2P	CART-based	CART-based
Language identify	-	open source model
English word G2P	-	g2p-en & phoneme mapping

2.3.1. Text analysis

Text analysis included G2P conversion and break prediction. Both modules received raw text as input. For task SH1 we used the same CART-based G2P as in section 2.1.2 and BERT-ES for break prediction. For task SS1 we used BERT-ML for break prediction, but we had no dedicated module for mixed-lingual G2P. Our strategy was to map English phonemes to Spanish counterparts, as explained below.

Table 4: *English (EN) phonemes to Spanish (ES) phonemes mapping. * is a vowel phoneme determined by Spanish letters*

EN	**ES**	**EN**	**ES**	**EN**	**ES**
AA0/2	a	ER0/2	* r	OW1	o1 w
AA1	a1	ER1	*1 r	OY0/2	o j
AE0/2	a	EY0/2	e j	OY1	o1 j
AE1	a1	EY1	e1 j	P	p
AH0/2	*1	F	f	R	r/rr
AH1	*	G	G/g	S	s/T
AO0/2	o	HH	x	SH	s/tS
AO1	o1	IH0/2	i	T	t
AW0/2	a w	IH1	i1	TH	T
AW1	a1 w	IY0/2	i	UH0/2	u
AY0/2	a j	IY1	i1	UH1	u1
AY1	a1 j	JH	x	UW0/2	u/w
B	B/b	K	k	UW1	u1
CH	tS	L	L/l	V	B/b
D	D/d	M	m	W	w
DH	D	N	n	Y	j/jj
EH0/2	e	NG	n	Z	s/T
EH1	e1	OW0/2	o w	ZH	s/x

Given an input text we first identified the English words from Spanish words using an open source language identification model [28, 29] [6]. This model predicts a probability distribution over languages for each word. If the predicted probability of the word being English was higher than Spanish, and the value was above a threshold, we identified it as a English word, otherwise it was Spanish. G2P for Spanish words was same as in task SH1. For English words, we used a python package 'g2p-en' [7] to get English G2P results, then mapped the English phonemes to Spanish counterparts according to table 4. Mapping rules in this table were composed by informally comparing the phone sets and listening to example SS1 sentences provided by the Challenge. Most phonemes made a one-to-one mapping, but some English phonemes might map to multiple or no Spanish counterparts. In this case the English letters corresponding to the phoneme in question were regarded as Spanish

[6]https://fasttext.cc/docs/en/language-identification.html
[7]https://pypi.org/project/g2p-en/

letters, and one-to-one mapped to Spanish phonemes according to basic pronunciation rules.

3. Evaluation results

12 teams submitted results for task SH1 and 10 teams submitted results for task SS1. Our system was identified as K. Submitted speech examples from the participants and natural speech (identified as R) were evaluated by three groups of listeners, including paid participants (denoted as SP) who are native Spanish speakers, self-identified speech experts, and volunteers.

The evaluation comprises 6 sections and includes 3 metrics, detailed in Table 5.

Table 5: *Evaluation sections and metrics for each task*

Section	Task SH1	Task SS1
section1	Similarity	Similarity
section2	Similarity	Naturalness
section3	Naturalness	Naturalness
section4	Naturalness	Acceptability
section5	Intelligibility	Acceptability
section6	Intelligibility	Acceptability

3.1. Naturalness and similarity

Naturalness and similarity were evaluated for both tasks. When evaluating naturalness, listeners were asked to choose a score which represented how natural or unnatural the sentence sounded on a scale of 1 [Completely Unnatural] to 5 [Completely Natural]. For similarity, a score that represented how similar the synthetic voice sounded to the voice in the reference samples on a scale from 1 [Sounds like a totally different person] to 5 [Sounds like exactly the same person] was chose. Figure 3 and Figure 4 show the scatter plot matching naturalness and similarity scores for task SH1 and task SS1, respectively.

In task SH1, our system achieved MOS of 3.67 and similarity score of 3.71. System F was obvious better than ours and on par with natural speech. Comparing our system to system J, G, and I, there was no significant difference. In task SS1, the MOS of our system was 3.68 and the similarity score was 4.16, which was higher than that of task SH1 though we used one system in both tasks.

The results validated the effectiveness of our method since we didn't understand Spanish and couldn't make any optimization in terms of language related issues during system development. We notice the MOS of SP listeners was 0.5 lower than that of other listeners, which may indicate the importance of language knowledge when building TTS system.

3.2. Intelligibility

Intelligibility was only evaluated for task SH1. Listeners were asked to listen to each sentence only once and type in what they heard. The word error rate (WER) of our system for Sharvard test was 4.5%, which was on par with natural speech, and for semantically-unpredictable sentences (SUS) test was 14%. Though the WER of our system was not the lowest, there was no significant difference compared our system to systems with lower WER.

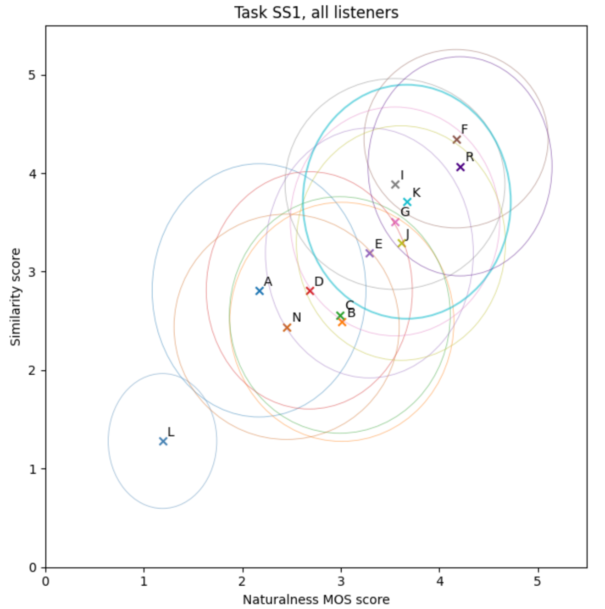

Figure 3: *Scatter plot matching naturalness and similarity scores for task SH1. K is our system.*

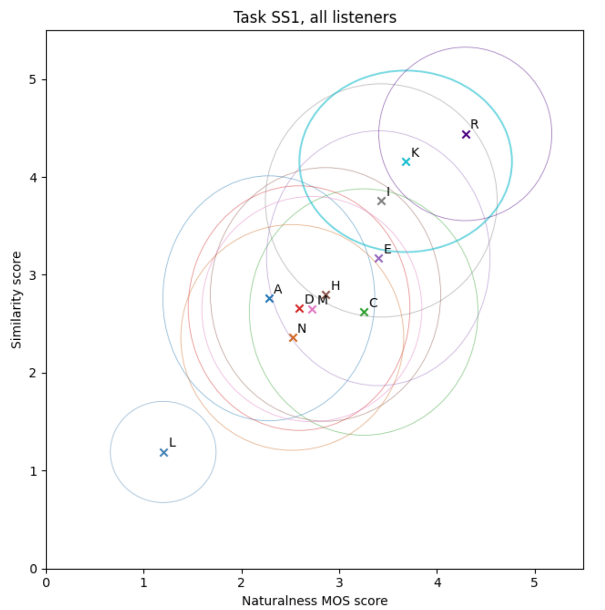

Figure 4: *Scatter plot matching naturalness and similarity scores for task SS1. K is our system.*

3.3. Acceptability

Acceptability was only evaluated for task SS1. Listeners were asked to choose a score that represented how acceptable or unacceptable of the English words in the sentence sounded on a scale from 1 [Not Intelligible] to 5 [Perfect].

The acceptability score of our system was 3.41, which suggests that our strategy mapping English phonemes to Spanish phonemes was acceptable. However, there was still a big gap between our system and natural speech. We may attribute it to the lack of language knowledge. We inferred the phoneme mapping rules only from 10 reference speech samples, which may result in wrong pronunciations.

4. Conclusions

In this paper, we present our TTS system developed for Blizzard Challenge 2021. The system was built following text analysis - acoustic model - vocoder pipeline. The text analyzer converted input text to a sequence of Spanish phonemes with prosodic boundary markers. English phonemes were mapped to their Spanish counterparts in spoke task. The acoustic model was built based on FastSpeech with fine-grained prosody modelling to capture local prosodic variations, followed by a HiFi-GAN vocoder. The same system was used for both tasks. Evaluation results showed that our system worked well by most of the criteria, but there was still much room for improvement in naturalness and English words acceptability, in which language knowledge may play an important role.

5. References

[1] R. Clark, K. Richmond, V. Strom, and S. King, "Multisyn voice for the blizzard challenge 2006," in *Blizzard Challenge Workshop*, 2006.

[2] K. Richmond, V. Strom, R. A. Clark, J. Yamagishi, and S. Fitt, "Festival multisyn voices for the 2007 blizzard challenge," in *Blizzard Challenge Workshop*, 2007.

[3] P. Scholtz, A. Visagie, and J. d. Preez, "Statistical speech synthesis for the blizzard challenge 2008," in *Blizzard Challenge Workshop*, 2008.

[4] Y.-F. Liao and M.-L. Wu, "The NTUT blizzard challenge 2009 entry," in *Blizzard Challenge Workshop*, 2009.

[5] J. Tao, R. Fu, and Z. Wen, "The NLPR speech synthesis entry for blizzard challenge 2019," in *Blizzard Challenge Workshop*, 2019.

[6] L. He, Q. Shi, L. Wu, J. Sun, R. He, Y. Long, and J. Liang, "The SHNU system for the blizzard challenge 2020," in *Blizzard Challenge Workshop*, 2020.

[7] Y. Wang, R. J. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. V. Le, Y. Agiomyrgiannakis, R. Clark, and R. A. Saurous, "Tacotron: Towards end-to-end speech synthesis," in *Interspeech*, 2017, pp. 4006–4010.

[8] 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 *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4779–4783.

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

[10] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," in *International Conference on Machine Learning (ICML)*, 2018, pp. 2410–2419.

[11] J.-M. Valin and J. Skoglund, "LPCNet: Improving neural speech synthesis through linear prediction," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 5891–5895.

[12] D. Griffin and J. Lim, "Signal estimation from modified short-time fourier transform," *IEEE Transactions on acoustics, speech, and signal processing*, vol. 32, no. 2, pp. 236–243, 1984.

[13] H. Kawahara, I. Masuda-Katsuse, and A. De Cheveigne, "Restructuring speech representations using a pitch-adaptive time–frequency smoothing and an instantaneous-frequency-based f0 extraction: Possible role of a repetitive structure in sounds," *Speech communication*, vol. 27, no. 3-4, pp. 187–207, 1999.

[14] M. Morise, F. Yokomori, and K. Ozawa, "WORLD: a vocoder-based high-quality speech synthesis system for real-time applications," *IEICE TRANSACTIONS on Information and Systems*, vol. 99, no. 7, pp. 1877–1884, 2016.

[15] 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 *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 3171–3180.

[16] Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech 2: Fast and high-quality end-to-end text to speech," in *International Conference on Learning Representations (ICLR)*, 2020.

[17] J. Kong, J. Kim, and J. Bae, "HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 17 022–17 033.

[18] "Sox, audio manipulation tool," http://sox.sourceforge.net.

[19] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, vol. 8, 2015, pp. 18–25.

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

[21] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*, 1984.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[23] J. Cañete, G. Chaperon, R. Fuentes, J.-H. Ho, H. Kang, and J. Pérez, "Spanish pre-trained bert model and evaluation data," in *PML4DC at ICLR 2020*, 2020.

[24] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations (ICLR)*, 2014.

[25] C. Lu, X. Wen, R. Liu, and X. Chen, "Multi-speaker emotional speech synthesis with fine-grained prosody modeling," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 5729–5733.

[26] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-VAE: Learning basic visual concepts with a constrained variational framework," in *International Conference on Learning Representations (ICLR)*, 2017.

[27] R. J. Skerry-Ryan, E. Battenberg, Y. Xiao, Y. Wang, D. Stanton, J. Shor, R. J. Weiss, R. Clark, and R. A. Saurous, "Towards end-to-end prosody transfer for expressive speech synthesis with tacotron," in *International Conference on Machine Learning (ICML)*, 2018, pp. 4700–4709.

[28] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov, "Fasttext.zip: Compressing text classification models," *arXiv preprint arXiv:1612.03651*, 2016.

[29] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," *arXiv preprint arXiv:1607.01759*, 2016.