The SCUT Text-To-Speech System for the Blizzard Challenge 2023

Qiaowei Ma¹, Weiheng Liu¹, Yitao Yang¹, Chao Xu¹, Hongtao Ling¹, Jinghui Zhong¹

¹South China University of Technology, China

Abstract

In this paper, we present the SCUT Text-To-Speech (TTS) system developed for the Blizzard Challenge 2023. Our system utilizes the FastSpeech2 acoustic model and the HiFiGAN vocoder to generate high-quality speech synthesis. We incorporate a prosody extractor based on VQ-VAE to enhance the expressive prosody of the synthesized speech. Experimental results demonstrate the effectiveness of our system in producing natural and fluent speech synthesis.

Index Terms: Blizzard Challenge 2023, speech synthesis, prosody modeling

1. Introduction

The Blizzard Challenge is an annually held seminar to evaluate corpus-based speech synthesizers on common databases. In this year, the challenge focuses on the French Text-to-Speech synthesis tasks, containing the following ones: (1) Hub task to build a French TTS system given 50 hours of speech data. (2) Spoke task to build a Voice that is closest to the provided 2 hours of speech data. We select the Hub task to build an expressive TTS system, which involves prosody modeling based on the semantic information from the pretrained language model.

The neural network-based Text-to-Speech method has been continuously breaking the boundary between human speech and robots. One of the earlier and most representative works is Tacotron [1, 2], which is characterized by the attentive encoder-decoder recurrent network. However, Tacotron encounters shortages of robustness and controllability, as well as low inference speed due to the autoregressive design. In recent years, Ren [3, 4] proposed a non-autoregressive model FastSpeech, which could speed up the inference by the parallel structure and also achieve competitive quality compared to the previous one. Thus, we choose the FastSpeech as the baseline model for the challenge.

Aiming at improving the naturalness of synthesized speech, the prosody modeling technique is a vital topic of TTS. The main challenge is that the fine-grained prosody information in speech audio is latent, which means that it could not be precisely labeled by humans. One of the approaches is to use Variational Autoencoder (VAE) to extract the hidden information from speech [5, 6, 7]. However, the VAE will often encounter the posterior collapse problem, which may increase the difficulty of model training. Chien et al. [8] proposed to use the Vector Quantization-VAE (VQ-VAE) [9] to extract the latent prosody information. The discrete space of VQ-VAE serves as an information bottleneck, allowing it to not only address the posterior collapse problem [9] but also achieve a higher quality in prosody naturalness [8]. In this paper, we mainly follow the method proposed in [8] to be the prosody modeling method in our system.

2. Method

2.1. Overall Architecture

Since we chose the Hub task, which involves only one speaker, we did not include speaker information in the model. In the front-end module, we extracted phonetic features of the sentences as input for the model. Additionally, we added prosodic features of the sentences to train the model and improve the quality of the generated speech. The structure of the model is shown in Figure 1. We selected FastSpeech2 [4] as the acoustic model, which consists of an encoder, a variance adapter, and a Mel-spectrogram decoder. Specifically, the encoder further extracts linguistic features from the phonetic features, the variance adapter models factors such as speech pitch and duration, and the Mel-spectrogram decoder converts intermediate feature maps into the final output, namely the Mel-spectrogram. Finally, we use HiFi-GAN, the vocoder, to convert Mel-spectrograms into waveform.

2.2. Data Selection

The Blizzard Challenge committee provides a French dataset of a female speaker, reading out the texts from several books characterized by rich prosody. We only use this material to model the French linguistic features in the acoustic model. In addition, in order to sufficiently train the vocoder, we externally utilize the IEMOCAP dataset [10] in the training phase of HiFi-GAN. There are 5 female and 5 male speakers in IEMOCAP dataset.

All wave files are resampled to 22.05kHz. In order to model the pause between sentences in the paragraph, we only trimmed the heading and trailing silence of each paragraph. Table 1 shows the dataset information.

Table 1: Dataset information

<table>
<thead>
<tr>
<th>dataset</th>
<th>speaker</th>
<th>utt.</th>
<th>para.</th>
<th>dur/hrs</th>
</tr>
</thead>
<tbody>
<tr>
<td>BC2023</td>
<td>1</td>
<td>44029</td>
<td>8632</td>
<td>51.5</td>
</tr>
<tr>
<td>IEMOCAP</td>
<td>10</td>
<td>4490</td>
<td>-</td>
<td>5.6</td>
</tr>
</tbody>
</table>

2.3. Front-end processing

This module aims to convert French characters to phonemes. In the training phase, we used Montreal Forced Aligner (MFA) to align speech data. Alignment refers to the process of finding the corresponding timestamps for each word or phoneme in the audio file. The alignment process requires a French dictionary, speech data, and text data. We use the pretrained dictio-
The overall architecture of our proposed method is as follows: (a) denotes the flow of our text-to-speech generation method, which mainly includes the acoustic model FastSpeech2 and the vocoder HifiGAN, (b) denotes the variances predictor used in the model, and (c) denotes the structure of the variance predictor and LN representing layer normalization.

Grapheme-to-phoneme(G2P) model [12] and acoustic model [13] provided by MFA. After using the MFA tool, we can convert the text into corresponding phoneme sequences. However, during the inference time, in order to better tackle the homographs, we switch to using the Phonemizer [14] as the G2P tool.

We simply replace the punctuation to phoneme ‘sil’ to model the pause between words and sentences.

2.4. Encoder

After the front-end processing, the phoneme embedding is fed into the transformer layer for further transformation into a hidden state. Specifically, the extracted embedding goes through multiple layers of multi-headed attention mechanisms and feed-forward neural networks. The multi-headed self-attentive mechanism assigns different weights to different parts of the input information to help the model focus on the key information in the input. The feedforward neural network, on the other hand, is responsible for the nonlinear transformation of processed information.

2.5. Variance Adaptor

The variance adaptor of FastSpeech2 employs a duration predictor, pitch predictor, and energy predictor to predict the duration of phonemes, the pitch of the audio, and the energy, respectively. For the prosody modeling, we follow the method proposed in [8], which adds a prosody predictor to the original variance predictor to enhance the expressive prosody of the synthetic speech. All the variance predictors are structures in Figure 1. (c), which consists of a 2-layer 1D-convolutional network with ReLU activation, each followed by layer normalization and the dropout layer, and an extra linear layer to project the hidden states into the output sequence.

2.5.1. Duration predictor

The Montreal Forced Aligner (MFA) tool is used to extract the ground truth phoneme duration from the audio. And the predictor is optimized by using mean square error (MSE) loss with the extracted duration as the training target.

2.5.2. Pitch predictor

The pitch predictor requires the pitch information of the speech as the training target. In FastSpeech2, the extracted pitch spectrogram is used as the training target. First, continuous wavelet transform (CWT) was used to decompose the continuous pitch series into pitch spectrogram, and then the predictor is trained to predict it. When synthesizing speech, the inverse CWT (iCWT), i.e., the inverse operation of CWT, is used to convert the pitch spectrogram into a pitch contour. In order to incorporate the pitch contour as input during both training and inference, a quantization process was used on the pitch F0 of each frame. This quantization maps the pitch F0 onto a logarithmic scale with 256 possible values. Subsequently, the quantized pitch F0 was converted into a pitch embedding vector, which is then added to the expanded hidden sequence.

2.5.3. Energy predictor

The Energy predictor first calculates the L2-norm of the amplitude of each short-time Fourier transform (STFT) frame as the energy. Then they are mapped to 256 values, encoded into energy embedding and the corresponding embedding is finally added to the hidden state. During training, the predictor directly predicts the original values of energy before mapping.

2.5.4. Prosody predictor

The model structure of the prosody predictor is the same as the other three predictors. The training process of the prosody predictor is shown in Figure 2. During training, fine-grained prosody labels are extracted from the ground-truth mel-spectrogram by the prosody extractor. The detail of the extractor is based on the Vector Quantised-Variational Autoencoder (VQ-VAE) [9], which uses the Vector Quantization (VQ) method to model the latent space with discrete variables in the VAE. We follow the architecture and the training strategy pro-
Figure 2: The training process of Prosody predictor, firstly, the extracted word embedding is input to prosody predictor, and the predicted labels are output by predictor to compare with the real labels extracted by prosody extractor, and the MSE loss is used to optimize the model.

posed in [8]. Briefly, in the training time, as the orange path in Figure 2, the prosody extractor extracts the latent prosody labels from the ground truth Mel-spectrogram, which is the ground truth prosody label. Meanwhile, these labels will also be utilized to train the prosody predictor, which generates the predicted prosody label from the pre-trained language model. In the inference time, as the green path in Figure 2, the model only uses the predicted prosody label output by the trained prosody predictor. In this way, the predictor could independently predict the prosody label only based on the semantic information in the word embedding.

We did not pay much attention to the syntactic analysis and the discourse analysis for expressive text-to-speech modeling. We just simply feed the whole paragraph to the pretrained BERT to provide this system minimal information of sentence-level context. The specific implementation is illustrated in Fig 3, the entire paragraph is input into the pre-trained BERT model to obtain BERT’s word embeddings. And then separate these embeddings based on the sentence. After processing in this way, each sentence contains paragraph information.

After getting the word level prosody label, each of them will expand to the size according to its phoneme number. Finally, the output prosody embedding will be added to the phoneme embedding.

2.6. Mel-spectrogram Decoder
The Mel-spectrogram decoder decodes the output of the variance adaptor, using it as input for several transformer layers. Finally, the decoded information is mapped to the mel-spectrogram through a linear layer.

2.7. HiFi-GAN Vocoder
The vocoder converts mel-spectrogram into raw waveform. To enhance the discriminator’s ability to distinguish synthesized and real audio, HiFi-GAN utilizes both a multi-scale discriminator and a multi-period discriminator. The multi-scale discriminator continuously performs average pooling on the speech sequence, gradually reducing its length. Then, it applies several convolutional layers on different scales of speech and flattens the output to serve as the output of the multi-scale discriminator. On the other hand, the multi-period discriminator folds the output of the multi-scale discriminator. The output of the HiFi-GAN is a two-dimensional feature map with a certain period and performs two-dimensional convolution on this feature map. Additionally, HiFiGAN incorporates a multi-receptive field fusion module in the generator to further improve the quality of the generated audio.

3. Experiment
The training of the aforementioned system mainly has two parts: the acoustic model and the vocoder. For the acoustic model, we only use the BC2023 dataset to train the Fastspeech2-based model, attached with the VQ-VAE-based prosody module. We randomly pick 600 split samples for validation set and the rest ones are for training set. For the model configuration of FastSpeech2, we set the encoder to 4 feed-forward Transformer blocks in the encoder, and 6 ones for the decoder. The size of French phoneme vocabulary is 45, including 2 punctuations “sil” and “sp”. The filter size of the prosody extractor is set to 64, with kernel sizes of 9 and 5. The prosody label in the VQ-VAE codebook is of size 256 with dimensions 3. The prosody predictor is in the same setting as the variance predictor in FastSpeech2. And we use the FlauBERT [15] as the pretrained language model to generate the word embedding. The remaining model architecture follows the setting in [4]. The total number of parameters for the acoustic model is 36.2M. For the training, the batch size is set to 32. The optimizer and the learning rate schedule are the same as the [4]. We trained the acoustic model for 440K steps, during which both the training and validation loss continuously declined.

For the vocoder, we follow the principle of fine-tuning. We pretrain the HiFi-GAN from scratch on the BC2023 dataset and the IEMOCAP dataset by feeding the ground truth Mel-spectrogram. In this stage, 5000 split samples are randomly picked for validation. And we finetune the vocoder with the synthesis Mel-spectrogram output by the trained Fastspeech2. In this stage, only 600 samples from BC2023 dataset are used for validation. During the training, we set the batch size to 32, with a learning rate of 0.0002. the parameters of the model architecture follow the setting of HiFi-GAN-V1 in [16]. The total number of parameters for the vocoder is 13.9M. The pretraining phase lasted for 2.2M steps, and 300K steps for fine-tuning.

All the experiments are run on one AMD Ryzen Threadripper 3975WX CPU. We assign one NVIDIA GTX3090 GPU to train the acoustic model and two of the ones to train the HiFi-GAN. The OS is Ubuntu 20.04 with Python 3.9 and Pytorch 1.13 installed. The average inference speed of the whole system

Figure 3: The process of extracting word embeddings using pretrained BERT.
is 610.5kHz, where \( nk\)kHz means \( n\) thousand raw audio frames per second. In other word, the system inference speed is 27.7 times real-time.

4. Results

4.1. Evaluation

The main evaluation metrics for the FH1 task are MOSquality, MOSsimilarity, MUSHRAquality, SUSintelligibility, and HOMOSintelligibility. Among them, the primary goal of MOSquality and MUSHRAquality is to assess the audio quality of the speech synthesis system. MOSsimilarity aims to evaluate the similarity between the audio samples and the reference speaker. SUSintelligibility focuses on evaluating the intelligibility of the speech samples, while HOMOSintelligibility assesses the intelligibility of homographs in speech synthesis.

4.2. Discussion

From the evaluation metrics, we can see that our system can synthesize speech with a certain degree of naturalness and intelligibility. The speaker’s timbre is quite similar to that of a real person. However, although we successfully incorporated prosody information compared to our work in the last challenge [17], our score on the MOS metric is still lower than our expectations. Possible reasons are: First, due to the lack of a native French speaker in the team, it is possible that the prosody generated by our model is still relatively flat and not expressive enough. Second, we did not put too much effort into the pauses in syntax, which may have greatly affected the naturalness of the speech. Third, in the processing of homographs, the word error rate still needs to be further reduced, and the G2P module needs to be improved more finely.

5. Conclusions

In this paper, we provide a detailed description of our participating model. The overall architecture of our model is based on FastSpeech2 and HiFi-GAN. For the speech synthesis task of FH1, we enhanced FastSpeech2’s original variance adaptor by incorporating a prosody predictor that may help with prosody analysis. By adding the prosody predictor’s output embedding to the phoneme embedding, we aimed to improve the speech quality and achieve a better representation of prosody in the generated audio. However, in the evaluation, our model still falls short in several metrics, such as audio naturalness. In future work, we will focus more on enhancing the naturalness of the synthesized audio.

6. Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant 62076098 and in part by the Program for Guangdong Introducing Innovative and Entrepreneurial Teams under Grant 2017ZT07X183.
7. References


