Neural Fusion for Voice Cloning

Bo Chen, *Student Member, IEEE*, Chenpeng Du, *Graduate Student Member, IEEE*, and Kai Yu, *Senior Member, IEEE*

***Abstract*—Voice cloning is a technique to build text-to-speech applications for individuals. When only very limited training data is available, it is challenging to preserve both high speech quality and high speaker similarity. We propose a neural fusion architecture to incorporate a unit concatenation method into a parametric text-to-speech model to address this issue. Unlike the hybrid unit concatenation system, the proposed fusion architecture is still an end-to-end neural network model. It consists of a text encoder, an acoustic decoder, and a phoneme-level reference encoder. The reference encoder extracts phoneme-level embeddings corresponding to the cloning audio segments, and the text encoder infers phoneme-level embeddings from the input text. One of the two embeddings is then selected and sent to the decoder. We use auto-regressive distribution modeling and decoder refinement after the selection stage to overcome the concatenation discontinuity problem. Experimental results show that the neural fusion system significantly improves the speaker similarity using the selected units with the highest probability. The speech naturalness remains similar to the directly decoded systems.**

***Index Terms*—Voice cloning, unit selection, FastSpeech2.**

I. Introduction

VOICE cloning and voice conversion are the two major techniques to build speech generation applications for individuals with small training data. Voice conversion focuses more on acoustic conversion, while voice cloning aims to build a text-to-speech (TTS) model. Voice cloning is usually required to build the TTS model with a small amount of training data for each speaker. The challenge of voice cloning is to guarantee the synthetic speech's speech quality, speaker similarity, expressiveness, and robustness. Much research has been done on voice cloning in recent years. It is summarized in [1] that there are two primary desired abilities of the voice cloning system: to imitate the speaking style of the cloned speaker and to build text-to-speech systems with limited data. For the speaking style extraction approach, some models directly model the prosody aspects, including the rhythms, pauses, intonations, stresses, etc. Directly modeling prosodic aspects of speech is beneficial for stylization [2], [3]. Some models present the global style modeling through variational auto-encoder (VAE) [4], [5] or global style tokens (GST) [6] extracted from reference audios. However, these methods attempt to extract the global style representations from a few audio clips. The representations map the target speaking style to the general speaking styles trained on a large corpus. Therefore, if the target speaker has distinctive prosodies, the similarity of the synthetic speech is not reliable.

Meanwhile, the research on building the target speaker's voice is more popular. Speaker adaptation in text-to-speech has been researched for decades. It is introduced in Hidden Markov Model (HMM) based parametric TTS model [7], [8] and Deep Neural Network (DNN) based TTS model [9], [10]. The average voice model (or multi-speaker model) is usually trained on a large corpus with adequate audios. Recent methods focus on accurately obtaining the speaker identity information using text-speaker disentanglement or phoneme-posteriorgram (PPG) [11]. The vector quantization (VQ) technology [12], [13] and U-net structure [14] are used to disentangle the speaker content and voice information. AdaSpeech [15] is proposed for high-quality and efficient adaptation. Most of the works focus on building voice cloning systems by using a few minutes of audio clips [16], [17]. Since voice cloning is usually used with real-world found audio, low-quality recordings and noisy speech is also an issue [18], [19].

Unlike typical speaker adaptation, the requirements of voice cloning are different, such that usually a small number of model variables are allowed to be updated (so-called "resources" in applications). Moreover, unlike typical recording speech, the collected audios from the clients are much more expressive and have richer speaking styles compared to the reading speech. It is still challenging to model the speaker's identity, including the speaking style, with only several minutes of audio data.

Since the audio data is small in voice cloning tasks, parametric methods are usually adopted. The synthesized audio usually achieves good quality and similarity with the speaker embeddings. However, the similarity relies on the database to train the multi-speaker model and the method to extract the speaker embeddings. The similarity might drop if the speaker's identity is distinctive to the training corpus. Based on these facts, this paper introduces a neural fusion architecture to incorporate a unit concatenation method in a parametric text-to-speech system. Unlike the hybrid unit concatenation system, the proposed neural fusion architecture is still a parametric text-to-speech system with a frame-level spectrogram decoder. The spectrogram decoder takes the text and phoneme-level prosody embeddings as the input and generates the frame-level spectrograms as the final

Manuscript received October 15, 2021; revised February 20, 2022 and April 6, 2022; accepted April 15, 2022. Date of publication May 9, 2022; date of current version June 17, 2022. This work was supported in part by Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX0102, and in part by the State Key Laboratory of Media Convergence Production Technology and Systems Project under Grant SKLMCPTS2020003. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Yu Tsao. *(Corresponding author: Kai Yu.)*

The authors are with X-LANCE Lab, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: bobmilk@sjtu.edu.cn; duchenpeng@sjtu.edu.cn; kai.yu@sjtu.edu.cn).

Digital Object Identifier 10.1109/TASLP.2022.3171971

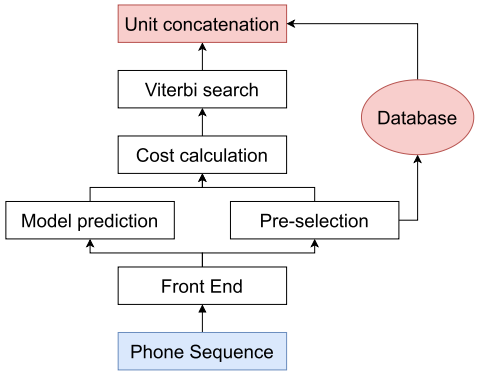


Fig. 1. The architecture of the traditional hybrid unit concatenation system.

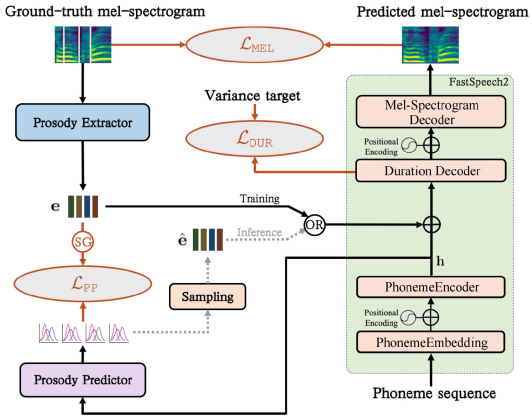


Fig. 2. The architecture of the GMM-based MDN phoneme conditioned FastSpeech2.

outputs. The phoneme-level prosody embeddings are encoded from natural phoneme segments. The distributions of the embeddings are explicitly modeled by a Gaussian mixture model (GMM). The phoneme-level prosody embedding sequences are first generated before the decoder in the inference stage. Then the embedding sequences are decoded into spectrograms. We introduce the fusion system at the prosody embedding level. The fusion method enhances the speaker similarity of the synthesized speech using the phoneme-level prosody embedding units. The output spectrograms are generated using parametric neural networks, but part of the phoneme-level prosody embeddings are the unit embeddings extracted from the real audios. We use an embedding distribution that helps to select the unit embeddings with the highest probability. The distribution is based on our previous work to model the phoneme-level speaking style using GMM [20]. After the unit embeddings are selected, the decoder decodes the mixed inference embeddings and unit embeddings into spectrograms.

The rest of the paper is organized as follows. Section II reviews the related works. Section III describes the method of fusion in the prosody embedding and the decoder refinement procedure to reduce the concatenation problem. Section IV explains the experiments and analyzes the results. Section V provides the conclusion of the work.

II. Related Work

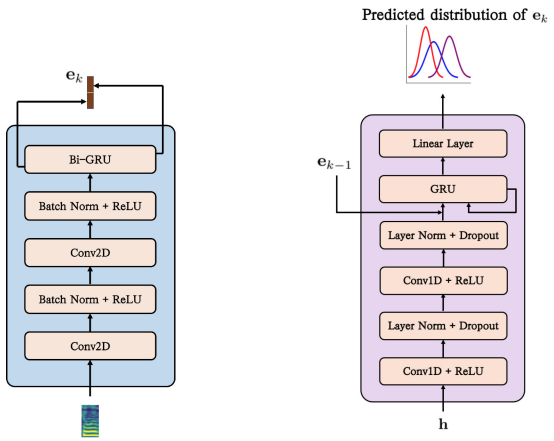
A. Unit Concatenation System

The hybrid unit concatenation system is a popular text-to-speech technique. There have been methods to build a unit concatenation system with the help of parametric models for decades. The hybrid system was first introduced to the HMM-based parametric text-to-speech model [21]. Some works have already combined the concatenative system and statistical systems with HMM models [22], [23]. The HMM-based hybrid system is replaced by the DNN-based hybrid system in [24], [25]. These hybrid systems largely follow the basic structure in Fig. 1 [25]. This method is also applied to end-to-end parametric systems. The unit embeddings are extracted from Tacotron encoders for unit selection text-to-speech system [26]. The UnitNet was recently proposed for concatenation text-to-speech [27]. It extracts a hierarchical architecture for representing phoneme-size units and also measures the dependency among consecutive units. In these end-to-end based concatenation speech synthesis (CSS) models [27], the phoneme-level unit embeddings are extracted from the audios and contexts as phoneme-level unit representations. The phoneme-level representations are then used in the dynamic processing search to select the unit sequence for concatenation. We take a similar approach to extract the unit embeddings using an end-to-end network. Different from that, the unit is introduced to enhance the speaker similarity in the voice cloning. Mixing the units of the parametric model in the synthesis stage has not been investigated before. We first run the selection stage in the prosody embedding level and fuse the unit embeddings with the inference embeddings.

B. Phoneme-Level Conditioned FastSpeech2

FastSpeech2 [28] is one of the state-of-the-art end-to-end text-to-speech models. The phoneme sequences are encoded into phoneme-level hidden embeddings to predict the phoneme-level duration. Then the hidden embeddings are extended to frame-level embeddings using the phoneme duration in order to progressively predict frame-level prosody features (pitch, energy) and spectrograms. This architecture achieves good performance in spectrogram modeling. Since the speakers have different speaking styles for the same phoneme and same text, Du *et al.* proposed a GMM-based mixture density network to predict the parameters of the GMM distribution, including means, variances, and mixture weights based on conditional FastSpeech2 [20]. It is under the assumption that phonemes' prosody is rich and requires sufficient complexity to be modeled. This assumption makes it possible that there are several possible phoneme prosodies that we could find the unit close to a specific prosody of the mixture distribution.

Fig. 2 shows the architecture of the GMM-based phoneme-level prosody modelling in conditioned FastSpeech2 [20] using mixture density network (MDN). The model consists of 3 parts: the phoneme encoder, prosody extractor/predictor and duration/mel-spectrogram decoder. The detail of the prosody extractor and prosody predictor is shown in Fig. 3.



(a) Prosody Extractor (b) Prosody Predictor

Fig. 3. Architecture of the prosody extractor and prosody predictor.

We define the input phoneme sequence as $\boldsymbol{l} = l_1, l_2, .., l_N$ and the mel-spectrograms sequence as $\boldsymbol{o} = o_1, o_2, .., o_T$ where N and T are the sequence lengths. The acoustic features are first assigned to the phonemes by the pre-aligned phoneme durations $\boldsymbol{d} = d_1, d_2, .., d_N$ from the automatic speech recognition (ASR) model. The phoneme-level prosody embeddings $\boldsymbol{e} = e_1, e_2, .., e_N$ are extracted from the ground-truth mel-spectrograms that:

$$\boldsymbol{e} = \texttt{ProsodyExtractor}(\boldsymbol{o}, \boldsymbol{d}). \tag{1}$$

The network encodes the input phoneme sequence into the general hidden phoneme embeddings $\boldsymbol{h} = h_1, h_2, .., h_N$

$$\boldsymbol{h} = \texttt{PhonemeEncoder}(\boldsymbol{l}). \tag{2}$$

The prosody predictor network models the distribution of the phoneme-level prosody embeddings. The distribution $p(\boldsymbol{e}|\boldsymbol{h})$ is modeled under GMM assumption conditioned on the phoneme embedding $\boldsymbol{h}$.

$$p(\boldsymbol{e}|\boldsymbol{h}) = \sum_{k=1}^{K} w_k \mathcal{N}(\boldsymbol{e}|\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k^2). \tag{3}$$

where

$$\boldsymbol{\mu}_k = \mathbf{m}_k, \tag{4}$$

$$\boldsymbol{\sigma}_k^2 = \exp(\mathbf{v}_k). \tag{5}$$

The $w_k, \mathbf{m}_k, \mathbf{v}_k$ are combined as the neural network outputs corresponding to the mixture weight, mean, and variance of the k-th Gaussian components. It should be noted that to constraint the sum of mixture weights w_k to be 1, the output of w_k is rescaled by the Softmax function

$$w_k = \frac{\exp(\alpha_k)}{\sum_{i=1}^{K} \exp(\alpha_i)}. \tag{6}$$

The loss function of the GMM-based MDN prosody predictor is defined as:

$$\mathcal{L}_{\texttt{PP}} = -\log p(\boldsymbol{e}|\boldsymbol{h})$$

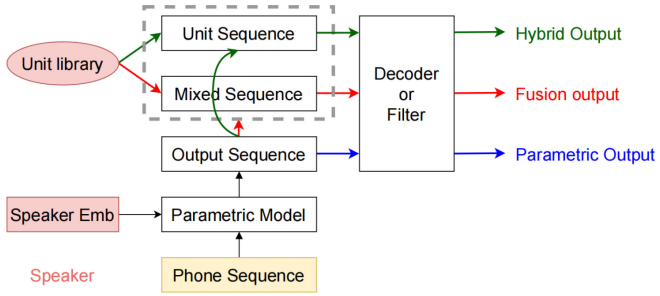


Fig. 4. The brief architecture of the fusion system compared to the hybrid concatenation system and the parametric system. The data flows are shown in green, red, and blue lines for different systems.

$$= -\log\left(\sum_{k=1}^{K} w_k \mathcal{N}(\boldsymbol{e}|\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k^2)\right) \tag{7}$$

The network decodes the prosody embeddings and the phoneme embeddings into the duration prediction $\hat{\boldsymbol{d}}$ and mel-spectrograms $\hat{\boldsymbol{o}}$ that

$$\hat{\boldsymbol{d}} = \texttt{DurDecoder}(\boldsymbol{e}, \boldsymbol{h}) \tag{8}$$

$$\hat{\boldsymbol{o}} = \texttt{MelDecoder}(\boldsymbol{e}, \boldsymbol{h}, \boldsymbol{d}) \tag{9}$$

The model takes the reference embedding to reconstruct the acoustic features. In general, the loss of the global training procedure is

$$\mathcal{L} = \mathcal{L}_{\texttt{PP}} + \mathcal{L}_{\texttt{DUR}} + \mathcal{L}_{\texttt{MEL}} \tag{10}$$

where

$$\mathcal{L}_{\texttt{DUR}} = \|\boldsymbol{d}, \hat{\boldsymbol{d}}\|_2, \tag{11}$$

$$\mathcal{L}_{\texttt{MEL}} = \|\boldsymbol{o}, \hat{\boldsymbol{o}}\|_1 \tag{12}$$

are the mean square error and mean absolute error. The pitches and energies are also modeled alongside the duration modeling following the FastSpeech2. In the inference stage, the $\hat{\boldsymbol{e}}$ is sampled from the distribution in (3) to predict the duration $\hat{\boldsymbol{d}}$ in (8). The $\boldsymbol{d}$ in (9) is replaced by $\hat{\boldsymbol{d}}$ to calculate the $\hat{\boldsymbol{o}}$ as the model outputs.

III. Neural Fusion Architecture

A. Fusion System

Our neural fusion system is a text-to-speech system that mixes unit concatenation and parametric systems. The traditional hybrid system is usually considered as a unit concatenation system with parametric selection models in recent years [25], [29], [30]. To distinguish our approach from the hybrid system, we call the proposed architecture a "fusion system". Different from the hybrid system, the outputs of the fusion system consist of both unit segments from audios and inference segments from parametric model [22], [23]. Fig. 4 briefly shows the difference of 3 types of text-to-speech systems. Given the input sequence and the speaker embedding, the parametric model predicts a sequence of outputs. The parametric TTS system directly decodes the output sequence into spectrograms. The hybrid system takes the output sequence or output distributions to select the units from the

unit library with other guidance. The units are then decoded or directly concatenated into audio outputs with a signal processing filter. The fusion system selects the units with the lowest cost from the library but keeps the rest parametric outputs. Then the mixed sequence is decoded into the spectrograms.

B. Unit Selection in Prosody Embedding

Though the multi-speaker TTS model is able to be adapted to the cloned speaker in the MDN distributions, the speaker identity of the synthetic speech sounds more like a general speaker with the targeting style. The rich prosody is difficult to be captured in the speaker embedding. As described in the previous section, we use neural fusion architecture, which incorporates the unit concatenation method in parametric text-to-speech architecture, to address this problem. The selection and fusion procedures work on the phoneme-level prosody embeddings. With the selected prosody embeddings from the natural speech, the generated audios contain clips from the natural speech, which sound more similar to the cloned speaker. It should be noted that the audio data for voice cloning is often limited. It is difficult to build a pure hybrid unit selection system based on the small amount of training data. This section describes the fusion method to take advantage of end-to-end text-to-speech and unit selection systems jointly.

Based on the framework in Section II, we build a phoneme-level unit embedding library from the audio data of the cloning speaker s using the prosody extractor in Fig. 2. The unit embedding library $U(s)$ is defined as the real embedding set of speaker s

$$U(s) = \{U_x(s)\}, \quad x \in P, \tag{13}$$

where P is the whole legal phoneme set and $U_x(s)$ is the unit embedding library of phoneme x for speaker s. For each audio with pre-aligned duration, the natural phoneme segment corresponding to the phoneme label l_j is encoded in to $\boldsymbol{e}_j$ by the prosody extractor. The $U_x(s)$ corresponding to the phoneme x and speaker s is defined as

$$U_x(s) = \{\boldsymbol{e}_j, \text{ if } l_j = x\} \text{for } \boldsymbol{l} = (l_1, l_2, .., l_N) \in \boldsymbol{L}(s), \tag{14}$$

where $\boldsymbol{L}(s)$ is all the sentence group of the cloning data from speaker s. The unit embedding library $U(s)$ is constructed to make the fusion sequence in the system. We define 3 types of prosody embeddings: the real embeddings $\boldsymbol{e}_j^r$ from the audio of the current training utterance, the sampled embeddings $\boldsymbol{e}_j^s$ from the distribution in (3), and the most probable unit embeddings $\boldsymbol{e}_j^u$ from the unit embedding library. The details of the 3 types of embeddings are defined as follows:

$$\boldsymbol{e}_j^r = \texttt{ProsodyExtractor}(\boldsymbol{o}_j, \boldsymbol{d}), \tag{15}$$

$$\boldsymbol{e}_j^s = \text{Sample} \sim p(\boldsymbol{e}|\boldsymbol{h}_j, s), \tag{16}$$

$$\boldsymbol{e}_j^u = \text{argmax}_{\boldsymbol{e}}\, p(\boldsymbol{e}|\boldsymbol{h}_j, s) \quad \text{for } \boldsymbol{e} \in U_x(s) \text{ if } \boldsymbol{e} \neq \boldsymbol{e}_j^r \text{ and } l_j = x. \tag{17}$$

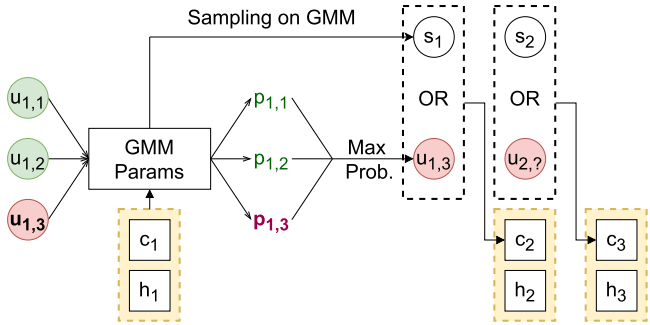


Fig. 5. The method of selecting the unit from the GMM distribution. The u indicates the unit embeddings, while s indicates the sampled embeddings.

Then the $\hat{\boldsymbol{e}} = \hat{e}_1, .., \hat{e}_N$ in the inference stage is obtained from the equation that

$$\hat{\boldsymbol{e}}_j = \begin{cases} \boldsymbol{e}_j^u & \text{if } p(\boldsymbol{e}_j^u|\boldsymbol{h}_j, s) \geq \hat{p} \\ \boldsymbol{e}_j^s & \text{if } p(\boldsymbol{e}_j^u|\boldsymbol{h}_j, s) < \hat{p} \end{cases} \tag{18}$$

where $\hat{p}$ is a lower bound of the probability.[1]

C. Concatenation of Unit in the Decoder

The concatenation gap between the neighboring units is a common problem in a pure unit selection system. In this fusion method, the unit embeddings and sampled embeddings are mixed in a sequence. Therefore, there are 3 types of possible concatenation gaps:

- model-unit gap: the gap between the sampled embeddings from the model and the unit embedding,
- unit-model gap: the gap between the unit embedding and the sampled embeddings from the model,
- unit-unit gap: the gap between the neighbor unit embeddings.

To address the concatenation problem, we proposed two strategies to reduce the gap between neighbor embeddings:

1) Auto-Regressive Embedding Modeling: We first introduce the auto-regressive embedding modeling. The prosody embedding distribution is conditioned on the previous embedding outputs. In the auto-regressive modeling, the (3) is changed into

$$p(\boldsymbol{e}_j|\boldsymbol{h}_j, s) = \sum_{k=1}^{K} w_k \mathcal{N}(\boldsymbol{e}_j|\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k^2, \boldsymbol{e}_{<j}, s). \tag{19}$$

Fig. 5 shows the procedure of auto-regressive embedding inference. The speaker-dependent GMM distribution $p(\boldsymbol{e}_j|\boldsymbol{h}_j, s)$ is first predicted with the text embeddings $\boldsymbol{h}_j$ and the previous GRU cell $\boldsymbol{c}_j$. Normally, the phoneme-level prosody embeddings are sampled from the GMM distribution as $\boldsymbol{e}_j^s$ for parametric inference. In our fusion method, we calculate the probability of the candidate unit embeddings $\boldsymbol{e}_{j,m}^u$ under the GMM distribution for the m-th candidate unit in $U_{l_j}(s)$. The $\boldsymbol{e}_{j,m}^u$ with the highest probability is chosen as the $\boldsymbol{e}_j^u$. In Fig. 5, we assume $\boldsymbol{e}_{1,3}^u$ is the unit embedding with highest probability for the phoneme with index "1". The final output $\hat{\boldsymbol{e}}_1$ is picked from $\boldsymbol{e}_1^s$ and

[1] $\hat{p}$ is an empirical value to adjust the unit/sampling rate. Normally the vowels are provided with a smaller value than the consonants.

Algorithm 1: Refinement Procedure.

Input:

phoneme input sequences: $\boldsymbol{l} = l_1, .., l_N$

prosody embeddings: $\boldsymbol{e} = \boldsymbol{e}_1, .., \boldsymbol{e}_N$

mel-spectrograms: $\boldsymbol{o} = o_1, .., o_T$

Output:

$\boldsymbol{h} = \text{PhonemeEncoder}(\boldsymbol{l})$

$\boldsymbol{e}_0 = 0$

for $j = 1$ to N **do**

  $\boldsymbol{w}, \boldsymbol{\mu}, \boldsymbol{\sigma} = \text{ProsodyPredictor}(\boldsymbol{h}, \boldsymbol{e}_{j-1}, s)$

  $k = \text{argmax}_i(w_i)$

  Compute $\boldsymbol{e}^r, \boldsymbol{e}^s, \boldsymbol{e}^u$ using (15)–(17)

  $\boldsymbol{e}_i^{\text{trn}}, t_i = \text{Random}((\boldsymbol{e}^s, 0), (\boldsymbol{e}^{\text{simu}}, 1))$

end for

prosody embeddings: $\boldsymbol{e}^{\text{trn}} = \boldsymbol{e}_1^{\text{trn}}, .., \boldsymbol{e}_N^{\text{trn}}$

type indicators: $\boldsymbol{t} = t_1, .., t_N$

$\hat{\boldsymbol{o}}, \hat{\boldsymbol{d}} = \text{Decoders}(\boldsymbol{e}^{\text{trn}}, \boldsymbol{t}, \boldsymbol{h})$

Back-propagate $\|\boldsymbol{o} - \hat{\boldsymbol{o}}\|_2$, $\|\boldsymbol{d} - \hat{\boldsymbol{d}}\|_1$

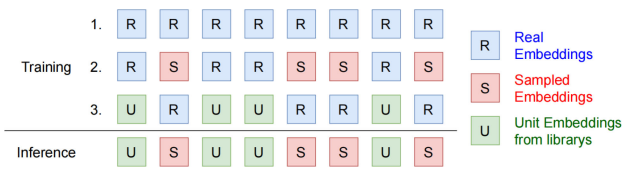


Fig. 6. The prosody embedding sequences in the decoder refinement procedure and the inference procedure.

$\boldsymbol{e}_{1,3}^u$. Then $\hat{\boldsymbol{e}}_1$ is fed into the network to predict the GMM distribution of the next phoneme auto-regressively. In practice, all the embeddings can be selected from the unit library, which will result in poor performance with limited data in the voice cloning task. Therefore, it is required to set a threshold on which unit is allowed to be chosen.

This procedure helps the model to make a better prediction if the previous embedding is from the unit library instead of sampling from the distribution. Therefore, the "unit-model gap" is reduced, and the currently selected units also guide the following sequence inference. However, the "unit-unit gap" and "model-unit gap" are not handled.

2) Decoder Refinement: We introduce the decoder refinement in this framework based on the "unit-unit gap" and "model-unit" gap. It should be noted that the decoder refinement is applied to general model training without the cloning speakers. Some parameters and indicators are added to the decoder during refinement training. The added parts are also used in the inference stage. It should be noted that the decoder parameters will not change for different speakers.

We recall the $\boldsymbol{e}_j^r, \boldsymbol{e}_j^s, \boldsymbol{e}_j^u$ in (15)–(17). Fig. 6 gives an example of the differences among the different types of embeddings sequences. It is required that the fusion method takes the mixed embedding sequence with both unit or sampled embeddings in the inference stage. Therefore, we managed to simulate the inference scenario in the refinement stage. To simulate that, we present 3 types of training strategies. Each row in Fig. 6 represents a type of embedding sequence in the training procedure.

Algorithm 2: Inference Procedure.

Input:

phoneme input sequences: $\boldsymbol{l} = l_1, .., l_N$

Speaker vector: s

Output:

$\boldsymbol{h} = \text{PhonemeEncoder}(\boldsymbol{l})$

$\boldsymbol{e}_0 = 0$

for $j = 1$ to N **do**

  $\boldsymbol{w}, \boldsymbol{\mu}, \boldsymbol{\sigma} = \text{ProsodyPredictor}(\boldsymbol{h}, \boldsymbol{e}_{j-1}, s)$

  $k = \text{argmax}_i(w_i)$

  $\boldsymbol{e}_j^u = \text{argmax}_{\boldsymbol{e}} \mathcal{N}(\boldsymbol{e}|\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k^2)$ for $\boldsymbol{e}$ in $U_{l_j}(s)$

  if $\mathcal{N}(\boldsymbol{e}_j^u|\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k^2) > \hat{p}$ **then**

    $\boldsymbol{e}_j = \boldsymbol{e}_j^u$

    $t_j = 1$ (as `"Unit"`)

  else

    $\boldsymbol{e}_j = \text{Sample} \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k)$

    $t_j = 0$ (as `"Sampled"`)

  end if

end for

prosody embeddings: $\boldsymbol{e} = \boldsymbol{e}_1, .., \boldsymbol{e}_N$

type indicators: $\boldsymbol{t} = t_1, .., t_N$

$\hat{\boldsymbol{o}}, \hat{\boldsymbol{d}} = \text{Decoders}(\boldsymbol{e}, \boldsymbol{t}, \boldsymbol{h})$

The first row represents the GMM-based FastSpeech2 training that all the embeddings are the real embeddings extracted from the audios. In this scenario, $\boldsymbol{e}_j^s$ and $\boldsymbol{e}_j^u$ are both assumed to be $\boldsymbol{e}_j^r$ in the training stage. The second row represents that some real embeddings are replaced by the sampled embeddings from the probability distribution. In this scenario, $\boldsymbol{e}_j^u$ is assumpted to be $\boldsymbol{e}_j^r$. The third row represents that some real embeddings are replaced by unit embeddings (embeddings from other segments). In this scenario, $\boldsymbol{e}_j^s$ is assumpted to be $\boldsymbol{e}_j^r$.

The 3 types of embeddings are introduced in the training procedure of the decoder refinement. Therefore, the model is optimized with embedding sequences similar to the inference stage such that unit embeddings and sampled embeddings are mixed together. Eq. (17) is designed for the purpose that the unit embeddings in the training procedure should not be too far away from the real embeddings. Therefore, we select the closest unit embedding from the unit library, which has the highest probability under the distribution but is not exactly the real embeddings of the current phoneme. During training, the prosody embedding $\boldsymbol{e}$ is mixed with randomly selected embedding types in a fixed sampling rate.

$$\boldsymbol{e}_j^{\text{trn}} = \text{Random}(\boldsymbol{e}_j^r, \boldsymbol{e}_j^{\text{simu}}) \tag{20}$$

where $\boldsymbol{e}_j^{\text{simu}}$ is $\boldsymbol{e}_j^s$ or $\boldsymbol{e}_j^u$ for different training strategies.

Then the decoder is optimized to minimize the loss that:

$$\mathcal{L}_{\text{DUR}} = \|\boldsymbol{d},\ \texttt{DurDecode}(\boldsymbol{e}^{\text{trn}}, \boldsymbol{h})\|_2, \tag{21}$$

$$\mathcal{L}_{\text{MEL}} = \|\boldsymbol{o},\ \texttt{MelDecode}(\boldsymbol{e}^{\text{trn}}, \boldsymbol{h}, \boldsymbol{d})\|_1. \tag{22}$$

We also introduce the type indicators $\boldsymbol{t}_j$ to indicate the type of $\boldsymbol{e}_j$ for the current phoneme l_j. The type indicators are global values shared by all phonemes. Each phoneme can be set as any type

of indicator in the training stage, depending on the provided data. $\boldsymbol{t}_j$ is fed into the network alongside $\boldsymbol{e}_j$. The details of the refinement procedure and inference procedure are shown in Algorithm 1 and Algorithm 2.

The GMM is required to better model the variances of different speakers or speaking styles. However, it is not necessary to sample from all the Gaussian components for the voice cloning task since the cloning audios are limited. Therefore, we make an approximation that the single Gaussian model is treated as the embedding distribution in the inference stage to reduce model variance. Therefore, (19) is approximated as

$$p(\boldsymbol{e}_j|\boldsymbol{h}_j, s) = \mathcal{N}(\boldsymbol{e}_j|\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k^2, \boldsymbol{e}_{<j}, s) \tag{23}$$

where w_k, the weight of the k-th Gaussian component, is the maximum weight of the GMM model. This approximation only makes influences in the refinement procedure and inference procedure.

IV. Experiment and Result

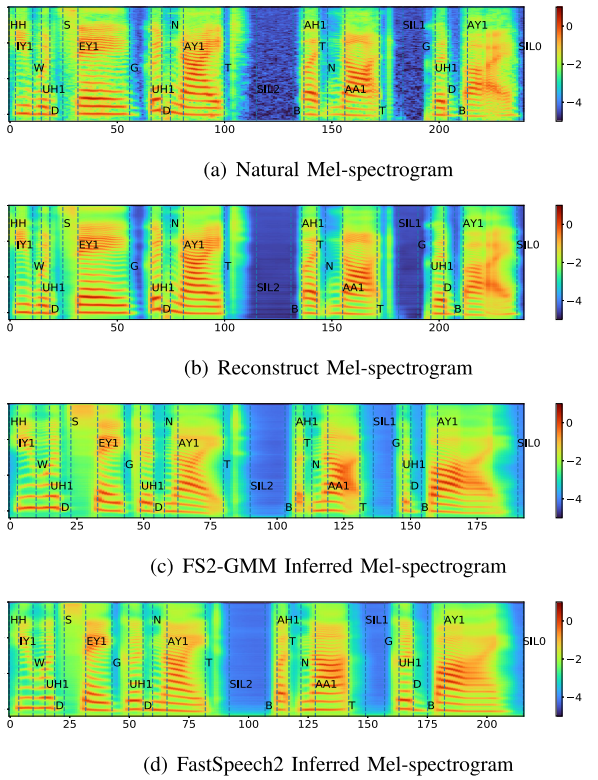
A. Experiment Setup

We conduct the experiments on the LibriTTS corpus [31]. The models are first trained using "train-clean-360" and "train-clean-100" data. A standard voice cloning method is to update a speaker embedding for a text-to-speech system. Therefore, We select two state-of-the-arts text-to-speech model with a trainable speaker embedding for evaluation, the FastSpeech2 (FS2) [28] and the GMM based phoneme-level conditioned FastSpeech2 (FS2-GMM) [20]. The speaker embeddings are back-propagated in the adaptation procedure for the cloning speakers. The "FS2-GMM" model is trained using the auto-regressive prosody modeling. The decoder is refined on the parameters of the "FS2-GMM" model using training data, which does not draw any information from the test speakers. During model training, the distribution of phoneme-level prosody embeddings of various speakers is modeled by a GMM with 20 Gaussian components. In the inference stage, we directly use the Gaussian component with the largest weight as the embedding distribution. The 320-dim mel-spectrograms are selected as the acoustic features, with pitches, duration, and energies as the prosody feature, following our previous setup in [20]. The mel-spectrograms are extracted with 50 ms window, 12.5 ms frameshift, 1024 FFT points, and 320 mel-bins using PyWorld [32] and librosa [33]. The phoneme duration is pre-generated from Kaldi [34] before the model training. The model first predicts the prosody components, including duration and full-time length pitches and energies. Then the decoder generates the frame-level acoustic features. The neural vocoder is trained on the clean audios of a female speaker in LJSpeech [35] corpus and the training audios in LibriTTS without a speaker specifier. We picked 4 female and 2 male speakers in the test set for objective and subjective measures. 100 sentences are sampled from each test speaker for voice cloning, and another 30 sentences are sampled for evaluation.[2]

[2]The audio samples and preliminary tests are presented in the webpage: https://bobmilk.github.io/neural-fusion.

TABLE I
Information of the Test Speakers

Gender	Female				Male	
SpeakerID	4970	5142	3570	1995	7127	2300
Audio length(sec)	592.4	410.1	989.4	398.2	628.3	709.1



(a) Natural Mel-spectrogram

(b) Reconstruct Mel-spectrogram

(c) FS2-GMM Inferred Mel-spectrogram

(d) FastSpeech2 Inferred Mel-spectrogram

Fig. 7. An example of the mel-spectrograms using different generating methods. The phoneme boundaries are presented as dashed lines with the phoneme labels. Text: "He would say good night, but not, goodbye.".

The information on the cloning data of the test speakers is listed in Table I.

B. Unit Embedding Reconstruction

Unlike the conventional unit selection system, the candidate audio units are the phoneme unit embeddings in this architecture. Therefore, we first examine that with the real embeddings extracted from the audio, the reconstructed mel-spectrograms are close enough to the natural mel-spectrograms.

The evaluation is only adopted on the test speaker, which is not seen in the multi-speaker model training. Therefore, we ensure that the reconstruction quality is from the model ability rather than the possibly overfitting problem. Fig. 7 gives an example of the natural and reconstructed mel-spectrograms. Fig. 7(a) represents the mel-spectrograms extracted from the natural speech. Fig. 7(b) represents the decoded mel-spectrograms with the real prosody embeddings. It is clear that the reconstructed mel-spectrograms largely follow that of the natural mel-spectrograms, though it is slightly smoother than the natural mel-spectrograms. Fig. 7(c) and (d) represents the mel-spectrograms directly inferred from the model parameters.

It is easily observed that the difference between natural mel-spectrograms and reconstructed mel-spectrograms is much more minor than inferred mel-spectrograms. It indicates that the phoneme-level prosody embeddings have the ability to reconstruct the mel-spectrograms. Therefore, it is possible to make the fusion on the prosody embedding level.

C. Embedding Selection and Fusion

This section presents the performance of directly decoding from the mixed embeddings without refinement. Operatically, the type indicators $\boldsymbol{t}$ are set as "natural" labels under the assumption that all the phoneme-level prosody embeddings are the natural embedding. The GMM models the prosody embeddings in the multi-speaker model to handle various speaking prosodies. The prosody distribution is the key component of the selection. However, the cloning data is too small to cover all the speaking prosodies. We examined the selection performance using the entire GMM or the Gaussian components with the top weights. It is observed in the preliminary test that some inappropriate unit embeddings might be selected with a high probability in GMM. Therefore, the Gaussian component with the highest weight is directly approximated as the prosody embedding distribution to stabilize the performance.

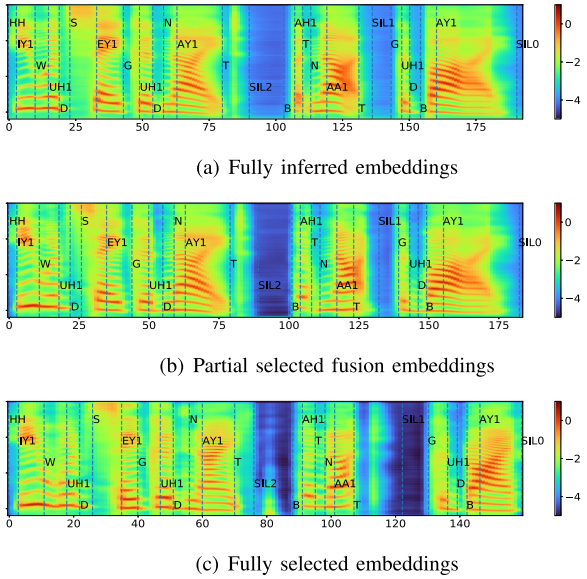
1) Selection Strategy: We then examine the performance of the selection and fusion without the decoder refinement. Empirically, some generated audios already achieve both naturalness and similarity improvements using direct selection and fusion strategy. However, the selection and fusion are not always stable since the candidate units are limited. Sometimes there are concatenation problems, or the quality of the unit is not good enough. Therefore, we select the units using the hierarchical selection to take advantage of the text.

- If the triphone is matched in the unit embedding library, we select the triphone with the highest probability using lower bound p_{tri}.
- If the previous biphone is matched in the unit embedding library, we select the previous biphone using lower bound p_{bi}

The lower bounds follows the rule $p_{\mathrm{tri}} < p_{\mathrm{bi}}$. We attempted to include the monophone in the hierarchical selection, but the boundary effect was too strong. Therefore, the units only matched with monophone are not considered as the unit embedding candidates.

2) Consonants and Vowels: Since units are segmented by the ASR alignments, there are boundary errors in the segmentation. The errors are usually 1 or 2 frames if they occur. The vowel commonly has a longer duration than the consonant. The boundary errors have more influence on the consonants than on the vowels if the segmentation is not sufficiently accurate. Therefore, we separately treat the vowels and consonants that the probability threshold to select the consonants is much higher than the vowels.

Fig. 8 shows an example of the generated mel-spectrogram using the same input phoneme sequence. In Fig. 8(b), about 40% of phoneme embeddings are replaced by the unit embeddings. It is clear that Fig. 8(b) is much more similar to Fig. 8(a) than Fig. 8(c). It shows that the parametric model still guides the synthetic speech if only a part of the embeddings is selected from the unit library. The parametric guidance vanishes if the proportion of unit embeddings is too high.



(a) Fully inferred embeddings

(b) Partial selected fusion embeddings

(c) Fully selected embeddings

Fig. 8. An example of the mel-spectrograms using different selection methods. The phoneme boundaries are presented as dashed lines with the phoneme labels. Text: "He would say good night, but not, goodbye.".

D. Decoder Refinement

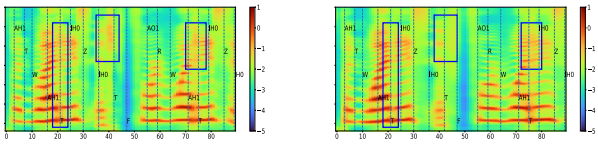
We then examine the performance of the decoder refinement using the same phoneme-level prosody embeddings sequence. In this case, we use the pre-generated mixed prosody embedding sequence with both sampled embeddings and unit embeddings. The mel-spectrogram difference is evaluated for the test speakers on different refinement systems defined as follows:

- FS2-GMM: The base model that all prosody embeddings are real embeddings in the training procedure.
- U/S refine: The model that unit embeddings and sampled embeddings are directly mixed as a sequence to refine the decoder.
- R/S refine: The model that real embeddings are treated as unit embeddings in the "U/S refine" system.
- R/U refine: The model that real embeddings are treated as sampled embeddings in the "U/S refine" system.

We propose a pseudo scenario in which the embedding sequences are provided with mixed real and unit embeddings to evaluate the refinement algorithm objectively. Therefore, the refinement performance can be evaluated by the mel-spectrogram difference (MSD) using the L1 distance between the natural mel-spectrograms and the generated mel-spectrograms. We randomly select 40% phonemes from the test utterances, and the prosody embeddings of these phonemes are replaced by the unit embeddings from other cloning utterances. The result is shown in Table II that the MSD always drops for different speakers after the R/U refinement. Fig. 9 also provides an example of the mel-spectrograms from the refinement. It is highlighted in the

TABLE II
MEL-SPECTROGRAM DIFFERENCE (MSD) OF DIFFERENT REFINEMENT METHODS FOR TEST SPEAKERS

SpeakerID	FS2-GMM	R/S refine	R/U refine	U/S refine
4970	0.5720	0.5441	**0.5188**	0.6062
5142	0.5410	0.5353	**0.5252**	0.6211
3570	0.5392	0.5105	**0.4890**	0.5539
1995	0.4957	0.4879	**0.4763**	0.5514
7127	0.4959	0.4839	**0.4706**	0.5315
2300	0.5212	0.5082	**0.4955**	0.5640



(a) Fusion without refinement. (b) Fusion with refinement.

Fig. 9. An example of mel-spectrograms segment from fusion models using the same embedding sequence. Text: "Yes, but what is it for? what is it all about?".

TABLE III
PREFERENCE TESTS ON SIMILARITY

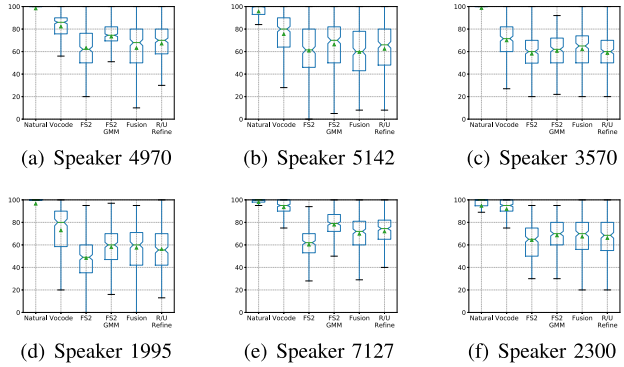
SpeakerID	FS2-GMM	R/U refine	undecided	p-value(T Test)
4970	27.17%	**48.17%**	24.67%	1.871×10^{-9}
5142	20.67%	**39.33%**	40.00%	2.178×10^{-9}
3570	**38.52%**	36.48%	25.00%	0.585
1995	26.00%	**46.17%**	27.83%	3.814×10^{-9}
7127	31.17%	**43.17%**	25.67%	6.223×10^{-4}
2300	30.19%	**47.59%**	22.22%	3.739×10^{-6}

squares that the boundaries are blurry or mismatched in Fig. 9(a), while the boundaries are clear in Fig. 9(b). It should also be noted that all the test speakers are unseen in the decoder refinement procedure.

E. Subjective Measure

We conduct the preference test to evaluate the performance improvement on the similarity and the MUSHRA test to evaluate the global speech quality using webMUSHRA [36]. The listeners are non-native English speakers. For each test group, the listeners were presented with 30 utterances in the evaluation set of the test speaker. Every speech was presented to at least 9 different listeners. The audios with the same text are presented in random orders. Especially in the preference test, each pair of generated speech are presented twice in different orders. In the preference test, the listeners were provided with the reference audios and the synthetic audios of the two different systems ("FS2-GMM" and "R/U refine"). It is examined in the preliminary tests that the speech naturalness seriously dropped by the concatenation problem in bad cases without refinement. Therefore it is less meaningful to measure the similarity of the fusion system without refinement. Since the vocoder is trained on LJSpeech, we provide the vocoded audios as the reference audios instead of the natural audios to avoid vocoding mismatch. The listeners were requested to label the speaker's similarity to the reference audios.[3] Table III shows the results that the "R/U Refine" system achieves better similarity than the "FS2-GMM" system for 4 speakers. At the same time, there is no significant difference for the result of speaker 3570, and the difference is minor for speaker 7127. It is reasonable that the baseline system has already achieved good similarity for some speakers.

[3]The listeners were required to ignore the general prosodies (rising/falling tones) since there is randomness for the general prosodies.



(a) Speaker 4970 (b) Speaker 5142 (c) Speaker 3570

(d) Speaker 1995 (e) Speaker 7127 (f) Speaker 2300

Fig. 10. The MUSHRA tests on naturalness. The triangles indicates the mean score. The green lines indicate the median score.

Fig. 10 shows the MUSHRA tests on the naturalness to examine the performance of the refinement network. It is observed in Fig. 10 that the "FS2-GMM" system achieves the best naturalness except for Fig. 10(c). The "Fusion" system achieves worse naturalness compared to "R/U Refine" in Fig. 10(a), (b), and (e), especially for the worst cases that receive extremely low scores. The performances are similar in Fig. 10(d) and (f) for "Fusion" and "R/U Refine" systems. Therefore, the refinement procedure achieves similar or better performance except for Speaker 3570. It indicates that the fusion system suffers the concatenation problems, but the refinement procedure has the ability to reduce the problem. Interestingly, in Fig. 10(c), the naturalness of the "Fusion" system archives the best performance but becomes worse after refinement. It is for the reason that the concatenation problem is not serious for speaker 3570. Therefore, directly decoding from the unit embeddings without refinement also contributes to speech naturalness, but the model refinement drops the speech naturalness obtained from unit embeddings. Also, both "Fusion" and "R/U refine" systems, taking advantage of the base model "FS2-GMM," achieve significantly higher scores than the "FS2" baseline.

V. CONCLUSION

This paper proposes a neural fusion architecture for small data voice cloning tasks. The architecture is a parametric text-to-speech model incorporating a unit concatenation method. The architecture is designed on the phoneme-level prosody conditioned FastSpeech2. The phoneme-level unit embeddings are extracted from the original speech using a prosody extractor. The prosody embedding sequence consists of both inferred embeddings and unit embeddings in the inference stage. The refined decoder makes fusion on the mixed embedding sequence to produce synthetic speech. The result shows that the incorporation of unit embeddings significantly improves speaker similarity. The refinement procedure has the ability to mitigate the concatenation problems.

REFERENCES

[1] Q. Xie *et al.*, "The multi-speaker multi-style voice cloning challenge 2021," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2021, pp. 8613–8617.

[2] A. Łaćucki, "Fastpitch: Parallel text-to-speech with pitch prediction," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2021, pp. 6588–6592.

[3] T. Raitio, R. Rasipuram, and D. Castellani, "Controllable neural text-to-speech synthesis using intuitive prosodic features," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 4432–4436.

[4] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Proc. Int. Conf. Learn. Representations*, 2014.

[5] K. Akuzawa, Y. Iwasawa, and Y. Matsuo, "Expressive speech synthesis via modeling expressions with variational autoencoder," in *Proc. Interspeech*, 2018, pp. 3067–3071.

[6] Y. Wang *et al.*, "Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5180–5189.

[7] M. Tamura, T. Masuko, K. Tokuda, and T. Kobayashi, "Speaker adaptation for HMM-based speech synthesis system using MLLR," in *Proc. 3rd ESCA/COCOSDA Workshop Speech Synth.*, 1998, pp. 273–276.

[8] J. Yamagishi and T. Kobayashi, "Average-voice-based speech synthesis using hsmm-based speaker adaptation and adaptive training," *IEICE Trans. Inf. Syst.*, vol. 90, no. 2, pp. 533–543, 2007.

[9] Z. Wu, P. Swietojanski, C. Veaux, S. Renals, and S. King, "A study of speaker adaptation for DNN-based speech synthesis," in *Proc. 16th Annu. Conf. Int. Speech Commun. Assoc.*, 2015, pp. 879–883.

[10] R. Doddipatla, N. Braunschweiler, and R. Maia, "Speaker adaptation in DNN-based speech synthesis using d-vectors," in *Proc. 21st Annu. Conf. Int. Speech Commun. Assoc.*, 2017, pp. 3404–3408.

[11] T. Wang, J. Tao, R. Fu, J. Yi, Z. Wen, and R. Zhong, "Spoken content and voice factorization for few-shot speaker adaptation," in *Proc. INTERSPEECH Conf.*, 2020, pp. 796–800.

[12] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6309–6318.

[13] A. Razavi, A. van den Oord, and O. Vinyals, "Generating diverse high-fidelity images with VQ-VAE-2," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 14 866–14 876.

[14] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.- Assist. Interv.*, 2015, pp. 234–241.

[15] M. Chen *et al.*, "AdaSpeech: Adaptive text to speech for custom voice," in *Proc. Int. Conf. Learn. Representations*, 2021.

[16] H.-T. Luong and J. Yamagishi, "NAUTILUS: A versatile voice cloning system," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 28, pp. 2967–2981, 2020.

[17] S. O. Arik, J. Chen, K. Peng, W. Ping, and Y. Zhou, "Neural voice cloning with a few samples," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 10 040–10 050.

[18] W.-N. Hsu *et al.*, "Disentangling correlated speaker and noise for speech synthesis via data augmentation and adversarial factorization," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2019, pp. 5901–5905.

[19] J. Cong, S. Yang, L. Xie, G. Yu, and G. Wan, "Data efficient voice cloning from noisy samples with domain adversarial training," in *Proc. Interspeech*, 2020, pp. 811–815.

[20] C. Du and K. Yu, "Rich prosody diversity modelling with phone-level mixture density network," in *Proc. Interspeech*, 2021, pp. 3136–3140.

[21] A. W. Black *et al.*, "CMU blizzard 2007: A hybrid acoustic unit selection system from statistically predicted parameters," in *Proc. Blizzard Challenge Workshop*, 2007, Paper 005.

[22] S. Tiomkin, D. Malah, S. Shechtman, and Z. Kons, "A hybrid text-to-speech system that combines concatenative and statistical synthesis units," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 5, pp. 1278–1288, Jul. 2011.

[23] M. P. Aylett and J. Yamagishi, "Combining statistical parameteric speech synthesis and unit-selection for automatic voice cloning," in *Proc. LangTech*, 2008.

[24] T. Merritt, R. A. Clark, Z. Wu, J. Yamagishi, and S. King, "Deep neural network-guided unit selection synthesis," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2016, pp. 5145–5149.

[25] T. Capes *et al.*, "Siri on-device deep learning-guided unit selection text-to-speech system," in *Proc. INTERSPEECH*, 2017, pp. 4011–4015.

[26] X. Zhou, Z.-H. Ling, and L.-R. Dai, "Extracting unit embeddings using sequence-to-sequence acoustic models for unit selection speech synthesis," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 7659–7663.

[27] X. Zhou, Z. Ling, and L.-R. Dai, "UnitNet: A sequence-to-sequence acoustic model for concatenative speech synthesis," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 2643–2655, 2021.

[28] Y. Ren *et al.*, "FastSpeech 2: Fast and high-quality end-to-end text to speech," in *Proc. Int. Conf. Learn. Representations*, 2021.

[29] F.-L. Xie, X.-H. Li, W.-C. Su, L. Lu, and F. K. Soong, "A new high quality trajectory tiling based hybrid TTS in real time," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2021, pp. 5704–5708.

[30] X. Yang, Z. Liu, Q. Sun, and H. Wang, "A novel unit selection and unit smoothing method for chinese concatenation speech," in *Proc. Int. Conf. Model. Anal. Simul. Technol. Appl.*, 2019, pp. 280–285.

[31] H. Zen, V. Dang, R. Clark, Y. Zhang, and Y. Wu, "LibriTTS: A. corpus derived from libriSpeech for text-to-speech," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2019.

[32] M. Morise, F. Yokomori, and K. Ozawa, "World: A vocoder-based high-quality speech synthesis system for real-time applications," *IEICE Trans. Inf. Syst.*, vol. 99, no. 7, pp. 1877–1884, 2016.

[33] B. McFee *et al.*, "librosa: Audio and music signal analysis in python," in *Proc. 14th Python Sci. Conf.*, 2015, pp. 18–25.

[34] D. Povey *et al.*, "The kaldi speech recognition toolkit," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding Conf. IEEE Signal Process. Soc.*, Dec. 2011.

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

[36] M. Schoeffler *et al.*, "webMUSHRA-A comprehensive framework for web-based listening tests," *J. Open Res. Softw.*, vol. 6, no. 1, p. 8, 2018.

Bo Chen (Student Member, IEEE) received the B.S. and M.Sc. degrees in computer science and technology in 2013 and 2016, respectively, from Shanghai Jiao Tong University, Shanghai, China, where he is currently working toward the Ph.D. degree with X-LANCE Laboratory, Department of Computer Science and Engineering. His research interests include voice cloning and voice conversion.

Chenpeng Du (Graduate Student Member, IEEE) received the B.S. degree in computer science and technology from the Harbin Institute of Technology, Harbin, China, in 2018. He is currently working toward the Ph.D. degree with X-LANCE Laboratory, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. His research interests include high-fidelity speech synthesis and speech recognition.

Kai Yu (Senior Member, IEEE) received the B.Eng. and M.Sc. degrees from Tsinghua University, Beijing, China, in 1999 and 2002, respectively, and the Ph.D. degree from Machine Intelligence Laboratory, Engineering Department, Cambridge University, Cambridge, U.K., in 2006. He is currently a Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. His main research interests include speech-based human machine interaction, including speech recognition, synthesis, language understanding and dialogue management. From 2017 to 2019, he was a member of the IEEE Speech and Language Processing Technical Committee.