Transfer Learning From Speech Synthesis to Voice Conversion With Non-Parallel Training Data

Mingyang Zhang, *Member, IEEE*, Yi Zhou, *Student Member, IEEE*, Li Zhao, and Haizhou Li, *Fellow, IEEE*

***Abstract*—We present a novel voice conversion (VC) framework by learning from a text-to-speech (TTS) synthesis system, that is called TTS-VC transfer learning or TTL-VC for short. We first develop a multi-speaker speech synthesis system with sequence-to-sequence encoder-decoder architecture, where the encoder extracts the linguistic representations of input text, while the decoder, conditioned on target speaker embedding, takes the context vectors and the attention recurrent network cell output to generate target acoustic features. We take advantage of the fact that TTS system maps input text to speaker independent context vectors, thus repurpose such a mapping to supervise the training of the latent representations of an encoder-decoder voice conversion system. In the voice conversion system, the encoder takes speech instead of text as the input, while the decoder is functionally similar to the TTS decoder. As we condition the decoder on a speaker embedding, the system can be trained on non-parallel data for any-to-any voice conversion. During voice conversion training, we present both text and speech to speech synthesis and voice conversion networks respectively. At run-time, the voice conversion network uses its own encoder-decoder architecture without the need of text input. Experiments show that the proposed TTL-VC system outperforms two competitive voice conversion baselines consistently, namely phonetic posteriorgram and AutoVC methods, in terms of speech quality, naturalness, and speaker similarity.**

***Index Terms*—Autoencoder, context vector, non-parallel, text - to-speech (TTS), transfer learning, voice conversion (VC).**

I. Introduction

VOICE Conversion (VC) takes speech of the source speaker as input and generates speech that sounds from a target speaker while maintaining the linguistic content. It is an enabling technology for many innovative applications, such as personalized or expressive speech synthesis [1], [2], speech enhancement [3], normalization of impaired speech [4], [5], speech-to-singing conversion [6], [7] and dubbing of movies and games. In general, voice conversion techniques are broadly grouped into parallel and non-parallel methods according to their use of training data [8]. Parallel voice conversion requires source-target pair of speech samples of the same linguistic content, while non-parallel voice conversion is trained on unpaired speech data.

Parallel voice conversion can be formulated as a regression problem where a mapping function is estimated between the source and target spectral features. Successful techniques include Gaussian mixture model (GMM) [9]–[11], frequency warping [12]–[14], and deep neural networks (DNN) [15]–[19]. Other techniques, such as sparse representation [20]–[22], and two-step adaptation [23]–[25], are studied to reduce the training data size. Non-parallel voice conversion techniques are certainly more attractive as parallel data is not easily available in practice. There has been prior studies on finding the optimal segments from unpaired source and target training data, such as frame mapping [26]–[28], clustering [29]–[31] and generative adversarial network methods [32]–[36].

Speech carries speaker-independent linguistic information as well as speaker characteristics and speaking style. If we can disentangle linguistic features from speaker representation, we may synthesize the linguistic content from one voice to another. Variational autoencoder (VAE) [37], [38] represents a technique in this direction. It is built of an encoder-decoder neural network, where the encoder learns a latent space to represent the speaker-independent linguistic information, while the decoder reconstructs the target speech features by conditioning on a target speaker representation. AutoVC is another successful attempt, that is trained with a carefully designed bottleneck layer forcing the encoder to disentangle the speaker-independent features only with the self-reconstruction loss [39], [40]. It outperforms the VAE-based techniques without the need of linguistic supervision. The success of the autoencoder framework is based on two assumptions: 1) the latent space only captures speaker-independent linguistic representation without a trace of the speaker information; 2) by conditioning on target speaker representation, a decoder is capable of generating the desired acoustic features for a target speaker. Unfortunately, the latent codes in autoencoder techniques are not trained to associate with linguistically motivated sound units. While autoencoder methods are effective in disentanglement, the lack of linguistic grounding could be a limitation for its performance.

Manuscript received September 25, 2020; revised January 1, 2021 and February 22, 2021; accepted March 7, 2021. Date of publication March 17, 2021; date of current version April 8, 2021. This work was supported by the National Research Foundation, Singapore under its AI Singapore Programme AISG Award AISG-GC-2019-002, and AISG Award AISG-100E-2018-006, and its National Robotics Programme under Grant 192 25 00054 and in part by RIE2020 Advanced Manufacturing, and Engineering Programmatic Grants A1687b0033, and A18A2b0046, and in part by the National Key Research, and Development Program of China 2018YFB1305203, 2020YFC2004003. The work of Yi Zhou was also supported by NUS research scholarship. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Heiga Zen. *(Corresponding author: Li Zhao.)*

Mingyang Zhang and Li Zhao are with the School of Information Science, and Engineering, Southeast University, Nanjing, Jiangsu, China (e-mail: zhangmy@seu.edu.cn; zhaoli@seu.edu.cn).

Yi Zhou is with the Department of Electrical, and Computer Engineering, National University of Singapore, Singapore (e-mail: yi.zhou@u.nus.edu).

Haizhou Li is with the Department of Electrical, and Computer Engineering, National University of Singapore, Singapore, and also with Machine Listening Lab, University of Bremen, Bremen 28359, Germany (e-mail: haizhou.li@nus.edu.sg).

Digital Object Identifier 10.1109/TASLP.2021.3066047

The recent advances in deep learning approaches open up new possibilities beyond the traditional parallel and non-parallel voice conversion paradigms. One is able to transfer knowledge from external resources, such as automatic speech recognition (ASR) and text-to-speech (TTS) synthesis, to voice conversion systems.

Phonetic PosterioGram (PPG) framework draws much attention in non-parallel voice conversion. PPG is an intermediate result from a speaker-independent ASR system, representing the posterior probability of phonetic classes of a speech frame [41]. The PPG-based voice conversion techniques [25], [42], [43] is an effective way to leverage ASR knowledge, that is learnt from a large speech corpus. Despite their success, PPG techniques still suffer from some inherent limitations. For example, the quality of PPG highly depends on the ASR system that is neither optimized for speech synthesis nor voice conversion.

Text-to-speech and voice conversion share a common goal to generate natural speech. However, TTS systems are usually trained with large speech corpora, while voice conversion systems face limited training data constraint. Therefore, transfer learning from TTS to voice conversion is naturally motivated in practice. Generally speaking, a neural TTS is trained to address two research problems, one is the ability to generate effective intermediate representations from the input text, another is the ability to align the attention to the intermediate representations to bridge between encoder and decoder. In short, the former is referred to as feature mapping, the latter is referred to as alignment. There is a general belief that voice conversion can benefit from TTS in one way or another. For feature mapping, voice conversion can learn from TTS to produce phonetically motivated intermediate representations, which are speaker-independent; For alignment, voice conversion can benefit from the learned TTS for natural speech rendering and text-to-speech alignment.

Park *et al.* [44] proposed a transcription-guided speech encoder as part of a sequence-to-sequence TTS model for any-to-many voice conversion, which requires both text and speech as input during run-time inference. On the other hand, Luong *et al.* [45] proposed to bootstrap a voice conversion system from a pre-trained speaker-adaptive TTS model, where both voice conversion and TTS share a common decoder. This method only handles feature mapping yet leaves the alignment task to an external system. Zhang *et al.* [46] proposed an architecture for joint training of voice conversion and text-to-speech. By taking text as an additional input, the voice conversion system improves voice quality during run-time inference. However, it relies on large parallel training data. Huang *et al.* [47] proposed a transformer architecture with TTS pre-training. The idea is to transfer knowledge from a pre-trained TTS model to a voice conversion model benefiting from large-scale, easily accessible TTS corpora. Though it attempts to handle alignment issues such as the conversion of articulation and prosody, this technique only works for parallel training data.

Building on the success of the prior studies on speech disentanglement and TTS-VC transfer learning, in this paper, we study a novel transfer learning technique from TTS to voice conversion. We adopt Tacotron-2 as the TTS framework [48]. In a two-step training strategy, we first train a standard multi-speaker Tactron-2 on a large database. We then transfer the TTS knowledge to an encoder-decoder architecture for voice conversion. We hypothesize that, 1) the context vector generated by the text encoder in a TTS system is speaker-independent representing the linguistic information; 2) the decoder in a TTS system, that constructs target acoustic features by conditioning on target speaker embedding, also works for voice conversion.

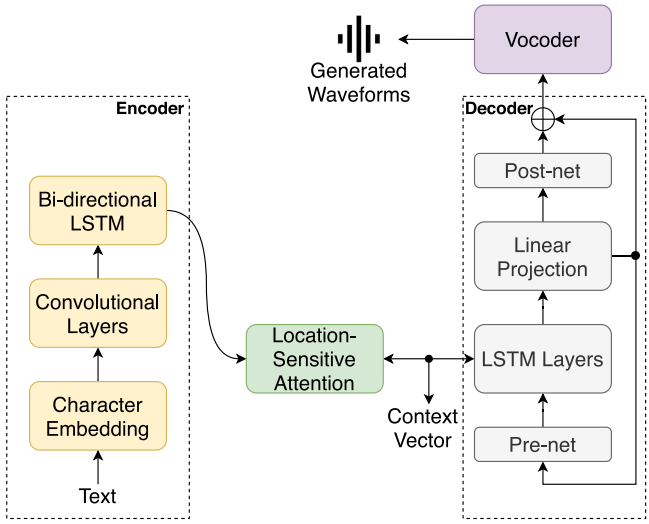

Fig. 1. Block diagram of the Tacotron-2 system. A text-speech alignment can be obtained by feedings input text to encoder, and target spectral frames to the decoder in a teacher-forcing mode. This process produces a sequence of speaker-independent context vectors, that are frame-aligned with target mel-spectrogram, to serve as the supervision target in TTS-VC transfer learning.

The main contributions of this paper include: 1) a novel transfer learning framework from TTS to voice conversion; 2) an integrated solution that benefits from transfer learning and speaker disentanglement; 3) an effective solution that deals with both feature mapping and alignment for voice conversion with non-parallel training data.

This paper is organized as follows. We introduce the related work that motivates our research in Section II. We formulate the proposed TTS-VC transfer learning framework in Section III. We present the experiments and discuss the results in Section IV. Section V concludes the study.

II. Related Work

A. Tacotron-2

Tacotron is an end-to-end text-to-speech (TTS) system [49] that has a sequence-to-sequence encoder-decoder architecture with attention mechanism [50]. Tacotron-2 is an updated version of Tacotron [48], as illustrated in Fig. 1. It includes an encoder that maps an input text sequence to a fixed-dimensional state vector, an attention-based decoder [51] that predicts a mel-spectrogram, and a neural vocoder [52] that reconstructs speech waveform from the predicted mel-spectrogram. For rapid turn-around, we use WaveRNN neural vocoder to generate speech waveform from mel-spectrogram in this paper [53].

The input to the encoder is a sequence of characters, which are first encoded into character embeddings. These embedded vectors are then processed by a stack of convolutional layers with batch normalization [54]. The outputs of convolutional layers

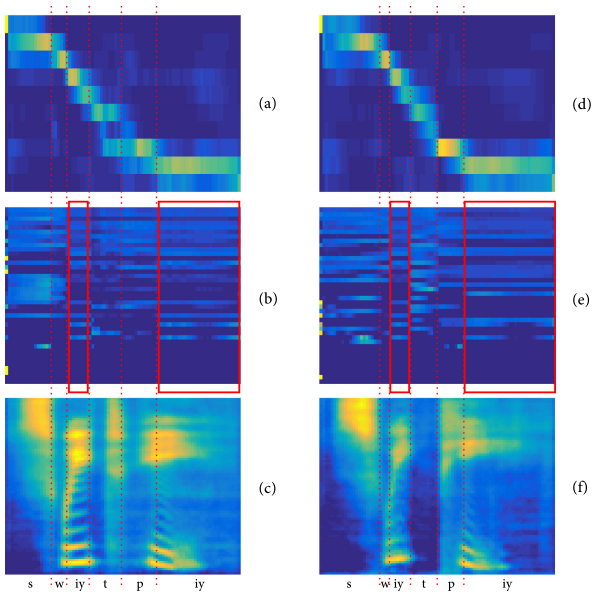

Fig. 2. Illustration of an utterance *'sweet pea'* in (a), (b), and (c) for Speaker 1, and in (d), (e), and (f) for Speaker 2. (a) temporal alignment weight between text and mel-spectrogram, (b) a stack of context vectors for the utterance (reduced to 40-dimension for ease of comparison), and (c) its mel-spectrogram with 80-dimensional spectral features. The horizontal axis all plots represents time, while vertical axis of (b), (c), (e) and (f) displays the dimension of vectors.

are taken by a bi-directional Long Short-Term Memory (LSTM) layer to produce text encoding.

The decoder is an auto-regressive recurrent neural network (RNN) that predicts a mel-spectrogram from the text encoding. A location-sensitive attention mechanism is used to summarize the text encoding into a sequence of fixed-length context vectors. At each prediction time step, the previous predicted spectral frame and the current context vector are both used as the input to the decoder to predict the target spectral frame. All spectral frames form an output mel-spectrogram.

B. Linguistic Representation

In Tacotron-2 training, the text encoder converts an input sentence to a stack of fixed-dimension text encodings. Then the location-sensitive attention in Fig. 1 learns to align the text encodings with a sequence of target spectral frames and generates the context vectors as a result. The context vectors are obtained by multiplying the alignment weights with text encodings. Fig. 2(a) and (d) show an example of the alignment between input text characters and mel-spectrogram learned by the attention mechanism.

During training, the text-speech alignment can be obtained from the trained model in the teacher-forcing mode. In practice, we present the input text and the ground truth target mel-spectrogram to the encoder and decoder, respectively. As the decoder generates output spectral features in an autoregressive manner, we use the ground truth target mel-spectrogram in place of the predicted frames from previous time steps to guide the feature generation. Fig. 2(b) and (e) illustrate a stack of generated context vectors, and Fig. 2(c) and (f) show their corresponding 80-dimensional mel-spectrogram. We note that the length of the context vector sequence is the same as that of the corresponding mel-spectrogram.

As text encodings are derived to represent input text, that seeks to represent linguistic information. Context vectors are weighted sums over encoder time steps of the text encodings, that could be influenced by the mel-spectrogram, or acoustic features, via the decoder. However, such influence from acoustic features is minimum. It is generally considered [44] that context vectors represent speaker-independent linguistic features. This is especially true when we train a multi-speaker TTS using an encoder-decoder architecture, where the decoder is conditioned on speaker embedding [55], [56]. In this case, the encoder-decoder architecture seeks to disentangle speaker from linguistic content.

It is important that the context vectors are associated with input characters and optimized for speech synthesis task during the decoder training. In other words, they are linguistically motivated and serve as an ideal candidate of intermediate representation for encoder-decoder style voice conversion. Park *et al.* [44] make use of context vectors from a TTS system as the linguistic features, that improve non-parallel voice conversion. In Fig. 2, we use an utterance to compare the plot of a context vector sequence with its corresponding spectrogram for two speakers. We have three observations: a) the patterns of context vectors within the same phoneme (e.g. 's', 'w', and 'iy') are rather stationary; b) the patterns for the phonemes are discrete and distinct from one another; c) the context vectors for the same phonemes are very close within the same speaker (see 'iy' in the red boxes in Fig. 2(b) or Fig. 2(e)) and across speakers (see 'iy' in the red boxes between Fig. 2(b) and Fig. 2(e)). All the observations point to the fact that the context vectors are the unique identifier of the phonemes, that are linguistically motivated. The context vectors don't reflect the spectral details of the actual acoustic rendering of the speech by individual speakers as illustrated in Fig. 2(c) and (f). Our analysis corroborates the observations in [44].

In voice conversion, there have been previous studies on the use of PPG from automatic speech recognition (ASR) for linguistic representation of speech content [25], [42], [43]. Context vector is a linguistic representation similar to PPG, but derived from text-to-speech synthesis (TTS) instead of ASR.

C. Leveraging Knowledge From Speech Synthesis

Traditionally, voice conversion operates at the signal level, while speech synthesis involves phonetic representation. Studies show that the use of linguistically informed features improves voice conversion. There have been studies to couple voice conversion with TTS training, that seeks to improve the training and run-time inference of voice conversion by adhering to linguistic content. However, such techniques usually require a large training corpus.

Zhang *et al.* [57] proposed to improve the sequence-to-sequence model [58] by using text supervision during training. A multi-task learning structure is designed which adds auxiliary classifiers to the middle layers of the sequence-to-sequence model to predict linguistic labels as a secondary task. The linguistic labels can be obtained with external alignment tools.

With the linguistic label objective, the encoder and decoder are expected to generate meaningful intermediate representations which are linguistically informed. The text transcripts are only required during training. Zhang *et al.* [46], and Luong *et al.* [45] proposed joint training of TTS and VC by sharing a common decoder. Park *et al.* [44] proposed to use the context vectors in Tacotron system as speaker-independent linguistic representation to guide the voice conversion.

Transfer learning is a technique to utilize knowledge from previously learned tasks and apply them to newer, related ones. A typical transfer learning involves pre-training of a base model, reusing the pre-trained model as the starting point for a model on the second task of interest, and refining the second model on input-output pair data for the second task. In a TTS-VC transfer learning scenario, usually we have significantly more data for TTS, and we would like to generalize the learned knowledge from TTS training for VC, which has significantly less training data. Huang *et al.* [47] proposed a technique to use a trained TTS decoder as the starting point of a VC decoder to train an encoder-decoder VC system. The study was focused on conversion of a specific source-target pair with parallel training data. It doesn't aim to disentangle speaker and linguistic information. Nonetheless, it represents a successful attempt in TTS-VC transfer learning.

All the studies suggest that voice conversion benefits from linguistically informed intermediate representations, and point to a direction for more studies on how voice conversion can benefit from TTS systems without involving large training data. TTS-VC transfer learning becomes a natural choice that will be the focus of this paper.

D. Speaker Disentanglement and Voice Cloning

Speaker disentanglement and voice cloning with autoencoder [37], [38] represents one of the successful techniques in voice conversion with non-parallel training data, where the encoder learns to disentangle speaker representation from speaker-independent linguistic representation, and the decoder reconstructs target speech with linguistic features, conditioning on target speaker representation. A speaker encoder is usually used to generate such speaker representation, e.g. speaker embeddings. Successful examples include i-vector [59], x-vector, and d-vector [60].

With speaker disentanglement, the decoder is able to reconstruct speech for any target speakers unseen during training, that we call voice cloning. Voice cloning has also been a successful technique in speech synthesis that takes text as input and generates voice of unseen speakers, when given a few speech samples [61], [62]. As the idea of voice cloning with speaker embeddings are proven effective in both TTS and VC, a common network architecture certainly facilitates the TTS-VC transfer learning.

III. TTS-VC Transfer Learning

Voice conversion is typically a research problem with scarce training data, however, deep learning techniques are typically data driven, that rely on big data. This is actually the strength of deep learning in voice conversion. Deep learning opens up many possibilities to benefit from abundantly available training data, so that the voice conversion task can be devoted to learning the mapping of speaker characteristics. For example, we wouldn't like the voice conversion task to infer low level detail during speech reconstruction, a neural vocoder can learn from a large database to do so [63]. We wouldn't like the voice conversion task to learn how to represent an entire phonetic system of a spoken language either, a general purpose acoustic model from a speech recognition or synthesis system can learn from a large database to do a better job.

The formulation of transfer learning aims to achieve just that. By leveraging the large database, we free up the conversion network from using its capacity to represent low level detail and general information, but instead, to focus on the high level semantics necessary for speaker identity conversion.

In this section, we will describe the proposed TTS-VC transfer learning architecture and a two-step training scheme. Fig. 3 illustrates the architecture of our proposed model and the loss function adopted during model training.

A. Pre-Training of the Multi-Speaker TTS Model

An encoder-decoder TTS model offers two useful properties: 1) The TTS encoder is trained to produce linguistic features from the text that is assumed speaker independent; 2) A multi-speaker TTS decoder provides a way to combine speaker-independent linguistic features and speaker embedding to produce speech in target voice. From voice conversion point of view, we would like to disentangle speaker-independent linguistic features from source speech and re-compose them with target speaker embedding to generate speech in target voice. The linguistic features and the decoder mechanism are the knowledge that voice conversion would like to learn from.

Tacotron-2 model was firstly studied for single speaker TTS [48]. To train a multi-speaker Tacotron model, we consider the use of the speaker embedding and where to apply the speaker embedding. We adopt the same speaker verification network [64] as in [55] to generate a fixed-dimensional speaker embedding vector. In [55], speaker embedding is applied on encoder output before the attention mechanism, hence, the resulting context vectors are speaker dependent. In this paper, we would like to generate context vectors that are sufficiently speaker independent. Therefore, we propose to incorporate speaker embeddings only after the attention mechanism as the controlling input to the TTS decoder. In this way, the encoder-decoder TTS architecture serves as a disentangling mechanism, which uses context vectors to represent speaker-independent linguistic features, and speaker embedding to represent the speaker's voice identity.

As shown in Fig. 3, the text encoder transform a sequence of text characters $\mathbf{X}_T = \{\mathbf{x}_T^1, \mathbf{x}_T^2, \ldots \mathbf{x}_T^M\}$ to a sequence of fixed-dimension embedding vectors:

$$\begin{aligned} \mathbf{O}_T &= Encoder_T(\mathbf{X}_T) \\ &= \{\mathbf{o}_T^1, \mathbf{o}_T^2, \ldots, \mathbf{o}_T^M\} \end{aligned} \tag{1}$$

where M denotes the length of the text sequence.

With the attention mechanism, we can obtain an alignment between the text embeddings and the mel-spectrogram features, that is described by a weight matrix $\mathbf{W}$. The context vectors

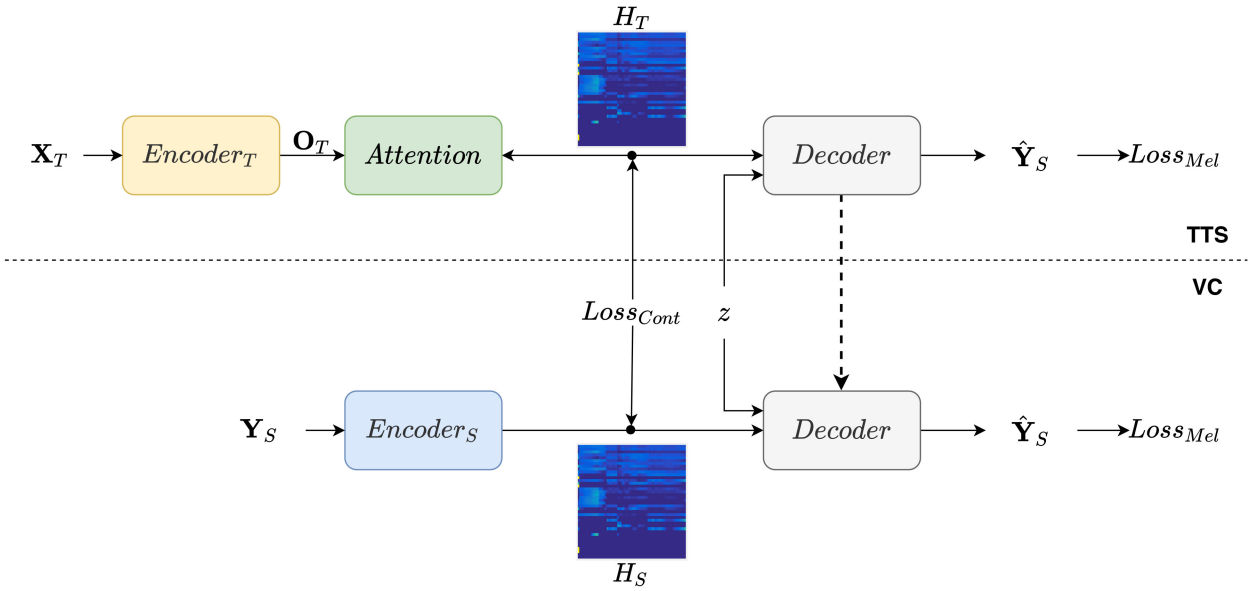

Fig. 3. Diagram of the proposed TTS-VC transfer learning architecture. The upper panel is a Tacotron TTS pipeline, and the lower panel is a voice conversion pipeline. $\mathbf{X}_T$ denotes the input text, $\mathbf{Y}_S$ and $\hat{\mathbf{Y}}_S$ are target mel-spectrogram and the mel-spectrogram generated by the pipelines; $\mathbf{O}_T$ denotes the text encoding, $\mathbf{H}_T$ denotes the context vectors from TTS pipeline, $\mathbf{H}_S$ denotes the context vectors equivalents from the VC pipeline; z denotes the speaker embedding.

$\mathbf{H}_T$ can be obtained by applying the weight matrix on the text embeddings:

$$\begin{aligned}\mathbf{H}_T &= \mathbf{W} \times \mathbf{O}_T \\ &= \{\mathbf{h}_T^1, \mathbf{h}_T^2, \ldots, \mathbf{h}_T^N\}\end{aligned} \tag{2}$$

where N denotes the length of the mel-spectrum frames. During training, we feed all mel-spectrum frames to the pre-net in a teacher-forcing mode, where N is the length of the training utterance. At run-time inference, N is predicted by the decoder.

The decoder takes the concatenation of context vectors $\mathbf{H}_T$ and speaker embedding z to generate the mel-spectrum features, $\hat{\mathbf{Y}}_S$:

$$\begin{aligned}\hat{\mathbf{Y}}_S &= Decoder(concat(\mathbf{H}_T, z)) \\ &= \{\hat{\mathbf{y}}_S^1, \hat{\mathbf{y}}_S^2, \ldots, \hat{\mathbf{y}}_S^N\}\end{aligned} \tag{3}$$

The loss function is defined as the mean square error (MSE) between the ground truth mel-spectrogram $\mathbf{Y}_S$ and the predicted one $\hat{\mathbf{Y}}_S$:

$$\begin{aligned}Loss_{Mel} &= MSE(\mathbf{Y}_S, \hat{\mathbf{Y}}_S) \\ &= \frac{1}{N}\sum_{n=1}^{N} ||\mathbf{y}_S^n - \hat{\mathbf{y}}_S^n||^2\end{aligned} \tag{4}$$

With a trained Tacotron-2 model, to get the context vector of the training data, we feed input text to the encoder and ground truth mel-spectrum to the decoder in a teacher-forcing mode. By doing this, we obtain a sequence of context vectors that has the same length as the sequence of mel-spectrum intended for voice conversion training.

B. Transfer Learning From TTS to VC

We propose an encoder-decoder voice conversion framework similar to those in [37], [47], [65] in terms of the system architecture. The VC encoder seeks to generate speaker-independent linguistic features from input spectral features, while the VC decoder reconstructs the mel-spectrum features from the linguistic features, conditioning on a speaker code. Studies show that voice conversion benefits from explicit phonetic modeling that ensure adherence to linguistic content during conversion [44], [45].

The question is how to establish the correspondence between input mel-spectrum features and the speaker-independent linguistic features. The PPG-based voice conversion techniques [25], [42], [43] require an external ASR system to work along side during training and inference. The autoencoder-style voice conversion frameworks learn the latent codes in an unsupervised manner, therefore, they are either speaker independent or phonetically motivated. Others rely on an explicit temporal alignment process [66].

Unlike the prior work, we propose to use the linguistic features, i.e. context vectors, from a trained TTS model, that are phonetically motivated, to serve as the supervision target of the VC latent code during training. By doing this, VC benefits from TTS model in many ways, as shown in Fig. 3. First, the trained TTS provides a temporal alignment between input text $\mathbf{X}_T$ and context vectors $\mathbf{H}_T$, the latter is frame-aligned with mel-spectrum features $\mathbf{Y}_S$; Second, the context vectors $\mathbf{H}_T$ represent speaker-independent linguistic features of input text, that are suitable to serve as the supervision targets of $\mathbf{H}_S$ for the voice conversion encoder; Third, the TTS decoder can be used as the initialization of VC decoder; Fourth, the same TTS training dataset can be used for VC training without the need of additional dataset. The voice conversion encoder can be described as follows,

$$\begin{aligned}\mathbf{H}_S &= Encoder_S(\mathbf{Y}_S) \\ &= \{\mathbf{h}_S^1, \mathbf{h}_S^2, \ldots, \mathbf{h}_S^N\}\end{aligned} \tag{5}$$

A loss function is introduced to minimize the distance between a VC latent code $\mathbf{H}_S$ and context vector $\mathbf{H}_T$:

$$Loss_{Cont} = MSE(\mathbf{H}_S, \mathbf{H}_T) = \frac{1}{N}\sum_{n=1}^{N} ||\mathbf{h}_S^n - \mathbf{h}_T^n||^2 \quad (6)$$

In this work, the VC decoder is similar to the TTS decoder functionally, that takes the concatenation of the linguistic features $\mathbf{H}_S$ and the speaker embedding z to generate the mel-spectrum features $\hat{\mathbf{Y}}_S$:

$$\hat{\mathbf{Y}}_S = Decoder(concat(\mathbf{H}_S, z)) = \{\hat{\mathbf{y}}_S^1, \hat{\mathbf{y}}_S^2, \ldots, \hat{\mathbf{y}}_S^N\} \quad (7)$$

The TTS decoder is trained with the TTS pipeline, that takes $\mathbf{H}_T$ as input. We propose to use the TTS decoder as the VC decoder and the VC encoder is trained to produced TTS latent codes. However, there could be a potential mismatch between the TTS decoder and $\mathbf{H}_S$. To minimize such mismatch, we use the TTS decoder as initialization, and refine the VC decoder through an adaptation process, with the same loss function as (4).

To summarize, the speech encoder and decoder are trained with the joint loss function as formulated in (4) and (6).

$$Loss_{Joint} = Loss_{Cont} + Loss_{Mel} = \frac{1}{N}\sum_{n=1}^{N}(||\mathbf{h}_S^n - \mathbf{h}_T^n||^2 + ||\mathbf{y}_S^n - \hat{\mathbf{y}}_S^n||^2) \quad (8)$$

The training procedure seeks to learn to disentangle linguistic information from speaker information and to optimize the representations for speech generation.

During the transfer learning, we use the same data as those for TTS pre-training. The difference is that the text transcript is no longer required in VC training. No additional speech data is required either.

C. Zero-Shot Run-Time Inference

Once the TTS-VC transfer learning is completed, the voice conversion pipeline is able to perform voice conversion independently without involving the attention mechanism of TTS. During run-time inference, both the source and the target speaker might be unseen speakers, therefore, the inference is referred to as any-to-any voice conversion. The proposed framework is able to perform such any-to-any voice conversion without further system training, that is also called zero-shot run-time inference.

To convert an utterance from source to target, we only need a speech sample, e.g. one utterance, from the target speaker. We use the speech sample to obtain a speaker embedding z_t from a speaker verification network. The run-time inference stage can be formulated as:

$$\hat{\mathbf{Y}} = Decoder(concat(Encoder_s(\mathbf{Y}), z_t)) \quad (9)$$

where $\mathbf{Y}$ denotes the source mel-spectrogram and $\hat{\mathbf{Y}}$ denotes the converted target mel-spectrogram.

D. Spectral and Prosodic Mapping

Traditionally voice conversion is performed by an analysis-mapping-reconstruction pipeline, where source speech is first decomposed into vocoding parameters such as spectral features, F_0 and aperiodicity indicators. Individual vocoding parameters are then mapped from source to target by respective conversion models. Statistical model, regression model, and deep learning model are commonly used. Recently, end-to-end TTS shows that it is possible to predict both spectral and prosodic features from a latent representation by a decoder [67]. This suggests that voice conversion can also be done in the same way if we are able to characterize the input speech with similar latent representation. In this paper, with TTS-VC transfer learning, we adopt the strategy in end-to-end TTS to model spectrum and prosody simultaneously. We consider $\mathbf{H}_T$ speaker-independent and F_0 agnostic. Hence, $\mathbf{H}_S$, which is learnt under the supervision of $\mathbf{H}_T$, is also expected to be F_0 agnostic. In this way, the decoder is capable to model F_0 without being influenced by F_0 from the source speaker. The decoder is followed by WaveRNN vocoder to reconstruct a speech signal.

E. Comparison With Other VC Systems

The proposed TTS-VC transfer learning framework, denoted as TTL-VC for short, represents a new way to leverage TTS knowledge. It employs a simple architecture to couple with TTS for knowledge transfer and an independent voice conversion pipeline for inference. To draw a clear distinction between TTL-VC and other prior work, we provide a comprehensive comparison in terms of data requirement, system configuration, and application scenarios in Table I. The four systems in Table I are discussed in Section II as the prior work. They represent the recent progress in TTS-assisted voice conversion. From Table I, we note that TTL-VC is unique in many ways. With transfer learning, TTL-VC doesn't rely on TTS encoder during run-time inference, it is trained solely on the TTS training data without the need of additional training data; With disentangled latent codes, TTL-VC is able to perform any-to-any voice conversion without involving model re-training or adaptation for unseen speakers.

The four systems in Table I do not employ transfer learning, furthermore, they have different requirements about training data. Therefore, a direct comparison of their performance is not meaningful. Instead, we benchmark the proposed TTL-VC with three competing systems, as summarized in Table II, that share the same decoding strategy, as formulated in (9), and are trained on the same dataset for a fair comparison.

IV. Experiments

A. Experimental Setup and Model Architecture

The three competing baselines in Table II include a multi-speaker TTS model, a PPG-VC model and an AutoVC model. They represent the state-of-the-art voice conversion performance. They also share the same idea, that is to use speaker-independent linguistic features as the latent codes, in order to support any-to-any voice conversion.

TABLE I
THE PROPERTIES OF VOICE CONVERSION SYSTEMS THAT LEVERAGE TTS KNOWLEDGE. TTL-VC TRANSFER LEARNING SUPPORTS FLEXIBLE VOICE CONVERSION TASK (FROM ONE UNSEEN SPEAKER TO ANOTHER) AT RUN-TIME, WHILE KEEPING MINIMUM DATA REQUIREMENT (NON-PARALLEL TRAINING DATA, NO ADAPTATION FOR UNSEEN SPEAKERS, ONLY SPEECH IS REQUIRED DURING INFERENCE)

	TTS-VC transfer learning	**source-target training data**	**VC inference input**	**TTS involved during inference**	**additional training data for VC**	**VC application**
Park et al. [44]	no	non-parallel	text and speech	encoder and decoder	yes	any-to-many
Luong et al. [45]	no	non-parallel	speech	decoder	yes	any-to-one
Zhang et al. [46]	no	parallel	text and speech	decoder	yes	many-to-one
Huang et al. [47]	no	parallel	speech	decoder	yes	one-to-one
TTL-VC	yes	non-parallel	speech	decoder	no	any-to-any

TABLE II
PERFORMANCE BENCHMARKING BETWEEN TTL-VC AND THREE COMPETITIVE BASELINES. ALL SYSTEMS ARE TRAINED ON THE SAME MULTI-SPEAKER NON-PARALLEL DATASET, AND SHARE A SIMILAR DECODING ARCHITECTURE THAT TAKES LATENT LINGUISTIC FEATURES, CONDITIONING ON SPEAKER EMBEDDING, TO GENERATE TARGET SPEECH. NOTES: MS-TTS AND TTL-VC USE TEXT TRANSCRIPTS OF SPEECH DATA DURING TRAINING; MS-TTS TAKES TEXT AS INPUT, WHILE OTHERS TAKE SPEECH AS INPUT

	latent code	**latent code training**	**training data**	**decoder**	**inference input**
MS-TTS [48]	phonetic context vectors	supervised	text-speech pair	autoregression w/ attention	text
PPG-VC [42]	phonetic posteriogram	supervised	PPG-decoded speech	autoregression w/o attention	speech
AutoVC [39]	content code	unsupervised	speech only	framewise mapping w/o attention	speech
TTL-VC	phonetic context vectors	supervised	text-speech pair	autoregression w/o attention	speech

The speaker verification network used to generate speaker embedding [64] is a 3-layer LSTM network with 768 hidden units followed by a linear projection layer with a size of 256. The resulting d-vector serves as speaker embedding z_t for all systems in Table II. Next, we briefly describe the experimental setup in this comparative study.

1) TTS-VC Transfer Learning (TTL-VC): We employ a two-step training scheme for TTL-VC system.

First, a multi-speaker TTS (MS-TTS) is trained as a teacher model, that follows Tacotron-2 architecture [48] as illustrated in the upper panel of Fig. 3. The encoder converts input text to a sequence of 512-dimensional character embeddings. These embeddings pass through 3 1-dimensional convolutional layers, each containing 512 filters with 5 kernel size, followed by batch normalization [54] and GELUs activation [68]. The convolutional layer output is taken by a single bi-directional LSTM layer with 512 hidden units. The pre-net of the decoder is a stack of 2 fully-connected layers with 256 hidden units followed by GELUs activation. The LSTM layers in the decoder contain 2 uni-directional LSTM layers with 1024 hidden units. The linear projection layer outputs the 80-dimensional mel-spectrogram. The post-net contains 5 1-dimensional convolutional layers each containing 512 filters with 5 kernel size, followed by batch normalization and *tanh* activation.

Second, the voice conversion pipeline in TTL-VC takes speech as input and generates speech as output, that is illustrated in the lower panel of Fig. 3, where input and output speech is represented as 80-dimensional mel-spectrum features. We conduct the transfer learning as discussed in Section III.B. The VC encoder is trained to generate context vectors that are similar to those in TTS pipeline; the VC decoder is trained to take the context vectors and speaker embedding z_t to recompose the target speech. Both encoder and decoder adopt the same architecture of the MS-TTS model. No attention mechanism is involved in voice conversion pipeline as it only performs a framewise mapping, thus no temporal alignment is required.

2) Multi-Speaker TTS Model (MS-TTS): The MS-TTS model is the teacher model in TTL-VC. We note that MS-TTS is not a voice conversion system, it takes the text and speaker embedding as input, and generates mel-spectrum features as output. We adopt it as a baseline for two reasons.

First, while the MS-TTS system shares the same decoder architecture as TTL-VC, its context vectors $\mathbf{H}_T$ are produced only from the text without influence from any source speaker. We expect that the synthesized speech to be highly similar to that of target speaker; Second, by comparing the prosodic patterns between MS-TTS and TTL-VC, we would like to observe whether the attention mechanism in MS-TTS has an advantage over the framewise mapping in TTL-VC.

3) PPG-VC Model: An ASR system is trained with the Kaldi toolkit [69]. We first process the input speech into a sequence of 40-dimensional MFCC features with 12.5 ms frame shift. The ASR system then converts the speech feature sequence into a phonetic posteriogram (PPG) sequence, where a PPG frame represents the probability distribution over 132 phonetic classes. In PPG-VC, a pre-trained ASR serves as the encoder. Only the VC decoder is involved in the training. The VC decoder is similar to the TTS decoder in Tacotron-2 [48]. The decoder LSTM layer takes PPG frames as input, and generate 80-dimensional mel-spectrum as output.

We adopt the PPG-VC model as one of the baseline models because it shares a similar encoder-decoder architecture as TTL-VC. Furthermore, PPG-based VC systems represent state-of-the-art performance in recent voice conversion challenges (VCC) [70].

4) AutoVC Model: If we consider TTL-VC as an extension to the autoencoder (AE) based techniques, AutoVC serves as a competitive reference model. AutoVC, a successful AE-based implementation, is known for its impressive performance in speaker disentanglement [39]. With the same training data, the difference between TTL-VC and AutoVC mainly lies in the adoption of transfer learning. TTL-VC employs linguistically motivated context vectors while AutoVC obtains latent codes in an unsupervised manner. Simply speaking, the lower panel of Fig. 3 resembles the AutoVC workflow except that TTL-VC introduces $Loss_{Cont}$ during the encoder-decoder pipeline

training process. We implement AutoVC as a baseline to observe the benefit of TTS knowledge transfer.

For a fair comparison between TTL-VC and AutoVC [39], our AutoVC configuration follows that of the TTL-VC network. The dimension of the latent code and the up/down sampling factor are set to 32, the AutoVC decoder performs frame-wise mapping without auto-regression. In this way, the decoder only contains LSTM layers, linear projection and post-net. Both TTL-VC and AutoVC take 80-dimensional mel-spectrum as input and generate mel-spectrum features as output.

B. Database and Feature Extraction

All systems in Table II and WaveRNN vocoder are trained on the same dataset from LibriTTS database [71]. We use the *train-clean-360* subset of the database that contains 191.29 hours of speech data from a total of 904 speakers, that consist of 430 female and 474 male speakers. The audio files are recorded at 24 kHz sampling rate. We didn't apply any pre-processing techniques on the audio data.

For system evaluation, we use speech data from the VCC2018 [70]. We use the evaluation subset of the dataset that contains 8 source speakers (4 female speakers and 4 male speakers) and 4 target speakers (2 female speakers and 2 male speakers). Each speaker provides 35 utterances. Each source-target group, namely Female-Female, Female-Male, Male-Male, and Male-Female, has 4 speakers as the sources and 2 speakers as the targets, consisting of a total of 280 conversion pairs. The audio files are recorded at 22.05 kHz sampling rate.

We train a speaker verification network as speaker encoder on AISHELL2 [72] corpus. AISHELL2 contains 1000 hours of speech data from 1991 speakers, including 845 male speakers and 1146 female speakers. We obtain an equal error rate (EER) of 2.31% for speech samples, each having 3 seconds on average, on a test set of 63.8 hours that consists of 200 unseen speakers. Using the speaker encoder, we derive a speaker embedding z_t from 5 seconds of speech sample for all 12 speakers in VCC 2018 database, and visualize them in Fig. 4 using t-distributed stochastic neighbor embedding (t-SNE) algorithm [73]. We observe a clear clustering of speech samples by speakers.

All speech data is resampled to 16 kHz for PPG extraction. The acoustic features are extracted with 12.5 ms frame shift and 50 ms frame length. The ASR model contains 4 bidirectional gated recurrent unit (GRU) layers with 512 hidden units in each layer. Followed by a softmax layer, the 256-dimensional probability output is taken as PPG features.

C. Results and Discussion

1) Objective Evaluation: We evaluate the systems in terms of mel-cepstrum distortion (MCD) and root mean square errors of $\log F_0$ (RMSE) between converted and reference speech utterances [74]. MCD is defined as

$$MCD[dB] = 10/\ln 10 \sqrt{2 \sum_{d=1}^{D} (\hat{Y}_d - Y_d)^2}, \qquad (10)$$

where D is the mel-cepstral coefficients (MCCs) feature dimension, $\hat{Y}_d$ and Y_d are the d^{th} coefficients of the converted and original MCCs, respectively. A lower MCD value accounts for a lower distortion [75].

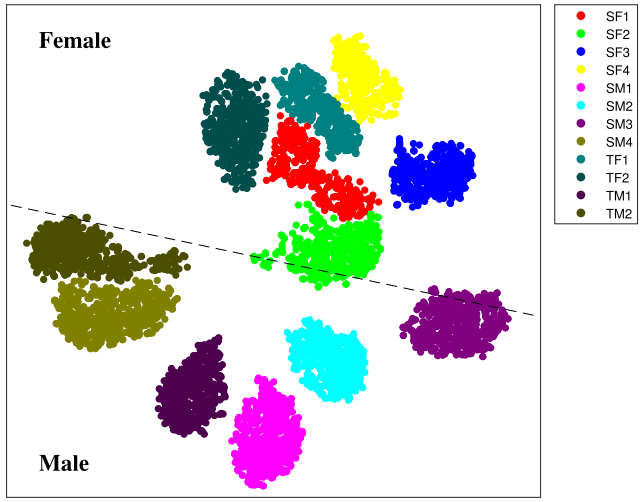

Fig. 4. Visualization of the speaker embedding clusters using t-SNE for 12 speakers (SF1-4, SM1-4, TF1, TF2, TM1, and TM2) in VCC 2018 dataset.

The $\log F_0$ RMSE is defined as

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (\log \hat{F}_{0_i} - \log F_{0_i})^2}, \qquad (11)$$

where N is the total number of frames, $\hat{F}_{0_i}$ and F_{0_i} represent the F_0 values at the i^{th} frame of the converted and reference speech, respectively.

As mel-spectrogram features are adopted as acoustic features, F_0 and MCCs are not readily available in the converted acoustic features. We extract F_0 and 25-dimensional MCCs using WORLD vocoder [76] from the reconstructed waveform for evaluation purpose. To account for the temporal difference, dynamic time warping is performed between the converted utterance and the target reference to compute MCD and $\log F_0$ RMSE, where $\log F_0$ RMSE is calculated only on the voiced frames in the reference utterances.

TABLE III summarizes the MCD and RMSE evaluation as an average over the 280 conversion pairs (4 source speakers $\times$ 2 target speakers $\times$ 35 utterances) for each source-target gender group, and 70 utterances (2 target speakers $\times$ 35 utterances) for MS-TTS speech synthesis. It is observed that TTL-VC outperforms all other voice conversion models consistently for both MCD and $\log F_0$ RMSE. We note that female speakers have a higher $\log F_0$ variance than male speakers, that is reflected in the $\log F_0$ RMSE of MS-TTS samples, and generated speech for female target speakers. For both male and female target speakers, TTL-VC shows a clear advantage in prosodic mapping. We are glad to see that TTL-VC significantly outperforms TTS systems, which suggests that TTL-VC not only delivers the exact linguistic content but also with improved prosody over TTS output.

By comparing TTL-VC and MS-TTS, we note that the difference lies in the encoder. The former takes speech as input, while the latter takes text as input. The results suggest that source

TABLE III
AVERAGE MCD AND $\log F_0$ RMSE OF BETWEEN THE CONVERTED SAMPLE AND THE TARGET REFERENCE FOR *AUTOVC*, *PPG-VC* AND *TTL-VC* MODELS. *SOURCE* DENOTES THE DISTORTION DIRECTLY BETWEEN SOURCE AND TARGET REFERENCE, WHICH COULD BE THE WORST SITUATION AS NO CONVERSION HAS TAKEN PLACE

Source-Target	Model	MCD (dB)	$\log F_0$ RMSE
Female-Female	Source	8.49	0.29
	AutoVC	4.03	0.30
	PPG-VC	4.16	0.33
	TTL-VC	**3.77**	**0.27**
Female-Male	Source	8.67	0.52
	AutoVC	3.46	0.42
	PPG-VC	3.47	0.37
	TTL-VC	**3.31**	**0.33**
Male-Male	Source	9.62	0.40
	AutoVC	3.54	0.29
	PPG-VC	3.55	0.37
	TTL-VC	**3.36**	**0.28**
Male-Female	Source	10.38	0.66
	AutoVC	4.19	0.32
	PPG-VC	4.25	0.34
	TTL-VC	**3.96**	**0.29**
Text-Female	MS-TTS	4.32	0.39
Text-Male	MS-TTS	3.85	0.35

speech is more informative than text input in providing speaker-independent prosodic patterns for target speech generation. This could be explained by the fact that the prosodic pattern of a sentence is a modulation between speaker-independent components, e.g. prosodic phrasing, intonation, and speaker-dependent components, e.g. accent, pitch level. Source speech provides essential speaker-independent components for the reconstruction of pitch contour, while text input doesn't.

To visualize the effect, Fig. 5 takes a male-female example to compare the spectrogram and F_0 of natural speech and generated speech from various models. We have three observations, 1) The duration of the TTS synthesized speech is predicted by an attention mechanism, it differs from that of either source or target natural speech. PPG-VC, AutoVC and TTL-VC generate speech that has the same duration as the source because they perform the framewise mapping. 2) The F_0 prosodic patterns of both PPG-VC and AutoVC are closer to the source, with a $\log F_0$ RMSE of 0.21 and 0.25, than to the target, with a $\log F_0$ RMSE of 0.23 and 0.26 respectively. This suggests that both PPG and autoencoder latent codes are influenced by the source speaker. 3) The F_0 prosodic pattern of TTL-VC is closer to the target ($\log F_0$ RMSE $= 0.08$) than to the source ($\log F_0$ RMSE $= 0.87$). This suggests that the context vectors are rather speaker independent, that allow the decoder to recompose speech well for target speaker using target speaker embedding. The observations in Fig. 5 are consistent with the performance statistics in Table III.

2) Subjective Evaluation: Subjective evaluations are performed through listening tests by human subjects. Both AB and XAB preference tests were conducted to assess speech quality and speaker similarity, respectively. In addition, to study listeners' preferences across all experimental systems, we further conducted best-worst scaling (BWS) and mean opinion score (MOS) tests [77] on speaker similarity and speech quality, respectively. 20 samples were randomly selected from the converted samples of each experimental system and provided to 15 participants for all the tests.[1] All listeners are university students and staff members, where English is their official language of instruction.

- AB preference tests: A and B were speech samples randomly selected from different systems. The listeners were asked to choose the sample having higher quality and naturalness. We have three comparison sets, where 7 samples were evaluated by each listener. The results are illustrated in Fig. 6. All experiment results are reported with p-value < 0.01, while the "TTL-VC vs. AutoVC" pair in female-female conversion is with p-value $= 0.017$. Therefore, the performance gains by TTL-VC over other methods are statistically significant. First, by comparing TTL-VC with PPG-VC, we observe that TTL-VC receives most of the preference votes for all source-target pairs. This suggests that context vectors outperform PPG as linguistic features; Second, between TTL-VC and AutoVC, we observe that TTL-VC also consistently outperforms AutoVC across all source-target pairs. This confirms the advantage of context vectors over AE latent codes. Last, when we focus on the comparison between TTL-VC and MS-TTS, it is found that TTL-VC outperforms baseline MS-TTS in Female-Female, Female-Male and Male-Female conversions, while it performs slightly worse than MS-TTS in Male-Male conversions. Overall, TTL-VC outperforms other competing VC models and performs comparably with MS-TTS on average.
- XAB preference tests: X was the reference target speech sample; A and B were speech samples randomly selected from different systems. The listeners were asked to listen both samples, and choose the one more similar to the target speaker. For each comparison pair, 7 sets of samples were evaluated by each listener on average. The results are illustrated in Fig. 7. All experiments are reported with p-values < 0.01, while the "TTL-VC vs. MS-TTS" pair in text-female conversion has p-value $= 0.033$. This suggests that the performance gains are statistically significant. By comparing TTL-VC with the competing models, we observe that TTL-VC obviously outperforms PPG-VC and AutoVC for all speaker pairs in terms of speaker similarity; furthermore, TTL-VC and MS-TTS are on a par with each other.
- BWS tests: We presented 4 samples of the same content from all four experimental systems to the listeners. The listeners were asked to pick the best and worst sample from the four. 20 scaling sets were evaluated by each listener. The detailed results are illustrated in Table IV. Overall, TTL-VC receives the highest best votes (43.05%) and the lowest worst votes (4.44%) respectively. The results indicate that the performance of all four experimental systems is ranked in the descending order as: TTL-VC, MS-TTS, AutoVC and PPG-VC.
- MOS tests: The listeners were asked to rate the speech quality and naturalness of the converted speech on a 5-point

[1]The generated speech samples for all the source-target group pairs of each system are available at https://arkhamimp.github.io/TTL-VC/

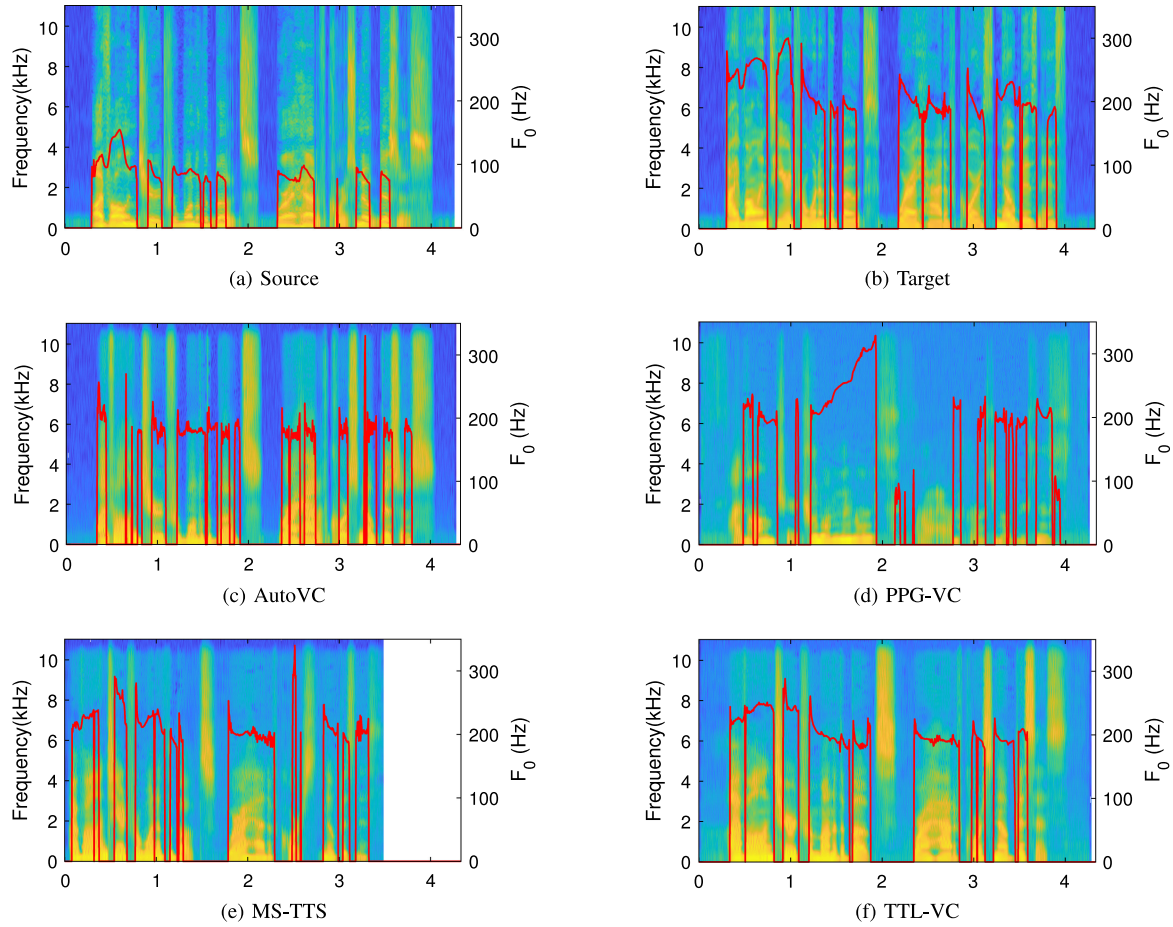

Fig. 5. The comparison of spectrogram and F_0 for a male-female voice conversion example '*I volunteered for one of the boats, where I had, of course, no business*' using four models. Horizontal axis (x-axis) displays time in second, and vertical axis (y-axis) represents spectral frequency and F_0 frequency respectively.

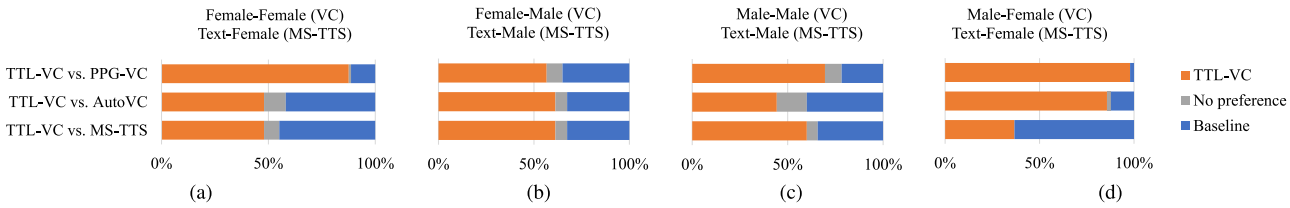

Fig. 6. AB test between TTL-VC and other models for four source-target groups.

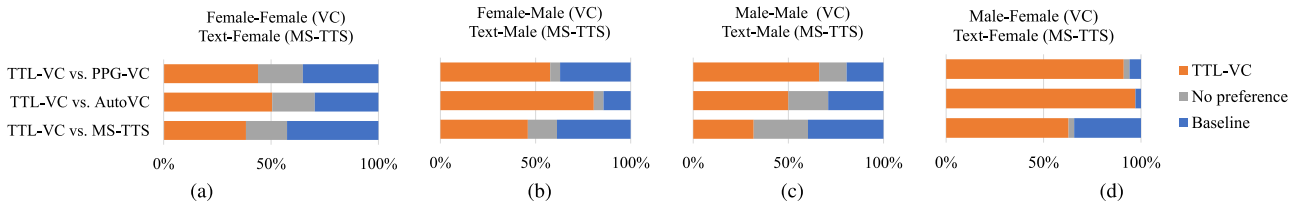

Fig. 7. XAB test between TTL-VC and other models for four source-target groups.

scale. A higher score indicates better quality. For each experimental system, 20 samples were rated by each listener. The average results are illustrated in Fig. 8. We find that TTL-VC receives the highest MOS score (3.75) among all systems after the natural target speech (4.64), that slightly outperforms MS-TTS (3.65). With slight variations in different speaker pairs, we could rank the systems from high to low performance in the following order, TTL-VC, MS-TTS, AutoVC, and PPG-VC. The observation is consistent with that in other listening tests.

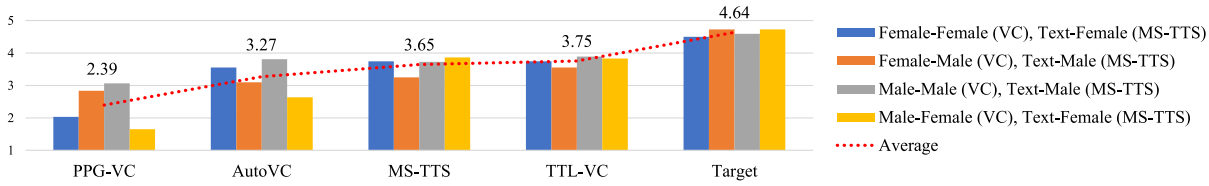

Fig. 8. MOS results of four models for four source-target groups and the average result.

TABLE IV
BEST-WORST SCALING TEST RESULTS FOR THE CONVERTED SAMPLES BY SOURCE-TARGET PAIR AND THEIR AGGREGATE. THE BEST% OR WORST% OVER FOUR MODELS SUM TO 100%

Source-Target	Model	Best%	Worst%
Female-Female	AutoVC	26.47	11.03
	PPG-VC	5.52	77.57
	TTL-VC	40.07	5.52
Text-Female	MS-TTS	27.94	5.88
Female-Male	AutoVC	7.57	57.37
	PPG-VC	25.90	29.88
	TTL-VC	40.63	2.79
Text-Male	MS-TTS	25.90	9.96
Male-Male	AutoVC	24.47	11.70
	PPG-VC	13.83	64.36
	TTL-VC	36.70	8.51
Text-Male	MS-TTS	25.00	15.43
Male-Female	AutoVC	3.85	28.85
	PPG-VC	0.96	67.31
	TTL-VC	54.81	0.96
Text-Female	MS-TTS	40.38	2.88
All	AutoVC	15.59	27.24
	PPG-VC	11.55	59.78
	TTL-VC	**43.05**	**4.44**
	MS-TTS	29.81	8.54

We observe that the proposed TTL-VC system clearly outperforms AutoVC and PPG-VC baselines in terms of speech quality, naturalness, and speaker similarity. In both MOS and BWS tests, it is encouraging to see that TTL-VC could succesully learn from TTS to achieve TTS quality without the need of text input at run-time inference.

V. CONCLUSION

This paper presents a novel strategy for TTS-VC transfer learning, which has a simpler run-time inference system architecture, yet achieves consistently higher performance than other systems of similar architecture. We have demonstrated the effectiveness of the transfer learning algorithm and the system architecture. It is particularly encouraging that we observe that the proposed system not only provides high quality spectral mapping, but also prosodic rendering.

REFERENCES

[1] A. Kain and M. W. Macon, "Spectral voice conversion for text-to-speech synthesis," in *Proc. Int. Conf. Acoust. Speech, Signal Process.*, vol. 1 1998, pp. 285–288.
[2] Chung-Hsien Wu, Chi-Chun Hsia, Te-Hsien Liu, and Jhing-Fa Wang, "Voice conversion using duration-embedded bi-HMMs for expressive speech synthesis," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 14, no. 4, pp. 1109–1116, Jul. 2006.
[3] T. Toda, M. Nakagiri, and K. Shikano, "Statistical voice conversion techniques for body-conducted unvoiced speech enhancement," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 20, no. 9, pp. 2505–2517, Nov. 2012.
[4] K. Nakamura, T. Toda, H. Saruwatari, and K. Shikano, "Speaking-aid systems using GMM-based voice conversion for electrolaryngeal speech," *Speech Commun.*, vol. 54, no. 1, pp. 134–146, 2012.
[5] L.-W. Chen, H.-Y. Lee, and Y. Tsao, "Generative adversarial networks for unpaired voice transformation on impaired speech," in *Proc. INTERSPEECH*, 2019, pp. 719–723.
[6] T. L. Nwe, M. Dong, P. Chan, X. Wang, B. Ma, and H. Li, "Voice conversion: From spoken vowels to singing vowels," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2010, pp. 1421–1426.
[7] T. Saitou, M. Goto, M. Unoki, and M. Akagi, "Speech-to-singing synthesis: Converting speaking voices to singing voices by controlling acoustic features unique to singing voices," in *Proc. IEEE Workshop Appl. Signal Process. Audio Acoust.*, 2007, pp. 215–218.
[8] A. Mouchtaris, Y. Agiomyrgiannakis, and Y. Stylianou, "Conditional vector quantization for voice conversion," in *Proc. IEEE Int. Conf. Acoust. Speech, Signal Process.*, 2007, pp. IV–505.
[9] T. Toda, A. W. Black, and K. Tokuda, "Voice conversion based on maximum-likelihood estimation of spectral parameter trajectory," *IEEE Trans. Audio, Speech, Language Process.*, vol. 15, no. 8, pp. 2222–2235, Nov. 2007.
[10] S. Takamichi, T. Toda, A. W. Black, and S. Nakamura, "Modulation spectrum-constrained trajectory training algorithm for gmm-based voice conversion," in *Proc. Int. Conf. Acoust. Speech, Signal Process.*, 2015, pp. 4859–4863.
[11] K. Tanaka, S. Hara, M. Abe, M. Sato, and S. Minagi, "Speaker dependent approach for enhancing a glossectomy patient's speech via gmm-based voice conversion," in *Proc. INTERSPEECH*, 2017, pp. 3384–3388.
[12] X. Tian, Z. Wu, S. W. Lee, and E. S. Chng, "Correlation-based frequency warping for voice conversion," in *Proc. IEEE Int. Sympo. Chin. Spoken Lang. Process.*, 2014, pp. 211–215.
[13] X. Tian, Z. Wu, S. W. Lee, N. Q. Hy, M. Dong, and E. S. Chng, "System fusion for high-performance voice conversion," in *Proc. INTERSPEECH*, 2015, pp. 2759–2763
[14] X. Tian, S. W. Lee, Z. Wu, E. S. Chng, and H. Li, "An exemplar-based approach to frequency warping for voice conversion," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 25, no. 10, pp. 1863–1876, Oct. 2017.
[15] L.-H. Chen, Z.-H. Ling, L.-J. Liu, and L.-R. Dai, "Voice conversion using deep neural networks with layer-wise generative training," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 22, no. 12, pp. 1859–1872, Dec. 2014.
[16] T. Nakashika, R. Takashima, T. Takiguchi, and Y. Ariki, "Voice conversion in high-order eigen space using deep belief nets," in *Proc. INTERSPEECH*, 2013, pp. 369–372.
[17] S. H. Mohammadi and A. Kain, "Voice conversion using deep neural networks with speaker-independent pre-training," in *IEEE Spoken Lang. Technol. Workshop*, 2014, pp. 19–23.
[18] L. Sun, S. Kang, K. Li, and H. Meng, "Voice conversion using deep bidirectional long short-term memory based recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust. Speech, Signal Process.*, 2015, pp. 4869–4873.
[19] M. Zhang, B. Sisman, L. Zhao, and H. Li, "Deepconversion: Voice conversion with limited parallel training data," *Speech Commun.*, vol. 122, pp. 31–43, 2020.
[20] Z. Wu, T. Virtanen, E. S. Chng, and H. Li, "Exemplar-based sparse representation with residual compensation for voice conversion," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 22, no. 10, pp. 1506–1521, Oct. 2014.
[21] B. Sisman, M. Zhang, and H. Li, "A voice conversion framework with tandem feature sparse representation and speaker-adapted wavenet vocoder," in *Proc. INTERSPEECH*, 2018, pp. 1978–1982.

[22] B. Sisman, M. Zhang, and H. Li, "Group sparse representation with wavenet vocoder adaptation for spectrum and prosody conversion," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 27, no. 6, pp. 1085–1097, Jun. 2019.

[23] T. Toda, Y. Ohtani, and K. Shikano, "Eigenvoice conversion based on gaussian mixture model," in *Proc. IEEE Int. Conf. Spoken Lang. Process.*, 2006, pp. 2446–2449,.

[24] R. Kuhn, J.-C. Junqua, P. Nguyen, and N. Niedzielski, "Rapid speaker adaptation in eigenvoice space," *IEEE Speech Audio Process.*, vol. 8, no. 6, pp. 695–707, Nov. 2000.

[25] X. Tian, J. Wang, H. Xu, E.-S. Chng, and H. Li, "Average modeling approach to voice conversion with non-parallel data," in *Proc. Odyssey: The Speaker Lang. Recognit. Workshop*, 2018, pp. 227–232.

[26] D. Erro, A. Moreno, and A. Bonafonte, "Inca algorithm for training voice conversion systems from nonparallel corpora," *IEEE Trans. on Audio, Speech, and Lang. Process.*, vol. 18, no. 5, pp. 944–953, Jul. 2010.

[27] D. Erro and A. Moreno, "Frame alignment method for cross-lingual voice conversion," in *Proc. INTERSPEECH*, 2007, pp. 1969–1972.

[28] Y. Qian, J. Xu, and F. K. Soong, "A frame mapping based HMM approach to cross-lingual voice transformation," in *Proc. IEEE Int. Conf. Acoust. Speech, Signal Process.*, 2011, pp. 5120–5123.

[29] M. Abe, S. Nakamura, K. Shikano, and H. Kuwabara, "Voice conversion through vector quantization," *J. Acoustical Soc. Jpn.*, vol. 11, no. 2, pp. 71–76, 1990.

[30] K. Shikano, S. Nakamura, and M. Abe, "Speaker adaptation and voice conversion by codebook mapping," in *Proc. IEEE Int. Symp. Circuits Syst.*, 1991, pp. 594–597.

[31] O. Turk and L. M. Arslan, "Robust processing techniques for voice conversion," *Comput. Speech Lang.*, vol. 20, no. 4, pp. 441–467, 2006.

[32] F. Fang, J. Yamagishi, I. Echizen, and J. Lorenzo-Trueba, "High-quality nonparallel voice conversion based on cycle-consistent adversarial network," in *Proc. IEEE Int. Conf. Acoust. Speech, Signal Process.*, 2018, pp. 5279–5283.

[33] T. Kaneko and H. Kameoka, "Cyclegan-vc: Non-parallel voice conversion using cycle-consistent adversarial networks,"*in Proc. Eur. Signal Process. Conf.*, 2018, pp. 2100–2104.

[34] T. Kaneko, H. Kameoka, K. Tanaka, and N. Hojo, "Cyclegan-vc2: Improved cyclegan-based non-parallel voice conversion," in *Proc. IEEE Int. Conf. Acoust. Speech, Signal Process.*, 2019, pp. 6820–6824.

[35] T. Kaneko, H. Kameoka, K. Tanaka, and N. Hojo, "CycleGAN-VC3: Examining and improving CycleGAN-VCs for mel-spectrogram conversion," in *Proc. INTERSPEECH*, 2020, pp. 2017–2021.

[36] H. Kameoka, T. Kaneko, K. Tanaka, and N. Hojo, "Stargan-vc: Non-parallel many-to-many voice conversion using star generative adversarial networks," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2018, pp. 266–273.

[37] C.-C. Hsu, H.-T. Hwang, Y.-C. Wu, Y. Tsao, and H.-M. Wang, "Voice conversion from non-parallel corpora using variational auto-encoder," in *Proc. IEEE Asia-Pacific Signal Inf. Process. Assoc.*, 2016, pp. 1–6.

[38] C.-C. Hsu, H.-T. Hwang, Y.-C. Wu, Y. Tsao, "Voice conversion from unaligned corpora using variational autoencoding wasserstein generative adversarial networks," in *Proc. INTERSPEECH*, 2017, pp. 3364–3368.

[39] K. Qian, Y. Zhang, S. Chang, X. Yang, and M. Hasegawa-Johnson, "AutoVC: Zero-Shot voice style transfer with only autoencoder loss," in *Proc. 36th Int. Conf. Mach. Learn., ser. Proc. Mach. Learn.* Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. Long Beach, California, USA: PMLR, 09-15 Jun 2019, pp. 5210–5219.

[40] K. Qian, Z. Jin, M. Hasegawa-Johnson, and G. J. Mysore, "F0-consistent many-to-many non-parallel voice conversion via conditional autoencoder," in *ICASSP 2020-2020 IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 6284–6288.

[41] T. J. Hazen, W. Shen, and C. White, "Query-By-Example spoken term detection using phonetic posteriorgram templates," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2009, pp. 421–426.

[42] L. Sun, K. Li, H. Wang, S. Kang, and H. Meng, "Phonetic posteriorgrams for many-to-one voice conversion without parallel data training," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2016, pp. 1–6.

[43] Y. Saito, Y. Ijima, K. Nishida, and S. Takamichi, "Non-parallel voice conversion using variational autoencoders conditioned by phonetic posteriorgrams and d-vectors," in *Proc. Int. Conf. Acoust. Speech, Signal Process.*, 2018, pp. 5274–5278.

[44] S.-w. Park, D.-y. Kim, and M.-c. Joe, "Cotatron: Transcription-guided speech encoder for any-to-many voice conversion without parallel data," 2020, *arXiv:2005.03295*.

[45] H. Luong and J. Yamagishi, "Bootstrapping non-parallel voice conversion from speaker-adaptive text-to-speech," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2019, pp. 200–207.

[46] M. Zhang, X. Wang, F. Fang, H. Li, and J. Yamagishi, "Joint training framework for text-to-speech and voice conversion using multi-source tacotron and WaveNet," in *Proc. INTERSPEECH*, 2019, pp. 1298–1302.

[47] W.-C. Huang, *et al.*, "Voice transformer network: Sequence-to-sequence voice conversion using transformer with text-to-speech pretraining," in *Proc. Interspeech*, 2020, pp. 4676–4680, doi: 10.21437/Interspeech.2020-1066.

[48] J. Shen *et al.*, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," in *Proc. IEEE Int. Conf. Acoust. Speech, Signal Process.*, 2018, pp. 4779–4783.

[49] Y. Wang *et al.*, "Tacotron: Towards end-to-end speech synthesis," in *Proc. INTERSPEECH*, 2017, pp. 4006–4010, doi: 10.21437/Interspeech.2017-1452.

[50] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*.

[51] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-Based Models for Speech Recognition," in *Adv. Neural Inf. Process. Syst.* 28, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 577–585.

[52] A. Van Den Oord *et al.*, "WaveNet: A. generative model for raw audio," in *Proc. 9th ISCA Speech Synthesis Workshop*, 2016, p. 125.

[53] N. Kalchbrenner *et al.*, "Efficient neural audio synthesis," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 2415–2424.

[54] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn., ser. Proc. Mach. Learn. Res.*, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07-09 Jul 2015, pp. 448–456.

[55] Y. Jia, Y. Zhang *et al.*, "Transfer Learning From Speaker Verification to Multispeaker Text-to-Speech Synthesis," in *Adv. Neural Inf. Process. Syst.* 31, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Curran Associates, Inc., 2018, pp. 4480–4490.

[56] E. Cooper *et al.*, "Zero-shot multi-speaker text-to-speech with state-of-the-art neural speaker embeddings," in *Proc. IEEE Int. Conf. Acoust. Speech, Signal Process.*, 2020, pp. 6184–6188.

[57] J.-X. Zhang, Z.-H. Ling, Y. Jiang, L.-J. Liu, C. Liang, and L.-R. Dai, "Improving sequence-to-sequence voice conversion by adding text-supervision," in *Proc. IEEE Int. Conf. Acoust. Speech, Signal Process.*, 2019, pp. 6785–6789.

[58] J.-X. Zhang, Z.-H. Ling, L.-J. Liu, Y. Jiang, and L.-R. Dai, "Sequence-to-sequence acoustic modeling for voice conversion," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 27, no. 3, pp. 631–644, Mar. 2019.

[59] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 4, pp. 788–798, May 2011.

[60] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *Proc. IEEE Int. Conf. Acoust. Speech, Signal Process.*, 2014, pp. 4052–4056.

[61] S. J. Arik, K. Chen Peng, W. Ping, and Y. Zhou, "Neural Voice Cloning With a Few Samples," in *Advances in Neural Information Processing Systems* 31, S. Bengio, H. Wallach, K. H. Larochelle, K. Grauman N. Cesa-Bianchi, and R. Garnett, Eds. Curran Associates, Inc., 2018, pp. 10019–10029.

[62] H.-T. Luong and J. Yamagishi, "Nautilus: A versatile voice cloning system," in *Proc. IEEE/ACM Trans. Audio, Speech, Lang. Process.*, 2017, pp. 2967–2981, doi: 10.1109/TASLP.2020.3034994.

[63] J. Chorowski, R. Weiss, S. Bengio, and A. Oord, "Unsupervised speech representation learning using wavenet autoencoders," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 27, no. 12, pp. 2041–2053, Dec. 2019.

[64] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, "Generalized end-to-end loss for speaker verification," in *Proc. IEEE Int. Conf. Acoust. Speech, Signal Process.*, 2018, pp. 4879–4883.

[65] J.-c.Chou, C.-c. Yeh, H.-y. Lee, and L.-s. Lee, "Multi-target voice conversion without parallel data by adversarially learning disentangled audio representations," in *Proc. INTERSPEECH*, 2018, pp. 501–505.

[66] J. Zhang, Z. Ling, and L. Dai, "Non-parallel sequence-to-sequence voice conversion with disentangled linguistic and speaker representations," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 28, pp. 540–552, 2020.

[67] R. Skerry-Ryan *et al.*, "Towards End-to-End prosody transfer for expressive speech synthesis with tacotron," in *Proc. Mach. Learn.Res.*, J. Dy and

A. Krause, Eds., vol. 80. Stockholmsmassan, Stockholm Sweden: PMLR, 10-15 Jul 2018, pp. 4693–4702.
[68] D. Hendrycks and K. Gimpel, "Bridging Nonlinearities and Stochastic Regularizers With Gaussian Error Linear Units," 2016, CoRR, vol. abs/1606.08415.
[69] D. Povey *et al.*, "The kaldi speech recognition toolkit," in *Proc. IEEE Workshop Auto. Speech Recognit. Understanding*, 2011.
[70] J. Lorenzo-Trueba *et al.*, "The voice conversion challenge 2018: Promoting development of parallel and nonparallel methods," in *Proc. Odyssey: The Speaker Lang. Recognit. Workshop*, 2018, pp. 195–202.
[71] H. Zen *et al.*, "LibriTTS: A corpus derived from LibriSpeech for text-to-speech," in *Proc. INTERSPEECH*, 2019, pp. 1526–1530.
[72] J. Du, X. Na, X. Liu, and H. Bu, "AISHELL-2: Transforming mandarin ASR research into industrial scale," 2018, *arXiv:1808.10583*.
[73] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *J. Mach. Learn. Res.*, vol. 9, no. 86, pp. 2579–2605, 2008.
[74] H. Ze, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2013, pp. 7962–7966.
[75] R. Kubichek, "Mel-cepstral distance measure for objective speech quality assessment," in *Proc. IEEE Pacific Rim Conf. Communications Comput. Signal Process.*, 1993, pp. 125–128.
[76] 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. E 99.D, no. 7, pp. 1877–1884, 2016.
[77] B. Sisman, H. Li, and K. C. Tan, "Sparse representation of phonetic features for voice conversion with and without parallel data," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2017, pp. 677–684.

Mingyang Zhang received the B.Eng. degree in electrical information engineering from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2014. He is currently working toward the Ph.D. degree with Key Laboratory of Underwater Acoustic Signal Processing of Ministry of Education, Southeast University, Nanjing, China. He is also with the Department of Electrical andComputer Engineering, the National University of Singapore, Singapore for a research attachment. In 2018, he was a Research Intern with the National Institute of Informatics, Tokyo, Japan. His current research interests include speech synthesis and voice conversion.

Yi Zhou received the B.Eng. degree in electrical and electronic engineering from Nanyang Technological University, Singapore, in 2015. She is currently a Research Scholar working toward the Ph.D. degree with Human Language Technology Lab, Department of Electrical and Computer Engineering, the National University of Singapore. Her research focuses on cross-lingual speech generation.

Li Zhao received the B.E. degree from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1982, the M.S. degree from Suzhou University, Suzhou, China, in 1988, and the Ph.D. degree from the Kyoto Institute of Technology, Kyoto, Japan, in 1998. He is currently a Professor with Southeast University, Nanjing, China. His research interests include speech signal processing and pattern recognition.

Haizhou Li (Fellow, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees in electrical and electronic engineering from the South China University of Technology, Guangzhou, China, in 1984, 1987, and 1990, respectively. He is currently a Professor with the Department of Electrical and Computer Engineering, National University of Singapore (NUS), Singapore. Prior to joining NUS, he taught in The University of Hong Kong, Hong Kong, from 1988 to 1990 and South China University of Technology, Guangzhou, China, from 1990to 1994. He was a Visiting Professor with CRIN in France from 1994 to 1995, the Research Manager with the Apple-ISS Research Centre from 1996 to 1998, the Research Director with Lernout & Hauspie Asia Pacific from 1999 to 2001, the Vice President of InfoTalk Corp. Ltd. from 2001 to 2003, and the Principal Scientist and Department Head of the Human Language Technology, Institute for Infocomm Research, Singapore from 2003 to 2016. His research interests include automatic speech recognition, speaker and language recognition, and natural language processing. He was the Editor-in-Chief of IEEE/ACM TRANSACTIONS ON AUDIO, SPEECH AND LANGUAGE PROCESSING from 2015 to 2018, a Member of the Editorial Board of Computer Speech and Language from 2012 to 2018. He was an elected Member of the IEEE Speech and Language Processing Technical Committee from 2013 to 2015, the President of the International Speech Communication Association from 2015 to 2017, the President of Asia Pacific Signal and Information Processing Association from 2015 to 2016, and the President of the Asian Federation of Natural Language Processing from 2017 to 2018. He was the General Chair of ACL 2012, INTERSPEECH 2014 and IEEE ASRU 2019. He is a Fellow of the ISCA. He was the recipient of the National Infocomm Award 2002 and the President's Technology Award 2013 in Singapore. He was named one of the two Nokia Visiting Professors in 2009 by the Nokia Foundation, and the Bremen Excellence Chair Professor in 2019.