Investigation of learning abilities on linguistic features in sequence-to-sequence text-to-speech synthesis

Yusuke Yasuda\textsuperscript{a,b,*}, Xin Wang\textsuperscript{a}, Junichi Yamagishi\textsuperscript{a,b,c}

\textsuperscript{a}National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, 101-8430, Japan
\textsuperscript{b}SOKENDAI (The Graduate University for Advanced Studies), Shonan Village, Hayama, Kanagawa, 240-0193, Japan
\textsuperscript{c}Centre for Speech Technology Research, University of Edinburgh, 10 Crichton Street Edinburgh, EH8 9AB, UK

Abstract

Neural sequence-to-sequence text-to-speech synthesis (TTS) can produce high-quality speech directly from text or simple linguistic features such as phonemes. Unlike traditional pipeline TTS, the neural sequence-to-sequence TTS does not require manually annotated and complicated linguistic features such as part-of-speech tags and syntactic structures for system training. However, it must be carefully designed and well optimized so that it can implicitly extract useful linguistic features from the input features.

In this paper we investigate under what conditions the neural sequence-to-sequence TTS can work well in Japanese and English along with comparisons with deep neural network (DNN) based pipeline TTS systems. Unlike past comparative studies, the pipeline systems also use neural autoregressive (AR) probabilistic modeling and a neural vocoder in the same way as the sequence-to-sequence systems do for a fair and deep analysis in this paper. We investigated systems from three aspects: a) model architecture, b) model parameter size, and c) language. For the model architecture aspect, we adopt modified Tacotron systems that we previously proposed and their variants using an encoder from Tacotron or Tacotron2. For the model parameter size aspect, we investigate two model parameter sizes. For the language aspect, we conduct listening tests in both Japanese and English to see if our findings can be generalized across languages.

Our experiments on Japanese demonstrated that the Tacotron TTS systems with increased parameter size and input of phonemes and accentual type labels outperformed the DNN-based pipeline systems using the complicated linguistic features and that its encoder could learn to compensate for a lack of rich linguistic features. Our experiments on English demonstrated that, when using a suitable encoder, the Tacotron TTS system with characters as input can disambiguate pronunciations and produce natural speech as good as those of the systems using phonemes. However, we also found that the encoder could not learn English stressed syllables from characters perfectly and hence resulted in flatter fundamental frequency. In summary, these experimental results suggest that a) a neural sequence-to-sequence TTS system should have a sufficient number of model parameters to produce high quality speech, b) it should also use a powerful encoder when it takes characters as inputs, and c) the encoder still has a room for improvement and needs to have an improved architecture to learn supra-segmental features more appropriately.

Keywords: Text-to-speech synthesis, deep learning, Sequence-to-sequence model, End-to-end learning, Tacotron

1. Introduction

Traditional text-to-speech synthesis (TTS) such as the deep neural network (DNN)-based statistical parametric speech synthesis (SPSS) framework \cite{1} converts input text into output waveforms by using modules in a pipeline: a text analyzer to derive linguistic features such as syntactic and prosodic tags from text, a duration model to predict the phoneme duration, an acoustic model to predict the acoustic features such as mel-cepstral coefficients and \( F_0 \), and a vocoder to produce the waveform from the acoustic features. Such a TTS system can produce reasonably good waveforms, but training the modules in the pipeline can be laborious. For example, the text analyzer requires manually annotated prosodic tags; the duration and acoustic models need alignment between the linguistic and acoustic feature sequences.

Sequence-to-sequence neural TTS is a recently developed framework that uses a single model to conduct the task of all or most modules in the SPSS-based TTS pipeline. For example, an ideal end-to-end sequence-to-sequence TTS uses a single neural network to directly convert the text into the waveform. Because such a TTS system is expected to implicitly learn the word pronunciation, prosodic realization, speaking styles, and the alignment between text and speech, it can be trained from many TTS databases with only the waveform and the text transcription. In practice, a sequence-to-sequence TTS system may still leverage a separate neural waveform generator and a grapheme-to-phoneme converter for ideographic languages such as Japanese and Chinese. Although it is not fully end-to-end, such a TTS system requires no alignment and simplifies the training process.

Several sequence-to-sequence based TTS methods have been...
proposed so far \cite{2,3,4,5,6,7}. Some of them have been evaluated against the DNN-based SPSS pipeline and unit-selection-based TTS systems, and it is widely agreed now that sequence-to-sequence based TTS can generally generate more natural synthetic speech. Particularly, Tacotron \cite{6}, which is a successor to Tacotron \cite{2}, and a Transformer-based TTS system \cite{7} have advanced the sequence-to-sequence based TTS methods to the human-level naturalness.

We applied Tacotron \cite{2} to Japanese in our previous research \cite{8}. Unlike English, Japanese is not a language to which sequence-to-sequence based TTS methods can be applied straightforwardly, mainly due to two issues: character diversity and pitch accent. Our modified Tacotron systems successfully produced synthetic speech with correct pitch accent by using accentual type labels along with phonemes as inputs. However, the synthetic speech of our proposed systems did not match the naturalness of those from comparable pipeline systems using an autoregressive probabilistic model and neural vocoder.

One possible way to fill the gap between our proposed systems and the pipeline systems is to introduce richer full-context label: the super set of phonemes and accentual type labels containing more complex features (e.g., part-of-speech tag and syntactic analysis). However, this strategy is the exact opposite of end-to-end sequence-to-sequence TTS strategies. Another approach is empowering sequence-to-sequence TTS models themselves to compensate for their lack of complex features by implicitly extracting more from simple inputs like texts. However, little known about what kind of linguistic features can be compensated for by a powerful model and which changes (e.g., parameter size, neural network structure, and input linguistic feature) contribute to the improvement to what extent. This is what we want to investigate in this paper.

In this paper, we improve our sequence-to-sequence TTS models by following the changes made by Tacotron2 \cite{6} and compare the effect of each group of the changes. Concretely, the configuration changes include increasing the model parameter size and simplifying the encoder’s network structure. Furthermore, we apply the best configurations to English TTS systems using character input and analyze if the systems can implicitly learn linguistic features such as phone and lexical stress.

We also compared our sequence-to-sequence TTS models with the new configuration against strong pipeline systems with autoregressive (AR) probabilistic models and WaveNet-based waveform generation. Note that, in many studies, the sequence-to-sequence based methods have been compared with pipeline systems using a non-AR acoustic model, a conventional vocoder for waveform generation, or both \cite{2,6,5}. Our comparison may be fairer because recent studies have shown that AR modeling is crucial for not only sequence-to-sequence \cite{9} but also the DNN-based pipeline TTS systems \cite{10}.

This paper is organized as follows. In section 2 we review the background of the pipeline based TTS method, sequence-to-sequence TTS methods, and our previous work about Tacotron based Japanese TTS methods. In addition, we summarize the transition from Tacotron to Tacotron2 to provide background for our new experimental conditions. In section 3 we describe our TTS systems used for this investigation. In section 4 we explain the new experimental conditions and their results. Section 5 concludes our findings.

2. Background

2.1. TTS frameworks and Terminology

This paper describes a sequence-to-sequence TTS framework and compares it with the pipeline TTS framework. Since machine learning based speech synthesis was long studied prior to the deep learning period, this section clarifies the definition of the two TTS frameworks and a taxonomy of TTS methods belonging to the frameworks. Figure 1 shows TTS frameworks appearing in this paper.

Pipeline TTS framework consists of independent front-end and back-end models. Pipeline TTS framework includes HSMM based SPSS, DNN based SPSS, and vocoder-free TTS. HSMM based SPSS and DNN based SPSS predict vocoder parameter with HSMM or DNN, respectively, and they use conventional vocoder for waveform generation. Vocoder-free TTS predicts power spectrogram with DNN, and it uses inverse STFT and phase recovery to generate waveform. Neural vocoder can also be used for waveform generation. Sequence-to-sequence TTS has a single model, and it may use either vocoder parameter or power spectrogram, and corresponding waveform generation method or neural vocoder.

This paper is partially based on our previous work published in \cite{8}. The main focus of this journal paper is an analysis of implicit learning abilities on linguistic features in sequence-to-sequence text-to-speech synthesis in Japanese and English and differs from that of previous work, where we proposed a new neural network architecture for Japanese sequence-to-sequence text-to-speech synthesis.

1This paper is partially based on our previous work published in \cite{8}.
acoustic model to convert linguistic features into vocoder parameters [12]. Such an HSMM treats the phoneme duration as a hidden variable and does not require an independent aligner or a phoneme duration model. HSMM-based SPSS uses a conventional vocoder to produce the waveform [13][14], which suffers from the artifact caused by minimum phase and other assumptions in the speech production process [15].

DNN-based SPSS [11] uses a DNN rather than an HSMM as the acoustic model. It outperforms the HSMM-based SPSS by switching from the state-level acoustic modeling to frame-level regression [16]. However, typical DNN-based SPSS relies on an external duration model to align linguistic features and target vocoder parameters, even though there has been an attempt to combine HSMM with DNN [17]. Traditional DNN-based SPSS has relied on the conventional vocoder used in HSMM-based SPSS. However, since the birth of neural vocoders [18][19], some DNN-based SPSS systems have adopted a neural vocoder for waveform generation. These systems can improve the quality of synthetic speech by directly modeling the waveform with the neural vocoder.

There are also TTS approaches that do not depend on vocoder parameters. In these approaches, an acoustic model predicts a power spectrogram [20], an acoustic feature that is obtained with a simpler analysis and is closer in form to a waveform than vocoder parameters. In this method, a waveform is generated by applying inverse Short-Time Fourier Transform (STFT) with phase recovery. This approach is normally distinguished from SPSS and is referred to as vocoder-free TTS [21]. It inherits the pipeline structure of DNN-based SPSS, but it can alleviate buzzes caused by the conventional vocoder. However, it causes another type of artifact when the prediction accuracy of power spectrogram is poor or when the hop size of the windows is too large.

DNN-based SPSS and vocoder-free TTS are major approaches in pipeline TTS frameworks in recent works. Although the distinction between the two approaches is based on the type of acoustic feature, the difference becomes less important once a neural vocoder is used to generate the waveform. A neural vocoder can take arbitrary acoustic features as input. Vocoder parameters [22], fundamental frequency and frame-level linguistic features [18], and mel-scale amplitude spectrograms [23] have been used as input of the neural vocoder in pipeline TTS. For more details on the pipeline TTS frameworks, see Appendix Appendix A.

A sequence-to-sequence TTS framework is the main focus of this paper. As figure 1 shows, a sequence-to-sequence TTS framework has a different structure from the pipeline one: it unifies the front-end, duration model, and acoustic model as one model by utilizing the DNN-based sequence-to-sequence framework that can map a source-to-target sequence with a length mismatch. As well as pipeline TTS, sequence-to-sequence TTS has variant systems using different types of acoustic features and waveform generation methods: vocoder parameter and conventional vocoder [24][4], vocoder parameter and neural vocoder [3], mel and linear spectrogram and invert STFT with phase recovery [2], and mel or linear spectrogram and neural vocoder [6].

In this paper, we use pipeline” to refer to all methods in a pipeline framework and DNN-based pipeline TTS” to refer to both DNN-based SPSS and vocoder-free TTS. We clearly mention a specific TTS method if the distinction is important. We use “conventional” or “traditional” for the pipeline TTS framework even though some methods in the pipeline framework are relatively new in TTS history and the first sequence-to-sequence TTS was proposed only a few years ago [24].

2.2. Sequence-to-sequence based TTS

2.2.1. Overview

Sequence-to-sequence based TTS is a new TTS framework using a sequence-to-sequence learning method [25]. The sequence-to-sequence based TTS unifies the linguistic model, acoustic model, duration model, and ultimately the waveform model from the conventional pipeline framework into a single model. It has provoked many studies because of its potential to unify all models into one and its handiness as it uses text and speech only for model training.

Many architectures have been proposed for sequence-to-sequence based TTS. Notable architecture examples are Tacotron [2], Char2Wav [3], VoiceLoop [4], DeepVoice3 [5], DCTTS [26], Tacotron2 [6], and Transformer [7]. All of them have an encoder-decoder [27] structure, which consists of an encoder to encode linguistic input and a decoder to decode acoustic features autoregressively. From the perspective of NN types to implement the sequence-to-sequence framework, those TTS architectures can be roughly classified into recurrent neural network (RNN)-based [2][3][6], convolutional neural network (CNN)-based [5][26], self-attention based ones [7] or memory buffer [4].

Note that, although sequence-to-sequence based TTS is a new framework, each key element such as phone and linguistic feature embedding [28][29], autoregressive decoder [23][30], and postnet [31][32] has already been investigated for DNN-based pipeline TTS methods separately. A notable difference between the sequence-to-sequence and pipeline TTS frameworks is not the elements themselves but joint training of the elements.

2.2.2. Linguistic modeling

The pipeline TTS frameworks typically use grapheme-to-phoneme conversion. The sequence-to-sequence TTS instead uses an encoder to transform input text into a hidden representation that is supposed to encode the pronunciation and possibly prosody information.

An initial proof-of-concept of sequence-to-sequence TTS was investigated in Chinese with phoneme input [24]. Then character level input was investigated in English [2][3] and was confirmed to be feasible in other alphabetical languages [33] and Chinese [34][35]. The naturalness of synthetic speech using character input can be quite high for English [6].

Although implicit linguistic modeling is possible when the encoder has a number of sufficient non-linear transformation layers [9] and when there is a sufficient amount of training data, even large-scale speech corpora such as LibriTTS [36]...
never match a lexicon dictionary in terms of word coverage \cite{37}. This suggests a potential limitation of end-to-end TTS for out-of-vocabulary words and out-of-domain text, especially in languages with highly nonphonemic orthography such as English and ideographic languages such as Chinese. It is reported that phoneme input gives better results than character input \cite{38} \cite{35} for sequence-to-sequence based TTS and many recent sequence-to-sequence based TTS studies avoid character input and use phonemes instead. Some studies use both character and phoneme \cite{5} \cite{39} or a random mix of them \cite{40}. It is also reported that pre-training an encoder to learn grapheme-to-phoneme mapping is also an effective approach \cite{34}. Nevertheless, the sequence-to-sequence based TTS is convenient to construct a multi-lingual model by handling various inputs with a single model, and it has been investigated with character, phoneme, and even byte input \cite{41} \cite{42} \cite{43} \cite{35}.

There are also attempts to use richer linguistic information as additional inputs for sequence-to-sequence based TTS for tonal languages such as Chinese \cite{44}, and pitch accent languages such as Japanese \cite{45}, where tone or accent and prosody information are indispensable yet hard to learn from graphemes or phonemes implicitly. However, some studies report excessive information negatively affects their results \cite{46} \cite{47}. Simpler linguistic features such as pinyin with tone \cite{48}, phones with tone \cite{49} \cite{46}, or phonemes with accentual type \cite{4} seems to be good choices for these languages.

2.2.3. Alignment modeling

As another difference from DNN-based pipeline TTS, the sequence-to-sequence TTS does not use the external aligner or duration model. Instead, many sequence-to-sequence TTS methods use an attention mechanism \cite{50} to implicitly align source and target sequences.

There are many types of attention mechanisms. Basic ones include Gaussian mixture model (GMM) \cite{51}, additive \cite{50}, dot-product \cite{52}, and location-sensitive \cite{53} attention mechanisms, which were designed for other tasks but applied to TTS as in \cite{3} \cite{4} \cite{2} \cite{5} \cite{7} \cite{24} \cite{4} \cite{6}. Several attention mechanisms have also been proposed specifically for TTS, where the alignment between source and target is monotonic and the target sequence is much longer than the source. They include forward attention \cite{49}, stepwise monotonic attention \cite{54}, a mixture of logistic attention \cite{55}, and dynamic convolution attention \cite{56}, all of which use a time relative approach to enforce monotonicity.

In addition, attention regularization has also been proposed for robust and fast alignment learning \cite{20} \cite{57}. Furthermore, instead of using an advanced attention mechanism, systems based on CNN or self-attention that enable parallel training have advanced to fast non-autoregressive inference using attention distribution distillation \cite{59} or duration model distillation \cite{58}. Meantime, Watts et al. \cite{9} found out that jointly modeled alignment by attention is not the source of the high naturalness of sequence-to-sequence based TTS.

2.2.4. Acoustic modeling

Typically, decoders in the sequence-to-sequence TTS adopt deep autoregressive neural networks to produce the target acoustic feature sequence on the basis of the attention and encoder outputs.

The autoregressive structure is introduced to model its statistical dependency across frames and is very important for improving the prediction accuracy Watts et al. \cite{9}. However, the autoregressive modeling has a common problem called exposure bias – The autoregressive decoder is inclined to excessively rely on feedback acoustic features while ignoring the encoder output to predict the next acoustic feature frame. Furthermore, the inaccurate predictions accumulate across the frames and lead to alignment errors and unacceptable outputs – To alleviate this problem, most architectures apply dropout \cite{59} regularization to feedback in layers called pre-net \cite{2} \cite{5} \cite{6} \cite{7}. There are also methods using a Generative Adversarial Network (GAN) \cite{60}, forward-backward decoding \cite{61}, and teacher-student training \cite{48} to alleviate the exposure bias.

As for acoustic feature, mel-scale spectrogram is widely used. The fine-grained vocoder parameter is less commonly used since it increases the length of output sequences and make decoding more difficult.

2.3. Transition from Tacotron to Tacotron2

Next, we review the differences between Tacotron and Tacotron2 since the change from Tacotron to Tacotron2 is known to be a successful example of improving the naturalness of synthetic speech in sequence-to-sequence based TTS and since we also compare these specific architectures in our comparisons.

Details in the transition from Tacotron to Tacotron2 can be classified into three groups of changes: 1) simplified neural network modules, 2) increased model parameter size, and 3) introduction of a neural vocoder. Table 1 summarizes the network structure and parameter sizes of Tacotron and Tacotron2. Tacotron2 generally has simpler network structures than Tacotron. Tacotron uses an encoder that consists of pre-net \cite{2} and CBHG modules, where CBH stands for Convolution Banks, Highway networks \cite{62} and G stands for bidirectional-Gated recurrent unit (GRU) \cite{63} RNN. In Tacotron2, the encoder does not have pre-net \cite{2}, and the CBH module is replaced with simple CNN layers. Unlike Tacotron, Tacotron2 does not have the decoder RNN and instead has one additional layer in the attention RNN, so Tacotron2 has two layers in the attention RNN. The CBHG module in post-net \cite{2} is also replaced with CNN layers, and the role of post-net is changed to improve possibly oversmoothed mel-spectrograms from the autoregressive decoder, instead of converting the scale of the spectrogram from mel into linear. Meanwhile, Tacotron2 has an advanced attention mechanism. Tacotron2 has adopted location-sensitive attention \cite{53}, an extended version of additive attention \cite{50}. Location-sensitive attention can consider location in addition to content to align the source and target, so it is suitable for speech tasks and was originally tested in speech recognition. In contrast, additive attention was proposed for machine translation.

In addition to the simplified network structure, the model parameter size is increased significantly in Tacotron2. Table 1 also compares the parameter sizes of neural network layers in
Tacotron and Tacotron2. Along with the expansion of parameter size, many regularization techniques have been introduced, probably because models with huge capacity are more likely to suffer from overfitting. Dropout [59] is applied to every CNN layer in the encoder and post-net. Zoneout [54] regularization is applied to long short-term memory (LSTM) layers in the encoder and decoder. The L2 regularization term is added in the loss term. In contrast, in Tacotron, regularization is conducted only in the encoder pre-net and decoder pre-net through dropout.

Tacotron2 uses WaveNet [18] for waveform synthesis instead of the Griffin-Lim algorithm [65]. Human-level naturalness of synthetic speech was achieved by training WaveNet with a predicted spectrogram from Tacotron2. In other words, WaveNet can correct errors in a predicted spectrogram from Tacotron2 during waveform generation in this configuration.

2.4. Japanese sequence-to-sequence TTS

Finally, we overview sequence-to-sequence TTS systems designed for Japanese since we use Japanese as well as English for our experiments.

In the case of Japanese, the first issue is the diversity of input text characters, i.e., thousands of ideograms called kanji (i.e., logographic Chinese characters) alongside two syllabic scripts called hiragana and katakana. The huge number of kanji characters poses a challenge for the sequence-to-sequence TTS to learn the pronunciation. Another challenging yet not well recognized issue is the pitch accent estimation. Unlike the pitch accent in English, Japanese pitch accent directly affects the meaning of words and the perceived naturalness of the speech. Although the pitch accent of individual words can be encoded in a lexicon, it is affected by adjacent words in an accentual phrase, i.e., pitch accent sandhi.

In our previous research [8], we investigated the Japanese TTS using Tacotron [2]. We used the phoneme as the input and focused on the issue with pitch accent. Specifically, we proposed modified Tacotron architectures to model the Japanese pitch accent, which will be detailed in section 3. Through a large-scale listening test, we found that using accentual type (i.e., pitch accent in the accentual phrase) as an additional input feature significantly improved the pitch accent and the naturalness of the generated speech for native listeners. However, our Tacotron system using the phoneme and the accentual type as input performed worse than DNN-based pipeline systems with an autoregressive acoustic model [22, 23] and WaveNet [18] vocoder. This could be due to the difference between Tacotron and Tacotron2 described earlier. As far as we know, human-level naturalness of synthetic speech has not been achieved for Japanese TTS using Tacotron yet.

3. Speech synthesis systems used in this investigation

3.1. Pipeline systems

We used two DNN-based pipeline systems: DNN-based SPSS and vocoder-free TTS. Both pipelines use an acoustic model with autoregressive density modeling and a neural vocoder.

Figure 2A shows the architecture of acoustic models in the DNN-based SPSS system used in this study. This system is used for Japanese TTS research [22] and consists of two acoustic models to predict two vocoder parameters. The first model in the DNN-based SPSS system is responsible for modeling the mel-generalized cepstral coefficient (MGC). It is based on a bidirectional recurrent mixture density network (RMDN) [66], but autoregressive feedback connection is introduced at the output layer to improve density modeling with AR [67]. The autoregressive feedback from past adjacent frames is linearly transformed to compute output density parameters. The second model in the DNN-based SPSS system is a $F_0$ model. The $F_0$ output modeled in this model is discrete, based on a just-noticeable difference in pitch in speech perception [65]. Thus, its output is a softmax distribution. This model also has also autoregressive feedback, but the feedback is nonlinearly transformed by a stack of neural network layers. In addition, this

---

Table 1: Comparison of structure and configuration between Tacotron and Tacotron2. Numbers in brackets are unit size of layers.

|                     | Tacotron |Tacotron2 |
|---------------------|----------|----------|
| Embedding           | (256)    | (512)    |
| Encoder pre-net     | FFN (256, 128) | - |
| Encoder             | CBHG (256) | CNN + LSTM (512) |
| Decoder pre-net     | FFN (256, 128) |FFN (256, 256) |
| Attention RNN       | GRU (256) | LSTM (1024, 1024) |
| Attention           | Additive (256) | Location-sensitive (128) |
| Decoder RNN         | GRU (256, 256) | - |
| Post-net            | CBHG (256) | CNN (512) |
| Total parameters    | 6.9 $\times$ 10$^6$ | 27.3 $\times$ 10$^6$ |
| Vocoder             | Griffin-Lim [65] | WaveNet [18] |

Figure 2: Acoustic models of pipeline TTS systems used in our experiments. A: DNN-based SPSS; left is MGC prediction model based on autoregressive recurrent mixture density network, right is $F_0$ prediction model based on deep autoregressive recurrent neural network. B: Mel-spectrogram prediction model of Vocoder-free TTS.
feedback scheme has an architecture consisting of two network modules. One module is bidirectional RNN and additional network layers that take linguistic input, and the other is a unidirectional RNN layer that runs an autoregressive loop. The former module corresponds to an encoder and the latter corresponds to a decoder in a sequence-to-sequence framework [27]. However, unlike a sequence-to-sequence framework, this architecture’s input is a frame level linguistic feature aligned by a duration model, so source and target mapping is frame-by-frame.

Figure 2-B shows the architecture of an acoustic model in the vocoder-free TTS system used in this study. This system was used for a voice cloning study in English [23], its target acoustic feature is the mel-spectrogram. The architecture of the acoustic model consists of two network modules: a bidirectional RNN based module that processes linguistic features, and a unidirectional RNN based module with a deep autoregressive feedback scheme of nonlinear transformation, as well as the F0 model in the DNN-based SPSS system.

Both the DNN-based SPSS and vocoder-free TTS systems use HMMs to predict phoneme duration and a WaveNet neural vocoder to generate waveforms. The WaveNet predicts a quantized waveform in 10 bits with μ-law coding at a 16 kHz sampling rate. Both systems showed high naturalness in a listening test, and their scores did not differ much [6].

3.2. Baseline Tacotron

We used a modified version of Tacotron [8] to handle pitch accent in Japanese. We refer to this system as baseline Tacotron in this paper. Figure 3-A shows its network structure. Our baseline Tacotron is based on Chinese Tacotron proposed by Zhang et al. [49]. To handle pitch accent, the baseline Tacotron has separate embedding tables with different dimensions for phonemes and accentual type labels. Its encoder has two pre-nets [2]: bottle-neck layers for embedding vectors from phonemes and accentual type labels. The two inputs are then concatenated and encoded by convolution banks, highway networks [62], and bidirectional-LSTM [69] modules. Note that we replace the GRU cell [63] in the CBHG module with a LSTM cell to apply zoneout regularization [64]. Zoneout regularization along with the LSTM cell is introduced in the successors of the original Tacotron [70,6], and we use LSTM with the zoneout regularization for all RNN layers including the attention RNN and decoder RNN. We refer to the modified encoder module as “CBHL” from here on.

At the decoder, encoded values are processed with an attention framework [50,22,71] to align with decoder outputs. The output from the decoder pre-net is first concatenated with the context vector from the previous time step and processed by the attention RNN. The attention mechanism relates the encoder output and output from the attention RNN by assigning probabilities the attention mechanism assigns as weights. The final output of the attention framework is processed by the context vector and the output from the attention RNN. We use the LSTM cell for the attention RNN. For an attention mechanism, we use forward attention without a transition agent [49] instead of additive attention [50]. As mentioned in [49], the forward attention accelerates the alignment learning speed. In addition, forward attention is proved to give fewer alignment errors than additive and location-sensitive attention, which are used in Tacotron and Tacotron2 [54], and we observed the same trends in our preliminary experiments.

The output of the attention framework is processed by the decoder RNN. We use the LSTM cell for the decoder RNN. The output of the decoder RNN is projected to the mel-spectrogram with a linear layer as the final output of the network. In addition to the mel-spectrogram, Japanese Tacotron predicts stop flags [5] to determine when to stop the prediction loop.

We use WaveNet [18] to synthesize waveforms from the

Figure 3: A: Neural network architecture of baseline Tacotron. B: Neural network architecture of self-attention Tacotron. Numbers in brackets are parameter size for layers. Arrows indicate feedback from the previous time step. Note that displayed parameter sizes are adjusted for experiments conducted in this paper.
mel-spectrogram. WaveNet is trained with the same mel-spectrogram used for training Japanese Tacotron. We use the same conditions for the mel-spectrogram as [2]: 80 bins, 50 ms frame length, and 12.5 ms frame shift.

3.3. Self-attention Tacotron

We use self-attention Tacotron proposed in [8] as the third system to be investigated. The motivation to use self-attention in Tacotron is to capture long-term dependency better to produce correct pitch accent of Japanese language, which is phrase level information. It is known that by directly connecting distant states, self-attention relieves the high burden placed on LSTM to learn long-term dependencies to sequentially propagate information over long distances [72).

Figure 3-B shows the network structure of self-attention Tacotron. The difference of self-attention Tacotron from the baseline Tacotron is the existence of self-attention layers in both the encoder and decoder. We used a self-attention block based on [73]. The self-attention has multiple heads, and a residual connection is placed after the self-attention layer. As in [72], the self-attention block is placed after the LSTM layers. The encoder has two outputs: output of the bidirectional LSTM layer and output of the self-attention layer. The two encoder outputs are aligned to output acoustic features with different attention mechanisms in dual source attention [73]. The output from the bidirectional LSTM layer is fed to forward attention [49], and the output from the self-attention layer is fed to additive attention [50]. By using attention for each encoder output, we can check how the encoder outputs are used from the decoder by visualizing the attention probability distribution. In our previous study, we discovered that the forward attention captured the source-target relationship because it showed monotonic alignment, and the additive attention captured the phrase structure. The outputs from two attention layers are concatenated and processed by the decoder RNN followed by self-attention.

4. Experiments

Using the four speech synthesis systems and their variants, we conducted two experiments to analyze several important factors for end-to-end speech synthesis. The first experiment is about Japanese TTS to answer the following questions:

- Q1) How much can increasing model parameter size improve naturalness?

- Q2) How much do the differences in network structures between the CBHL encoder [2] and CNN-based encoder [6] affect naturalness in Japanese?

- Q3) How much can the presence of self-attention improve naturalness in Japanese?

- Q4) How much do our sequence-to-sequence based systems differ from comparable pipeline systems using full-context label in terms of naturalness of synthetic speech in Japanese, and where do the differences possibly come from?

The second experiment is about English TTS using raw text to answer the following questions:

- Q5) How much is character input inferior to phone input in our Tacotron systems?

- Q6) How much does the difference in encoder network structures affect naturalness in English, and can the encoders reduce the gap between character and phone input?

- Q7) How much can the presence of self-attention improve naturalness in English, and can it reduce the gap between character and phone input?

- Q8) How much do our sequence-to-sequence based Tacotron systems differ from comparable pipeline systems using full-context labels in terms of naturalness of synthetic speech in English, and where do the differences possibly come from?

4.1. Experimental conditions for Japanese systems

**Database.** For the Japanese speech corpus, we chose the ATR Ximera corpus [74], the same Japanese speech corpus used in the previous experiment [8]. This corpus contains 28,959 utterances or around 46.9 hours in total duration from a female speaker. We also used the same settings as in the previous research. For linguistic features, we used phonemes and accentual type labels from manually annotated labels [22]. We trimmed the beginning and ending silences from the utterances, after which the duration of the corpus was reduced to 33.5 hours. We used 27,999 utterances for training, 480 for validation, and 141 for testing.

**Tacotron configurations.** We used the baseline Tacotron and self-attention Tacotron described in section [8]. To construct models with large parameter size, we followed configurations of Tacotron2 [6]. We used 64 dimensions for accentual type embedding and 448 dimensions for phoneme embedding. Encoder pre-net had 448 and 224 dimensions for phoneme input and 64 and 32 dimensions for accentual type input. The CBHL encoder has 256 units, and its output from the bidirectional RNN is 512 dimensions. The decoder pre-net has 256 dimensions for each layer. The attention RNN has 128 dimensions, and decoder RNN has 1024 dimensions each. For the self-attention layer in self-attention Tacotron, we used 64 dimensions for the encoder, and 1024 dimensions for the decoder. These parameter configurations are also shown in figure 4-A for baseline Tacotron and figure 3-B for self-attention Tacotron. We also constructed variant models of Japanese Tacotron and self-attention Tacotron with the CNN based encoder from Tacotron2 [6] instead of the CBHL encoder to investigate the effect of the network structure change as model parameter size increases. We added CNN-based post-net of Tacotron2 to all systems for a fair comparison.

---

2Audio samples for our experiments can be found at https://nii-yamagishilab.github.io/yasuda-csl-tacotron-audio-samples/.
**Tacotron and Wavenet training.** We used L1 loss for the mel-spectrogram, and binary cross entropy loss for the stop flag. The models were optimized with an Adam optimizer [75], with exponential learning rate decay from $10^{-4}$ to $10^{-5}$. We applied L2 regularization with a weight of $10^{-6}$ for baseline Tacotron and $10^{-7}$ for self-attention Tacotron. We used a reduction factor of two, the same value as the previous work [8], instead of the reduction factor of one used in Tacotron2 [6], to avoid increasing training time. We did not enable dropout during inference unlike Tacotron2 [6]. For waveform synthesis, we used the same µ-law WaveNet [18] model from the previous research, which was trained with ground truth mel-spectrograms of the same condition as acoustic features used by our Tacotron systems: 50 ms frame length and 12.5 ms frame shift.

**Listening test.** We conducted a listening test about naturalness of synthetic speech. We recruited 399 native Japanese listeners via crowdsourcing. Listeners were asked to evaluate 36 samples using a five-grade scoring system in one set. One listener could evaluate at most ten sets. We collected 33,840 data points in total. We checked the statistical significance of the scores between systems with a Mann-Whitney rank test [76].

**Systems.** The listening test contained 12 systems including natural samples, three analysis-by-synthesis systems, two pipeline TTS systems, and six Tacotron systems. The three analysis-by-synthesis systems were WaveNet models used for the DNN based SPSS system using the vocoder parameter, for the vocoder-free TTS system using mel-spectrogram with 5 ms frame shift, and for the Tacotron systems using the mel-spectrogram with 12.5 ms frame shift. The two pipeline systems were the DNN based SPSS and vocoder-free TTS systems using an autoregressive acoustic model as described earlier. These pipeline systems used manually annotated full-context labels [22], of which the subsets are the phoneme and accentual type labels used for all the Tacotron systems. The six Tacotron systems are the original Tacotron and self-attention Tacotron with two parameter size configurations. We refer to the parameter size configurations of Tacotron and Tacotron2 as small” and large,” respectively. Tacotron2 with a CNN encoder and its self-attention variant were also evaluated.

**4.2. Results of Japanese systems**

Table 1 shows the number of samples containing fatal alignment errors from each system. There are three type of fatal alignment errors: discontinuous alignment, incomplete alignment, and overestimated duration, which can be easily determined with software or human eyes by looking at the attention probability distribution. The discontinuous alignment error is represented in a skip or repeat of a transcription and can be determined by finding non-monotonic jumps of the attention distribution mode. The incomplete alignment error is represented in a premature termination of a transcription and can be determined by finding an alignment distribution mode that does not reach the final input position at the final time step. The overestimated duration error is represented in an excessively prolonged phoneme and can be determined by finding an attention distribution mode staying in the same position for a long time. We detected the fatal alignment errors automatically from an extended test set containing 480 samples.

As can be seen from the table, all systems except one had no fatal alignment errors. We see that the system using small parameter size with self-attention had fatal alignment error rate of 7.9 %. Small parameter size seems to be sufficient to learn alignment for this corpus but somewhat sensitive. On the other hand, all systems with large parameter size have no fatal errors.

**Figure 4** shows the results of the listening test in Japanese. Please refer to Table 3 for a statistical significance test. We can see a significant score gap between Tacotron systems with small and large parameter sizes. Both baseline Tacotron and self-attention Tacotron systems with small parameter size had low scores of 2.98±0.03 and 2.88±0.03, respectively. We investigated samples that had average mean opinion scores (MOSs) of less than 2.5 from these systems by listening and found incorrect pitch accents from the lowly rated samples. Some lowly rated samples from self-attention Tacotron with small parameter size contained fatal alignment errors, which was suggested by the alignment error detector. On the other hand, Tacotron systems with large parameter size had high MOSs of about 3.8. There were no statistically significant differences among these systems according to the statistical significance test, so the presence of self-attention and difference in encoder net-
work structure did not majorly affect the naturalness of synthetic speech. These Tacotron systems had similar scores to the pipeline system using the mel-spectrogram, which had an MOS of 3.74 ± 0.03. The Tacotron systems using the CBHL encoder had statistically significant differences from the pipeline system using mel-spectrogram, but the Tacotron systems using the CNN based encoder did not. The pipeline system using the vocoder parameter had a lower MOS (3.48 ± 0.03) than the pipeline system using the mel-spectrogram and the Tacotron system with large parameter size.

4.3. Discussion on results in Japanese

This experiment showed: Q1) Increasing parameter size greatly improved naturalness. Q2) The difference in encoder structure did not make much difference in Japanese. Q3) Self-attention did not improve naturalness for models with large parameter size. Self-attention was expected to improve naturalness for models with small parameter size according to our previous research [8], but any improvement was undercut by alignment errors caused by poor training in this experiment. Q4) The Tacotron system with large parameter size using the CBHL encoder slightly outperformed the pipeline system using the same mel-spectrogram and the same neural vocoder.

In the previous research [8], our Tacotron systems did not match the pipeline systems in terms of the quality of synthetic speech, and we hypothesized that one possible reason was linguistic feature limitation, but this hypothesis turns out to be incorrect. The main reason is simply the parameter size.

When a model has sufficient parameter capacity, we confirmed that the Tacotron systems can learn to compensate for the lack of other rich information and accurately predict mel-spectrogram without requiring additional contextual labels, which are used in the pipeline systems such as part-of-speech tags and syntactic analysis results. The score of the best Tacotron system (large parameter size, the CBHL encoder) is better than that of the pipeline system, but, as expected, the difference is smaller than what the literature reported in the past since both systems use autoregressive modeling and the same neural vocoder.

There are many differences between Tacotron and Tacotron 2 apart from the use of neural vocoders as we described earlier. Our result, however, reveals that such changes are not fundamental differences and do not bring about significant improvements, at least, for our Japanese TTS systems using phonemes. Our analysis results indicate that the major improvements simply come from the model capacity.
However, we also observed that the linguistic feature limitation affected accuracy of pitch accent prediction for a few systems. More specifically, we found incorrect pitch accent from audio samples generated using the systems with small parameter size. For pitch accent information, we fed accentual type labels, which indicate the accentual phrase boundary and type of pitch accent but do not explicitly indicate where to drop pitch. Therefore, we think that the systems with small parameter size were occasionally not able to resolve the position to drop pitch given accentual type labels alone. This is consistent with results reported in Fujimoto et al. [47], where an abstract pitch contour pattern is used such as high and low labels plus an accentual phrase boundary.

### 4.4. Experimental conditions for English systems

**Database.** We chose the Blizzard 2011 corpus [77] as a speech corpus to construct English TTS models. This corpus contains 12,092 utterances or around 17 hours in total duration from a female speaker. This time we used characters or phones for input. The Blizzard 2011 corpus is suitable to compare character and phone input because it has a wider variety of unique word types (18,695 unique words) [37], so we can thoroughly check the generalizability of our systems using character input. We obtained phone labels by using Flite [78]. All characters were converted into lowercase, but no elaborate text normalization was performed. We trimmed the beginning and end silences from the utterances, after which the duration of the corpus was reduced to 15.7 hours. We used 11,092 utterances for training, 500 for validation, and 500 for testing.

**Tacotron configurations.** We used the same network architectures as Japanese experiments, but we chose large parameter size configuration only for all Tacotron models as we learned from section 4.1. Note that accentual type embedding is unnecessary for English TTS, so we removed it. Thus, we allocated 512 dimensions to character or phone embedding, 512 and 256 dimensions to two layers in encoder pre-net.

**Listening test.** We conducted a listening test about naturalness of synthetic speech in a similar way to the Japanese test. We recruited 49 native English listeners via crowdsourcing. Listeners were asked to evaluate 36 samples using a five-grade scoring system in one set. One listener could evaluate at most 20 sets. We collected 17,928 evaluations in total.

**Systems.** The listening test contained 12 systems including natural samples, two analysis-by-synthesis systems, one DNN-based SPSS system from pipeline TTS, and eight Tacotron systems with combinations of conditions of self-attention, input feature, and encoder network structure.

### 4.5. Results of English systems

Table 4 shows the number of samples containing fatal alignment errors out of 500 samples in a test set.

| Input  | Encoder | Self-attention | Alignment error |
|--------|---------|----------------|-----------------|
| Character | CBHL | - | 5 (1.0 %) |
| Character | CBHL | ✓ | 5 (1.0 %) |
| Character | CNN | - | 1 (0.2 %) |
| Character | CNN | ✓ | 2 (0.4 %) |
| Phone     | CBHL | - | 3 (0.6 %) |
| Phone     | CBHL | ✓ | 1 (0.2 %) |
| Phone     | CNN  | - | 2 (0.2 %) |
| Phone     | CNN  | ✓ | 6 (1.2 %) |

However, we think that insights obtained from Q5 and Q6 are interesting. According to our result, the simplification of encoder architecture adopted for Tacotron 2 does not improve the performance. Instead, it worsens the performance when the inputs are no clear relationship between the fatal alignment error rate and model configuration like network structure and input linguistic feature. In addition, samples containing fatal alignment errors looked random, so there were no specific sentences that were difficult to align in the test set.

Figure 5 shows the results of the listening test in English. Please refer to Table 5 for the statistical significance of the test. First, we can see that when the encoder was CNN based, both systems with and without self-attention using phones had higher scores than corresponding systems using characters. The baseline Tacotron with the CNN based encoder had 3.08 ± 0.05 when characters were used and 3.32 ± 0.05 when phones were used. Self-attention Tacotron with the CNN based encoder had 3.01 ± 0.05 when characters were used and 3.19 ± 0.05 when phones were used.

Second, interestingly, we can see that when the CBHL encoder was used, the score gap caused by the difference in input features vanished. The baseline Tacotron with the CBHL encoder had 3.50 ± 0.04 when characters were used and 3.49 ± 0.04 when phones were used. Self-attention Tacotron with the CBHL encoder had 3.57 ± 0.04 both when characters and phones were used.

Third, self-attention helped slightly improve scores when the CBHL encoder was used. On the other hand, self-attention negatively affected systems using the CNN based encoder. Finally, the Tacotron systems did not match the pipeline system in terms of naturalness.

### 4.6. Discussion on results in English

This experiment showed: Q5) When the CNN-based encoder was used, models using character input were inferior to models using phone input. Q6) However, when the CBHL encoder was used, models using character input could perform as well as models using phone input. Q7) Self-attention slightly improved naturalness of models using the CBHL encoder but not naturalness of models using the CNN-based encoder. The improvements happened for both character and phone inputs, rather than reducing gap between them. Q8) Our Tacotron systems were outperformed by the pipeline system.

We think that insights obtained from Q5 and Q6 are interesting. According to our result, the simplification of encoder architecture adopted for Tacotron 2 does not improve the performance. Instead, it worsens the performance when the inputs are...
Table 5: Mann-Whitney rank test for English listening test. The symbol * means statistical significance by $p \leq 0.01$.

| System      | Input | Encoder          | Self-att. | System | ABS | Pipeline | Tacotron | Phone |
|-------------|-------|------------------|-----------|--------|-----|----------|----------|-------|
| Natural     | -     | -                | -         | -      | -   | -        | -        | -     |
| ABS         | -     | -                | -         | -      | -   | -        | -        | -     |
| Pipeline    | -     | -                | -         | -      | -   | -        | -        | -     |
| Tacotron    | -     | -                | -         | -      | -   | -        | -        | -     |

Figure 5: Box plot for results of English listening test. Red dots indicate mean, and black bars indicate median.

The reason English Tacotron systems were outperformed by the pipeline system unlike the Japanese Tacotron systems is unnatural prosody. Similar problems are reported from a few studies using Tacotron based systems. According to Shen et al. [6], unnatural prosody including unnatural pitch was the most common error in their manual analysis for 100 challenging English sentences. We investigated samples from the two best Tacotron systems: self-attention Tacotron using characters and self-attention Tacotron using phones. We listened to their sam-

other encoder structures such as the CBHL encoder.

When the CBHL encoder was used, systems using characters gave the same quality as systems using phones as input. This suggests that the particular network structure in the CBHL encoder can learn to disambiguate underlying phones given characters. A similar result was reported by [7] in that the CBHL encoder worked better than a CNN plus self-attention encoder for a transformer-like sequence-to-sequence system using character input in the same corpus. The CBHL encoder was originated from a word embedding method that relies on only character-level input [79]. They compared nearest neighbor words on the basis of cosine distance of word vectors obtained before and after highway layers and found that the highway layers could encode semantic features that were not discernible from orthography, whereas representations before highway layers encoded surface form features. A similar mechanism was expected to happen in highway layers in the CBHL encoder in our experiment, encoding pronunciations that were not discernible from orthography. Further research is required to generalize the conclusion by investigating other languages, corpora, and encoder structures.

The reason English Tacotron systems were outperformed by the pipeline system unlike the Japanese Tacotron systems is unnatural prosody. Similar problems are reported from a few studies using Tacotron based systems. According to Shen et al. [6], unnatural prosody including unnatural pitch was the most common error in their manual analysis for 100 challenging English sentences. We investigated samples from the two best Tacotron systems: self-attention Tacotron using characters and self-attention Tacotron using phones. We listened to their sam-

other encoder structures such as the CBHL encoder.

When the CBHL encoder was used, systems using characters gave the same quality as systems using phones as input. This suggests that the particular network structure in the CBHL encoder can learn to disambiguate underlying phones given characters. A similar result was reported by [7] in that the CBHL encoder worked better than a CNN plus self-attention encoder for a transformer-like sequence-to-sequence system using character input in the same corpus. The CBHL encoder was originated from a word embedding method that relies on only character-level input [79]. They compared nearest neighbor words on the basis of cosine distance of word vectors obtained before and after highway layers and found that the highway layers could encode semantic features that were not discernible from orthography, whereas representations before highway layers encoded surface form features. A similar mechanism was expected to happen in highway layers in the CBHL encoder in our experiment, encoding pronunciations that were not discernible from orthography. Further research is required to generalize the conclusion by investigating other languages, corpora, and encoder structures.

The reason English Tacotron systems were outperformed by the pipeline system unlike the Japanese Tacotron systems is unnatural prosody. Similar problems are reported from a few studies using Tacotron based systems. According to Shen et al. [6], unnatural prosody including unnatural pitch was the most common error in their manual analysis for 100 challenging English sentences. We investigated samples from the two best Tacotron systems: self-attention Tacotron using characters and self-attention Tacotron using phones. We listened to their sam-

characters. Pronunciation cannot always be accurately inferred from characters in English, whereas phones enable pronunciation ambiguities to be avoided. Therefore, it is reasonable that systems using phones as inputs scored higher than systems using characters, and this is confirmed by other sequence-to-sequence based TTS research [5,35]. A reasonable conclusion that phones work better than characters was confirmed in an experiment using large-scale corpora in three languages including English [35]. However, their system used the CNN based encoder from Tacotron2 [35], so their conclusion may not hold for

other encoder structures such as the CBHL encoder.

When the CBHL encoder was used, systems using characters gave the same quality as systems using phones as input. This suggests that the particular network structure in the CBHL encoder can learn to disambiguate underlying phones given characters. A similar result was reported by [7] in that the CBHL encoder worked better than a CNN plus self-attention encoder for a transformer-like sequence-to-sequence system using character input in the same corpus. The CBHL encoder was originated from a word embedding method that relies on only character-level input [79]. They compared nearest neighbor words on the basis of cosine distance of word vectors obtained before and after highway layers and found that the highway layers could encode semantic features that were not discernible from orthography, whereas representations before highway layers encoded surface form features. A similar mechanism was expected to happen in highway layers in the CBHL encoder in our experiment, encoding pronunciations that were not discernible from orthography. Further research is required to generalize the conclusion by investigating other languages, corpora, and encoder structures.

The reason English Tacotron systems were outperformed by the pipeline system unlike the Japanese Tacotron systems is unnatural prosody. Similar problems are reported from a few studies using Tacotron based systems. According to Shen et al. [6], unnatural prosody including unnatural pitch was the most common error in their manual analysis for 100 challenging English sentences. We investigated samples from the two best Tacotron systems: self-attention Tacotron using characters and self-attention Tacotron using phones. We listened to their sam-

other encoder structures such as the CBHL encoder.

When the CBHL encoder was used, systems using characters gave the same quality as systems using phones as input. This suggests that the particular network structure in the CBHL encoder can learn to disambiguate underlying phones given characters. A similar result was reported by [7] in that the CBHL encoder worked better than a CNN plus self-attention encoder for a transformer-like sequence-to-sequence system using character input in the same corpus. The CBHL encoder was originated from a word embedding method that relies on only character-level input [79]. They compared nearest neighbor words on the basis of cosine distance of word vectors obtained before and after highway layers and found that the highway layers could encode semantic features that were not discernible from orthography, whereas representations before highway layers encoded surface form features. A similar mechanism was expected to happen in highway layers in the CBHL encoder in our experiment, encoding pronunciations that were not discernible from orthography. Further research is required to generalize the conclusion by investigating other languages, corpora, and encoder structures.

The reason English Tacotron systems were outperformed by the pipeline system unlike the Japanese Tacotron systems is unnatural prosody. Similar problems are reported from a few studies using Tacotron based systems. According to Shen et al. [6], unnatural prosody including unnatural pitch was the most common error in their manual analysis for 100 challenging English sentences. We investigated samples from the two best Tacotron systems: self-attention Tacotron using characters and self-attention Tacotron using phones. We listened to their sam-

other encoder structures such as the CBHL encoder.

When the CBHL encoder was used, systems using characters gave the same quality as systems using phones as input. This suggests that the particular network structure in the CBHL encoder can learn to disambiguate underlying phones given characters. A similar result was reported by [7] in that the CBHL encoder worked better than a CNN plus self-attention encoder for a transformer-like sequence-to-sequence system using character input in the same corpus. The CBHL encoder was originated from a word embedding method that relies on only character-level input [79]. They compared nearest neighbor words on the basis of cosine distance of word vectors obtained before and after highway layers and found that the highway layers could encode semantic features that were not discernible from orthography, whereas representations before highway layers encoded surface form features. A similar mechanism was expected to happen in highway layers in the CBHL encoder in our experiment, encoding pronunciations that were not discernible from orthography. Further research is required to generalize the conclusion by investigating other languages, corpora, and encoder structures.

The reason English Tacotron systems were outperformed by the pipeline system unlike the Japanese Tacotron systems is unnatural prosody. Similar problems are reported from a few studies using Tacotron based systems. According to Shen et al. [6], unnatural prosody including unnatural pitch was the most common error in their manual analysis for 100 challenging English sentences. We investigated samples from the two best Tacotron systems: self-attention Tacotron using characters and self-attention Tacotron using phones. We listened to their sam-

characters. Pronunciation cannot always be accurately inferred from characters in English, whereas phones enable pronunciation ambiguities to be avoided. Therefore, it is reasonable that systems using phones as inputs scored higher than systems using characters, and this is confirmed by other sequence-to-sequence based TTS research [5,35]. A reasonable conclusion that phones work better than characters was confirmed in an experiment using large-scale corpora in three languages including English [35]. However, their system used the CNN based encoder from Tacotron2 [35], so their conclusion may not hold for

other encoder structures such as the CBHL encoder.

When the CBHL encoder was used, systems using characters gave the same quality as systems using phones as input. This suggests that the particular network structure in the CBHL encoder can learn to disambiguate underlying phones given characters. A similar result was reported by [7] in that the CBHL encoder worked better than a CNN plus self-attention encoder for a transformer-like sequence-to-sequence system using character input in the same corpus. The CBHL encoder was originated from a word embedding method that relies on only character-level input [79]. They compared nearest neighbor words on the basis of cosine distance of word vectors obtained before and after highway layers and found that the highway layers could encode semantic features that were not discernible from orthography, whereas representations before highway layers encoded surface form features. A similar mechanism was expected to happen in highway layers in the CBHL encoder in our experiment, encoding pronunciations that were not discernible from orthography. Further research is required to generalize the conclusion by investigating other languages, corpora, and encoder structures.

The reason English Tacotron systems were outperformed by the pipeline system unlike the Japanese Tacotron systems is unnatural prosody. Similar problems are reported from a few studies using Tacotron based systems. According to Shen et al. [6], unnatural prosody including unnatural pitch was the most common error in their manual analysis for 100 challenging English sentences. We investigated samples from the two best Tacotron systems: self-attention Tacotron using characters and self-attention Tacotron using phones. We listened to their sam-
samples that had relatively low average MOS whereas corresponding samples from the pipeline had high MOS of over 4.5. We found that the samples from the Tacotron systems had relatively flat pitch change, which resulted in unnatural prosody. Indeed, the average standard deviations of $F_0$ in the voiced region of generated samples in a test set were 48 Hz for self-attention Tacotron using characters, 46 Hz for self-attention Tacotron using phones, and 59 Hz for the pipeline system. Figure 6 shows standard deviations of $F_0$ against the output waveform length for samples from our Tacotron systems and the pipeline system. We can clearly see that samples predicted from our Tacotron system have smaller standard deviations than samples predicted from the pipeline across the whole output length. In addition, we found that the low-score samples contain errors such as mis-predictions or wrongly stressed syllables. This would be one reason the English Tacotron systems have lower scores than the pipeline system unlike the Japanese Tacotron systems.

5. Conclusion

In this paper, we investigated under what conditions sequence-to-sequence based text-to-speech (TTS) could work well given simple input such as text or phonemes in Japanese and English, along with comparing them with comparable deep neural network (DNN) based pipeline TTS systems using complex full-context labels. We empowered models of our sequence-to-sequence based TTS methods instead of enriching linguistic features to see how much enforced models could overcome the linguistic feature limitation.

We upgraded configurations for our Tacotron based methods from our previous research. We increased the parameter size of models for our Tacotron methods. In addition, we tested a convolutional neural network (CNN) based encoder from Tacotron2, along with the CBHL (convolution banks, highway networks and long short-term memory) encoder from the original Tacotron for our TTS systems. Unlike other studies, our baseline pipeline systems used autoregressive probabilistic modeling and a neural vocoder as the sequence-to-sequence based methods do, so the differences in the two methods were mainly about framework approaches.

Our experiment showed that increasing parameter size enabled the sequence-to-sequence based methods using phonemes and accentual-type labels as inputs to outperform the comparable pipeline systems in Japanese. This suggested that a powerful sequence-to-sequence TTS model could learn to compensate for a lack of rich linguistic features. We further investigated the upgraded TTS systems using characters as input in English. We found that the CBHL encoder could learn to dis-ambiguate pronunciation ambiguities given characters as well as phone input better than the CNN encoder. However, we also observe that the CBHL encoder could not learn English stressed syllables from characters perfectly and hence resulted in flatter fundamental frequency.

Our future work includes improvements of network architectures and exploring a new way for learning supra-segmental features more appropriately.

6. Acknowledgments

This work was partially supported by a JST CREST Grant (JPMJCR18A6, VoicePersonae project), Japan, MEXT KANKENHI Grants (16H06302, 17H04687, 18H04120, 18H04112, and 18KT0051, 19K24371, 19K24373), Japan, and a Google AI Focused Research Awards Program, Japan. The numerical calculations were carried out on the TSUBAME 3.0 supercomputer at the Tokyo Institute of Technology.

The authors thank Dr. Erica Cooper from NII for her useful feedback on our synthetic speech.

References

[1] H. Zen, A. Senior, M. Schuster, Statistical parametric speech synthesis using deep neural networks, in: Proc. ICASSP, 2013, pp. 7962–7966.
[2] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgiannakis, R. Clark, R. A. Saurous, Tacotron: Towards end-to-end speech synthesis, in: Proc. Interspeech, 2017, pp. 4006–4010.
[3] J. Sotelo, S. Mehri, K. Kumar, J. F. Santos, K. Kastner, A. Courville, Y. Bengio, Char2wav: End-to-end speech synthesis, in: Proc. ICLR (Workshop Track), 2017.
[4] Y. Taigman, L. Wolf, A. Polyak, E. Nachmani, VoiceLoop: Voice fitting and synthesis via a phonological loop, in: Proc. ICLR, 2018.
[5] W. Ping, K. Peng, A. Gribiansky, S. O. Arik, A. Kannan, S. Narang, J. Raiman, J. Miller, Deep Voice 3: Scaling text-to-speech with convolutional sequence learning, in: Proc. ICLR, 2018.
[6] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Ryan, R. A. Saurous, Y. Agiomyrgiannakis, Y. Wu, Natural TTS synthesis by conditioning WaveNet on Mel spectrogram predictions, in: Proc. ICASSP, 2018, pp. 4779–4783. URL: https://doi.org/10.1109/ICASSP.2018.8461368.
[7] N. Li, S. Liu, Y. Liu, S. Zhao, M. Liu, Neural speech synthesis with transformer network, in: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019, AAAI Press, 2019, pp. 6706–6713. URL: https://doi.org/10.1609/aaai.v33i01.33016706.
Appendix A. Pipeline TTS framework

Appendix A.1. Front-end in pipeline based TTS

All TTS methods in the pipeline framework have a front-end that uses multiple knowledge-based or statistical models to extract linguistic features from the input text. These linguistic features are usually represented as full-context labels, which encode the segmental (e.g., phoneme identity) and suprasegmental information (e.g., phrase-level intonation) to utter the text. For example, a typical front-end for English may use a pronunciation dictionary and a decision-tree-based model to conduct the letter-to-sound (or grapheme-to-phoneme) conversion \([63]\). It may use other hand-crafted rules or statistical models to infer the phase breaks \([82]\) and pitch accent \([83]\). These modules in the front-end are usually language dependent, and the statistical ones require supervised training and hand-annotated datasets.

Note that there are also methods to build a front-end with less language-specific knowledge or supervised training. For example, the vector space model may be used to derive linguistic features from input text in an unsupervised way \([28,29]\). However, such a method is not applicable to ideographic languages such as Japanese due to the sparse distribution of written characters. On the other hand, there are also methods to jointly optimize statistical models in the front-end with the acoustic model in the back-end \([84]\). However, this method was only proposed for HMM-based SPSS and cannot be applied to DNN-based SPSS directly.

Appendix A.2. Back-end in pipeline based TTS

The back-end in pipeline based TTS has two models: acoustic and waveform. The acoustic model has been advanced by replacing the hidden Markov models (HMMs) with DNNs \([10]\). On the basis of the vanilla feed-forward and recurrent NNs, researchers have further proposed many NN-based acoustic models with improved probabilistic modeling capability.

Particularly, the vanilla feed-forward and recurrent NNs assume statistical independence among the acoustic feature frames \([30]\). To amend the independence assumption, autoregressive NN models were proposed to model the acoustic feature distribution of one frame conditioned on the previous frames, which can be implemented by feeding back the previous frames linearly \([67]\) or non-linearly \([30,22]\). A similar idea can be used to capture the causal dependence between different acoustic feature dimensions \([85]\). Meanwhile, unlike the autoregressive NNs, there also other NN-based models using intractable or implicit density functions. Examples of the former case include the post-filtering approach using a restricted Boltzmann machine (RBM) \([31]\) and the trajectory DNN \([86]\). The latter case is the generative adversarial network (GAN) \([32]\). All the recently proposed NN-based acoustic models alleviate the over-smoothing effect in the vanilla NNs.

There are two types of waveform model in back-end: algorithmic method or neural method. Conventional vocoder is an algorithmic method that has long been a core component of SPSS. However, the artifacts in generated waveforms from the vocoder limit the quality of SPSS, which motivates the development of waveform generation methods such as GriffinLim \([65]\) in vocoder-free TTS and the neural waveform models \([13]\). Compared with the Griffin-Lim and related algorithms, the neural waveform models show better performance \([10]\). They are also more flexible as they can work on various types of input.
acoustic features including mel-spectrogram and vocoder parameters, be implemented with different types of NNs such as GAN [87] and normalizing flow [88], and incorporate classical signal processing algorithms like linear prediction coefficients (LPC) [89] and harmonic-plus-noise model [90].

Appendix A.3. Pipeline vs sequence-to-sequence TTS

The DNN-based pipeline and sequence-to-sequence based TTS have a few aspects in common. Both methods can benefit greatly from autoregressive modeling of the acoustic features sequences [10, 9] and neural waveform generators [10, 5, 6]. Although it is agreed that sequence-to-sequence based TTS can generally generate more natural speech than traditional methods such as pipeline SPSS, few studies have fully investigated this under the same conditions of density modeling and a waveform generation method. For example, Tacotron [2] and Tacotron2 [6] were evaluated against a HMM-based unit selection system [91] and a DNN-based pipeline system without either a neural waveform generator or an autoregressive acoustic model [92]. Some recent studies included a DNN-based pipeline using neural waveform generators, but the non-autoregressive acoustic models still affect the fairness of the comparison [92, 5].

Therefore, if we want to compare pipeline based TTS and sequence-to-sequence based TTS from a framework point of view, it is desirable to select comparable pipeline systems that use an autoregressive decoder with the same waveform generation method as sequence-to-sequence based TTS. Otherwise, differences in the experimental results for the two different approaches can be expected to mainly be caused by the difference in waveform generation techniques and the assumption of probabilistic modeling.

The largest scale experiment to compare various TTS systems including sequence-to-sequence based methods and pipeline methods is perhaps Blizzard challenge 2019 [94], in which the task is single-speaker Chinese TTS. Among 24 participating systems, at least 14 were sequence-to-sequence based TTS methods, 5 were SPSS methods, and 2 were unit selection. In addition, one system was natural speech and one was Merlin benchmark [95]. The challenge did not intend fair comparison, because the participants could use various methods of waveform generation and even manual labeling or data argumentation with other corpora. Nevertheless, sequence-to-sequence based TTS systems using a neural vocoder obtained relatively high ranks, and the top score was achieved by a DNN-based SPSS system using a neural vocoder, BERT (Bidirectional Encoder Representations from Transformers) based front-end, autoregressive duration model, and non-autoregressive acoustic model with a GAN post-filter [96].