INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

First Step Towards End-to-end Parametric TTS Synthesis: Generating Spectral Parameters with Neural Attention

Wenfu Wang, Shuang Xu, Bo Xu

Interactive Digital Media Technology Research Center
Institute of Automation, Chinese Academy of Sciences, Beijing, P.R.China
{wangwenfu2013, shuang.xu, xubo}@ia.ac.cn

Abstract

In conventional neural networks (NN) based parametric text-to-speech (TTS) synthesis frameworks, text analysis and acoustic modeling are typically processed separately, leading to some limitations. On one hand, much significant human expertise is normally required in text analysis, which presents a laborious task for researchers; on the other hand, training of the NN-based acoustic models still relies on the hidden Markov model (HMM) to obtain frame-level alignments. This acquisition process normally goes through multiple complicated stages. The complex pipeline makes constructing a NN-based parametric TTS system a challenging task. This paper attempts to bypass these limitations using a novel end-to-end parametric TTS synthesis framework, i.e. the text analysis and acoustic modeling are integrated together employing an attention-based recurrent neural network. Thus the alignments can be learned automatically. Preliminary experimental results show that the proposed system can generate moderately smooth spectral parameters and synthesize fairly intelligible speech on short utterances (less than 8 Chinese characters).

Index Terms: parametric TTS synthesis, end-to-end, attention-based recurrent neural network, acoustic modeling

1. Introduction

A conventional Text-to-speech (TTS) synthesis [1] system is typically partitioned into two parts: text analysis and speech synthesis, which are processed separately. The text analysis part converts a text string to a sequence of linguistic features comprising whatever contextual factors that might affect the acoustic realization of the speech sounds. Figuring out which factors matter normally requires much significant expertise, resulting in a laborious task for researchers. The speech synthesis part, of which statistical parametric speech synthesis (SPSS) [2–4] is a typical representative, uses the converted linguistic features to generate waveform. Recently, neural networks (NN) based acoustic modeling in SPSS have achieved state-of-the-art performance. Despite the advances, in these approaches, a prerequisite is that a set of alignments between linguistic and acoustic features are required. A Hidden Markov model (HMM) is normally used to achieve this. But the training procedure typically includes multiple awkward iterative stages such as context-independent and context-dependent training, in which expert knowledge is also indispensable. All these limitations could result in a hard work to build a state-of-the-art TTS system.

In this paper, we propose a novel end-to-end TTS synthesis framework that employs an attention-based recurrent neural network architecture to integrate text analysis and acoustic modeling together. This end-to-end framework simplifies the existing pipeline to build a NN-based TTS system and models complex sequential mapping directly from a text sequence (e.g. phonemes, syllables or words) to its acoustic trajectory. The term "end-to-end" in this work means that text analysis and acoustic modeling are accomplished together by an attention-based recurrent network, which has the capacity to learn the relevant contextual factors embedded in the text sequence. Thus we successfully bypass the trouble that a conventional TTS system always encounters in text analysis. Another advantage is that the model can learn alignments between the discrete text sequence and real-valued acoustic feature sequence automatically during training. As a result, the frame-by-frame alignments required by almost all NN-based acoustic models are no more a necessity. Preliminary experimental results show that the proposed model can generate moderately smooth spectral parameters and synthesize fairly intelligible speech on short utterances (less than 8 Chinese characters).

1.1. Relation to prior work

The emerging neural networks (NN) have prevailed in SPSS and have become a dominant acoustic model, thanks to their more powerful capacities than HMMs. In [5–9], feedforward deep neural networks (DNN) are used to directly model the complex, nonlinear mapping from linguistic inputs to acoustic outputs. However, the sequential nature of speech is not explicitly modeled in the DNN architecture. To include internal temporal dependencies across sequences, some novel training criteria, such as minimum sequence error [10] and trajectory error training [11], have been proposed to minimize sentence-level error rather than frame-by-frame error on the DNN framework. Recently, recurrent neural networks (RNNs) with long short term memory (LSTM) [12, 13] units or gated recurrent units (GRU) [14], which can capture long-span context information, have been employed to acoustic modeling in SPSS [15–17]. A further extension is to improve the accuracy of acoustic feature distribution using mixture density networks (MDNs) [15, 18]. However, all the NN-based SPSS approaches listed above can be classified into the conventional TTS framework, which has some limitations. First, an explicit text analysis stage that requires much effort and expertise is a prerequisite. Second, training of the neural networks demands a set of frame-by-frame alignments, which are usually obtained through an awkward iterative procedure with an HMM. An end-to-end attention-based recurrent neural architecture can circumvent these limitations.

Recently, attention-based recurrent networks [19], which will be detailed in next section, have been successfully applied to various sequence-to-sequence (*seq2seq*) tasks, such as ma-

The work is supported by 973 Program in China, Grant No. 2013CB329302.

http://dx.doi.org/10.21437/Interspeech.2016-134

chine translation [19, 20], image caption generation [21], handwriting synthesis [22] and speech recognition [23–25]. Such models iteratively process their input by selecting relevant content at every generation step. This basic idea makes them applicable to TTS synthesis, which can be viewed as learning to generate a sequence (acoustic trajectory) given another sequence (text). However, compared to machine translation, TTS synthesis principally differs in generating much longer real-valued sequence (thousands of frames instead of tens of words), in which the frames are less distinctive than words. It is also different from handwriting synthesis, since the outputs contain multiple streams and the dimension is much higher. For these reasons, it is a challenge to apply attention-based architecture to TTS synthesis. The attempts made in this paper are an important step towards building fully end-to-end TTS synthesis system, which is an exciting area of research.

2. Attention-based recurrent network for parametric TTS synthesis

2.1. General framework

The architecture we introduce here is an attention-based recurrent sequence transducer (ARST) that iteratively generates an output sequence $\boldsymbol{y} = (\boldsymbol{y}_1, \cdots, \boldsymbol{y}_T)$ through an attentive selection from the relevant content of the input sequence $\boldsymbol{x} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_L)$, where $T \gg L$. In practice, $\boldsymbol{x}$ is usually encoded into an embedded representation $\boldsymbol{h} = (\boldsymbol{h}_1, \ldots, \boldsymbol{h}_L)$ that is more suitable for the attention mechanism to work with. Here, the input $\boldsymbol{x}$ is a sequence of phonemes representing the text string and the output $\boldsymbol{y}$ is a sequence of real-valued acoustic frames. At each time-step t, an ARST generates an output $\boldsymbol{y}_t$ by focusing on the relevant elements of $\boldsymbol{h}$, specifically,

$$\boldsymbol{s}_t = RNN(\boldsymbol{s}_{t-1}, \boldsymbol{c}_{t-1}, \boldsymbol{y}_{t-1}) \tag{1}$$

$$\boldsymbol{c}_t = AttendContext(\boldsymbol{s}_t, \boldsymbol{h}) \tag{2}$$

$$\boldsymbol{y}_t = Generate(\boldsymbol{s}_t, \boldsymbol{c}_t) \tag{3}$$

where $\boldsymbol{s}_t$ is the first-layer output state of the transducer which is realized as a two-layer LSTM architecture here. At each time-step t, the attention mechanism $AttendContext$ makes an attentive selection and produces a context vector $\boldsymbol{c}_t$ through understanding the context of the current focused phoneme, specifically,

$$e_{t,i} = \boldsymbol{v}^{\mathrm{T}} \tanh(\boldsymbol{W}\boldsymbol{s}_t + \boldsymbol{V}\boldsymbol{h}_i + \boldsymbol{b}) \tag{4}$$

$$w_{t,i} = \exp(e_{t,i}) \Big/ \sum_{j=1}^{L} \exp(e_{t,j}) \tag{5}$$

$$\boldsymbol{c}_t = \sum_{i=1}^{L} w_{t,i} \boldsymbol{h}_i \tag{6}$$

where $\boldsymbol{v}$, $\boldsymbol{b}$ are parameter vectors, $\boldsymbol{W}$, $\boldsymbol{V}$ are parameter matrices. $w_{t,i}$ is the normalized attention weight for $\boldsymbol{h}_i$, which is the score[1] measuring how closely the output state $\boldsymbol{s}_t$ is matched to the content $\boldsymbol{h}_i$. So the attention is content-based. On convergence, the distribution of $\boldsymbol{w}_t \in \mathbb{R}^L$, often called the alignment, is typically very sharp, with its focus sliding gradually from left to right across $\boldsymbol{h}$. Intuitively, the score can be interpreted as the attention's belief that it is generating $\boldsymbol{h}_i$'s acoustic trajectory. $\boldsymbol{w}_t$ is then used to produce $\boldsymbol{c}_t$ by linearly blend each $\boldsymbol{h}_i$ of $\boldsymbol{h}$.

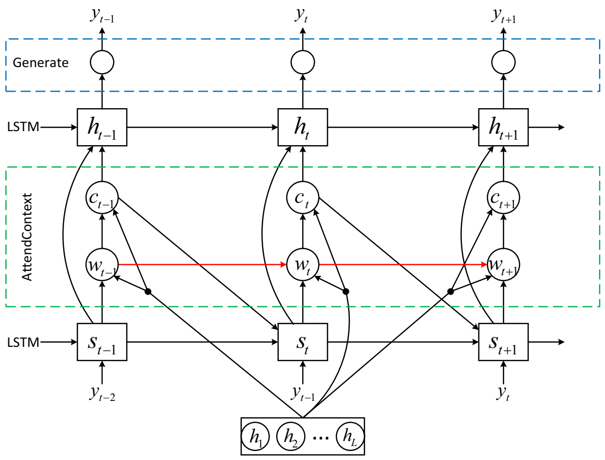

Figure 1: *Schematic diagram of the Attention-based Recurrent Sequence Transducer for TTS. Note that the encoder is not illustrated. The* $AttendContext$ *layer is nested in the transducer to offer attentive selection each time the ARST runs. The red arrows form a feedback from the previous alignment to 1) provide location-related information; 2) serve as a modulator when computing attention weights.*

The $Generate$ function read $\boldsymbol{s}_t$, $\boldsymbol{c}_t$ indirectly as input and generate an output frame aligned automatically with the focused phoneme. The process is graphically illustrated in Figure 1.

Note that the output frame $\boldsymbol{y}_t$ is actually a function of the input sequence $\boldsymbol{x}$ and all the frames generated so far. In a *seq2seq* task, given a training pair $(\boldsymbol{x}, \boldsymbol{y})$, we aim at maximizing the conditional probability $p(\boldsymbol{y}|\boldsymbol{x})$, which factorizes as

$$p(\boldsymbol{y}|\boldsymbol{x}) = p(\boldsymbol{y}_1 \cdots \boldsymbol{y}_T|\boldsymbol{x}) = \prod_{t=1}^{T} p(\boldsymbol{y}_t|\boldsymbol{y}_{<t}, \boldsymbol{x}) \tag{7}$$

where $p(\boldsymbol{y}_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$ can be efficiently modeled by an ARST.

2.2. Additional techniques with TTS synthesis

The content-based attention mechanism described above has demonstrated success on machine translation, where the input and output sequences contain just tens of words and the semantic distances between words in high-dimension space (embedded representation) are relatively large compared to that between frames, making the attention mechanism more discriminating. However, a direct application of this attention to TTS synthesis is problematic. Some remedies are taken into account.

2.2.1. Feedback of alignment

In TTS, the given text and the speech to be synthesized are of very different length. Typically, each phoneme spans around 25 frames. That means in a range of successive output timesteps, the attention should focus on only one phoneme with its implicit context. Thus the desired movement of alignment peak should be first up then down in a slow way. A location modulator[2] can be beneficial. On the other hand, TTS is a strict left-to-right *seq2seq* task. The attention focus should be ensured to move forward all the way across the text sequence. We attempt to address the two issues jointly through incorporating a location-based attention mechanism into the content-based one, as done

[1] This is the *concat*-attention using the terminology from [20]. We also tried the *dot*-attention, but preliminary results are not desirable.

[2] A similar modulator using LSTM was proposed to compute attention scores in [25].

in [23]. The “location” simply means the alignment produced at the previous step is taken into consideration when computing attention weights, showed as the red arrows in Figure 1. Thus Eq. (4) is extended as

$$\boldsymbol{F} = \boldsymbol{Q} * \boldsymbol{w}_{t-1} \tag{8}$$

$$e_{t,i} = \boldsymbol{v}^{\mathrm{T}} \tanh(\boldsymbol{W} s_t + \boldsymbol{V} h_i + \boldsymbol{U} f_i + \boldsymbol{b}) \tag{9}$$

where * denotes convolution and $\boldsymbol{Q}$, $\boldsymbol{U}$ are additional parameter matrices. The matrix $\boldsymbol{F}$ comprises all convolutional feature vectors $\boldsymbol{f}_i$ which are extracted for every position i of the previous alignment $\boldsymbol{w}_{t-1}$.

2.2.2. *Windowing*

Theoretically in TTS, the ARST can extract sentence-level linguistic context that contains potential contextual factors affecting prosody if the entire input text sequence is considered when making attentive selection (Eq. 9 and 6). But it was found ineffective in a preliminary experiment. We conjecture this might be overkill since the context vector $\boldsymbol{c}_t$ is likely to include noisy information from irrelevant phonemes, especially when the text is long. So we make a compromise by using a local context of the text though it is a little harmful to the prosody. A simple practice making the attention local is a windowing technique, which has proved its effectiveness in speech recognition [26]. At each timestep t, the attention mechanism considers only a subsequence $\hat{\boldsymbol{h}} = (\boldsymbol{h}_{p_t-d}, \cdots, \boldsymbol{h}_{p_t+d})$ of the entire sequence $\boldsymbol{h}$, where p_t is the window center and the window width is $2d+1$. The advantages of applying windowing technique are twofold. First, it helps to align between text and acoustic trajectory. Second, it can reduce the computational complexity [26].

2.2.3. *Input quantification & Sampling*

The ARST conditions the next step generation on the previous outputs. During training, the ARST maximizes the probability in Eq. (7), where $\boldsymbol{y}_{<t}$ is the groundtruth of the previous outputs. However during testing, the groundtruth is missing and the generation process can suffer because the model was not trained to be resilient to feeding in bad generations. Once mistakes occur, they can be quickly amplified in the subsequent generation. This is the *issue of error accumulation*, which is very serious in TTS synthesis. Unlike in speech recognition characters are predicted, we predict real-valued frames here. That means at every output step the errors almost always exist during training. To reduce these errors, we propose the *input quantification* technique analogous to analog-to-digital (AD) conversion. During both training and testing, the conditioned inputs ($\boldsymbol{y}_{t-1}$ at timestep t) are first quantified into a lattice-like space then fed in the ARST. To alleviate the mismatch between training and testing in *seq2seq* tasks, we employ the sampling technique proposed in [27] that samples from previous generated distribution at a scheduled rate during training, making the model more robust to correct its own mistakes at testing as it has learned to do so during training.

3. Experiments

3.1. Experimental setups

A Mandarin speech database recorded by a female professional speaker, both phonetically and prosodically rich, was used in our experiments. The database consisted of 7266 training utterances (around 7 hours, divided into three subsets: training, development and testing, with 6540, 686 and 40 utterances respectively) and was consistent in terms of recording quality and speaking style. The speech data was downsampled from 44.1 kHz to 16 kHz, then 41-dimensional line spectral pairs (LSPs), 25 band aperiodicities (BAPs) and logarithmic fundamental frequency (log F_0) were extracted every 5-ms using STRAIGHT [28]. In our experiments, static LSPs[3] were modeled as our acoustic features because the ARST can smooth the acoustic trajectory. We selected the untoned phonemes[4] as elements of the text string to make sure each phoneme corresponded to sufficient data. If the speech database is large, syllables or Chinese characters may be an alternative. Before training, the acoustic features were normalized to the range of [0.01, 0.99].

The ARST had 2 layers of LSTM; each layer contained 800 memory blocks with 512 recurrent projection units [13]. The dimension of context vector $\boldsymbol{c}_t$ is equal to that of embedded representation $\boldsymbol{h}_i$, which is the output of a unidirectional or bidirectional encoder. But in this paper, we omitted the encoder because there were no significant improvements found. Thus the $\boldsymbol{h}_i$ was just one-hot representation and the training speed was accelerated consequently. The convolution in Eq. (8) had 10 filters with each size of 5. The convolutional step was 1.

3.2. Training procedure

We first sorted the training utterances by length in an ascending order, as done in [29]. To start training, a zero vector was fed in to predict the first frame. Then the ARST was trained with a mini-batch stochastic gradient descent (SGD)-based algorithm with an initial learning rate of 0.008, and momentum of 0.9. For software implementation, the Kaldi toolkit [30] was used and training was conducted on a Tesla K40 GPU.

From our experience, it was hard to train an ARST completely from scratch on long sequences (e.g. thousands of frames). We found that providing an initial rough alignment at the early training stage was a very helpful way to quickly bring the model parameters in a good range. Rough alignments of speech units (phonemes or syllables) to segments of speech can be obtained easily with orthogonal methods such as manual labeling or performing forced alignments with an existing model. In our experiments, a trained HMM was employed to align between phonemes and their corresponding frames. The initial alignments, along with windowing, helped the training procedure. Specifically, at each output timestep t, we forced the attention mechanism to care about just the phonemes within a window centering on the roughly aligned phoneme. The window width was set to 5, with 2 phonemes on each side of the center. The window ensured the currently generated frame $\boldsymbol{y}_t$ was uttered by the phonemes within the window. On the other hand, the window was somewhat similar to the quin-phone in HMM-based SPSS. On convergence, the sampling method mentioned in Section 2 with a constant rate was then applied to refine the model.

The phoneme aligned automatically at t-th timestep was the one with the maximum attention weight within the window. The aligned phoneme was then treated as the new window center. During testing, we moved the window center one step forward whenever the weight of the right-hand phoneme adjacent to the center became maximum. We must also decide when the ARST has finished generating acoustic features. A heuristics is that as

[3] Initially the BAPs and log F_0 were also incorporated, but the alignment was hard to learn possibly due to the interactions of multiple streams, so they were excluded.

[4] If the acoustic features contain fundamental frequency, toned phonemes might be a better choice.

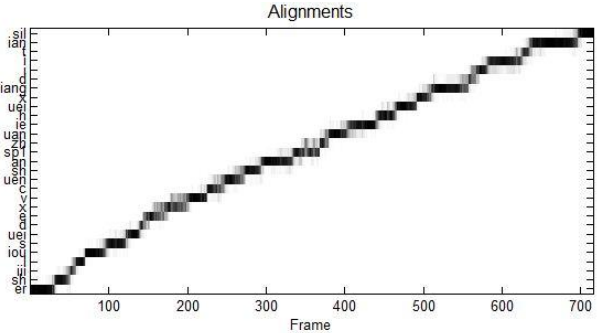

Figure 2: *Alignments between the phonemes (y-axis) and the corresponding acoustic frames (x-axis) produced by the ARST. The stair-like trajectory formed by the dark dots represents the alignments, which are monotonic in general. Notice that the up-and-down light dots near the dark ones to some extent make up the context affecting the prosodic trajectory. The blurs at the boundaries between phonemes represent transitions.*

soon as the last phoneme (silence) of the text became the attentive focus (i.e. window center), the generation process stopped.

3.3. Experimental results

The attention mechanism can learn an explicit alignment between phoneme text and the corresponding acoustic trajectory. An example of alignments is visualized in Figure 2.

We evaluated the *issue of error accumulation* existing in the ARST objectively. Three systems with different training strategies were trained respectively:

- **w/o IQ & w/o Samp**: The system trained without input quantification and sampling techniques.
- **w/ IQ & w/o Samp**: The system trained with input quantification but without sampling techniques.
- **w/ IQ & w/ Samp**: The system trained with input quantification and sampling techniques.

In order to reduce the influence of the data itself, the whole training set was involved in the assessment. Each of the utterances was generated respectively by the three systems. Then each generated acoustic sequence was split into two parts of equal length: the former part and the latter part, of which a comparison was made using the criterion of mean Euclidean distance (MED). The sequence of which the former part's MED was greater than the latter's was denoted as "F-gt-L" sequence, otherwise as "L-gt-F" one. For each system, we calculated the respective percentages of the "F-gt-L" and "L-gt-F" sequences, which are presented in Table 1. It is statistically significant that there were more "L-gt-F" sequences regardless of the systems. As we expected, the generation errors were gradually accumulated as the generation proceeded. It also can be seen that the percentages in "L-gt-F" column move downwards, indicating that the *issue of error accumulation* can be alleviated by the progressive training with the input quantification and sampling techniques, whereas it still remains. So it is not surprising that the best system (w/ IQ & w/ Samp) struggled to synthesize intelligible speech on long utterances in our experiments.

For comparison, our previous DNN-based system built in [15] was used as a baseline. The DNN had 5 hidden layers with each layer of 1024 hidden units. Figure 3 plots the trajectories of 3rd LSP coefficients of natural speech and generated by the ARST and DNN-based system. It can be seen from the figure that the ARST could generate moderately smooth speech parameter trajectory for a given text. But compared to DNN, the predictive accuracy is still inferior in general. To subjectively evaluate the performance of the best system, mean opinion score (MOS) test was also conducted. 20 short utterances with length less than 8 Chinese characters from the testing set, were selected for evaluation[5]. To synthesize speech, speech parameters of BAPs and log F_0 generated by the DNN baseline, along with the total durations of the original speech were used. 10 native listeners participated in the evaluation using headphones. After listening to each synthesized speech, the subjects were asked to rate the intelligibility of the speech in a 5-scale score (1: Bad, 2: Poor, 3: Fair, 4: Good, 5: Excellent). On average, our best system achieved an intelligibility score of 3.125 on short utterances, showing that our first attempts on the proposed system can synthesize fairly intelligible short speech. Nevertheless, the AB preference test results, which are not listed due to limited space, indicate that the best system is still not comparable with the DNN-based baseline system.

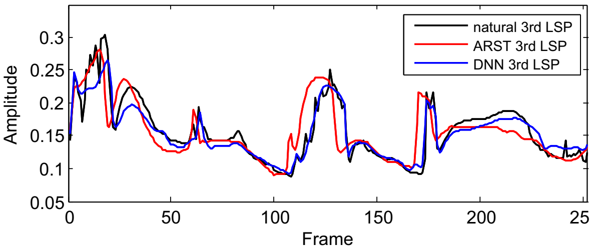

Figure 3: *Trajectories of 3rd LSP coefficients of natural speech and generated by the ARST and DNN-based system.*

Table 1: *The respective percentages of "F-gt-L" and "L-gt-F" sequences for three different systems. A t-test with confidence level of 0.95 was conducted.*

System	F-gt-L	L-gt-F	p-value
w/o IQ & w/o Samp	10.6%	89.4%	$< 10^{-6}$
w/ IQ & w/o Samp	20.2%	79.8%	$< 10^{-6}$
w/ IQ & w/ Samp	28.4%	71.6%	$< 10^{-6}$

4. Conclusions

This paper proposes a novel end-to-end parametric TTS synthesis framework that maps sequences directly from a text string to acoustic trajectory employing the attention-based recurrent sequence transducer (ARST). The "end-to-end" means the text analysis and acoustic modeling are integrated together into a unified model. Thus the trouble a conventional TTS synthesis encounters is avoided. Preliminary experimental results show that the ARST can generate moderately smooth spectral parameters and synthesize fairly intelligible speech on short utterances, whereas it still struggles on long speech possibly due to the *issue of error accumulation*. Although the proposed end-to-end system performs still not competitive with the DNN baseline, our first attempts made in this paper indicate an exciting research area of investigating fully end-to-end TTS synthesis.

More research will be conducted in the future, including explorations of incorporating fundamental frequency into acoustic features, investigating attention mechanism more suitable for TTS synthesis and experimenting on a larger dataset.

[5] Long utterances were not rated because most of the synthesized long speech in the back section sounds unintelligible.

5. References

[1] S. King, "An introduction to statistical parametric speech synthesis," *Sadhana*, vol. 36, no. 5, pp. 837–852, 2011.

[2] H. Zen, K. Tokuda, and A. W. Black, "Statistical parametric speech synthesis," *Speech Communication*, vol. 51, no. 11, pp. 1039–1064, 2009.

[3] H. Zen, "Acoustic modeling in statistical parametric speech synthesis–from HMM to LSTM-RNN," 2015.

[4] O. Watts, G. E. Henter, T. Merritt, Z. Wu, and S. King, "From H-MMs to DNNs: Where do the improvements come from?" 2016.

[5] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Proc. ICASSP*. IEEE, 2013, pp. 7962–7966.

[6] Y. Qian, Y. Fan, W. Hu, and F. K. Soong, "On the training aspects of deep neural network (dnn) for parametric tts synthesis," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 3829–3833.

[7] Z. Wu, C. Valentini-Botinhao, O. Watts, and S. King, "Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis," in *Proc. ICASSP*. IEEE, 2015, pp. 4460–4464.

[8] O. Watts, Z. Wu, and S. King, "Sentence-level control vectors for deep neural network speech synthesis," in *Proc. Interspeech*, 2015.

[9] C. Valentini-Botinhao, Z. Wu, and S. King, "Towards minimum perceptual error training for DNN-based speech synthesis," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[10] Y. Fan, Y. Qian, F. K. Soong, and L. He, "Sequence generation error (SGE) minimization based deep neural networks training for text-to-speech synthesis," in *Proc. Interspeech*, 2015.

[11] Z. Wu and S. King, "Minimum trajectory error training for deep neural networks, combined with stacked bottleneck features," in *Proc. Interspeech*, 2015.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[13] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," *arXiv preprint arXiv:1402.1128*, 2014.

[14] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[15] W. Wang, S. Xu, and B. Xu, "Gating recurrent mixture density networks for acoustic modeling in statistical parametric speech synthesis," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5520–5524.

[16] H. Zen and H. Sak, "Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis," in *Proc. ICASSP*. IEEE, 2015, pp. 4470–4474.

[17] Z. Wu and S. King, "Investigating gated recurrent networks for speech synthesis," in *Proc. ICASSP*. IEEE, 2016.

[18] H. Zen and A. Senior, "Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis," in *Proc. ICASSP*. IEEE, 2014, pp. 3844–3848.

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

[20] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[21] K. Xu, J. Ba, R. Kiros, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," *arXiv preprint arXiv:1502.03044*, 2015.

[22] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[23] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in Neural Information Processing Systems*, 2015, pp. 577–585.

[24] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, attend and spell," *arXiv preprint arXiv:1508.01211*, 2015.

[25] N. Jaitly, Q. V. Le, O. Vinyals, I. Sutskeyver, and S. Bengio, "An online sequence-to-sequence model using partial conditioning," *arXiv preprint arXiv:1511.04868*, 2015.

[26] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," *arXiv preprint arXiv:1508.04395*, 2015.

[27] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 1171–1179.

[28] H. Kawahara, I. Masuda-Katsuse, and A. De Cheveigne, "Restructuring speech representations using a pitch-adaptive time–frequency smoothing and an instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sounds," *Speech communication*, vol. 27, no. 3, pp. 187–207, 1999.

[29] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos *et al.*, "Deep speech 2: End-to-end speech recognition in English and Mandarin," *arXiv preprint arXiv:1512.02595*, 2015.

[30] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," 2011.