The TL-NTU Text-to-speech System for the Blizzard Challenge 2018

Junchao Wang[1], *Xiaohai Tian*[2], *Minghui Dong*[3], *Haihua Xu*[4]

[1]College of Information Science and Engineering, Xinjiang University, China
[2]School of Computer Engineering, Nanyang Technological University (NTU), Singapore
[3]Institute for Infocomm Research, A*STAR, Singapore
[4] Temasek Laboratories @ Nanyang Technological University Singapore

wangjunchao@mipitalk.com, haihuaxu@ntu.edu.sg

Abstract

In this work, we describe the TL-NTU's text-to-speech (TTS) system for the Blizzard Challenge 2018. Our efforts are mainly focused on two aspects, which are the front-end text analysis and back-end model training. For the front-end text analysis, we include phonetic, syllable, and word-level linguistic features using lexicon and word-level text analysis for full-context feature extraction. For back-end model training, a feed-forward Deep Neural Network (DNN) based phone duration model and a bidirectional long short-term memory (BLSTM) based acoustic model are trained. The performance of our system is assessed by reporting the results of listening tests provided by the challenge organizer.

Index Terms: text-to-speech, DNN, BLSTM

1. Introduction

Text-to-speech (TTS) is a technique to generate speech signal for given input text [1]. TTS can be realized by either selecting appropriate sub-word units from a database of natural speech, namely unit-selection, or generating speech from a statistical parametric model, namely statistical parametric speech synthesis (SPSS) [2]. Due to the flexibility to change its voice characteristics and robustness, SPSS is growing in popularity. This is also the focus of this work.

A typical SPSS system generally contains two parts, the front-end text analysis and back-end model training. The aim of front-end text analysis is to analyze the word sequence into parametric features. Then the back-end models (duration and acoustic model) are built mapping between text features and duration/acoustic features. During synthesis stage, given a word sequence, the speech waveform is reconstructed from the predicted parametric features of back-end models.

With the development of the deep learning techniques, i.e, Deep Neural Network (DNN) [3–6], the DNN based SPSS performance has been significantly improved. Here, the prior decision-tree based context-dependent HMM is replaced by DNN, which improves both naturalness and intelligibility [7]. To capture the information of long-span features, Long short-term memory (LSTM) and Bidirectional LSTM (BLSTM) were also employed in [8]. More recently, WaveNet [9–12] based TTS methods was proposed. The WaveNet is a convolutional autoregressive model which produces high quality speech. However, it generates synthesized speech with sample-by-sample manner, which makes both off-line training and run-time synthesis very slow. Another branch of recent breakthrough in TTS is end-to-end learning [13–15], that directly learns text features (character or letter) to wave audio (spectral frequency) mapping. Compared with the conventional DNN based TTS method, this approach abandon the front-end text analysis. Moreover, by using an attention mechanism [16, 17], both duration and acoustic features are modeled together.

For the Blizzard Challenge 2018, our submitted results are BLSTM based SPSS method built on Merlin toolkit [18]. Specifically, we include phonetic, syllable, and word-level linguistic features using lexicon and word-level text analysis for front-end full-context feature extraction. For back-end, we use a small feed-forward DNN for duration modeling and a BLSTM for acoustic modeling[1].

2. TL-NTU Speech Synthesis System

2.1. Data

2.1.1. Data Description

In the Blizzard Challenge 2018, the organizer released about 8.5 hours of a female British English data for training. The recording consists of 56 children story books with the types of ”mp3”, ”m4a” and ”wma”. For most of audio books, utterance level time-boundary segmentations were provided by the organizer.

2.1.2. Data preprocessing

The data provided by organizer cannot be directly applied for system training. It contains long speech segments and non-speech contents, e.g. ringing bell clips and over-long silence parts. Besides, the audio data format is also inappropriate for the acoustic feature extraction. To make the data appropriate for training, we first normalize all the audio files into standardized 16 KHz and 16 bit waveform files. Then we cut the audio and the corresponding text into standalone pair-wise utterances, according the time-boundary information provided by the organizer. To align the audio segments do not have time-boundary information, we conduct force alignment using our automatic speech recognition (ASR) system. Finally the overall data is about 6.5 hours for training.

2.2. Front-end Generation

2.2.1. Full-context label generation

To train DNN models, we need to generate the front-end features from given word-level utterances. Such features are normally named as the full-context label features, which is made up of phonetic features, syllable features, word features, as well

[1]Actually, we also tried the end-to-end based TTS method, however we were failed to generate quality synthesized speech for long utterances. Besides, we also tried with WaveNet, which is used as a Vocoder to generate speech from predict Vocoder features. It can consistent improve the quality of synthesized speech. However, we abandon it due to the time constraint.

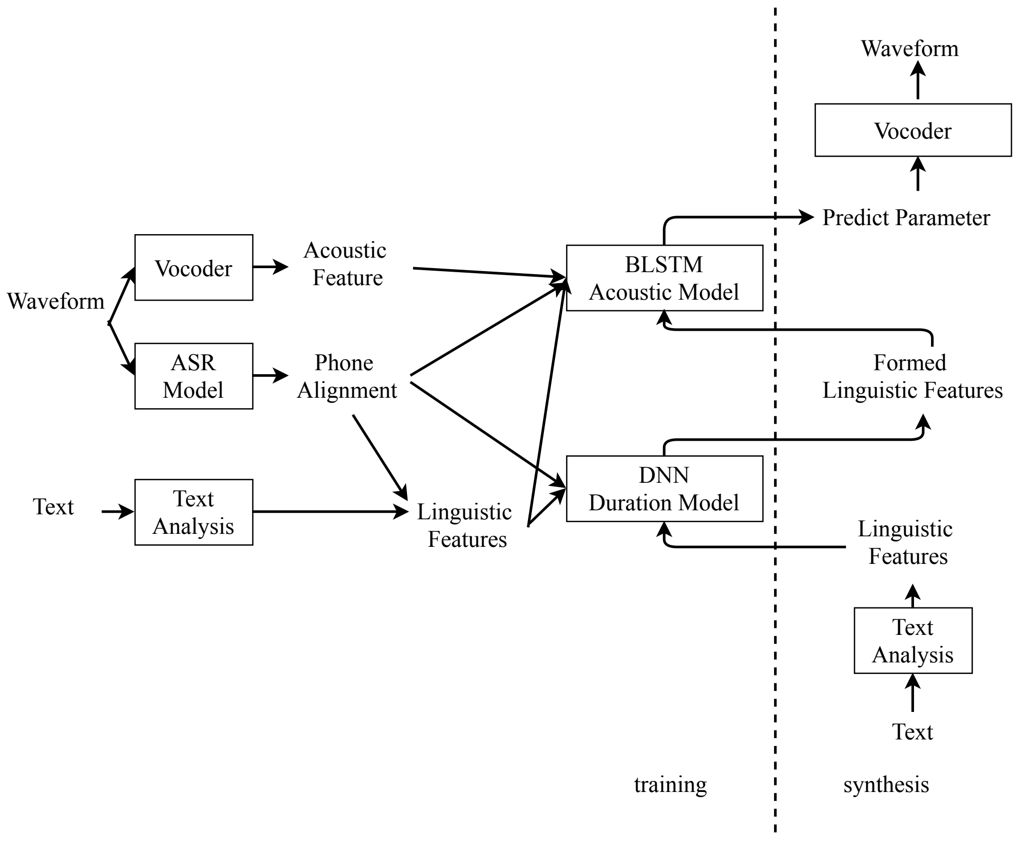

Figure 1: *Illustration of the training and speech synthesis procedures for deep neural network based text-to-speech system*

as features including part-of-speech and syntactic phrase etc. The phonetic and syllable features are from lexicon, either by ASR forced alignment or by lexicon looking-up, while the part-of-speech and syntactic phrase are from syntactic parser as indicated in [19]. There are overall 647 dimensional full-context label features.

2.2.2. Frame-level feature generation

In order to train DNN acoustic models, the front-end features contain both full-context label features as introduced in 2.2.1, and frame-level features, indicating the position and duration of the current phone [7]. Our frame-level features are four dimensional, including 3-dim position features and 1-dim duration features.

2.3. TL-NTU Overall TTS System Architecture

The overall TTS system architecture is illustrated in Figure 1. We train two DNN models to realize speech synthesis. One is a feed-forward DNN for duration modeling, and the other is a BLSTM for acoustic modeling. Before training, we make most of efforts on linguistic feature preparation. Except for the text analysis, we also use our ASR system to obtain the phone-level alignment as mentioned in 2.2.1. We use linguistic features and phone durations to train duration model. Meanwhile, we use linguistic plus frame-level features and acoustic features to train acoustic model. The acoustic features are extracted from the WORLD vocoder [20].

During the speech synthesis stage, we first do text analysis to generate linguistic features, while use duration model to generate frame-level features. The two features are then combined as input to the BLSTM acoustic models to predict the acoustic features that are utilized by the WORLD vocoder to synthesize the final waveform.

2.4. Duration model training

The feed-forward DNN is configured with 6 hidden layers with 1024 neurons for each layer. We use $\tanh$ as activation function.

To train the duration model, the input features are 647-dim binary and numeric features converted from the full-context label, the output features are one dimensional indicating the duration of the present phone in terms of frame number.

In total we have 7118 utterances, we randomly choose 218 utterances for validation and the rest is for training, we choose the Adam [21] algorithm to optimize neural network training, and the initial learning rate is set with 0.002. For the actual training, min-batch is fixed with 64. Before training the input features are min-max normalized.

2.5. Acoustic model training

We use the BLSTM to realize linguistic to acoustic feature mapping. The BLSTM neural network has two forward-backward recurrent layers topped with one full-connected layer. For the recurrent layers, there are 512 forward neurons and 512 backward neurons. The final full-connected layer has 127 neurons.

The input features are 651 dimensional. They are 647-dim full-context label features plus 4-dim frame-level features. The output features are 127 dimensional acoustic features, includ-

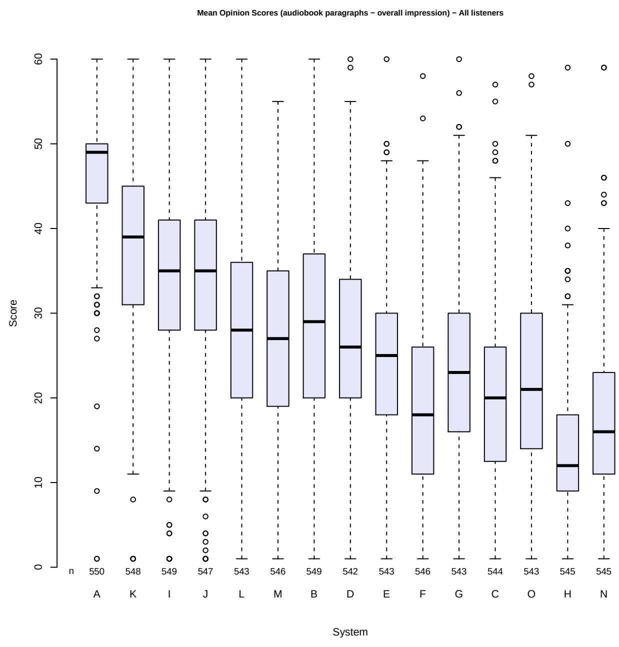

Figure 2: *Mean Opinion Scores of overall impression, our system is represented with "O".*

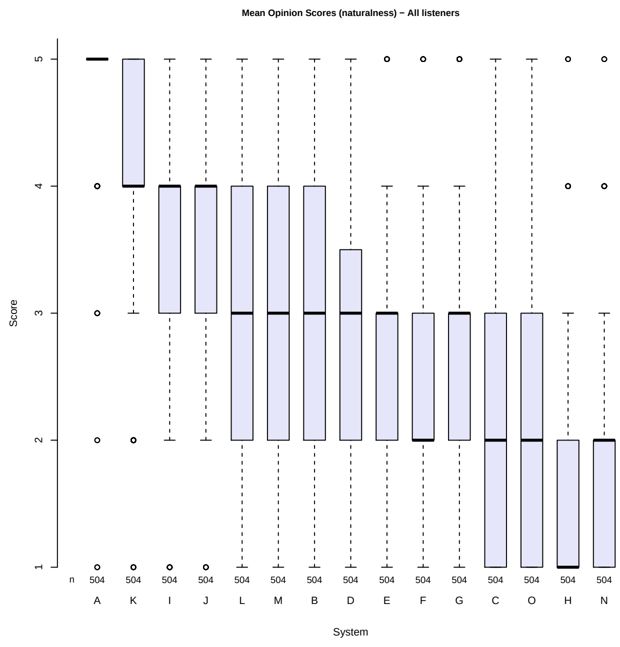

Figure 3: *Mean Opinion Scores for naturalness evaluation, our system is represented with "O".*

ing 40-dim Mel-generalized Cepstral (MGC), 1-dim log-f0(lf0), plus 1-dim band aperiodicity coefficients (BAP) as static features, and their corresponding delta and accelerate features, plus 1-dim voice/unvoice (vuv) features. Before training, both input and output features are normalized. The input features are min-max normalized, while the output features are global mean-variance normalized.

For training, we also use the Adam as the optimizer and set the initial learning rate with 0.0001 and set the batch size with 20.

2.6. Waveform synthesis

Once the acoustic features are predicted by the BLSTM acoustic models, we first denormalize the acoustic features with global mean and variance. We then use maximum likelihood parameter generation (MLPG) algorithm to get the static features. After that, we apply post-filtering on the static MGC features. The final features are taken as input to the WORLD vocoder to synthesize the waveform.

3. Results

3.1. Results

This section presents the official evaluation results, which are based on the subjective listening tests conducted by the organizer. Our system is labelled as ”O” in this year challenge.

Three kinds of listeners were participated in the listening tests, including paid listeners who is native speakers of English, online volunteers and speech experts.

There are 15 systems in evaluation test including 4 benchmark baselines, 10 participating teams and natural speech. System A is natural speech. System B is the Festival benchmark, which is a standard unit-selection system. System C is the HMM benchmark built using the HTS toolkit [22]. System D is the first DNN benchmark built using the HTS toolkit. System E is the second DNN benchmark built using the HTS toolkit, which employs trajectory training [23].

Four types of evaluation were conducted in this year, including Mean opinion scores for paragraphs (MOSpara) , MOS for sentences (MOSsentence), Similarity with original speaker (SIM) and Semantically unpredictable sentences (SUS). MOSpara eveluates the synthetic paragraph on the aspects of the overall impression, pleasantness, Speech Pauses, Stress, Intonation, Emotion and Listening effort, with a score scale of 1 to 60, the higher the better. MOSsentence evaluates the naturalness of the synthetic sentence with a score scale of 1 to 5. SIM represents how similar the synthetic voice is sounded to the reference samples on a scale from 1 to 5. SUS represents the intelligibility of the synthetic speech, and it is evaluated with word error rate (WER).

Overall, our system does not perform well. As shown in Figure 2, 3, 4, according to MOSpara, MOSsentence and SIM, our system just slightly outperforms the HMM based text-to-speech baseline system, while worse than the "D" and "E" DNN baselines. According to the SUS results, as shown in Figure 5, the WER our system is about 36%, which is around the average of all the systems. We are discussing the possible reasons in the next section.

4. Discussion & Conclusion

From Section 3.1, we can see our submitted results ("O") are even worse than the baseline systems of "D" and "E" for all the evaluations. In this section, we are discussing some reasons that probably affect our final results.

To build a TTS system from scratch, it mainly consists of four parts, data pre-processing for segmentation, alignment and cleaning, front-end text analysis for the full-context feature extraction, the back-end models (duration and acoustic modeling),

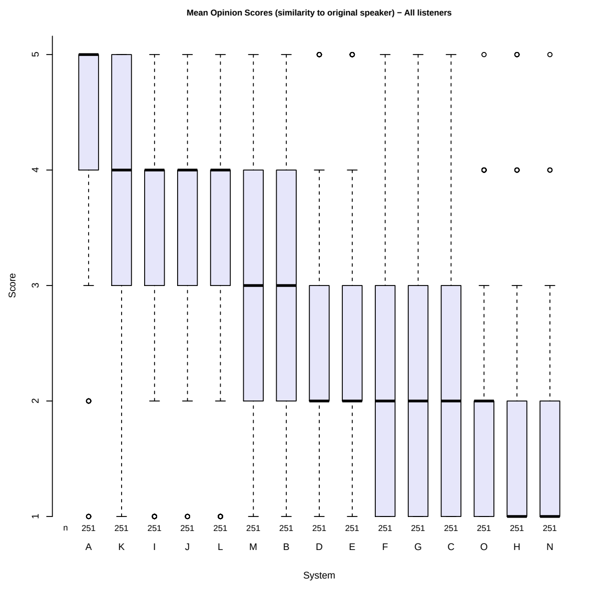

Figure 4: *Mean Opinion Scores for similarity to original speakers, our system is represented with "O".*

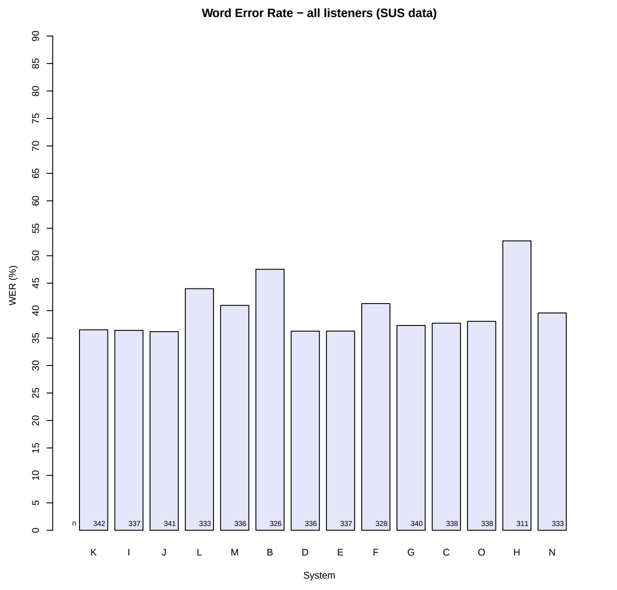

Figure 5: *Word Error Rate for intelligibility Evaluation, "O" is our system.*

as well as the vocoder responsible for the final speech generation. To train a TTS system with decent quality voice, it is indispensable to make comprehensive efforts on these four aspects. The failure for any of these four parts will lead to a underqualified TTS system. We are discussing problems of our system for these parts as below.

4.1. Data pre-processing

In this year Challenge, the organizer provided data that contains longer audio segments and corresponding transcriptions. To prepare data for the duration and acoustic model training, participants need to cut the audio segments into utterances and clean the dataset. In our implementation, we directly used the organizer provided time-boundary and our ASR recognizer to cut the audio segments. No extra boundary detection and data cleaning process was conducted. However, there are some over-expressive audio files, which appears abnormal in terms of pitch level fluctuation. This may have bad effect on the model training, yielding worse performance.

4.2. Back-end model training

For the back-end model training, we have not conducted extensive experiments on the challenge data, except for tuning the learning rate. For instance, we have not even changed the DNN topology architecture, due to limited time available.[2]

4.3. Initial Attempt Using WaveNet as Vocoder

After evaluation result submission, we also tried to learn a WaveNet using the learned acoustic features as input and the raw wave sample as output. As a result, the WaveNet here actually acts as a learned vocoder. On the third-party data, We observed obvious quality improvement compared the the case of using the conventional vocoder for waveform synthesis. In future, we are planning to conduct the WaveNet experiments on this year challenge data.

4.4. Conclusion

In this paper, we report our TL-NTU's text-to-speech system performance with the statistical parametric speech synthesis based DNN method in the Blizzard Challenge 2018. Due to various reasons, our system performance on the evaluation data is not good as that of the corresponding DNN baseline system released by the organizer, we also give an appropriate analysis and due discussion.

[2]Previously, we spent too much time on end-to-end attention based TTS system training using TACOTRON1. However, we found our end-to-end models cannot generalize to the unseen evaluation data, particularly it failed to synthesize speech for those longer sentences. Consequently, we gave it up and resorted the Merlin to train our TTS system only in the final remaining week.

5. References

[1] S. King, "Measuring a decade of progress in text-to-speech," *Loquens*, vol. 1, no. 1, p. 006, 2014.

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

[3] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[4] A.-r. Mohamed, G. E. Dahl, G. Hinton *et al.*, "Acoustic modeling using deep belief networks," *IEEE Trans. Audio, Speech & Language Processing*, vol. 20, no. 1, pp. 14–22, 2012.

[5] Z.-H. Ling, L. Deng, and D. Yu, "Modeling spectral envelopes using restricted boltzmann machines and deep belief networks for statistical parametric speech synthesis," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 10, pp. 2129–2139, 2013.

[6] H. Zen, "Deep learning in speech synthesis." in *SSW*, 2013, p. 309.

[7] H. Ze, A. Senior, and M. Schuster, “Statistical parametric speech synthesis using deep neural networks,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 7962–7966.

[8] Y. Fan, Y. Qian, F.-L. Xie, and F. K. Soong, “Tts synthesis with bidirectional lstm based recurrent neural networks,” in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[9] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, “Wavenet: A generative model for raw audio,” *CoRR abs/1609.03499*, 2016.

[10] A. v. d. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. v. d. Driessche, E. Lockhart, L. C. Cobo, F. Stimberg *et al.*, “Parallel wavenet: Fast high-fidelity speech synthesis,” *arXiv preprint arXiv:1711.10433*, 2017.

[11] A. Tamamori, T. Hayashi, K. Kobayashi, K. Takeda, and T. Toda, “Speaker-dependent wavenet vocoder,” in *Proc. Interspeech*, vol. 2017, 2017, pp. 1118–1122.

[12] T. L. Paine, P. Khorrami, S. Chang, Y. Zhang, P. Ramachandran, M. A. Hasegawa-Johnson, and T. S. Huang, “Fast wavenet generation algorithm,” *arXiv preprint arXiv:1611.09482*, 2016.

[13] J. Sotelo, S. Mehri, K. Kumar, J. F. Santos, K. Kastner, A. Courville, and Y. Bengio, “Char2wav: End-to-end speech synthesis,” 2017.

[14] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio *et al.*, “Tacotron: Towards end-to-end speech synthesis,” *arXiv preprint arXiv:1703.10135*, 2017.

[15] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan *et al.*, “Natural tts synthesis by conditioning wavenet on mel spectrogram predictions,” *arXiv preprint arXiv:1712.05884*, 2017.

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

[17] 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.

[18] Z. Wu, O. Watts, and S. King, “Merlin: An open source neural network speech synthesis system,” *Proc. SSW, Sunnyvale, USA*, 2016.

[19] Y. Lu, Z. Zhang, C. Yang, H. Ming, X. Zhu, Y. Zhang, S. Yang, D. Huang, L. Xie, and M. Dong, “The i2r-nwpu text-to-speech system for blizzard challenge 2017,” in *Proc. Blizzard Challenge Workshop*, 2017.

[20] M. Morise, F. Yokomori, and K. Ozawa, “World: a vocoder-based high-quality speech synthesis system for real-time applications,” *IEICE TRANSACTIONS on Information and Systems*, vol. 99, no. 7, pp. 1877–1884, 2016.

[21] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[22] H. Zen, T. Nose, J. Yamagishi, S. Sako, T. Masuko, A. W. Black, and K. Tokuda, “The hmm-based speech synthesis system (hts) version 2.0.” in *SSW*. Citeseer, 2007, pp. 294–299.

[23] Z. Wu and S. King, “Minimum trajectory error training for deep neural networks, combined with stacked bottleneck features,” in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.