# COMBINING UNIDIRECTIONAL LONG SHORT-TERM MEMORY WITH CONVOLUTIONAL OUTPUT LAYER FOR HIGH-PERFORMANCE SPEECH SYNTHESIS

*Wenfu Wang, Bo Xu*

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

## ABSTRACT

In this paper, we target improving the accuracy of acoustic modelling for statistical parametric speech synthesis (SPSS) and introduce the convolutional neural network (CNN) due to its powerful capacity in locality modelling. A novel model architecture combining unidirectional long short-term memory (LSTM) and a time-domain convolutional output layer (COL) is proposed and employed to acoustic modelling. The two components complement each other and result in a high-performance synthesis system. Specifically, the unidirectional LSTM can learn expressive feature representations from history context and the COL ingeniously absorbs some of these representations within a look-ahead window to advance predictions. This complementary mechanism significantly improve the predictive accuracy and the quality of synthetic speech. In addition, the unique operation mechanism of convolution makes COL a fine parameter trajectory smoother between consecutive frames. Subjective preference tests show that the proposed architecture can synthesize natural sounding speech without dynamic features.

***Index Terms***— Statistical parametric speech synthesis, LSTM, convolutional output layer, high-performance, trajectory smoother

## 1. INTRODUCTION

Recently the deep neural networks (DNNs) have greatly advanced the perceived naturalness of synthetic speech in statistical parametric speech synthesis (SPSS) [1, 2, 3, 4, 5, 6]. Further improvements were reported by using more advanced training criteria [7, 8, 9, 10, 11, 12] and more powerful models such as long short-term memory (LSTM) [13, 14] and gated recurrent unit (GRU) [15] recurrent neural networks (RNNs) [8, 16, 17, 18]. However, the acoustic modelling accuracy still remains a key factor [19] that limits the quality of synthetic speech.

In this paper, we explore more advanced acoustic modelling techniques for SPSS. Convolutional neural networks (CNNs) are an alternative type of neural network that can be used to model locally spatial and temporal correlation in sequential structure through weight sharing across local regions of input space. They have been explored extensively in the image recognition [20] and speech recognition [21, 22] fields, offering improvements over DNNs on many tasks. Motivated by the success, this paper investigates the application of CNNs to acoustic modelling in SPSS. Specifically, a novel model architecture (see Fig.1) using unidirectional LSTM-RNNs as its base and a simplified time-domain convolutional network as its output layer, is proposed and employed to acoustic modelling. Intuitively, the unidirectional LSTM (ULSTM) can learn expressive representations from history context and the convolutional output layer (COL) just ingeniously utilizes some of the representations within a look-ahead window to make a better prediction at each generation step. Experimental results both subjectively and objectively show that our proposed model can achieve high-performance SPSS. Specifically, the *advantages* of combination of ULSTM and COL (referred to as ULSTM-COL) are threefold. *First*, it can significantly improve the predictive accuracy of acoustic features over both ULSTM and latency-controlled bidirectional LSTM (LC-BLSTM) [23] models, and best perceived naturalness is also achieved. *Second*, the unique operation mechanism of the convolutional output layer makes itself serve as a fine parameter trajectory smoother between consecutive frames of acoustic parameters; hence, the dynamic feature constraints and maximum likelihood parameter generation (MLPG) algorithm [24] used to produce smooth trajectories are not required any more. *Third*, the unidirectional nature with negligible latency of the proposed architecture allows low-latency synthesis in real-time application.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 describes the ULSTM-COL architecture for acoustic modeling in SPSS. Experimental results and analysis are presented in Section 4, and Section 5 gives the conclusions.

## 2. RELATED WORK

The most popular way to apply neural networks to SPSS is use a feedforward neural network (FNN) or recurrent neural network (RNN) as a deep regression model to map linguistic features directly to acoustic features (e.g. LSP). However, a limitation of these two kinds of architectures is that the mapping performed is in essence still a one-to-one problem, though the RNN internally considers the dependencies on history or future context when propagating information. Strictly speaking, only the output of last or next timestep is explicitly used. Thus a potential problem is that the outputting representations of DNN/RNN in a range of successive timesteps are underused when performing a prediction at each timestep. However, the convolutional operation in our proposed ULSTM-COL architecture has the capacity to absorb multiple high-level representations each time to make a better prediction through introducing a slight but negligible delay. So the ULSTM-COL model actually performs a many-to-one mapping as opposed to DNN/RNN.

Dynamic features (deltas, delta-deltas) are usually used to smooth parameter trajectories during generation. But in a typical conventional implementation of neural networks-based SPSS, the relationships between static and dynamic features are ignored during training. New training criteria such as minimum trajectory error [9] and minimum sequence error [10] are proposed to explicitly take into account dynamic constraints in the training phase. Anoth-

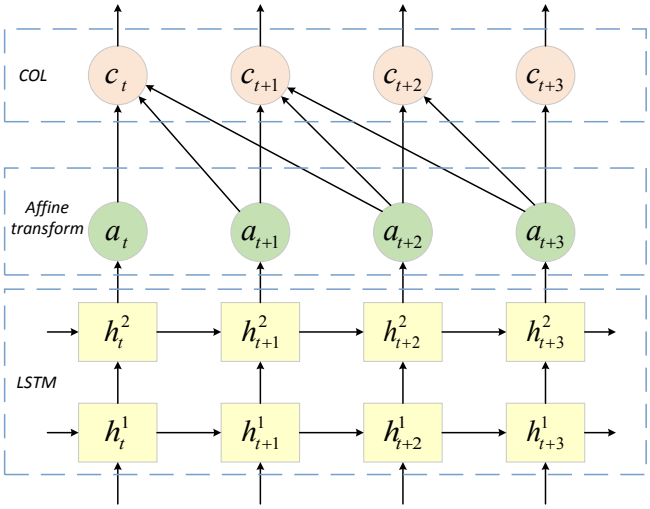



**Fig. 1**. Schematic diagram of the proposed ULSTM-COL architecture for SPSS.

er existing problem is the high-latency that MLPG algorithm brings (case 1 in [24]) during generation. Some solutions to addressing the high-latency are listed in [16]. But the most direct way to bypass these troubles is to remove the dynamic features during modelling. Zen et al. [16] proposed a recurrent output layer (ROL) based on unidirectional LSTM to achieve smooth transitions between consecutive frames and accordingly the MLPG is replaced. But it is still faced with the limitation that not such enough local information is utilized that inadequate smooth may exist when performing a prediction. However, the unique operation mechanism of COL makes itself a fine trajectory smoother without suffering from the above problem. In addition, the ULSTM-COL architecture with a negligible delay maintains the unidirectional property, thus allowing low-latency synthesis as ROL does in [16].

## 3. MODEL ARCHITECTURE

This section describes the ULSTM-COL architecture, illustrated in Fig. 1, for acoustic modelling in SPSS.

### 3.1. Long Short-Term Memory with Projection

The LSTM was initially proposed in [13] to solve the gradient vanishing problem in RNNs. Several minor modifications to the original LSTM unit has been made. In this paper, we adopt the implementation version as used in [14], where a recurrent projection layer is appended after the LSTM cells. The iterating equations are as follows:

$$\boldsymbol{i}_t = sigm(\boldsymbol{W}_{ix}\boldsymbol{x}_t + \boldsymbol{W}_{ir}\boldsymbol{r}_{t-1} + \boldsymbol{W}_{ic} \odot \boldsymbol{c}_{t-1} + \boldsymbol{b}_i) \quad (1)$$

$$\boldsymbol{f}_t = sigm(\boldsymbol{W}_{fx}\boldsymbol{x}_t + \boldsymbol{W}_{fr}\boldsymbol{r}_{t-1} + \boldsymbol{W}_{fc} \odot \boldsymbol{c}_{t-1} + \boldsymbol{b}_f) \quad (2)$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot g(\boldsymbol{W}_{cx}\boldsymbol{x}_t + \boldsymbol{W}_{cr}\boldsymbol{r}_{t-1} + \boldsymbol{b}_c) \quad (3)$$

$$\boldsymbol{o}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + sigm(\boldsymbol{W}_{ox}\boldsymbol{x}_t + \boldsymbol{W}_{or}\boldsymbol{r}_{t-1} + \boldsymbol{W}_{oc} \odot \boldsymbol{c}_t + \boldsymbol{b}_o) \quad (4)$$

$$\boldsymbol{m}_t = \boldsymbol{o}_t \odot h(\boldsymbol{c}_t) \quad (5)$$

$$\boldsymbol{r}_t = \boldsymbol{W}_{rm}\boldsymbol{m}_t \quad (6)$$

where the $\boldsymbol{W}$ terms denote weight matrices, the $\boldsymbol{b}$ terms denote bias vectors, $\boldsymbol{i}$, $\boldsymbol{f}$ and $\boldsymbol{o}$ are vectors of the input gate, forget gate and output gate respectively, and $\boldsymbol{c}$, $\boldsymbol{m}$ are cell activations and cell output vectors, $\boldsymbol{r}$ is the output vector via projecting $\boldsymbol{m}$, as formulated by Eq. 6. Basically the projection can reduce the output dimension, resulting in fewer parameters than the standard LSTM.

The LSTM in the ULSTM-COL works as a base to capture long-term dependencies across the linguistic input and to offer better representations for subsequent use as input to the convolution output layer.

### 3.2. Convolutional Output Layer

Since CNN has powerful capabilities to model locally spatial and temporal correlation in sequential structure through weight sharing across local regions of input space, a simplified time-domain convolutional output layer is designed to complement the ULSTM. Specifically, the COL focuses on only a small portion of the future information and absorb it to perform current prediction, as shown in Fig. 1. Suppose at timestep $t$, we consider a look-ahead window with width of $N$. We now have a feature representation matrix $\boldsymbol{a}_{t:t+N}$ of size $(N+1) \times d$, where $d$ is the dimension of acoustic features. Then a convolutional template $\boldsymbol{W}$ of the same size as $\boldsymbol{a}_{t:t+N}$ is defined. The output vector $\boldsymbol{c}_t$ of the COL at timestep $t$ is:

$$\boldsymbol{c}_t = \sum_{i=0}^{N} \boldsymbol{w}_i \odot \boldsymbol{a}_{t+i} \quad (7)$$

where $\boldsymbol{w}_i$ corresponds to the $i$-th row of the convolutional template $\boldsymbol{W}$ and $\odot$ denotes element-wise multiplication.

Note that our intention is to improve the acoustic modelling accuracy in a low-latency setting on the basis of ULSTM, so the convolutional template targets a certain amount of context information within a look-ahead window. As a result, the straightforward way to gather information makes ULSTM-COL perform even better than a latency-controlled bidirectional LSTM, which will be described in next subsection. The convolutional template can also be interpreted as a sequential memory block, which reads, at a time, a small fragment of the input sequence and writes a record (frame). We place the convolutional template at the output layer because experiments demonstrate it can serve as a fine parameter trajectory smoother. Thus dynamic features can be removed during modelling.

### 3.3. Latency-controlled Bidirectional LSTM

The latency-controlled bidirectional LSTM (LC-BLSTM) was proposed in [23] to lower the latency existing in a standard BLSTM in speech recognition. In LC-BLSTM, the forward sub-layer exploits the past history in the same way as ULSTM does. However, for the backward sub-layer, the LC-BLSTM only looks ahead a fixed number of frames instead of seeing the whole utterance during generation. Therefore, the LC-BLSTM can be much more efficiently trained in a mini-batch fashion without performance loss. This paper employs the LC-BLSTM as a benchmark to be compared with our proposed ULSTM-COL model.

### 3.4. Low-latency Synthesis using ULSTM-COL

The low-latency synthesis using ULSTM-COL can be outlined as follows. First, a text to be synthesized is converted into a sequence of phoneme-level linguistic features through text analysis. Next, the frame-level features of each phoneme are predicted using the duration model. Then the phoneme-level and frame-level features are spliced together as inputs to the well-trained ULSTM-COL. A fixed

number of input features are delayed at start when outputting the first frame. Then the propagation and waveform synthesis can be performed sequentially in a streaming manner. Note that the first step (text analysis) is processed in a sentence-level, whereas the remaining steps are streaming processing, as text analysis is usually significantly faster than the remaining ones. Here, the low-latency synthesis runs similar to that in [16] apart from a few frames of delay.

## 4. EXPERIMENTS

### 4.1. Experimental Setups

A Chinese 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 6550, 686 and 30 utterances respectively). The speech data was downsampled from 44.1 kHz to 16 kHz, then 40-order line spectral pairs (LSPs) plus a gain, 25 band aperiodicities (BAPs) and logarithmic fundamental frequency (log $F_0$) were extracted every 5-ms using STRAIGHT [25].

For the training of neural networks, the speech data and its associated transcriptions were time-aligned using an HMM aligner, which was first trained using maximum likelihood criterion and then refined by minimum generation error (MGE) training to minimize the generation error between predicted and original parameter trajectories of the training data. The phoneme-level feature vector contained 462 binary features for categorical linguistic contexts (e.g. phonemes identities) and 64 numeric features for numerical linguistic contexts (e.g. the number of phonemes in the current word). Then five binary features for state index and a numeric feature for the position of a frame in the current state were appended to the phoneme-level feature vectors to form frame-level linguistic features. The acoustic feature consisted of 41 LSPs, 25 BAPs and an interpolated log $F_0$, and optionally their dynamic counterparts. A voiced/unvoiced flag was also added to the output vector to indicate the voicing condition of the current frame. Both the input and output features were normalized to the range of [0.01, 0.99].

For comparison, three types of systems, which were ULSTM, LC-BLSTM and ULSTM-COL respectively, were established. The ULSTM served as a baseline to evaluate the other two models. All the architectures had two hidden layers. For the ULSTM architecture, each layer contained 800 memory blocks with 512 recurrent projection units, while the LC-BLSTM used an asymmetric architecture. Specifically, the forward sub-layer had 600 memory block with 384 recurrent projection units; and the backward sub-layer had 200 memory blocks with 128 recurrent projection units. The fixed number for latency control is set to 10. The window width $N$ of convolutional template in COL is set to 5[1] unless otherwise explicitly stated in the experiments. The parameters of all the models were first pretrained using layerwise backpropagation, and then optimized with a mini-batch stochastic gradient descent (SGD)-based algorithm with an initial learning rate of 0.004, and momentum of 0.5. For software implementation, the Kaldi toolkit [26] was used and training was conducted on a Tesla K80 GPU. Training ULSTM, LC-BLSTM and ULSTM-COL took about respectively 50 minutes, 85 minutes and 65 minutes every epoch.

At synthesis time, if the acoustic features contained dynamic features, the speech parameter generation algorithm (case 1 in [24]) was used to generate smooth acoustic trajectories. LSP based formant enhancement [27] was used to improve the quality of synthesized speech.

We evaluated the performance of the systems both objectively and subjectively. 30 utterances were tested. To objectively evaluate the synthetic quality, log spectral distance (LSD), BAPs error, voiced/unvoiced error rate and root mean squared error (RMSE) of log $F_0$ were measured. The subjective evaluation was an AB preference test. 15 native listeners with no hearing difficulties participated in the evaluation using headphones. Each subject evaluated 20 pairs of synthesized utterances and each pair was evaluated by 10 subjects at most. After listening to each pair of synthesized utterances, the subjects were asked to choose their preferred one; they could choose "neutral" if they had no preference.

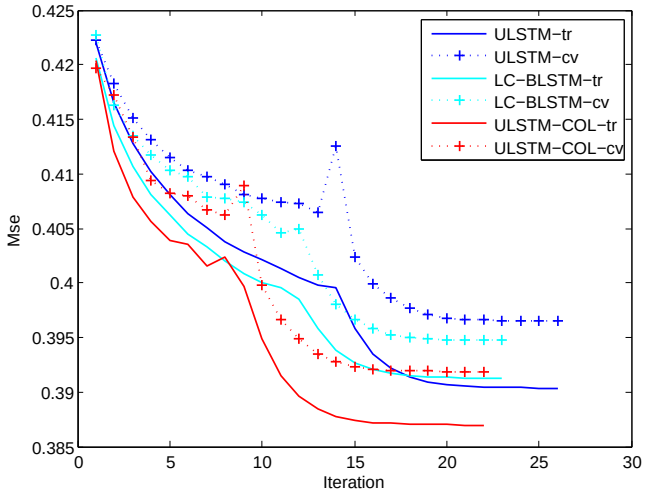



**Fig. 2**. Iteration curves of different systems on training set and development set.

### 4.2. Experimental Results and Analysis

The three systems were modeled with and without dynamic features respectively, whereas they used the same linguistic features throughout the experiments. We demonstrated the ULSTM-COL performed quite well under both these two situations.

#### *4.2.1. With dynamic features*

**Table 1**. Preference scores (%) between different systems modeled with dynamic features. The confidence level of $t$-test is 0.95.

| ULSTM | LC-BLSTM | ULSTM-COL | Neutral | $p$-value |
|---|---|---|---|---|
| 17.0 | – | **72.3** | 10.7 | $< 10^{-6}$ |
| – | 20.0 | **65.0** | 15.0 | $< 10^{-6}$ |
| 27.3 | **45.0** | – | 27.7 | $< 10^{-5}$ |

Fig. 2 shows the iteration curves on training set and development set during model training. It can be clearly seen that the ULSTM-COL system converges faster than both ULSTM and LC-BLSTM, and it achieves significantly the minimum mean square error (MSE) on development set.[2] We conjecture that the ULSTM layers have learned good feature representations from history context, so the

[1]This setup will be explained in the following experiments.

[2]This also leads to the best objective measurements among all the systems, which is not listed here due to limited space.

COL simply absorbs the appropriate local information within a look-ahead window to make a better prediction. This complementary mechanism makes ULSTM-COL a more powerful architecture to predict acoustic features. The subjective test in Table 1 also confirms this superiority. Significant preference to ULSTM-COL is given when comparing against the other two systems.

#### 4.2.2. *Without dynamic features*

**Table 2**. Objective results for variable super-parameter $N$ of the look-ahead window.

| $N$ | LSD (dB) | BAP Error (dB) | V/UV Error Rate (%) | RMSE of log $F_0$ |
|---|---|---|---|---|
| 3 | 2.2836 | **2.4154** | 5.108 | 0.1076 |
| 5 | **2.2823** | 2.4202 | 5.021 | **0.1051** |
| 7 | 2.2854 | 2.4222 | **5.017** | 0.1069 |
| 9 | 2.2854 | 2.4272 | 5.281 | 0.1066 |
| 15 | 2.2868 | 2.4255 | 5.212 | 0.1069 |

**Table 3**. Statistics of the database. A 5-state HMM was used.

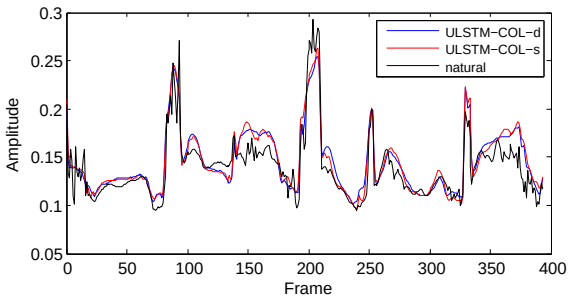
| # Frames | # Phonemes | # States | Avg. frames/state |
|---|---|---|---|
| 5819252 | 204552 | 1022760 | 5.69 |

First, we investigated the effect of varying the number of frames (super-parameter $N$) within the look-ahead window in COL on objective evaluation, shown in Table 2. From the table we can see that as N goes to 5, the LSD and RMSE of log $F_0$ reaches the minimum at the same time, and BAP error and V/UV error rate also gets approximately to their minimums respectively. But as N goes larger, these objective measurements all show rising trends in general. To account for this, simple statistical analyses of the training database were conducted. Table 3 lists some statistics. By calculation each HMM state lasts about 5.69 frames on average. It is interesting that this value is very close to the best width (**5** in our experiments) of the look-ahead window. We conjecture that the convolutional operation may include redundant information or even noise when looking ahead more than one state. This suggest a small amount of future information at one-state level is enough to help to advance prediction for the ULSTM-COL system. Note that this very small width brings just hundreds of parameters to COL but with great performance improvements. It also results in a negligible delay in a real-time synthesis application.

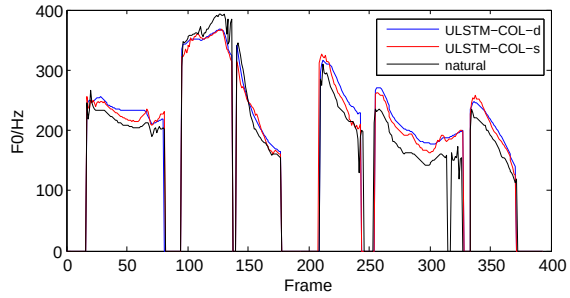
**Table 4**. Preference scores(%). The confidence level of $t$-test is 0.95.

| ULSTM-COL-d | ULSTM-COL-s | Neutral | $p$-value |
|---|---|---|---|
| 35.3 | **39.0** | 25.7 | 0.354 |

The unique work mechanism of convolution also makes itself a fine parameter trajectory smoother when placed at the output layer. This can be demonstrated from two aspects. First, we compared the ULSTM-COL system modeled with and without dynamic features (denoted as ULSTM-COL-d and ULSTM-COL-s respectively) using subjective preference test. As can be seen from Table 4, not statistically significant preference is showed to the either one of the cases, indicating that whether there are dynamic features has no significant effect on the perceived quality[3]. Second, Fig. 3 visualized the trajectories of 3rd LSP coefficients and $F_0$ contours of natural speech and generated by ULSTM-COL-d and ULSTM-COL-s. We can see in both cases the ULSTM-COL can generate smooth trajectories and $F_0$ contours. It' especially clear that ULSTM-COL-s can predict more detailed information which may be easily wiped away by the MLPG algorithm used with dynamic features.




(a) Trajectories of 3rd LSP coefficients




(b) F0 contours

**Fig. 3**. Trajectories of 3rd LSP coefficients and F0 contours for one test utterance.

Since the ULSTM-COL exhibits high-performance with just static acoustic parameters, it is applicable to a low-latency, real-time synthesis application.

## 5. CONCLUSIONS

This paper investigates the application of convolutional neural networks to acoustic modelling for SPSS. We propose a high-performance synthesis architecture called ULSTM-COL that takes advantages of the complementarity of unidirectional LSTM and convolutional output layer by combining them together. Specifically, the unidirectional LSTM works as base to offer expressive feature representations by capturing long-term history dependencies across linguistic input and the COL just ingeniously absorbs some of these representations within a look-ahead window to advance predictions. In addition, the unique operation mechanism of COL makes itself a fine trajectory smoother between consecutive acoustic frames. Experimental results both subjectively and objectively demonstrated that the ULSTM-COL trained with only static features can synthesize natural sounding speech. The generation process demands just a few frames of delay and maintains the unidirectional property. All these merits make the ULSTM-COL a high-performance synthesis system applicable to low-latency, real-time applications.

Our future work will focus on the investigation of convolutional neural networks as input layer for acoustic modelling.

[3]We also modeled ULSTM and LC-BLSTM with just static features, but noticeable discontinuities can be discerned in preliminary listening tests.

## 6. REFERENCES


[1] Heiga Zen, Andrew Senior, and Mike Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Proc. ICASSP*. IEEE, 2013, pp. 7962–7966.

[2] Yao Qian, Yuchen Fan, Wenping Hu, and Frank 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.

[3] Zhizheng Wu, Cassia Valentini-Botinhao, Oliver Watts, and Simon King, "Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis," in *Proc. ICASSP*. IEEE, 2015, pp. 4460–4464.

[4] Oliver Watts, Zhizheng Wu, and Simon King, "Sentence-level control vectors for deep neural network speech synthesis," in *Proc. Interspeech*, 2015.

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

[6] Shinji Takaki, Junichi Yamagishi, and Zhenzhou Wu, "A function-wise pre-training technique for constructing a deep neural network based spectral model in statistical parametric speech synthesis," *Machine Learning in Spoken Language Processing (MLSLP)*, 2015.

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

[8] Wenfu Wang, Shuang Xu, and Bo Xu, "Gating recurrent mixture density networks for acoustic modeling in statistical parametric speech synthesis," in *Proc. ICASSP*. IEEE, 2016.

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

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

[11] Zhizheng Wu and Simon King, "Improving trajectory modelling for DNN-based speech synthesis by using stacked bottleneck features and minimum trajectory error training," *arXiv preprint arXiv:1602.06727*, 2016.

[12] Benigno Uria, Iain Murray, Steve Renals, Cassia Valentini-Botinhao, and John Bridle, "Modelling acoustic feature dependencies with artificial neural networks: Trajectory-rnade," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4465–4469.

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

[14] Haşim Sak, Andrew Senior, and Françoise Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," *arXiv preprint arXiv:1402.1128*, 2014.

[15] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[16] Heiga Zen and Hasim 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] Yuchen Fan, Yao Qian, Fenglong Xie, and Frank K Soong, "TTS synthesis with bidirectional LSTM based recurrent neural networks," in *Proc. Interspeech*, 2014, pp. 1964–1968.

[18] Zhizheng Wu and Simon King, "Investigating gated recurrent networks for speech synthesis," in *Proc. ICASSP*. IEEE, 2016.

[19] Heiga Zen, Keiichi Tokuda, and Alan W Black, "Statistical parametric speech synthesis," *Speech Communication*, vol. 51, no. 11, pp. 1039–1064, 2009.

[20] Yann LeCun, Fu Jie Huang, and Leon Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*. IEEE, 2004, vol. 2, pp. II–97.

[21] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, and Gerald Penn, "Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4277–4280.

[22] Tara N Sainath, Oriol Vinyals, Andrew Senior, and Hasim Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4580–4584.

[23] Yu Zhang, Guoguo Chen, Dong Yu, Kaisheng Yao, Sanjeev Khudanpur, and James Glass, "Highway long short-term memory RNNs for distant speech recognition," in *Proc. ICASSP*. IEEE, 2016.

[24] Keiichi Tokuda, Takayoshi Yoshimura, Takashi Masuko, Takao Kobayashi, and Tadashi Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," in *Proc. ICASSP*. IEEE, 2000, vol. 3, pp. 1315–1318.

[25] Hideki Kawahara, Ikuyo Masuda-Katsuse, and Alain 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.

[26] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukáš Burget, Ondřej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlíček, Yanmin Qian, Petr Schwarz, et al., "The Kaldi speech recognition toolkit," 2011.

[27] Zhen-Hua Ling, Yi-Jian Wu, Yu-Ping Wang, Long Qin, and Ren-Hua Wang, "Ustc system for Blizzard Challenge 2006 an improved HMM-based speech synthesis method," in *Blizzard Challenge Workshop*, 2006.