Contextual Representation using Recurrent Neural Network Hidden State for Statistical Parametric Speech Synthesis

Sivanand Achanta, Rambabu Banoth, Ayushi Pandey, Anandaswarup Vadapalli, and Suryakanth V Gangashetty

Speech and Vision Laboratory, IIIT, Hyderabad, India.

{sivanand.a,rambabu.b,ayushi.pandey,anandaswarup.vadapalli}@research.iiit.ac.in

svg@iiit.ac.in

Abstract

In this paper, we propose to use hidden state vector obtained from recurrent neural network (RNN) as a context vector representation for deep neural network (DNN) based statistical parametric speech synthesis. While in a typical DNN based system, there is a hierarchy of text features from phone level to utterance level, they are usually in 1-hot-k encoded representation. Our hypothesis is that, supplementing the conventional text features with a continuous frame-level *acoustically guided* representation would improve the acoustic modeling. The hidden state from an RNN trained to predict acoustic features is used as the additional contextual information. A dataset consisting of 2 Indian languages (Telugu and Hindi) from Blizzard challenge 2015 was used in our experiments. Both the subjective listening tests and the objective scores indicate that the proposed approach performs significantly better than the baseline DNN system.
Index Terms: speech synthesis, recurrent neural network, deep neural network

1. Introduction

Text-to-speech synthesis using statistical parametric approach has received a great deal of attention in the last decade [1]. This trend can be observed in Blizzard challenge results in recent years [2], where statistical parametric speech synthesis systems (SPSS) have outperformed unit-selection counterparts. However, the naturalness of SPSS based systems needs to be improved. In [1], authors have identified the causes leading to the unnatural sounding in SPSS as: (1) Acoustic modeling (2) Vocoding and (3) Parameter generation. In this paper, we attempt to improve the acoustic modeling in a deep neural network (DNN) based SPSS.

DNNs have been used for acoustic modeling in both automatic speech recognition and SPSS [3] [4]. They have shown consistent improvements over hidden Markov model based approach. While the success of DNNs for acoustic modeling has been encouraging, there are several aspects that are not particularly suited for modeling speech parameters in synthesis domain. We highlight three aspects that may need to be addressed in a DNN based SPSS framework.

- Smooth parameter generation
- Context representation
- Avoid over-smoothing

The first problem has been addressed in the recent years either by the use of recurrent neural networks (RNNs) [5] [6] or by modifying the cost function to include the dynamic constraints during training [7]. The latter two problems received much lesser attention and has been attempted be very few [8] [9]. We focus on the second problem in this paper.

In a typical DNN/RNN based SPSS systems, text features at the input are composed of: (1) categorical (eg. penta-phone identities, identity of the vowel in the current syllable, etc,), (2) numerical features (# of syllabels in word, # words in phrase and so on), and (3) duration features like frame index, duration of phone etc,. The categorical features are usually 1-hot-k encoded representations. However a more suitable form of representation would be to use continuous valued representations as described in [10]. Learning embeddings for the phones/words/utterances as in [9] can also be considered as one alternate approach. In this paper, we train an RNN for predicting acoustic features and append the hidden-state of the RNN as the *acoustically guided* contextual representation for the conventional text-features. One advantage of this method over the embeddings is that these are adaptive for every new test utterance while embeddings once learnt remain fixed.

The paper is organized as follows: Section 2 relates the current work to previous works in the literature. In Section 3 a detailed description of the RNN architecture is given. The proposed method along with experiments and results is presented in Sections 4 and 5 respectively. The conclusions of the paper are drawn in Section 6 and possible directions for future work are given in Section 7.

2. Relation to Prior Work

In [11] [12], the authors have proposed the use of bottleneck features for improving DNN based speech synthesis. However our approach differs in some principal ways. Both the bottleneck and synthesis networks are DNNs in [11], while we use RNN for Context Representation (CR) learning network. Using DNN for CR network introduces two parameters (1) the bottleneck layer size, and (2) number of frames to be stacked. Extensive experiments have been reported in [11] to determine these two parameters. However, using a RNN hidden state as additional contextual representation, the problem of optimization of these hyper-parameters can be circumvented as will be shown in the later sections. The recurrent hidden state memorizes the past and hence avoids the need to stack. Also [11] uses a combined acoustic representation at the output of both CR and synthesis systems while we train independent systems for spectrum, f_0 and aperiodicity parameters.

3. RNNs for SPSS

In this section we briefly review the RNN based SPSS and the choice of our recurrent unit.

3.1. Elman RNN

An Elman RNN is a simple RNN with hidden-to-hidden recurrent connections and can be formally described as follows:

$$h_t = f(W_i x_t + W h_{t-1} + b_h) \quad (1)$$

$$y_t = g(U h_t + b_o) \quad (2)$$

The forward propagation of Elman RNN unrolled over time is shown in Fig. 1. W_i represents the input to hidden weights, W the recurrent weights of hidden layer and b_h the hidden biases, U the hidden to output weights and b_o the biases of output neurons. f, g are the activation functions at hidden and output layers respectively. Where x_t, h_t, y_t are input, state and output at time t respectively and h_{t-1} is the state at previous time instant $t-1$. In our case, since it is a regression task output layer is kept linear.

The parameters of the output layer can be learned using normal back-propagation, and other parameters of the model are learned using back-propagation through time. The recursion for computing the error signal at hidden layer is given as [13]

$$\begin{aligned} e_t &= y_t - d_t \\ \delta_t &= f^{'} * (W^T \delta_{t+1} + U^T e_t) \end{aligned} \quad (3)$$

e_t, δ_t represent error signal at time t at output layer and hidden layer respectively. However, this naive implementation can cause gradient explosion or vanishing phenomenon.

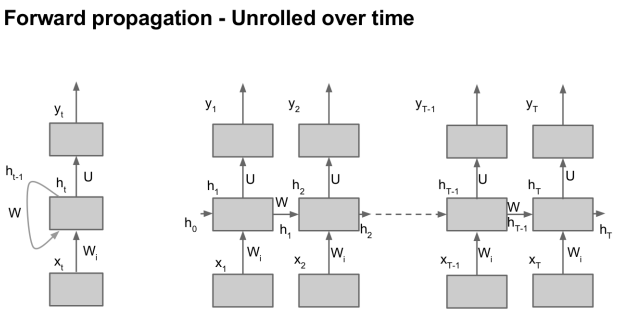

Figure 1: Elman RNN architecture.

3.2. Gated RNNs

Typically, the RNN is implemented using either long short-term memory (LSTM) or gated-recurrent units as RNN units [5] [6] [14], combinedly referred to as gated RNNs (GRN), to avoid the vanishing gradients problem [15]. However, there are two reasons for simple RNNs to be chosen over GRN: (1) The number of parameters in GRN are far more for a given hidden state size compared to simple RNN. Also the number of computational steps are much higher as the recurrent unit has more gates and (2) it is not clear as to what components of the complex architecture play a critical role.

Recently, in [16][17][18], experiments have been conducted to determine what components are contributing most to the performance. Through experimental results on various datasets and task ranging in different domains, it has been indicated that forget gate is the most important of all the gates in [16]. In [17], an architecture search was conducted to determine whether a better architecture than standard LSTM can be found. Experimental results concluded that with correct setting of forget gate bias the original LSTM formulation is uniformly better across all the tasks considered in that paper. More specifically, in the case of SPSS the relative importance of gates has been studied in [14] and a simplified gated recurrent unit optimal for SPSS has been proposed. However, whether gating at all is required or not has not been studied so far in the literature.

Owing to the above cited reasons we prefer to use simple RNNs. In [19], we have shown that simple RNNs can be successfully trained for the task of SPSS using sparse initialization technique. In this work, we use a more advanced form of sparse initialization namely "diagonal initialization" proposed in [20] for recurrent weight matrix initialization.

4. Experiments

In this section we describe the database used and experimental setup for building SPSS systems.

4.1. Database

We use a subset of the Blizzard challenge 2015 database for our experiments. The Blizzard challenge 2015 database contains about 4 hours of speech data in each of three Indian languages (Hindi, Tamil and Telugu), and about 2 hours of speech data in each of other three Indian languages (Marathi, Bengali and Malayalam), all recorded by native professional speakers in high quality studio environments. We used Telugu and Hindi language datasets for our experiments. The speech recordings released were sampled at 16kHz. Phone-level alignments were performed using the EHMM tool [21].

4.2. Experimental Setup

50 dimensional Mel-general cepstral (MGC) features and 26 dimensional band-aperiodicities (BAP) were extracted with a frame-shift of 5 ms for all the speech utterances along with their deltas and double-deltas. This feature extraction is followed from HTS-STRAIGHT demo available online. During synthesis natural durations were used.

We use DNN with conventional text features as our baseline system. Note that since we train a DNN separately for spectrum, f_0 and aperiodicity (i.e., DNN-MGC, DNN-f_0 and DNN-BAP respectively) DNN below implies the combination of these three DNN systems. For RNN, the past context was removed and only the future context was retained. Both the input and output features were mean variance normalized. Fig. 2 shows the block diagram for DNN/RNN based SPSS. The architectures of DNN and RNN used are given in Table 1.

For DNNs we use normalized initialization [22], and for RNNs we use the *diagonal initialization* proposed in [20]. In *diagonal initialization*, the recurrent weight matrices are initialized to scaled Identity matrices. This gives the benefit of orthogonality as well as the ability to set the spectral radius (i.e., the maximum eigen value) for stability. The scale of the recurrent Identity weight matrix was very important for learning and was set to 0.01 in all our experiments. Setting the scale to a lower value will make RNN not memorize anything at all and a higher scale will lead to explosion gradients or memorizing more than necessary thereby leading to a poorer local optimum. All the non-recurrent weight values were drawn from random Gaussian distribution. All the non-linearities used are rectified linear units (ReLU) [23].

4.3. Proposed Method

The proposed method for synthesis is shown in Fig. 3. Firstly we train an RNN as a CR system with conventional text features as input and either spectrum or f_0 as the output features. Then subsequently for training the synthesis system, we append the conventional text features with the hidden state from the trained RNN. The

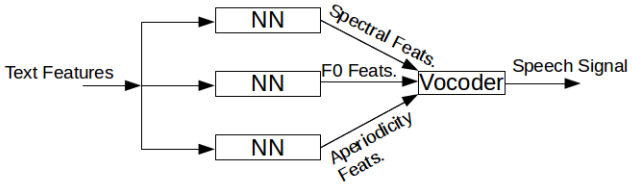

Figure 2: Block diagram of DNN/RNN based SPSS system.

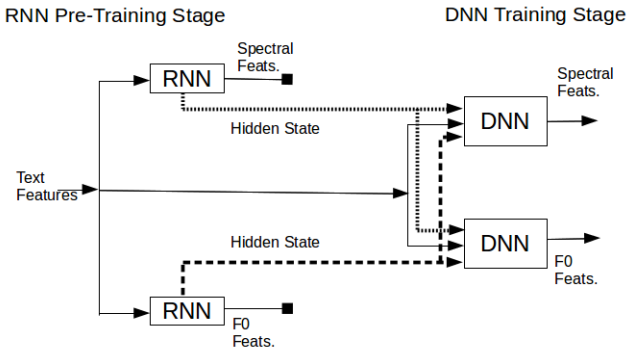

Figure 3: Block diagram of DNN with RNN hidden state as auxiliary context text features.

hidden state can either be from RNN-MGC or RNN-f_0 or from both networks. The resulting DNN is trained for predicting acoustic features stream-wise. In this study, we have restricted to using RNN-MGC network as CR network. Note here that CR networks are not used for predicting BAP features. Instead, the aperiodicities predicted from the baseline DNN system were used during synthesis. During the pilot experiments we found that BAPs predicted from RNN or from a hybrid network did not improve over using the BAPs from baseline DNN.

4.3.1. Normalization of Hidden State

The hidden state vectors were appended to the conventional text features in two ways: (1) mean and variance normalization was applied and (2) without any normalization. This was done primarily to study the effect of normalization on the appended features.

4.4. System Description

A brief description of systems trained is given below:

- DNN: Baseline system with conventional text features (DNN-MGC,DNN-f_0,DNN-BAP)
- RNN: RNN system trained for predicting spectral features (RNN-MGC,RNN-f_0,RNN-BAP)
- DNN-RNN-MGC-N: DNN trained with appending hidden state of RNN-MGC network with hidden state mean and variance normalized

Table 1: DNN and RNN architectures. The 'R' below indicates ReLU non-linearity.

Architecture	# Layers
DNN-MGC	500R 500R 500R
DNN-f_0	500R 100R
DNN-BAP	500R 500R
RNN-MGC	500R
RNN-f_0	300R
RNN-BAP	300R
DNN-MGC/RNN	1000R500R
DNN-f_0/RNN	1000R100R

- DNN-RNN-MGC: DNN trained with appending hidden state of RNN-MGC network without hidden state mean and variance normalized

The code for replicating the experiments can be found online [1].

Table 2: Performance of Telugu SPSS systems in terms of objective metrics.

System	MCD (dB)	f_0 (RMSE)	VUV (% error)	BAP (dB)
DNN	4.36	33.93	6.8	**23.84**
RNN	4.39	33.32	6.67	26.57
DNN-RNN-MGC-N	4.29	32.81	7.12	–
DNN-RNN-MGC	**4.24**	**31.12**	**6.64**	–

Table 3: Performance of Hindi SPSS systems in terms of objective metrics.

System	MCD (dB)	f_0 (RMSE)	VUV (% error)	BAP (dB)
DNN	4.11	17.12	8.04	**20.27**
RNN	4.11	16.85	7.9	23.14
DNN-RNN-MGC-N	3.99	17.97	7.69	–
DNN-RNN-MGC	**3.98**	**16.3**	**7.66**	–

5. Results

In this section we describe the objective and subjective evaluations of baseline and our proposed SPSS systems.

5.1. Objective Evaluation

The objective measures include Mel-cepstral distortion (MCD) for spectrum, RMSE for f_0, percentage error in frames for voiced/unvoiced and euclidean distortion for band-aperiodicity. In Tables 2 and 3 one can see that RNN improves f_0 modeling over the baseline DNN. Whilst, in case of spectral parameter prediction both DNN and RNN perform equally well as can be seen from the MCD scores. During the calculation of MCD scores spectral enhancement using global variance was not applied but was done before subjective listening tests. However, MLPG smoothing was applied for all the parameter tracks.

As for the effect of CR networks, we can see that appending RNN-MGC hidden state improves all the objective measures for both the languages. However the improvement for RNN-MGC is better when the hidden state features are left un-normalized. This result is contrary to expectation, since we would expect the DNN to optimize better if the input features are all normalized. Fig. 4 shows the Mel-general cepstral trajectories of 2 through 5 coefficients as predicted from various systems along with the natural trajectory. It can be seen that the DNN-RNN-MGC system is able to better approximate the natural contours especially at the phone-boundaries (for instance approximately at 3.8 sec, 4 sec and 7.3 sec in first panel in Fig. 4). The rnn-dnn in the legend (Fig. 4) implies the DNN-RNN-MGC system.

5.2. Subjective Evaluation

We have conducted MUSHRA (MUltiple Stimuli with Hidden Reference and Anchor) subjective listening tests with 10 listeners for Telugu language. There is reference speech file which is the natural speech signal and it is hidden amongst the test cases. In total, each subject had to listen to 20 samples synthesized from the held out test set. The listener had to rate how natural sounding the test system is with respect to the reference. The scale was from 0 to 100 (0 implying very unnatural and 100 for highly natural synthesis). The subject was asked to give maximum rating to atleast one of the 5 test wavefiles which sounded closest to the reference file.

Fig. 5 shows the mean opinion scores (MOS) of various systems. The results clearly indicate that the subjects prefer the DNN-RNN-MGC system the most for Telugu. From the informal listening tests, we found that the sounds in general and consonantal clusters like *tra* in particular, were more clearly intelligible when using the additional context from the CR networks. This could have contributed to the preference of the proposed systems. We could not conduct the subjective listening tests for Hindi and hence only objective results are given.

Samples can be heard online [2].

6. Conclusions

In this paper, we have proposed to use the RNN hidden state as the contextual feature for improving the DNN based synthesis. The proposed method was tested on a dataset consisting of 2 Indian languages from Blizzard

[1] https://github.com/SivanandAchanta/SSW9

[2] http://goo.gl/7l1Nwo

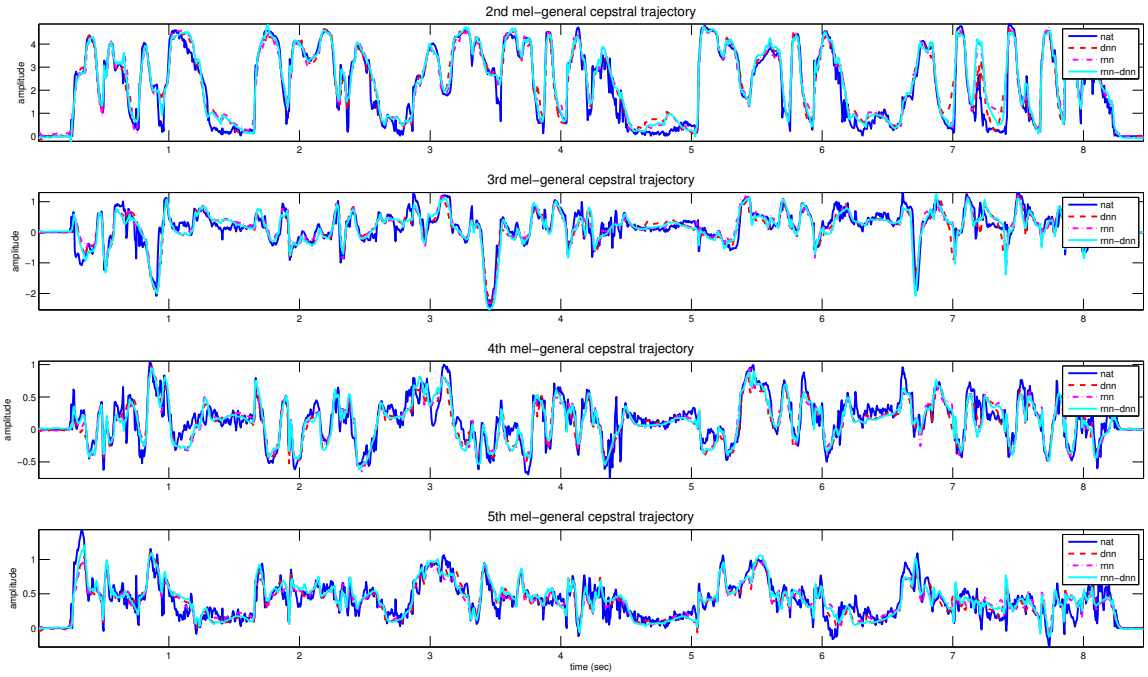

Figure 4: Mel-general cepstral trajectories of 2nd, 3rd, 4th and 5th coefficients from various systems (Best viewed in color).

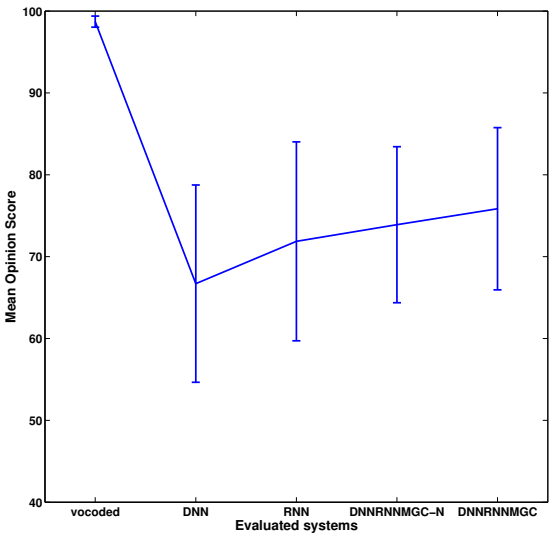

Figure 5: Subjective listening test results for Telugu.

challenge 2015 data. It is clear from both subjective listening test and objective scores that adding the contextual features substantially improves over the baseline. From the informal subjective listening tests the improvements seem to come from better intelligibility of difficult sounds which is quite encouraging for pursuing this line of thought further.

7. Future Work

Training an RNN using multi-lingual data for contextual feature extraction is one direction. Also, in this work we have limited to using uni-directional RNN however taking future context may also prove to be helpful in predicting spectrum and f_0 and hence Bi-RNNs have to be explored. While this work was limited to using CR network trained on MGCs alone, the effect of appending features from a CR network trained on f_0 needs to be studied. An exploration of training the CR network with combined acoustic features will also be carried out. The synthesis network can also be another RNN instead of DNN as used in this work. We also plan to conduct extensive listening tests for all Telugu as well as Hindi.

8. Acknowledgements

The first and last authors like to acknowledge TCS for partially funding the first author for his PhD. Also we would like to thank all the participants who have enthusiastically participated in the listening tests.

9. References

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

[2] S. King, "Measuring a decade of progress in text-to-speech," *Loquens*, vol. 1, no. 1, p. e006, 2014.

[3] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, and T. N. Sainath, "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] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Proc. ICASSP*, 2013, pp. 7962–7966.

[5] 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*, 2015, pp. 4470–4474.

[6] Y. Fan, Y. Qian, F.-L. Xie, and F. K. Soong, "TTS Synthesis with Bidirectional LSTM Based Recurrent Neural Networks," in *Proc. INTERSPEECH*, 2014, pp. 1964–1968.

[7] Z. Wu and S. King, "Minimum trajectory error training for deep neural networks, combined with stacked bottleneck features," in *Proc. INTERSPEECH*, 2015, pp. 309–313.

[8] K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda, "Trajectory training considering global variance for speech synthesis based on neural networks," in *Proc. ICASSP*, 2016, pp. 5600–5604.

[9] O. Watts, Z. Wu, and S. King, "Sentence-level control vectors for deep neural network speech synthesis," in *Proc. INTERSPEECH*, 2015, pp. 2217–2221.

[10] H. Lu, S. King, and O. Watts, "Combining a vector space representation of linguistic context with a deep neural network for text-to-speech synthesis," in *8th ISCA Workshop on Speech Synthesis*, Barcelona, Spain, Aug 2013, pp. 281–285.

[11] Z. Wu and S. King, "Improving trajectory modelling for dnn-based speech synthesis by using stacked bottleneck features and minimum generation error training," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. PP, no. 99, pp. 1–1, 2016.

[12] 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*, April 2015, pp. 4460–4464.

[13] R. J. Williams and J. Peng, "An Efficient Gradient-Based Algorithm for On-line Training of Recurrent Network Trajectories," *Neural Computation*, vol. 2, pp. 490–501, 1990.

[14] Z. Wu and S. King, "Investigating gated recurrent neural networks for speech synthesis," *arXiv preprint arXiv:1601.02539*, 2016.

[15] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[16] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *arXiv preprint arXiv:1503.04069*, 2015.

[17] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *Proc. ICML*, 2015, pp. 2342–2350.

[18] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014. [Online]. Available: http://arxiv.org/abs/1412.3555

[19] S. Achanta, T. Godambe, and S. V. Gangashetty, "An investigation of recurrent neural network architectures for statistical parametric speech synthesis," in *Proc. INTERSPEECH*, 2015, pp. 2524–2528.

[20] Q. V. Le, N. Jaitly, and G. E. Hinton, "A simple way to initialize recurrent networks of rectified linear units," *arXiv preprint arXiv:1504.00941*, 2015.

[21] K. Prahallad, "Automatic building of synthetic voices from audio books," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, USA, 2010.

[22] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. International conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[23] M. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q. Le, P. Nguyen, A. Senior, V. Vanhoucke, J. Dean, and G. Hinton, "On rectified linear units for speech processing," in *Proc. ICASSP*, May 2013, pp. 3517–3521.