INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

A Hierarchical Encoder-Decoder Model for Statistical Parametric Speech Synthesis

Srikanth Ronanki, Oliver Watts, Simon King

The Centre for Speech Technology Research (CSTR), University of Edinburgh, UK
srikanth.ronanki@ed.ac.uk

Abstract

Current approaches to statistical parametric speech synthesis using Neural Networks generally require input at the same temporal resolution as the output, typically a frame every 5ms, or in some cases at waveform sampling rate. It is therefore necessary to fabricate highly-redundant frame-level (or sample-level) linguistic features at the input. This paper proposes the use of a hierarchical encoder-decoder model to perform the sequence-to-sequence regression in a way that takes the input linguistic features at their original timescales, and preserves the relationships between words, syllables and phones. The proposed model is designed to make more effective use of supra-segmental features than conventional architectures, as well as being computationally efficient. Experiments were conducted on prosodically-varied audiobook material because the use of supra-segmental features is thought to be particularly important in this case. Both objective measures and results from subjective listening tests, which asked listeners to focus on prosody, show that the proposed method performs significantly better than a conventional architecture that requires the linguistic input to be at the acoustic frame rate.

We provide code and a recipe to enable our system to be reproduced using the Merlin toolkit.

Index Terms: Speech synthesis, prosody modeling, hierarchical models and sequence modeling.

1. Introduction

Statistical parametric speech synthesis (SPSS) continues to make significant improvements every year, yet is rarely convincingly natural [1]. In this paper, we focus on the absence of variation in the predicted intonation, which makes synthetic speech dull and therefore inappropriate for a range of appealing applications, such as audiobooks or interactive systems. We propose a new hierarchical encoder-decoder model which aims to improve intonation modelling, leading to improvements in overall synthesized speech naturalness.

1.1. Relation to prior work

Neural networks are rapidly coming to dominate SPSS, for both duration and acoustic modelling (including F0). Zen et al. [2] re-introduced neural networks for speech synthesis. In that paper, a Deep Neural Network (DNN) was used as a fairly straightforward drop-in replacement for the usual regression tree of HMM-based SPSS, performing the complex mapping (i.e., regression) from linguistic features to speech parameters on a frame-by-frame basis. Later, others [3, 4, for example] explored variants on this basic idea. Given that speech is a time series, and SPSS is a sequence-to-sequence regression problem, it was then natural to apply various recurrent architectures, including Long Short-Term Memory (LSTM) units and Gated Recurrent Units (GRUs) [5, 6, 7, 8, for example], which we can group under the general category of Recurrent Neural Networks (RNNs). Nevertheless, these approaches still all require input at the same frame-rate as the output, which is determined by the needs of the vocoder.

Focusing on intonation, DNNs offer improvements over regression trees [9, 10] and RNNs improve over DNNs [11, 12]. It seems likely that long-range dependencies are of greater importance in predicting intonation than when predicting the spectral envelope. By "long-range" we mean over sequences of linguistic units such as syllables, words and phrases This suggests the use of a hierarchical model that directly exploits such linguistic structure, instead of merely "flattening" it, first on to the segment (phone) and then further to the acoustic frame.

We identify two potential limitations of the usual approach taken in prior work [2, 4]. First, the use of frame-level inputs is highly redundant because the majority of input features remain constant for 10-100 consecutive frames, incurring unnecessary computations both in training and when performing synthesis. Second, representing supra-segmental features associated with syllables, words or phrases, at segmental or sub-segmental timescales may actually hinder the model from learning to use such features [13].

There are other recent attempts at hierarchical modelling, although only of F0 contours. [14] proposes two different model structures – cascade and parallel DNNs – to embody hierarchical and additive properties. [10] also proposed parallel and cascaded DNNs to model supra-segmental and segmental features separately. Both approaches use multiple networks, incurring additional computational cost compared to non-hierarchical approaches.

1.2. Novelty of this work

We present a hierarchical framework which exploits linguistic structure in order to model long-term dependencies, whilst at the same time being computationally efficient.

New techniques have recently emerged for mapping between two sequences of arbitrary lengths and potentially unknown alignment, including sequence-to-sequence neural networks [15], attention-based models [16], and the sequence transduction networks [17] that we investigate in this work.

We combine a hierarchical *encoder* to model linguistic structure at multiple time-scales, with a *decoder* that predicts speech parameters for each output acoustic frame.

Through experiments, we first compare our proposed hierarchical encoder-decoder (HED) architecture with the usual approach. We then examine the effectiveness of supra-segmental word-level features, when added to a standard set of linguistic features, comparing conventional frame-level combination with their use directly at the word level in our proposed hierarchical encoder.

http://dx.doi.org/10.21437/Interspeech.2017-628

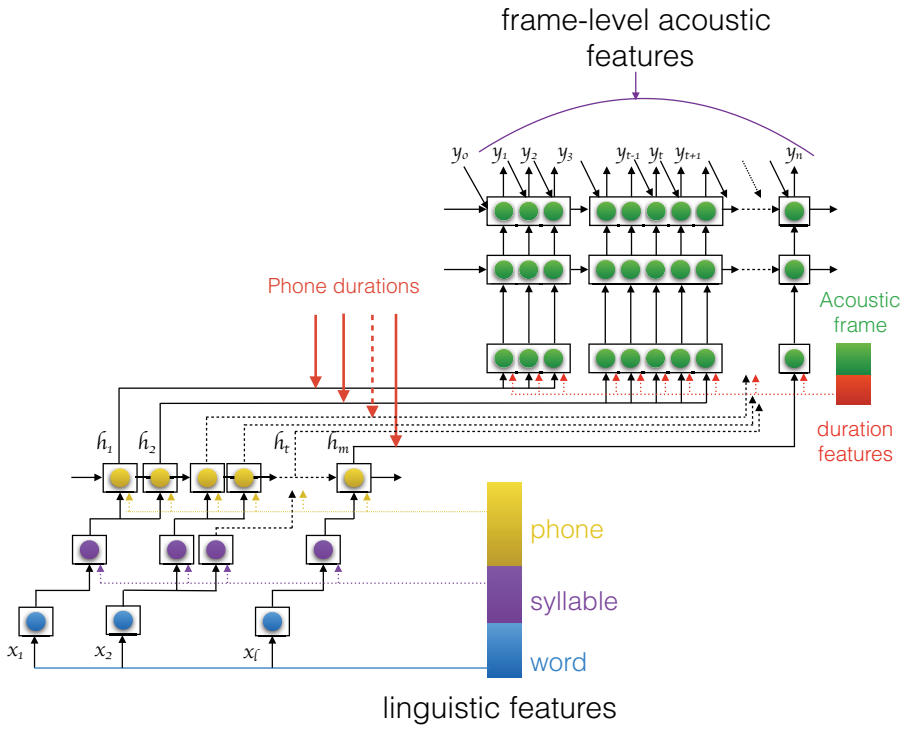

Figure 1: *Schematic diagram of a hierarchical encoder-decoder for SPSS. The lower part of the network is the hierarchical encoder, with each layer operating at a particular linguistic level, and phone-level recurrence as its final encoded output. The upper part of the network is the decoder, generating speech parameters using frame-level recurrence. Solid black lines indicate the propagation of hidden activations between layers, and dashed colored lines indicate the injection of linguistic features at the appropriate level. The patterns of connections between word, syllable and phone layers is determined by the known linguistic structure of the current utterance. Each block of green units represents a phone, with the number of units corresponding to its duration in frames (although not drawn to scale).*

2. The Usual Approach

2.1. Pre-processing input features

The alignments between words, syllables, and phones are given by the linguistic specification, provided by the TTS front end. The usual approach in neural network-based TTS – regardless of neural network architecture – is to pre-process the input representation by **flattening** followed by **upsampling** [2, 4].

Flattening: attaching linguistic features to the phone, creating a linear sequence of context-dependent phones, and discarding explicit structure (e.g., that phones belong to syllables).

Upsampling: duplicating linguistic features for a number of consecutive acoustic frames, to map from linguistic timescale to vocoder frame rate (or possibly to waveform sampling rate, if directly generating a waveform). Note that upsampling cannot *add* information; in fact, it results in the same amount of information being represented less efficiently.

It is common practice to add within-phone positional features, derived from existing features, when upsampling to compensate for limitations of the regression model.

2.2. Input-to-output alignment in the usual approach

By making an assumption that is almost universal in speech technology – viz. that a speech signal is a sequence of non-overlapping units – the input-output alignment can be pre-computed using HMM-based forced alignment for the training data, and can be determined during synthesis of test utterances using a duration model learned from the same data.

2.3. Regression

There are many possibilities for the architecture of the network used to perform regression from flattened-and-upsampled linguistic features to either frame-level vocoder speech parameters [2, 7, 9, for example] or to waveform samples [18]. Nevertheless, what all of these architectures have in common is the requirement for input and output to be at the same rate, meaning that the input must be upsampled.

3. Proposed Hierarchical Encoder-Decoder

Figure 1 provides a schematic diagram of the proposed hierarchical encoder-decoder neural network. The key ideas are to avoid the flattening pre-processing step entirely, and to integrate the upsampling into the model itself.

3.1. Hierarchical encoder

Figure 1 shows how the proposed method employs a hierarchical encoder that accepts input at the original linguistic timescales of word, syllable and phone. The upsampling between these levels is performed progressively, rather than all at once. Features at each level are injected into the model at the appropriate timescale by appending them to the hidden representation at that level. This has a variety of possible advantages over the usual approach.

Features from longer timescales (e.g., word), have a potentially weakened or diluted effect in the usual approach because they are constant for many consecutive frames, yet must share

the same projections (hidden layers) as features that are locally more predictive of the output. In the proposed encoder architecture, there is a hierarchy of projections (layers) that introduces information from one linguistic timescale at a time.

The proposed architecture substantially reduces the number of inputs because the upsampling takes places within the model architecture rather than the pre-processing of features in the usual approach. This reduces the number of model parameters and consequently the computational cost.

3.2. Recurrent decoder

The output from the hierarchical encoder is at phonetic timescale (h_t in Figure 1). Using externally-provided duration information, this is upsampled to the acoustic frame rate used by the vocoder and then augmented by appending frame-level features derived from duration.

When training the recurrent decoder, past output (y_{t-1} in Figure 1) can either be ground-truth from the training data (a method known as "teacher forcing" [19]) or the prediction of the network itself. y_0 is initialized as the zero vector.

3.3. Input-to-output alignment

In machine translation, where encoder-decoder architectures are more commonly used [15, 20], alignment between input and output sequences is non-trivial: it may be non-monotonic because of word re-ordering. The situation in speech synthesis is more straightforward because the alignment is strictly monotonic. However, unlike machine translation, in speech synthesis the input and output sequences are at different timescales.

In the work presented here, we employ methods from the usual approach for neural network-based TTS (Section 2.2): forced alignment during training, and externally-provided durations for synthesis. This is done to measure the contribution of our proposed architecture independently of issues of duration. However, upsampling is done progressively within the model hierarchy.

Integration of alignment (during training) [21] or duration prediction (for synthesis) [22] are possible extensions in future work, and the proposed encoder-decoder architecture may offer ways to do this that are not available in the usual approach.

3.4. Additional word-level features

In addition to the usual linguistic and positional features, other features can be derived from text that may improve the naturalness of generated speech. Embeddings [23] are one of the most promising and generally-applicable ways to derive new features from text. For example, bottleneck features [4] (a kind of supervised embedding) work well for TTS.

Unsupervised word embeddings [13], learned only from text, are another promising idea. Unlike bottleneck features, which are extracted at frame level, word embeddings are derived for each word. As discussed earlier, upsampling word-level features to the vocoder frame rate is rather unsatisfactory and can be ineffective for learning. In the proposed hierarchical encoder-decoder architecture it is simple to input these features directly at the word level, without upsampling.

4. Experimental Setup

4.1. Data

We tested our proposed method on the speech database released for the 2016 Blizzard Challenge [24] consisting of the written and spoken versions of 50 children's books. The speaker is British and female. We used a segmentation of the audiobooks generously made available by a participant[1] from Blizzard Challenge workshop 2016. The total duration of the audio after segmentation is approximately 4.33 hours. For present purposes, 4% of this training data were set aside for testing. Our test set consists of three entire stories: *Goldilocks and the Three Bears*, *The Boy Who Cried Wolf* and *The Enormous Turnip*, having a total combined duration of approximately 10 minutes.

4.2. Feature extraction

Phone sequences were obtained from the text using Festival [25]. Festvox's `ehmm` method [26] was used to modify the phone sequences by the insertion of acoustically-motivated pauses; a forced alignment of these phone sequences with the sentence-segmented audio was then obtained using context-independent HMMs. Each phone was then characterised by a vector of 481 text-derived binary and numerical features – a subset of the features used as decision-tree clustering questions in the HTS demo [27], adapted for our phoneset.

These questions included linguistic contexts such as quinphone identity which are added at phone-level, and part-of-speech, positional information relating to syllables, words, phrases, *etc*. which are given as input at syllable and word-level. All numerical features are input (after appropriate normalisation) directly to the network, and not encoded as (for example) 1-of-K. Similar to [2], three numerical features for coarse-coded position of the current frame in the current phoneme and duration are computed and appended as frame-level additional features. All inputs were normalised to the range $[0.01,\ 0.99]$.

Word embeddings and word-quotation features were used as additional word-level features in some of the systems (with *WF* in the system name). We used 300-dimensional word embeddings[2] obtained as described in [28]. We also added a word-quotation feature, to words within double quotes; this generally indicates direct speech vs. narration, in the data we used.

The speech data was analysed with STRAIGHT [29], and each 5ms frame was represented using 60 mel cepstral coefficients (MCC), measures of aperiodicity in 25 frequency bands (BAP), logarithmic F_0 interpolated through unvoiced regions, and a binary voicing feature. These 87 static features were supplemented with delta and delta-delta features, and per-component mean and variance normalisation was performed.

4.3. System training

Four systems were trained: identifiers listed in the left-hand column of Table 1. *Frame-LSTM* is a baseline system typical of the usual method. It was configured with two feed-forward layers of 1024 nodes and three uni-directional simplified long short-term memory (SLSTM) [7] hidden layers consisting of 512 nodes – which provide frame-rate recurrence – and a final linear output layer. It is trained to regress directly from a sequence of flattened-and-upsampled frame-level linguistic feature vectors to a sequence of vocoder speech parameters.

HED-LSTM is a hierarchical encoder-decoder which implements our proposed idea. The encoder was configured with five feed-forward layers of 1024 nodes each and a uni-directional SLSTM. As shown in figure 1, the decoder has one hidden recurrent layer and an output layer whose predicted output at time

[1] `https://www.innoetics.com`

[2] We made use of the embeddings from 6B tokens made available at `http://nlp.stanford.edu/projects/glove/`

Phonetic class	Spectral		F0 Measure		
	MCD	BAPD	RMSE	Corr.	V/UV
Frame-LSTM	5.44	0.075	51.85	0.432	5.49%
HED-LSTM	5.48	0.074	50.20	0.453	5.48%
Frame-LSTM + WF	5.50	0.075	50.22	0.454	5.44%
HED-LSTM + WF	5.53	0.075	49.68	0.473	5.47%

Table 1: *Objective results.*

t is given as additional input for predicting frame $t+1$. Both layers have 512 uni-directional SLSTM units.

Two variant systems were built to test whether the proposed hierarchical encoder can better exploit word-level features than the usual approach. *Frame-LSTM+WF* and *HED-LSTM+WF* are identical to *Frame-LSTM* and *HED-LSTM* respectively, except that distributional word representations were added (Section 4.2). In *Frame-LSTM+WF* these were added to every frame of input, whereas in the hierarchical encoder-decoder *HED-LSTM+WF*, they were added at the word level, with other word-level linguistic and positional features.

All networks were initialised using small random weights; no pre-training was used. Each system was trained with a fixed learning rate, manually tuned to yield close-to-optimal results on the development set in 25 epochs or less. Early stopping was used to avoid overfitting, by aborting training once the objective function on the development set had failed to improve for five consecutive epochs.

4.4. Synthesis

At synthesis time, `ehmm` phone sequences derived from the test data were used as input to each model. This corresponds to an oracle pausing strategy. For all systems, natural durations were used during testing, since we are interested only in regression from linguistic features to speech parameters. In the proposed hierarchical encoder-decoder systems, durations are used when upsampling from phones to acoustic frames at the interface between encoder and decoder. Replacing oracle durations with predictions by an external duration model would be trivial.

In all systems, maximum likelihood parameter generation (MLPG) [30] using variances computed from the training data was applied to output features; spectral enhancement post-filtering was applied to the resulting MCC trajectories [4]. The STRAIGHT vocoder [29] was used to synthesize the waveform which is then normalised according to ITU P.56 [31].

5. Results

5.1. Objective measures

We objectively compared the speech parameters generated by each system for the test set against held out natural examples. We calculated Mel cepstral distortion (MCD), band aperiodicity distortion (BAPD), and root mean square error of F_0 (RMSE), Pearson correlation F_0 (Corr.) and voiced/unvoiced error rate (V/UV). Results are shown in Table 1.

Although *HED-LSTM* slightly increases MCD from 5.44 dB to 5.48 dB compared to *Frame-LSTM*, it makes more accurate predictions of F_0: RMSE dropped from 51.85 Hz to 50.20 Hz and Corr. increased from 0.432 to 0.453. When word-level features are added, both frame-level and hierarchical systems' F_0 predictions improved, while the MCD deteriorated slightly.

Usefully, *HED-LSTM* is computationally cheap, both in training and generation, compared to *Frame-LSTM*. We computed generation time for both the systems, which is the total time to generate all the 387 utterances in both development and testing sets. The generation time for *Frame-LSTM* with two feed-forward layers and three recurrent layers is 1910 seconds while *HED-LSTM* with same architecture took 1566 seconds.

5.2. Subjective evaluation

We assessed the naturalness of the synthesised speech via pairwise preference listening tests. Three pairs of systems were considered: *Frame-LSTM* vs. *HED-LSTM*, *Frame-LSTM* vs. *HED-LSTM+WF*, and *HED-LSTM* vs. *HED-LSTM+WF*. 16 paid native English speakers with no known hearing impairments, participated in the listening test. Each listener was asked to listen to 75 randomly selected pairs. Thus each condition received a total of 400 judgements(16*25). Listeners were told that the synthesized audio files were from children's story books. within each pair, the listener was played two synthesised versions of the same sentence and asked to choose which one sounded more natural. They were asked to focusing on prosody (tune) and only to select the "no preference" option when both sounded the same.

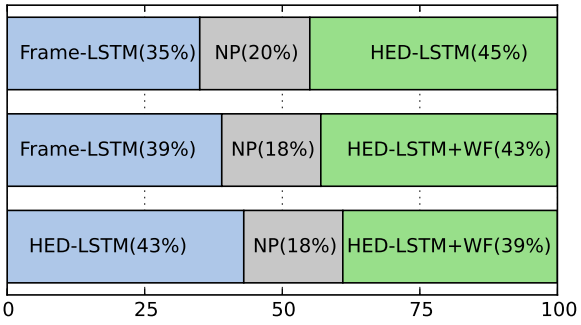

Figure 2: *Preference test results for naturalness with percentage in brackets, where NP denotes "no preference"*

Results of the preference test are presented in Figure 2. First, examine the effectiveness of the proposed hierarchical encoder-decoder: *HED-LSTM* sounds significantly more natural than *Frame-LSTM* with $\alpha = 0.05$ under a binomial test with an expected 50% split. *HED-LSTM+WF* is slightly, but not significantly preferred over *Frame-LSTM*.

6. Conclusion

We have described a hierarchical encoder-decoder to improve intonation modeling in an effective and computationally-efficient way. Unlike the usual approach – which requires linguistic feature flattening and upsampling – our proposed encoder accepts linguistic features at their natural timescales, and performs upsampling within the model architecture.

Objective results show an improvement in F0 measure and subjective results show that the proposed method performs significantly better than a strong baseline. Future work includes integration of duration prediction during synthesis, and perhaps alignment during training. It would also be possible to replace the decoder with direct waveform generation as in [32].

Reproducibility: We used the Open Source Merlin toolkit [33]. URL: `https://github.com/CSTR-Edinburgh/merlin/`

Acknowledgement: Watts and King were supported in this research by EPSRC Standard Research Grant EP/P011586/1, *Speech Synthesis for Spoken Content Production (SCRIPT)*.

7. References

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

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

[3] Y. Qian, Y. Fan, W. Hu, and F. K. Soong, "On the training aspects of deep neural network (DNN) for parametric TTS synthesis," in *Proc. ICASSP*. IEEE, 2014, pp. 3829–3833.

[4] 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*, 2015, pp. 4460–4464.

[5] 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.

[6] 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.

[7] Z. Wu and S. King, "Investigating gated recurrent networks for speech synthesis," in *Proc. ICASSP*, 2016, pp. 5140–5144.

[8] H. Zen, Y. Agiomyrgiannakis, N. Egberts, F. Henderson, and P. Szczepaniak, "Fast, compact, and high quality LSTM-RNN based statistical parametric speech synthesizers for mobile devices," in *Proc. Interspeech*, 2016.

[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] M. S. Ribeiro, O. Watts, and J. Yamagishi, "Parallel and cascaded deep neural networks for text-to-speech synthesis," in *Proc. SSW*, 2016, pp. 107–112.

[11] R. Fernandez, A. Rendel, B. Ramabhadran, and R. Hoory, "Prosody contour prediction with long short-term memory, bi-directional, deep recurrent neural networks," in *Proc. Interspeech*, 2014, pp. 2268–2272.

[12] S. Ronanki, G. E. Henter, Z. Wu, and S. King, "A template-based approach for speech synthesis intonation generation using LSTMs," in *Proc. Interspeech*, 2016.

[13] P. Wang, Y. Qian, F. K. Soong, L. He, and H. Zhao, "Word embedding for recurrent neural network based TTS synthesis," in *Proc. ICASSP*, April 2015, pp. 4879–4883.

[14] X. Yin, M. Lei, Y. Qian, F. K. Soong, L. He, Z.-H. Ling, and L.-R. Dai, "Modeling F0 trajectories in hierarchically structured deep neural networks," *Speech Commun.*, vol. 76, pp. 82–92, 2016.

[15] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[16] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. ICASSP*, 2016, pp. 4960–4964.

[17] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711 (not peer reviewed)*, 2012.

[18] 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," *ArXiV (not peer reviewed)*, 2016.

[19] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks." *ICML (3)*, vol. 28, pp. 1310–1318, 2013.

[20] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[21] W. Wang, S. Xu, and B. Xu, "First step towards end-to-end parametric TTS synthesis: Generating spectral parameters with neural attention," in *Proc. Interspeech*, 2016, pp. 2243–2247.

[22] S. Ronanki, O. Watts, S. King, and G. E. Henter, "Median-based generation of synthetic speech durations using a non-parametric approach," in *IEEE workshop on Spoken Language Technology*, San Diego, California, 2016.

[23] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: A simple and general method for semi-supervised learning," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ser. ACL '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 384–394.

[24] S. King and V. Karaiskos, "The Blizzard Challenge 2016," in *Proceedings of Blizzard Challenge Workshop*, 2016.

[25] A. Black, P. Taylor, R. Caley, R. Clark, K. Richmond, S. King, V. Strom, and H. Zen, "The festival speech synthesis system, version 1.4. 2," *Unpublished document available via http://www. cstr. ed. ac. uk/projects/festival. html*, 2001.

[26] K. Prahallad, A. W. Black, and R. Mosur, "Sub-phonetic modeling for capturing pronunciation variations for conversational speech synthesis," in *Proc. ICASSP*, 2006, pp. I–853–I–856.

[27] H. Zen, T. Nose, J. Yamagishi, S. Sako, T. Masuko, A. Black, and K. Tokuda, "The HMM-based speech synthesis system (HTS) version 2.0," in *Proc. SSW*, vol. 6, 2007, pp. 294–299.

[28] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," in *Proc. EMNLP*, 2014, pp. 1532–1543.

[29] H. Kawahara, "STRAIGHT, exploitation of the other aspect of VOCODER: Perceptually isomorphic decomposition of speech sounds," *Acoust. Sci. Technol.*, vol. 27, no. 6, pp. 349–353, 2006.

[30] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," in *Proc. ICASSP*, vol. 3, 2000, pp. 1315–1318.

[31] *Objective measurement of active speech level*, ITU Recommendation ITU-T P.56, International Telecommunication Union, Telecommunication Standardization Sector, Geneva, Switzerland, March 2011.

[32] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "SampleRNN: An unconditional end-to-end neural audio generation model," in *Proc. ICLR*, Toulon, France, 2017.

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