Neural speech synthesis for resource-scarce languages

Johannes A Louw

Digital Audio-Visual Technologies Research Group,
Next Generation Enterprises and Institutions, CSIR,
Pretoria, South Africa
jalouw@csir.co.za

Abstract. Recent work in sequence-to-sequence neural networks with attention mechanisms, such as the Tacotron 2 and DCTTS architectures, have brought on substantial naturalness improvements in synthesised speech. These architectures require at least an order of magnitude more data than is generally available in resource-scarce language environments. In this paper we propose an efficient feed-forward deep neural network (DNN)-based acoustic model, using stacked bottleneck features, that together with the recently introduced LPCNet vocoder can be used in resource-scarce language environments, with corpora less than 1 hour in size, to build text-to-speech systems of high perceived naturalness. We compare traditional hidden Markov model (HMM)-based acoustic modelling for speech synthesis with the proposed architecture using the World and LPCNet vocoders, giving both objective and MUSHRA based subjective results, showing that the DNN LPCNet combination leads to more natural synthesised speech that can be confused with natural speech. The proposed acoustic model provides for an efficient implementation, with faster than real time synthesis.

Keywords: HMM · DNN · Speech synthesis · LPCNet · acoustic modelling · resource-scarce languages

1 Introduction

The advent of neural network-based text-to-speech (TTS) systems has brought on dramatic improvements in the naturalness and intelligibility of synthesized speech. The success of these architectures can be broadly attributed to the attention based models (such as Tacotron [23] and Deep Convolutional TTS (DCTTS) [17]) as well as the use of neural network based vocoders (such as WaveNet [22]) [24].

Traditional TTS architectures are usually based on a pipeline of a linguistic front-end and a waveform generation back-end. The newer deep neural network (DNN)-based architectures "learn" the linguistic front-end by operating directly on characters and learning embeddings from the characters which can then be used to extract higher level linguistic knowledge and features. One challenge

of this approach is the data requirements in that the traditional hand-crafted features need to be learned from the data. The baseline Tacotron system has been trained with 40 hours of text and audio pairs [3]. These end-to-end architectures also perform poorly on non-alphabetic-, or pitch accent and tone-based languages (for example Japanese and Chinese) in comparison to architectures with a traditional linguistic front-end [4]. Another major challenge of the end-to-end architectures are the computational requirements that need to be taken into consideration, for example the Tacotron 2 architecture [14] takes on average 234 hours[1] to train whilst the WaveGlow vocoder [14] takes on average 768 hours[1] to train.

For most world languages there are no high quality, single speaker, recorded corpora available which will satisfy the data requirements of end-to-end TTS systems. There are attempts at creating large corpora from found data, such as the *CMU Wilderness Multilingual Speech Dataset*[1], but these datasets are usually of a lesser quality in that there are multiple speakers and they may contain background noise.

In this work we aim to develop a DNN-based speech synthesis architecture that can at the very least exceed the quality of hidden Markov model (HMM)-based approaches as generally used for resource-scarce languages. The goal is an architecture with low computational costs (in terms of training and synthesis time) which can be trained with relatively low data requirements. Our focus is only on the acoustic modelling. The organisation of the paper is as follows: in Section 2 we give the two acoustic models used in this work, while in Section 3 we give the two vocoders used in this work. Section 4 details our experiments and results, and lastly a discussion and conclusion is presented in Section 5.

2 Statistical Parametric Speech Synthesis

Concatenative synthesis, where units of speech from a recorded database are concatenated to form the target utterance, had been the commercially dominant TTS technology since the late 1990s. The size of the recorded databases for commercial systems contained over 100 hours of recorded speech [2]. From 2005 on-wards *statistical parametric speech synthesis* (SPSS) systems (which include HMM- and later DNN-based systems) have been steadily adopted after the success shown by HMM-based systems [7]. A SPSS system can simply be described as a model that can generate speech parameters, given an input target specification, from a statistical model (usually learned) of said speech parameters.

In the next sections we briefly described the two models used for acoustic modelling in this work.

2.1 HMM-Based Synthesis

Most HMM-based synthesizer implementations in the literature are based on the *HMM-based Speech Synthesis System* (HTS) [33], which is in fact a hidden semi-

[1] `https://github.com/NVIDIA/DeepLearningExamples/tree/master/PyTorch/\SpeechSynthesis/Tacotron2#expected-training-time`

Markov model (HSMM) because an explicit duration model is used for state duration determination. The full working of the HTS HMM-based synthesizer is out of the scope of this paper, but interested readers can see [28] and [18] for a detailed handling of the subject. In the next section we will highlight the parts that are important to the context of this paper, framed in terms of the training and synthesis parts of an HTS HMM-based synthesizer.

Training Speech features are extracted by a vocoder. A linguistic description is generated by the TTS front-end, this describes each phoneme, syllable, word and phrase in terms of it's context within the utterance (the generally used set of descriptors are given in [18]). A baseline monophone model is estimated, after which this model is used to estimate a context-dependent HSMM (using the linguistic description). A decision tree is generated by clustering the different contexts and then the HMM states of the leaf nodes are shared (to overcome the problem of data sparsity due to the modelling of the full contexts). State durations are modelled by a multivariate Gaussian distribution from the aligned linguistic descriptions.

Synthesis The synthesis procedure is then as follows: the input target text is converted to a linguistic description by the TTS front-end. The state duration decision tree is traversed to get the state durations for each full context label as it is in the linguistic description. The speech parameter decision tree(s) are traversed to get the HMM states for the linguistic description, these are then concatenated to form the HMM target sequence.

Parameter Generation With the state durations and the HMM target sequence the parameters can be generated simply as the means and variances of each HMM state (most likely observation is the mean of the Gaussian in that state) with the duration specifying the number of frames generated per HMM state. The maximum likelihood parameter generation (MLPG) [19] algorithm is then used to smooth the generated parameter sequence between the frames. The smoothed generated parameters are then fed back into the vocoder in order to generate the speech waveform.

2.2 DNN-Based Synthesis

In this work our goal is to define an acoustic model which can be used for resource-scarce languages and with low computational overhead. We opted for a basic feed-forward neural network (FFNN) as was used in [32] but with added *stacked bottleneck* features as was implemented in [25]. The architecture is shown in figure 1.

The scheme consists of two acoustic models, one for predicting the bottleneck features (bottleneck model, on the left in Fig. 1) and one for prediction of the acoustic features (acoustic model, on the right in Fig. 1). The bottleneck model is

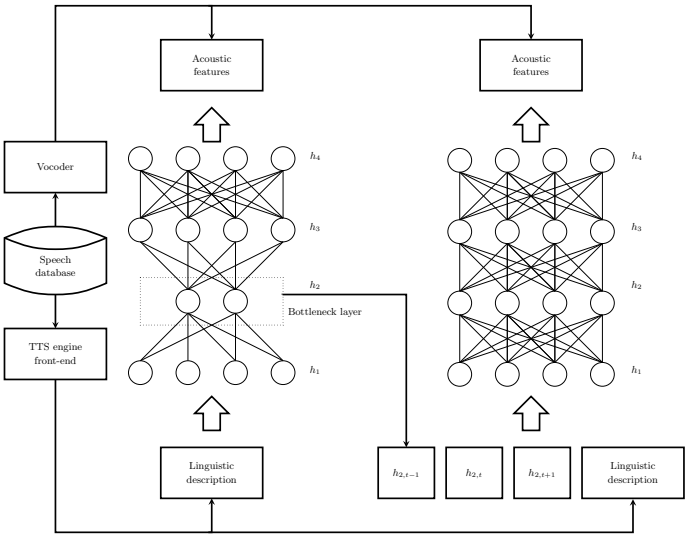

Fig. 1. A stacked bottleneck feed-forward acoustic model. On the left is the bottleneck model, and on the right the acoustic model.

trained as a normal FFNN, but with one bottleneck layer that forces the network to learn an efficient representation of the linguistic to acoustic mapping. During synthesis time the bottleneck model is used to generate a *linguistic-acoustic* context feature of the surrounding linguistic full context labels of the input to the acoustic model, thereby overcoming the context-independent frame modelling limitation of a FFNN. The network is forced to to take the surrounding contexts into account when predicting the acoustic features. The predicted features are also smoothed with the MLPG algorithm as is done for HMM-based synthesis (Section 2.1) before being passed to a vocoder for synthesis.

3 Vocoders

The parametric speech model in a SPSS system is represented by a vocoder (*voice + encoder*), which can extract parametric features describing the speech signal, which in turn are used by the SPSS system to statistically model the features. Different vocoders employ different parametric models of speech, but the important point is that it is reversible, i.e. the vocoder extracts the features, which are modelled by the SPSS system, which can generate new features based on an input target, which can then be converted back into speech by the vocoder.

In [34] Zen *et al.* identified the vocoding component of an SPSS system as a major factor that degrades the quality of synthesized speech. This is especially pronounced for resource-scarce corpora due to the inability of the acoustic model to learn a representative statistical distribution of the vocoded speech features because of the data sparsity problem.

In this work we compare the *World* [12] vocoder, the current state-of-the-art in terms of conventional digital signal processing (DSP) based vocoders, and the recently introduced efficient neural vocoder called *LPCNet* [20,21], in a resource-scarce setting. In the next sections we give a brief description of these two vocoders.

3.1 World

In the World vocoder speech is represented as a traditional source-filter model [16] for speech production, where for each frame of speech a *spectral envelope*, an *aperiodicity estimation* and an *excitation* (fundamental frequency f_0) are extracted. These three components combined give a complete parameterization of a frame of speech. The extraction and synthesis process happens with traditional DSP techniques.

3.2 LPCNet

LPCNet is a variant of WaveRNN [6], where instead of employing a neural network for the full speech spectrum (vocal source signal) modelling (like WaveRNN) it uses conventional signal processing. The spectral features are still predicted by a neural network. This change has a dramatic improvement in run-time computational requirements, being $5\times$ faster than real time on a 2.4 GHz Intel Broadwell architecture (whilst requiring only 20% of the CPU) [20]. The features extracted per frame of speech by LPCNet are: an *18-th order Bark-scale spectral envelope*, *pitch period* and *pitch correlation*. The neural network in the LPCNet vocoder can be trained on *found* data, thereby relieving a speech synthesis acoustic model of its stringent data requirements.

4 Experimental Setup

4.1 Data

The data used in this work is a *subset* of an in-house single speaker Afrikaans female TTS corpus of duration 12:08:15.89. The corpus was recorded in a professional studio at a 44.1 kHz sampling rate with 16 bits precision. The subset used are recordings of the text of the *Lwazi II Afrikaans TTS Corpus* [13], consisting of 763 utterances of duration 00:56:30.29. The utterances were randomly split into 715 utterances for training (00:53:12.09), 38 utterances for validation (00:02:37.99) and 10 utterances for testing (00:00:40.21). All audio was down-sampled to 16 kHz at 16 bits per sample and each utterance was normalised to the average power level of the subset (the 763 utterances).

Linguistic Descriptions The text annotations were tokenized and normalised with the Speect TTS engine front-end [10]. The linguistic descriptions of each utterance was also extracted using Speect. The set of linguistic descriptions were

the same as defined in [18], except for syllable stress, accent and ToBI (*Tones and Break Indices*) [15] tones which were not included due to it most probably not being available in resource-scarce settings.

Speect creates the pronunciation dictionaries which were used to force-align the linguistic descriptions and audio with the HTK toolkit [30] at a resolution of 10ms. A *silence* state was added between all words in order to capture any potential pauses in the recorded database which were not specifically annotated in the text with punctuation marks (this has been shown to improve the alignments [11], especially on small corpora). The alignments were done on a phone level for HMM-based synthesis and on a state level for DNN-based synthesis.

Acoustic Features Acoustic features for the corpus were extracted using the World and LPCNet vocoders. Both the World and LPCNet features were extracted at their default frame rates, which are 5ms and 10ms respectively. The spectral envelope extracted with the World vocoder was modelled with 60-th order (including energy) Mel-cepstral coefficients as follows:

$$H(z) = exp \sum_{m=0}^{M} c_\alpha(m))\tilde{z}^{-m} \tag{1}$$

where $\tilde{z}^{-1}$ is the first order all-pass function:

$$\tilde{z}^{-1} = \frac{z^{-1} - \alpha}{1 - \alpha z^{-1}}, \quad \alpha < 1 \tag{2}$$

and α is a frequency warping scale factor. The spectral aperiodicity estimation was modelled in the same fashion but with 5-th order (including energy) Mel-cepstral coefficients. For the World vocoder voice the fundamental frequency (f_0) estimation was converted to the log-domain as follows:

$$logF_0(f_0) = \begin{cases} log(f_0) & f_0 > 0 \\ -1 \times 10^{10} & f_0 = 0 \end{cases} \tag{3}$$

The reason for the logarithmic scale modelling of the fundamental frequency is the fact that $logF_0$ has a more Gaussian distribution than f_0 [29]. The LPCNet pitch period feature was not converted to the logarithmic domain due to the quantization of the feature and the unknown distribution. The deltas and delta-deltas of all features were also calculated.

4.2 HMM-Based Voices

The voice was based on the standard architecture of five-state, left-to-right HSMM. All continuous features were modelled by single-component Gaussians. For the LPCNet voice the pitch period feature was modelled as a normal continuous variable (voicing is captured in the pitch correlation feature) whereas the World $LogF_0$ feature was modelled as a 3-dimensional *multi-space probability*

distribution HMM (MSDHMM) [18], as is standard practice [33]. The decision trees state clustering was done using a minimum description length (MDL) factor of 1.0. For both voices global variance [33] was included.

Training of the HMM-based voices was done via custom scripts based on the standard demonstration script[2] available as part of HTS [33] (version 2.2). The custom scripts allow for the parallelization of the embedded re-estimation step on multi-core processors. Both voices train on an Intel i7 4-core, 8-thread CPU (2.80GHz) in about 1.5 hours. MLPG was used to smooth the generated features before synthesis, using the predicted means and variances of the acoustic features and their delta and delta-deltas. No post-filtering was applied.

4.3 DNN-Based Voices

The extracted linguistic descriptions were converted to a vector containing a combination of binary encodings (for the phoneme identities and features) and positional information (as is done in [27]).

Frame level positional information was also added to the state aligned linguistic descriptions in order to improve the granularity of the linguistic descriptions at the speech frame level. The positional information consisted of: *the frame position within the HMM state and phoneme*, *the state position within the phoneme*, and *state and phoneme durations*, as defined in [26].

The input linguistic descriptions and frame level positional information vector consisted of 384 features and was normalised to the range of [0.01, 0.99], whilst the output vectors (the vocoder features) were normalised to zero mean and unit variance. For the World vocoder the output vector consisted of 198 features (60-th order spectral envelope, 5-th order aperiodicity estimation and f_0 and their delta and delta-deltas), whilst for the LPCNet vocoder the output vector consisted of 60 features (18-th order Bark-scale spectral envelope, pitch period and pitch correlation and their delta and delta-deltas).

A hyperparemeter search for the basic architecture was conducted and we settled on a 4 hidden layer network, with 512 units per layer, for both the bottleneck model and the acoustic model in Fig. 1. The *rectified linear unit* (ReLU) activation function was used for the hidden layers, whilst the output layer was linear. The Adam optimisation algorithm [8] was used together with a learning rate scheduler that lowers the learning rate when the validation loss reaches a plateau (the Adam optimisation algorithm adjusts the learning rate, it is the upper bound that we reduced). Our loss function was the mean squared error on the predicted acoustic features. The starting learning rate was fixed at 0.004.

For the bottleneck model we built 25 variations of voices with different configurations of the position of the hidden bottleneck layer, the size of the bottleneck layer and the bottleneck features context size used in the acoustic model. The test set was synthesized with each of these voices and objective measures (see Section 4.4) applied in order to select the best model. MLPG was also used to

[2] `http://hts.sp.nitech.ac.jp/`

Table 1. A comparison between the objective results of the different configurations of the bottleneck model. MCD: Mel-cepstral distortion. f_0 RMSE: Fundamental frequency root mean squared error (linear scale). f_0 MAE: Fundamental frequency mean absolute error (linear scale). f_0 VCE %: Fundamental frequency voicing classification percentage error.

Layer index	Bottleneck size	Context size	MCD (dB)	f_0 RMSE (Hz)	f_0 MAE (Hz)	f_0 VCE %
N/A	N/A	N/A	5.7008	12.63	7.39	7.59
0	32	11	5.6569	12.64	6.94	7.86
		23	**5.6314**	12.11	6.82	**7.48**
		35	5.6663	13.54	7.16	8.03
	64	11	5.6426	**11.72**	**6.46**	7.59
		23	5.6449	13.31	7.27	7.89
		35	5.6600	12.54	7.03	7.50
1	32	11	5.7118	13.40	7.60	7.52
		23	5.6984	12.62	6.91	8.08
		35	5.6887	12.35	6.84	8.09
	64	11	5.6925	12.92	7.35	7.98
		23	5.6907	14.55	7.92	8.34
		35	5.6731	13.28	7.58	8.01
2	32	11	5.7322	14.87	8.40	7.86
		23	5.7259	13.25	7.45	7.86
		35	5.7339	12.69	7.60	8.01
	64	11	5.7173	14.26	8.28	7.86
		23	5.7084	13.04	7.55	7.95
		35	5.6943	33.36	8.57	7.82
3	32	11	5.7093	12.39	6.82	8.31
		23	5.7070	12.88	6.99	7.60
		35	5.6959	14.53	7.96	7.90
	64	11	5.6731	12.23	7.14	7.60
		23	5.6867	12.48	7.18	8.13
		35	5.7038	14.87	7.10	8.06

smooth the generated features before synthesis, using the predicted means and their delta and delta-deltas, but in contrast to the HMM-based voices, a global pre-computed variance is used (because the model just predicts the means and not the variances). No post-filtering was applied.

Table 1 gives the objective results of the 25 voices built in order to select the best combination of the *position of the hidden bottleneck layer* ("Layer index"), the *size of the bottleneck layer* ("Bottleneck size") and the *bottleneck features context size* ("Context size") as used in the acoustic model. The first entry in the table is a normal FFNN (without utilising the bottleneck scheme).

We settled on a hidden bottleneck in the first layer (Layer index = 0) with a size of 64 units and a context size of 11. Even though the voicing classification percentage error and Mel-cepstral distortion of the first layer model with bottleneck size 32 and context size 23 was marginally less than our selected model,

Table 2. Results of objective measures of the final four voices used in the perceptual evaluation. MCD: Mel-cepstral distortion. f_0 RMSE: Fundamental frequency root mean squared error (linear scale). f_0 MAE: Fundamental frequency mean absolute error (linear scale). f_0 VCE %: Fundamental frequency voicing classification percentage error.

System	MCD (dB)	f_0 RMSE (Hz)	f_0 MAE (Hz)	f_0 VCE %
DNN LPCNet	5.6426	**11.72**	**6.46**	**7.59**
DNN World	**5.3870**	34.15	7.53	10.93
HMM LPCNet	6.6260	23.07	9.88	10.00
HMM World	6.5512	28.00	11.07	9.46

in-house testing could not discern the difference between the synthesized test sets of the two models and a model with a smaller context size should synthesize faster.

Based on this we built voices with the World and LPCNet features and synthesized the test set. Both voices train on an Intel i7 4-core, 8-thread CPU (2.80GHz) in about 1.5 hours (the same as the HMM-based voices), but we did the bulk of the training of the 25 voices used for determining the model parameters on Google Colab[3] (which provides free GPUs) in about 35 minutes per voice. We did not train a new model for the LPCNet vocoder, but rather used one of the pre-trained models[4]. This is significant because it shows that the vocoder can be trained on speech that is not from the same language as used during synthesis time.

4.4 Results

The synthesized test set for each of the final four voices were done against the natural durations extracted from the alignments (see Section 4.1).

Objective Measures In order to quantitatively compare our search for the bottleneck layer position, size and context size (see Section 4.3), as well as the final voices we used the following objective measures:

- Mel-cepstral distortion (MCD), as defined in [9].
- f_0 Root mean squared error (RMSE), as defined in [31].
- f_0 Mean absolute error (MAE), this is similar to RMSE, but the MAE is less influenced by large outliers than RMSE.
- f_0 Voicing classification error (VCE), as defined in [31].

Table 2 gives the objective measures of the final four voices as defined in Sections 4.2 and 4.3.

[3] `https://colab.research.google.com`

[4] `https://media.xiph.org/lpcnet/data/`

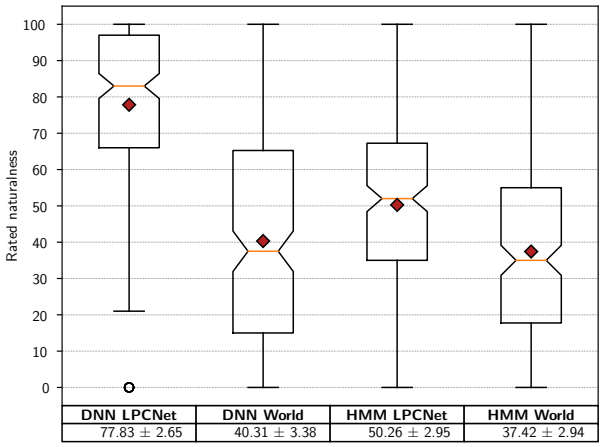

Fig. 2. Aggregated MUSHRA test results. Box edges are at 25 and 75% quantiles. The notches represent the confidence interval around the median, while the diamond markers (⋄) are the means. The values at the bottom are the mean MUSHRA naturalness scores of the four compared voices (at a 95% confidence level).

Table 3. Statistically significant differences in naturalness between the voice pairs are indicated by a solid black circle (●), while non-significant differences are indicated by a non-solid circle (○).

	DNN LPCNet	DNN World	HMM LPCNet	HMM World
DNN LPCNet		●	●	●
DNN World	●		●	○
HMM LPCNet	●	●		●
HMM World	●	○	●	

Subjective Measures For subjective evaluation we ran a web-based *MUltiple Stimuli with Hidden Reference and Anchor* (MUSHRA) [5] listening test[5]. The participants were asked to rate the stimuli in terms of naturalness, given the reference speech recording. 34 people participated, and after post-screening 20 participants remained (the minimum to be significant [5]). Participants who scored the hidden reference stimuli less than 90 for more than 20% of the test set were removed. The results are given in Figure 2.

We also calculated the Wilcoxon signed-rank test to determine if the results in Figure 2 are statistically significant between the voice pairs. The results are given in Table 3.

[5]Speech samples available at `https://abylouw.github.io/fair2019_samples.html`

5 Discussion and Conclusion

It is interesting to note that the DNN World voice had the lowest MCD in comparison to the reference test set and that this does not translate into better perceived quality. The RMSE value of the DNN World voice is relatively high in comparison to the other voices, but this is due to one outlier in the test set, and therefore we also calculated the MAE. In total there were 200 combined observations (20 participants and 10 audio sequences) of the DNN LPCNet voice and the hidden reference (the MUSHRA subjective assessment has a hidden reference waveform). Of the 200 combinations, there were 52 instances where the DNN LPCNet voice was rated equal or better than the reference recording. Although not statistically significant, it is still remarkable given that the corpus is less than 1 hour in duration and that the synthesized test samples were not seen by the model during the training stage.

In this work we found that the possible gains brought on by the move from an HMM-based acoustic model to a DNN-based acoustic model seem to be dependent on the specific vocoder, as there is not a statistically significant difference in the subjective evaluation results between the HMM World and DNN World voices, but there is a significant difference between the HMM LPCNet and DNN LPCNet voices. It might be that our chosen architecture was too small for the size of the feature set of the World vocoder (198 features) and this will require more tests in the future. Another possible reason is that the DNN architecture can learn the LPCNet features better than the HMM architectures, as the difference as found in the subjective naturalness tests between DNN LPCNet and DNN World are much larger than between HMM LPCNet and HMM World.

In terms of training time the difference between the traditional HMM-based voices and the proposed architecture for the DNN-based voices are negligible on CPUs, but the DNN-based voices can utilise GPUs for training which more than halves the training time. In terms of run-time we found that the HMM-based voices are about 27.83 times faster than real time, whilst the proposed architecture for the DNN-based voices is about 5.25 times faster than real time, both on a Intel i7 4-core, 8-thread CPU (2.80GHz). The HMM-based voices are thus about 5.3 times faster than the DNN-based voices. In our view this is not a significant hindrance towards the adoption of the proposed DNN-based architecture, as the naturalness improvements brought on outweigh the speed deficit and the proposed DNN-based architecture can still be utilised in real time settings. In preliminary tests we also found that by using a sliding-window approach to the MLPG parameter smoothing calculation utilising only the bottleneck context size for the number of acoustic frames we can synthesize speech in a streaming fashion with minimal loss in synthesized speech quality (imperceivable) versus using the all the acoustic frames of the whole utterance.

In future work we will focus on the duration model as well as intonation. The tonal languages in South Africa represent a significant challenge for speech synthesis, especially given the resource-scarce environment. We will also explore using variational autoencoders (VAE) for the bottleneck model, given their ability to learn a latent space.

References

1. Black, A.W.: CMU Wilderness Multilingual Speech Dataset. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 5971–5975. IEEE (2019)
2. Campbell, N.: Developments in corpus-based speech synthesis: Approaching natural conversational speech. IEICE transactions on information and systems **88**(3), 376–383 (2005)
3. Chung, Y.A., Wang, Y., Hsu, W.N., Zhang, Y., Skerry-Ryan, R.: Semi-Supervised Training for Improving Data Efficiency in End-to-End Speech Synthesis. arXiv e-prints arXiv:1808.10128 (Aug 2018)
4. Fujimoto, T., Hashimoto, K., Oura, K., Nankaku, Y., Tokuda, K.: Impacts of Input Linguistic Feature Representation on Japanese End-to-End Speech Synthesis. In: 10th ISCA Speech Synthesis Workshop. ISCA, Vienna, Austria (2019)
5. ITU-R, R.: BS. 1534-1. Method for the subjective assessment of intermediate sound quality (MUSHRA). International Telecommunications Union, Geneva (2001)
6. Kalchbrenner, N., Elsen, E., Simonyan, K., Noury, S., Casagrande, N., Lockhart, E., Stimberg, F., van den Oord, A., Dieleman, S., Kavukcuoglu, K.: Efficient Neural Audio Synthesis. arXiv e-prints arXiv:1802.08435 (Feb 2018)
7. King, S.: Measuring a decade of progress in text-to-speech. Loquens **1**(1), 2386–2637 (2014)
8. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
9. Kominek, J., Schultz, T., Black, A.W.: Synthesizer voice quality of new languages calibrated with mean mel cepstral distortion. In: Spoken Languages Technologies for Under-Resourced Languages (2008)
10. Louw, J.A.: Speect: a multilingual text-to-speech system. In: Proceedings of the Nineteenth Annual Symposium of the Pattern Recognition Association of South Africa. pp. 165–168. Cape Town, South Africa (November 2008)
11. Louw, J.A., Moodley, A.: Speaker Specific Phrase Break Modeling with Conditional Random Fields for Text-to-Speech. In: Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech). pp. 1–6. IEEE, Stellenbosch, South Africa (December 2016)
12. Morise, M., Yokomori, F., Ozawa, K.: WORLD: a vocoder-based high-quality speech synthesis system for real-time application. IEICE TRANSACTIONS on Information and Systems **99**(7), 1877–1884 (2016)
13. van Niekerk, D., de Waal, A., Schlünz, G.: Lwazi II Afrikaans TTS Corpus. `https://hdl.handle.net/20.500.12185/443` (November 2015), ISLRN: 570-884-577-153-6
14. Shen, J., Pang, R., Weiss, R.J., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerry-Ryan, R., Saurous, R.A., Agiomyrgiannakis, Y., Wu, Y.: Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions. arXiv e-prints arXiv:1712.05884 (Dec 2017)
15. Silverman, K., Beckman, M.E., Pitrelli, J., Ostendorf, M., Wightman, C., Price, P., Pierrehumbert, J., Hirschberg, J.: ToBI: a standard for labeling English prosody. In: Proceedings of the Second International Conference on Spoken Language Processing (ICSLP). pp. 867–870. Alberta, Canada (October 1992)
16. Sproat, R., Olive, J.: An Approach to Text-to-Speech Synthesis. In: Kleijn, W., Paliwal, K. (eds.) Speech Coding and Synthesis. pp. 611–633. Elsevier, Amsterdam, Netherlands (1995)

17. Tachibana, H., Uenoyama, K., Aihara, S.: Efficiently Trainable Text-to-Speech System Based on Deep Convolutional Networks with Guided Attention. arXiv e-prints arXiv:1710.08969 (Oct 2017)
18. Tokuda, K., Nankaku, Y., Toda, T., Zen, H., Yamagishi, J., Oura, K.: Speech synthesis based on hidden Markov models. Proceedings of the IEEE **101**(5), 1234–1252 (2013)
19. Tokuda, K., Yoshimura, T., Masuko, T., Kobayashi, T., Kitamura, T.: Speech parameter generation algorithms for HMM-based speech synthesis. In: 2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 00CH37100). vol. 3, pp. 1315–1318. IEEE (June 2000)
20. Valin, J.M., Skoglund, J.: A Real-Time Wideband Neural Vocoder at 1.6 kb/s Using LPCNet. arXiv preprint arXiv:1903.12087 (2019)
21. Valin, J.M., Skoglund, J.: LPCNet: Improving neural speech synthesis through linear prediction. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 5891–5895. IEEE (2019)
22. van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., Kavukcuoglu, K.: WaveNet: A Generative Model for Raw Audio. arXiv e-prints arXiv:1609.03499 (Sep 2016)
23. Wang, Y., Skerry-Ryan, R., Stanton, D., Wu, Y., Weiss, R.J., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S., Le, Q., Agiomyrgiannakis, Y., Clark, R., Saurous, R.A.: Tacotron: Towards End-to-End Speech Synthesis. arXiv e-prints arXiv:1703.10135 (Mar 2017)
24. Watts, O., Henter, G.E., Fong, J., Valentini-Botinhao, C.: Where do the improvements come from in sequence-to-sequence neural TTS? In: 10th ISCA Speech Synthesis Workshop. ISCA, Vienna, Austria (September 2019)
25. Wu, Z., King, S.: 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 (TASLP) **24**(7), 1255–1265 (2016)
26. Wu, Z., Valentini-Botinhao, C., Watts, O., King, S.: Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis. In: 2015 IEEE international conference on acoustics, speech and signal processing (ICASSP). pp. 4460–4464. IEEE (2015)
27. Wu, Z., Watts, O., King, S.: Merlin: An Open Source Neural Network Speech Synthesis System. In: SSW. pp. 202–207 (2016)
28. Yamagishi, J.: An Introduction to HMM-Bbased Speech Synthesis. Tech. rep., The Centre for Speech Technology Research, The University of Edinburgh (October 2006)
29. Yamagishi, J., King, S.: Simple methods for improving speaker-similarity of HMM-based speech synthesis. In: 2010 IEEE International Conference on Acoustics, Speech and Signal Processing. pp. 4610–4613. IEEE (2010)
30. Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X., Moore, G., Odell, J., Ollason, D., Povey, D., et al.: The HTK book. Cambridge University Engineering Department **3**, 175 (2002)
31. Yu, K., Young, S.: Continuous f0 modeling for hmm based statistical parametric speech synthesis. IEEE Transactions on Audio, Speech, and Language Processing **19**(5), 1071–1079 (2010)
32. Ze, H., Senior, A., Schuster, M.: Statistical parametric speech synthesis using deep neural networks. In: 2013 ieee international conference on acoustics, speech and signal processing. pp. 7962–7966. IEEE (2013)

33. Zen, H., Tokuda, K., Masuko, T., Kobayasih, T., Kitamura, T.: A Hidden Semi-Markov Model-Based Speech Synthesis System. IEICE Transactions on Information and Systems **E90-D**(5), 825–834 (May 2007)
34. Zen, H., Tokuda, K., Black, A.W.: Statistical parametric speech synthesis. speech communication **51**(11), 1039–1064 (2009)