Interspeech 2018
2-6 September 2018, Hyderabad

Spanish Statistical Parametric Speech Synthesis using a Neural Vocoder

Antonio Bonafonte, Santiago Pascual, Georgina Dorca

Universitat Politècnica de Catalunya, Barcelona, Spain

antonio.bonafonte@upc.edu, santi.pascual@upc.edu, georgina.dorca@gmail.com

Abstract

During the 2000s decade, unit-selection based text-to-speech was the dominant commercial technology. Meanwhile, the TTS research community has made a big effort to push statistical-parametric speech synthesis to get similar quality and more flexibility on the synthetically generated voice. During last years, deep learning advances applied to speech synthesis have filled the gap, specially when neural vocoders substitute traditional signal-processing based vocoders. In this paper we propose to substitute the waveform generation vocoder of MUSA, our Spanish TTS, with SampleRNN, a neural vocoder which was recently proposed as a deep autoregressive raw waveform generation model. MUSA uses recurrent neural networks to predict vocoder parameters (MFCC and logF0) from linguistic features. Then, the Ahocoder vocoder is used to recover the speech waveform out of the predicted parameters. In the first system SampleRNN is extended to generate speech conditioned on the Ahocoder generated parameters (mfcc and logF0), where two configurations have been considered to train the system. First, the parameters derived from the signal using Ahocoder are used. Secondly, the system is trained with the parameters predicted by MUSA, where SampleRNN and MUSA are jointly optimized. The subjective evaluation shows that the second system outperforms both the original Ahocoder and SampleRNN as an independent neural vocoder.

Index Terms: SPSS, neural vocoder, SampleRNN, Spanish TTS

1. Introduction

Speech synthesis is the technique to make machines generate speech signals, and text-to-speech (TTS) is the generation of these signals conditioned on linguistic contents. With the new technology trends focusing on human-machine natural interactions, these systems are gaining relevance because speech is the most intuitive interface with which we can communicate with these devices. A key factor in the advancement of these interactive technologies has been the evolution of deep learning in the TTS field. Deep neural networks (DNNs) have been applied over the past few years solving challenging tasks in the process of generating speech towards an end-to-end paradigm.

First deep learning TTS models worked in a two-stage fashion [1]. In this setup, the first step is to predict the number of frames (duration) of a phoneme to be synthesized with a duration model, where the input are linguistic features extracted from text with hand-crafted rules. Then, the acoustic parameters of every frame are estimated by the so called acoustic model, where we inject the linguistic features in addition to the relative duration of the generated phoneme. Different works use this design, outperforming previously existing statistical parametric speech synthesis (SPSS) systems with new prosodic and linguistic features, and also with perceptual losses in the acoustic mapping cost [2, 3, 4, 5, 6].

Because of the sequential nature of the TTS problem, recurrent neural networks (RNN) have been also used as deep architectures that effectively predict either prosodic features [7, 8] or duration and acoustic features [9, 10, 11, 12], including the studies of different variants of recurrent cells, like long short term memory (LSTM) or gated recurrent unit (GRU) modules.

So far these systems mapped linguistic features coming from text into acoustic features extracted from speech with some vocoder, but the latest deep TTS trend is going more end-to-end in predicting directly the waveform. Wavenet [13] is the first deep generative model that directly predicts waveform samples. This system is very good at representing very fine-grained waveform amplitudes, having the same type of input linguistic features as in two-stage models. There are new advances in going even from raw text to raw audio, like the recent Deep voice [14] and Char2wav [15]. These systems include some textual processing neural modules along with a waveform generator block. In the case of Deep voice they use a modified version of the Wavenet, and Char2wav includes the sampleRNN [16], which is another type of deep generative model for raw audio with lower computational and memory requirements than Wavenet.

In this work we explore the possible improvement in generated speech quality by inserting a sampleRNN as a neural vocoder on top of our two-stage RNNs acoustic model. This two-stage system is derived from our previous work on multi speaker adaptation (MUSA) [17], although here we will only focus on M1 speaker from our previous works.

We will explore three different configurations for the attachment of both systems: (1) training sampleRNN with real audio features and decoding the waveform out of the acoustic model prediction. (2) training sampleRNN with the acoustic model predictions themselves and then attaching the vocoder to the acoustic model. (3) training sampleRNN with the acoustic model predictions and fine tune the model end-to-end from the input to the acoustic model to the waveform.

This paper is structured as follows. In section 2 we describe our MUSA two-stage TTS system. Then sampleRNN is described in section 3, with few details on how we improved the base implementation we had and how we attached the conditioning from our acoustic model to it. In section 4 we describe the experiments performed, followed by our conclusions and some future lines of work in section 5.

2. MUSA text-to-speech

MUSA is a two stage RNN-LSTM model influenced by the work in Zen and Sak [12] in the use of LSTM models to build the duration model and the acoustic model without the need of predicting dynamic acoustic features. A key difference from [12] is the capacity to model many speakers and adapt the acoustic mapping among them with different output branches. Nonetheless in this work we did not use this capability and focused on just one speaker for the vocoding experiments.

The linguistic and prosodic features injected as inputs to both the acoustic and the duration model are extracted from text with the Ogmios front-end [18].

The duration model has a mixture of approximately 360 linguistic features in categorical, real and boolean form. These inputs are processed by an LSTM layer with 256 hidden units, and then a final linear unit converts the 256-dimensional hidden features into the real valued prediction of the duration. This duration is normalized within [0, 1] range for better backpropagation purposes.

Regarding the acoustic model, its inputs are the same linguistic features as in the duration model with 2 additional terms: (1) absolute phoneme duration and (2) current relative position inside the absolute duration. These features are then mapped into 43 acoustic features coming from the Ahocoder [19] vocoder: 40 MFCC coefficients, 1 voiced frequency, 1 logF0 and 1 voiced/unvoiced flag that masks out logF0 and voiced frequency values if these fall into an unvoiced region. The basic version of the model is composed of two fully connected layers of 256 units each with tanh activation functions. These are used as embedding layers to condense the representation of the mixed sets of linguistic and duration features at the input. These features are then injected directly into 2 LSTM hidden layers of 256 units that we call the trunk of the model. Finally the output layer is an LSTM layer with 43 units that generates the acoustic parameter trajectories for the Vocoder. This acoustic model supports the addition of multiple output layers emerging from the central LSTM trunk, such that many speakers can be represented with the same trunk by deriving one output layer per speaker. This allows us to perform speaker adaptation easily by adding new branches [20], or even performing interpolations of different speaker acoustic properties to create new voices [21].

3. SampleRNN

SampleRNN [16] is a deep autoregressive generative model for raw audio. It consists of an implementation of several recurrent layers at different scales, which are capable of modeling long term dependencies in audio waveforms while training on short sequences, which results in memory efficiency during training.

As a result, this recurrent architecture models the probability of a sequence of waveform samples by following the probability chain rule, as stated in equation 1, with an explicitly parameterized form of the distribution that can have any shape.

$$P(\boldsymbol{X}) = \prod_{t=1}^{T} p(x_t | x_1, ..., x_{t-1}) \tag{1}$$

In order to make sampleRNN work as a vocoder we provide it conditioning representations of the contents to generate in form of acoustic vectors coming from the vocoder parameterization.

$$P(\boldsymbol{X}|\boldsymbol{L}) = \prod_{t=1}^{T} p(x_t | x_1, ..., x_{t-1}, \boldsymbol{l_1}, ..., \boldsymbol{l_i}) \tag{2}$$

Where $\boldsymbol{l_i}$ stands for the 43-dimensional acoustic vector corresponding to the analysis window of the current sample x_t.

4. Experimental Setup

4.1. Dataset

The experiments are performed with a professional male speaker from the TCSTAR [22] dataset, thus with recording studio quality. We have 7.5 hours of speech from this speaker, which are divided in 80% for training, 10% for validation and 10% for test.

4.2. Neural Vocoder Architecture Setup

Several experiments with the unconditioned SampleRNN have been done to define the architecture and the optimizer parameters to be used in the remaining experiments in our TTS system. All experiments are conducted with a batch size of 128 sequences of length 1040.

As shown in figure 1, the selected architecture has three tiers. The higher tier is composed of 2 GRU layers. The input is 80 samples (5 ms. at 16kHz) and the output is the condition to the mid tier. The conditioner is expanded into 5 conditioning vectors through a 1D convolutional layer. The mid tier is also composed of 2 GRU layers. In this case the input is formed by 16 samples in addition to the previous conditioning vector. Finally, the lowest tier contains a MLP with one hidden layer. Owing to the fact that SampleRNN predicts discrete categories corresponding to quantized sample values at 8 bits, we have an output layer of 256 units. Note that all recurrent and fully connected layers contain 1024 units.

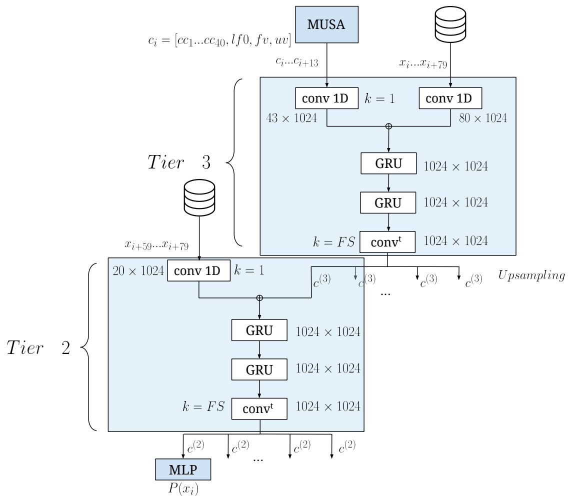

Figure 1: *Structure of the first two tiers of the system, with two RNN of 1024 hidden units. , a* $FS = [20 \quad 4]$ *and an upsampling ratio of* $r = 4$. *In the first tier we have an entry of 80 real valued samples and an activated conditioner flag. Both the input and the conditioner pass through their respective 1D convolution layer, and these outputs are linearly added. This will be the entrance to the first GRU, which will store the hidden state* $h(k)$ *to use it in the next tier as a conditioner. Then the output goes through the transposed convolution, performing the process of upsampling. Finally these output enters to the second frame-level. The input* $inp(k = 2)$ *is conditioned under the previous stored memory in the upper tier,* $h(k)_{t-1}$ *state. From here is followed the same process as the previous tier.*

The original SampleRNN used uniform quantization, so we decided to test the possible performance increase when we apply μ-law transformation to reduce the quantization noise. Listening to the newly generated samples clearly show the advantage of the μ-law transformed distribution over the raw one.

As a final remark, the proposed vocoder system was built on top of a publicly available implementation of sampleRNN[1] in PyTorch [23].

4.3. TTS System: MUSA and SampleRNN

As we are focusing on the acoustic model, all the durations are forced to be the annotated ones in the ground truth.

We want to test the performance of the coupling under three different configurations:

- Independent Neural Vocoder (INV). In this case SampleRNN is trained with the acoustic parameterization from original speech samples.
- Independently-trained MUSA-based Neural Vocoder (IMNV). For each utterance in the training corpus, we infer the acoustic parameter with the MUSA prediction. These parameters are used as training samples for our neural vocoder replacing previous ground truth samples coming from the original speech samples.
- Jointly-trained MUSA-based Neural Vocoder (JMNV). Same case as previous one but adding an additional training stage where the full joint model (MUSA+SampleRNN) is tuned end-to-end.

4.4. Objective Evaluation

We observe an abrupt behavior of the all losses after certain epochs. We believe that the learning rate could be too big after that epoch, thus we decide to decrease the learning rate monotonically with a scheduler. This decays the initial learning rate by a γ factor every E epochs. Specifically, we set up an initial learning rate (lr) of 0.001 with $\gamma = 0.1$. This way we have:

$$\begin{cases} \text{lr} = 10^{-3} & \text{epoch} < 15 \\ \text{lr} = 10^{-4} & 15 \leq \text{epoch} < 35 \\ \text{lr} = 10^{-5} & \text{epoch} > 35 \end{cases} \tag{3}$$

In table 1 we show the difference in test performance among three different optimizer executions. The test metric is the negative log-likelihood (NLL). These results confirm the effectiveness of the designed scheduler, being Adam the preferred option over RMSprop and so the one with which we combine the scheduler to obtain the best result.

Table 1: *Comparison of the different tested optimizers in terms of the negative log-likelihood (NLL) (lower is better).*

Optimizer	Average NLL Test
RMSprop	3.19
Adam	2.56
Adam+Scheduler	**2.41**

As shown in the table 2, when we substitute the ground truth features (INV) with the ones predicted by MUSA (IMNV) there is a noticeable degradation in the test performance, as expected. We can slightly improve it by facilitating the convergence in a setup were IMNV is initialized with INV weights. Last row in the table shows how we can close the gap by tuning the joint structure (MUSA and SampleRNN) end-to-end.

[1]https://github.com/deepsound-project/samplernn-pytorch

Table 2: *Comparison of SampleRNN test performance under different configurations in terms of the negative log-likelihood (NLL) (lower is better).*

Model Configuration	Average NLL Test
INV	2.41
IMNV	2.67
IMNV pre-trained	2.66
JMNV	2.47

4.5. Subjective Evaluation

Objective tests performed well to compare different architectures and to suggest how well does the TTS work. But when analyzing the complete system, that is, the conditioned SampleRNN under the MUSA characteristics, we obtained similar results in all the performed experiments. To get a more in-depth evaluation of the model a subjective test is conducted to evaluate the naturalness of the TTS developed in this work and also to make a comparison with the baseline system, that is, Ahodecoder vocoder under MUSA parameterization.

A web based application was developed with the selected audio files obtained from the experiments, explained below, and volunteers were asked to rate the voices in a scale from one to five, one being bad and five being excellent. In the test each listener was asked to evaluate 5 sentences, randomly selected from the test set. In total 14 subjects took the test. The participants could listen the different recordings as many times as required to make comparisons between the different systems and select which system or systems produce the most natural speech. For every sentence, the listeners evaluated 3 different versions generated with the following 3 different systems:

- Baseline: using Ahodecoder to generate speech from the acoustic parameters predicted by MUSA [17] (MAHO)
- Independently-trained MUSA-based Neural Vocoder (IMNV).
- Jointly-trained MUSA-based Neural Vocoder (JMNV).

Table 3 shows the results of the subjective test.

Table 3: *Preference test showing the best selected system or systems (in percentage).*

MAHO	IMNV	JMNV	JMNV OR MAHO	JMNV IMNV
21.4	4.3	51.4	14.3	8.6

As it can be seen, independent training of MUSA and SampleRNN (IMNV) produces much worse results that the baseline system that uses the original decoder of Ahocoder (MAHO). MAHO is one of the preferred system in 50.0% of the 70 evaluations, while IMNV is only one of the preferred systems in 12.9% of the cases. However, the joint training of MUSA and SampleRNN effectively integrates both systems producing the best option. As we can see, JMNV is one of the systems in 74.3% of the cases.

The results IMNV are much worse than expected. An analysis of the result showed that there was a normalization problem that produced an inconsistency between MUSA prediction and the ground truth acoustic features. However, the results show that the joint training is able to overcome such limitation and is more robust to inconsistencies.

5. Conclusions

In this paper we have proposed a complete Spanish TTS using deep learning. It is based on MUSA, our previous SPSS that used RNN to predict a sequence of acoustic features which are converted into speech using Ahocoder, a traditional high quality vocoder. In this work, the vocoder has been substituted by SampleRNN, a novel neural vocoder. First MUSA predicts the duration of the phonemes and the acoustic parameters from the input text. In the second step, SampleRNN converts the parameters into the respective waveform.

First of all, some variants have been introduced that have proven to be beneficial on the initial SampleRNN model. SampleRNN discretizes the signal with 8 bits to have a more flexible model of its pdf. The original work used a uniform quantization while here, influenced by the proposed Wavenet system [13], the μ-law quantification has been successfully proposed, significantly improving the quality of the generated signal. The architecture and the training parameters have been determined using SampleRNN to generate unconditioned speech.

Three configurations have been considered to integrate MUSA and SampleRNN. The joint optimization of MUSA and SampleRNN (JMNV) clearly outperforms the baseline that uses the decoder of the Ahocoder vocoder and the independent estimation of MUSA and the Neural Vocoder.

Currently we are working on integrating the multispeaker capability of MUSA with the neural vocoder. In this way, the MUSA intermediate layers can be used as speaker independent acoustic representation and the speaker identity can be imposed directly through the neural vocoder.

6. Acknowledgements

This research was supported by the project TEC2015-69266-P (MINECO/FEDER, UE).

7. References

[1] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7962–7966.

[2] 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," *Proc. ISCA SSW8*, pp. 281–285, 2013.

[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 *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 3829–3833.

[4] Q. Hu, Z. Wu, K. Richmond, J. Yamagishi, Y. Stylianou, and R. Maia, "Fusion of multiple parameterisations for dnn-based sinusoidal speech synthesis with multi-task learning," in *Proc. INTERSPEECH*, 2015, pp. 854–858.

[5] Q. Hu, Y. Stylianou, R. Maia, K. Richmond, J. Yamagishi, and J. Latorre, "An investigation of the application of dynamic sinusoidal models to statistical parametric speech synthesis." in *Proc. INTERSPEECH*, 2014, pp. 780–784.

[6] S. Kang, X. Qian, and H. Meng, "Multi-distribution deep belief network for speech synthesis," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 8012–8016.

[7] S. Pascual and A. Bonafonte, "Prosodic break prediction with rnns," in *International Conference on Advances in Speech and Language Technologies for Iberian Languages*. Springer, 2016, pp. 64–72.

[8] S.-H. Chen, S.-H. Hwang, and Y.-R. Wang, "An rnn-based prosodic information synthesizer for mandarin text-to-speech," *IEEE Transactions on Speech and Audio Processing*, vol. 6, no. 3, pp. 226–239, 1998.

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

[10] Z. Wu and S. King, "Investigating gated recurrent networks for speech synthesis," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5140–5144.

[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] H. Zen and H. Sak, "Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4470–4474.

[13] 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 preprint arXiv:1609.03499*, 2016.

[14] S. O. Arik, M. Chrzanowski, A. Coates, G. Diamos, A. Gibiansky, Y. Kang, X. Li, J. Miller, J. Raiman, S. Sengupta *et al.*, "Deep voice: Real-time neural text-to-speech," *arXiv preprint arXiv:1702.07825*, 2017.

[15] J. Sotelo, S. Mehri, K. Kumar, J. F. Santos, K. Kastner, A. Courville, and Y. Bengio, "Char2wav: End-to-end speech synthesis," in *International Conference on Learning Representations (ICLR): Workshop Track*, Toulon, France, Apr. 2017.

[16] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. C. Courville, and Y. Bengio, "Samplernn: An unconditional end-to-end neural audio generation model," *CoRR*, vol. abs/1612.07837, 2016. [Online]. Available: http://arxiv.org/abs/1612.07837

[17] S. Pascual, "Deep learning applied to speech synthesis," Master's thesis, Universitat Politècnica de Catalunya, 2016.

[18] A. Bonafonte, P. D. Agüero, J. Adell, J. Pérez, and A. Moreno, "Ogmios: The UPC text-to-speech synthesis system for spoken translation," in *TC-STAR Workshop on Speech-to-Speech Translation*, 2006, pp. 199–204.

[19] D. Erro, I. Sainz, E. Navas, and I. Hernaez, "Harmonics Plus Noise Model Based Vocoder for Statistical Parametric Speech Synthesis," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 2, pp. 184–194, Apr. 2014.

[20] S. Pascual and A. Bonafonte, "Multi-output rnn-lstm for multiple speaker speech synthesis and adaptation," in *Signal Processing Conference (EUSIPCO), 2016 24th European*. IEEE, 2016, pp. 2325–2329.

[21] S. Pascual and A. Bonafonte Cávez, "Multi-output rnn-lstm for multiple speaker speech synthesis with a-interpolation model," in *SSW9: 9th ISCA Workshop on Speech Synthesis: proceedings: Sunnyvale (CA, USA): September 13-15, 2016*. IEEE, 2016, pp. 112–117.

[22] A. Bonafonte, H. Höge, I. Kiss, A. Moreno, U. Ziegenhain, H. van den Heuvel, H.-U. Hain, X. S. Wang, and M.-N. Garcia, "Tc-star: Specifications of language resources and evaluation for speech synthesis," in *Proc. of LREC Conf*, 2006, pp. 311–314.

[23] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.