Neural VTLN for Speaker Adaptation in TTS

Bastian Schnell[1,2], *Philip N. Garner*[1]

[1]Idiap Research Institute, Switzerland
[2]École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
{bastian.schnell, phil.garner}@idiap.ch

Abstract

Vocal tract length normalisation (VTLN) is well established as a speaker adaptation technique that can work with very little adaptation data. It is also well known that VTLN can be cast as a linear transform in the cepstral domain. Building on this latter property, we show that it can be cast as a (linear) layer in a deep neural network (DNN) for speech synthesis. We show that VTLN parameters can then be trained in the same framework as the rest of the DNN using automatic gradients. Experimental results show that the DNN is capable of predicting phone-dependent warpings on artificial data, and that such warpings improve the quality of an acoustic model on real data in subjective listening tests.

Index Terms: VTLN, Speaker Adaptation, Deep Learning, TTS, Statistical Parametric Speech Synthesis

1. Introduction

Vocal Tract Length Normalization (VTLN) is a technique first developed in Automatic Speech Recognition (ASR) to normalize against different speakers [1, 2, 3, 4]. The technique was inspired by the observation that one significant difference between speakers is the length of their vocal tract. This length can vary from around 18 cm in males to around 13 cm in females. The length of the vocal tract is inversely proportional to the formant frequency positions. This leads to a variation of around 25% in formant center frequencies among speakers. VTLN addresses this issue by normalizing the speaker's speech feature vectors to an average vocal tract. Intuitively this is a warping of the spectrum; however, Pitz and Ney [5] showed that the vocal tract length normalization can be expressed as a linear transformation in the cepstral space, and can therefore be expressed as a matrix multiplication in that space. A significant amount of research has gone into selecting an appropriate normalization function, which includes piecewise-linear, power, quadratic, and bilinear functions. In this paper we use a bilinear transform, i.e., a non-complex all-pass transformation. This common implementation of VTLN is the same bilinear transform used in the generalized cepstral analysis [6].

ASR systems use VTLN to normalize different speakers to a neutral speaker before processing their speech with the same backbone system. It is well known that TTS systems can use the same technique to adapt a neutral voice to a specific voice, also called reverse VTLN. The number of parameters required for VTLN is much smaller compared to model based adaptations, therefore VTLN requires less adaptation data.

Some research has already investigated the use of VTLN for speaker adaptation in TTS and achieved good performance: In [7] voice conversion is performed by first subdividing source and target speaker's speech material into artificial phonetic classes by clustering frequency spectra of period-synchronous frames. Then, for each source class the most similar target class is determined. Finally, the warping parameters for each source class are estimated by minimizing the Euclidean distance of all frames of the source class to all frames of the mapping target class. In [8] a single warping parameter per speaker is estimated by line search. Speaker dependent speech is then synthesized with an average model and a speaker specific warping over the whole phrase. In [9] two DNNs are trained to model the VTLN and reverse VTLN step for each speaker. As the normalized features are unknown, the authors propose an iterative unsupervised algorithm: 1. Train a speaker-independent GMM, 2. estimate the warping parameters with Maximum Likelihood Estimation (MLE) between input features and predicted normalized features, 3. retrain the GMM with warped input features, 4. repeat step 2 and 3 five times. The predicted normalized features are then used to train the two DNNs. This differs from our approach as it models the VTLN step only implicitly and estimates the warping parameters with a GMM instead of a DNN. In previous work at Idiap [10, 11] the warping parameters are estimated with the expectation maximization (EM) algorithm for different classes. Those classes are based on a regression task tree which is developed from decision tree questions. The EM parameter estimation was conducted with grid [10] and Brent's search [11], essentially in the mathematical framework of hidden Markov models. The authors were able to show that VTLN does indeed lead to faster adaptation that is more natural than unconstrained linear transforms. Closest to our current work is that of Kotani *et. al.* [12]. They use two DNNs to predict a time-dependent linear conversion matrix and bias respectively. However, as they are predicting the full transformation matrix, without constraining it to VTLN warping matrices, the benefit of a small parameter space, i.e., a single time-dependent warping parameter, is lost.

In this paper, we propose a neural implementation of the VTLN technique where its time-dependent warping parameters are internally predicted by the neural network. A key contribution is that the warping parameters are trained via backpropagation in the same framework as the other network parameters. In section 2, we describe VTLN and how cepstral vectors can be warped with a single matrix multiplication. In section 3, we describe our neural implementation of VTLN, especially how we design it to allow efficient back-propagation and inference. In section 4, we explain the experimental design which in particular splits into three parts. First, in 4.1 we prove that our model is able to learn a particular phoneme-dependent warping by adapting to an artificially created speaker with known warping parameter per phoneme. Second, in 4.2 we compare our model to a strong baseline system in objective scores when used as a multi-speaker Text-To-Speech (TTS) system. Third, in 4.3 we compare two systems in objective and subjective scores on a speaker adaptation task. We conclude the paper in section 5 and give future objectives.

2. Vocal Tract Length Normalization

The different lengths of the vocal tracts of different speakers result in a shift of the formant frequencies. Pitz and Ney [5] have shown that this shift equals a linear transform, i.e., warping in the cepstral space. A common implementation of VTLN performs a warping of N mel-cepstral coefficients with an $N \times N$ warping matrix $\boldsymbol{A}$. In this work we use a bilinear transform therefore $\boldsymbol{A}$ only depends on the warping parameter α, expressed as $\boldsymbol{A}_\alpha$. The element in the k-th row and l-th column of matrix $\boldsymbol{A}_\alpha$ can be computed in two ways, 1) recursively [13, 10] by

$$\boldsymbol{A}_{k,l} = \begin{cases} \alpha^k & \text{if } l = 0 \\ 0 & \text{if } l > 0, k = 0 \\ \boldsymbol{A}_{k-1,l-1} & \text{otherwise} \\ +\alpha[\boldsymbol{A}_{k,l-1} - \boldsymbol{A}_{k-1,l}] & \end{cases} \tag{1}$$

or 2) explicitly (equation (15) in [5]) by

$$\boldsymbol{A}_{k,l} = \frac{1}{(l-1)!} \times \sum_{\substack{n= \\ \max(0,l-k)}}^{l} \binom{l}{n} \frac{(k+n-1)!}{(k+n-l)!} (-1)^{n \overbrace{+l+k}^{\text{added}}} \alpha^{2n+k-l} \tag{2}$$

Equation (2) was originally developed in [5] for positive alphas only. We extended it by the part marked *added* to make it also valid for negative alphas. The form of the resulting warping matrix $\boldsymbol{A}_\alpha$ for different alphas is qualitatively expressed in Figure 1. Most of the matrix is zero and the cepstral value after warping is a linear combination of the coefficients around the diagonal with alternating signs. Once $\boldsymbol{A}_\alpha$ is computed, a mel-cepstral coefficient vector $x = (c_1, \ldots, c_K)^T$ of a single frame (parallelizable for multiple frames) is warped to $x_\alpha = (\tilde{c}_1, \ldots, \tilde{c}_K)^T$ by multiplying it from the left with matrix $\boldsymbol{A}_\alpha$.

$$\boldsymbol{x}_\alpha = \boldsymbol{A}_\alpha \boldsymbol{x}$$

Cepstral vectors with deltas (Δ) and double deltas (Δ^2) can be warped in the same way by creating a block diagonal matrix from $\boldsymbol{A}_\alpha$ and multiplying the whole vector with it.

$$\begin{bmatrix} \boldsymbol{x}_\alpha \\ \Delta \boldsymbol{x}_\alpha \\ \Delta^2 \boldsymbol{x}_\alpha \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_\alpha & 0 & 0 \\ 0 & \boldsymbol{A}_\alpha & 0 \\ 0 & 0 & \boldsymbol{A}_\alpha \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \Delta \boldsymbol{x} \\ \Delta^2 \boldsymbol{x} \end{bmatrix} \tag{3}$$

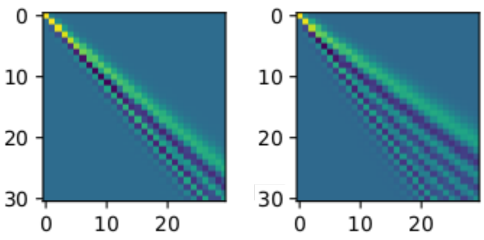

Figure 1: *Qualitative representation of a VTLN warping matrix for a bilinear transform (left: $\alpha = 0.1$, right: $\alpha = 0.2$).*

3. Neural VTLN Implementation

To use VTLN within the modern state-of-the-art deep learning frameworks a neural implementation is required. Ideally we like to use a neural network (called pre-network in the following) to predict mel-cepstral coefficients for a neutral speaker and then warping those features to another speaker. This makes the VTLN layer the last component of the network (see Figure 2). The pre-network also needs to predict the warping parameter α on a frame-wise basis. This can be seen as a fully-connected layer with a single output neuron. We pass the predicted unbounded value of α through a TanH non-linearity and multiply it with the maximum allowed value for α. In this work we use 0.2, but lower bounds as 0.15 or 0.12 can be used as well. From the now bounded α a warping matrix $\boldsymbol{A}_\alpha$ is created per frame which is used to warp the already predicted mel-cepstral coefficients ($\otimes$ in Figure 2). In a multi-speaker context the VTLN layer should have a speaker embedding input, which can be used as well in the pre-network (dotted line) to improve the prediction of the other speech features (LF$_0$, *etc.*).

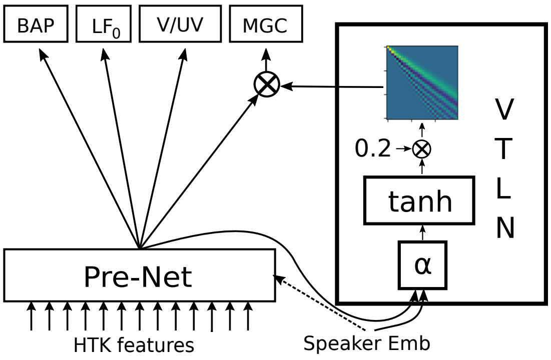

Figure 2: *Network structure with a VTLN layer. The α parameter is estimated per frame from the pre-network. The layer also has access to the speaker embedding.*

In the following we explain our implementation of a neural VTLN layer. For simplicity we express all the equations for a single input frame of mel-cepstral coefficients.

Implementing a VTLN layer in a neural network based on (1) or (2), that allows back-propagation, comes with a huge computational cost. We found that using the recursive formula (1) in the backwards pass prevents any efficient training. However, computing all the elements of matrix $\boldsymbol{A}_\alpha$ explicitly with (2) recomputes a lot of factorials each time, even though they can be treated as constants. A possible solution is to precompute $\boldsymbol{A}_\alpha$ for a range of α with a certain precision. In the forward pass we then use the weighted sum of the two precomputed matrices with the α value closest to the current input α. Even though this implementation gives good results we rejected it because it only approximates $\boldsymbol{A}_\alpha$.

The implementation we propose computes the exact $\boldsymbol{A}_\alpha$. Equation (2) is simply the sum of multiplications of constant values with a polynomial of α^k with $k = (0, 1, 2, \ldots, 2N)$, where N is again the number of mel-cepstral coefficients. This can be expressed as a dot-product of the polynomials vector $\boldsymbol{\alpha} = (1\ \alpha\ \alpha^2\ \alpha^3\ \ldots\ \alpha^{2N})$ along the third dimension of $\boldsymbol{A}^{3D}_{k,l}$, which has the size (N x N x 2N).

$$
\begin{aligned}
\boldsymbol{A}_{k,l} &= \frac{1}{(l-1)!} \sum_{\substack{n= \\ \max(0,l-k)}}^{l} \binom{l}{n} \frac{(k+n-1)!}{(k+n-l)!} (-1)^{n+l+k} \alpha^{2n+k-l} \\
&= \boldsymbol{A}^{3D}_{k,l} \boldsymbol{\alpha}
\end{aligned}
$$

$$
\boldsymbol{A}^{3D}_{k,l,2n+k-l} = \begin{cases} \frac{1}{(l-1)!} \binom{l}{n} \frac{(k+n-1)!}{(k+n-l)!} (-1)^{n+l+k} & \text{if } l\text{-}k \leq n \leq l, \\ & n \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{4}
$$

The matrix $\boldsymbol{A}^{3D}$ is constant for all α and is pre-computed when the VTLN layer is created. In the forward pass we create the polynomials vector $\boldsymbol{\alpha}$ and compute its dot-product along the third dimension of matrix $\boldsymbol{A}^{3D}$. This gives the warping matrix $\boldsymbol{A}_\alpha$ for the given α value, which is then used to warp one frame of mel-cepstral coefficients $\boldsymbol{x}$. We are using the PyTorch back-end and because we rely only on tensor operations to create $\boldsymbol{\alpha}$ and treating $\boldsymbol{A}^{3D}$ as a constant the gradient is automatically computed by Autograd. As we have mentioned in the beginning these equations hold for a single frame of mel-cepstral coefficients. However, note that the same computations apply for multiple frames and can efficiently be done in parallel for a varying number of frames and mini batch size. This is often implemented as highly parallelized operations called batched matrix-vector and batched matrix-matrix operations in modern matrix computation frameworks. Our implementation contains a single solely sequential operation, which is the creation of the $\boldsymbol{\alpha}$ vector. Even though its creation is sequential it is only sequential per frame but parallel throughout all the frames.

Our implementation becomes numerically unstable for big numbers of mel-cepstral coefficients. In particular the matrix $\boldsymbol{A}^{3D}$ contains very high values, due to the factorials, which are then multiplied by very small values in $\boldsymbol{\alpha}$, due to the high polynomials. We compared the $\boldsymbol{A}_\alpha$ matrix computed by our method with the same matrix computed recursively with (1) and also their gradient. Up to $N = 35$ mel-cepstral coefficients the error of our implementation is $< 10^{-8}$ for the values in $\boldsymbol{A}_\alpha$ and $< 10^{-5}$ for the gradients based on floating point precision. For higher numbers of mel-cepstral coefficients the errors quickly explode. We suggest to use double precision computation in those cases.

4. Experimental Proof of Concept

For our experiments we use the VCTK database [14], but only the 33 speakers (~11.5h) with English/no accent. The recordings are at 16kHz and have ~400 utterances per speaker. The phone sequences are extracted from text with Festival [15] and force-aligned by context-independent HMMs using HTK [16]. The inputs are 425 text-derived binary and numerical features normalized to [0.01, 0.99]. We use the WORLD vocoder [17] (D4C edition [18]) for the extraction of LF_0, 30-dimensional MGC, and one Band Aperiodicity (BAP) at 5 ms frame step. LF_0 is interpolated before training and a binary V/UV flag is used to capture voicing information. Additionally we compute dynamic features for all but V/UV. Output features, except V/UV, are speaker-dependent mean/variance normalized. Whenever we use speaker embeddings we use 128 dimensions.

We hypothesize in general that the VTLN layer gives the network an efficient tool for speaker adaptation because of the small parameter space. With the experiments we mean to test three more specific hypotheses:

4.1 We test if the VTLN layer is capable of learning a specific phoneme- and time-dependent warping parameter and that this learned parameter actually follows the ground truth.

4.2 We test if our model achieves higher performance compared to a strong baseline system when trained on a multi-speaker database with speaker embeddings in all layers.

4.3 We test if the proposed system achieves a higher speaker similarity in a speaker adaptation task with few data compared to the same baseline system.

In the long run we are more interested in the application of VTLN in affective speech synthesis. However, as a natural first step towards this higher goal we are using it for speaker adaptation. Hence, we are not conducting an excessive comparison with state-of-the-art techniques in speaker adaptation.

4.1. Adaptation to Artificial Speaker

In this experiment we prove that the proposed model is capable of learning a specific phoneme-dependent warping parameter α_t. Estimating α_t between phonemes of different speakers is difficult and would also require to consider preceding and succeeding phonemes as well as the mood of the speakers. To make sure we know what is the desired warping parameter we create an artificial speaker from our base speaker (speaker p276, female). For that we randomly select a warping parameter between -0.2 and $+0.2$ for each phoneme and warp the MGC features of the base speaker belonging to that phoneme with it. We first train the pre-network with the base speaker samples (~20 minutes) for 25 epochs, 0.05 dropout on all layers (PyTorch implementation is used for recurrent layers), a batch size of 32, and a learning rate of 0.001. In all experiments we use a plateau scheduler with a patience of five. We then stack the VTLN layer on top of it, keep the pre-network weights fixed, and train only the VTLN layer on the artificial speaker samples for 15 epochs and the same hyper-parameters as before but with a batch size of two due to the high memory requirement of our VTLN implementation. The artificial samples are the same as used before but warped (so again ~20 minutes). We can now compare the α_t predicted internally by our model to the ground truth. Our experiments show that the VTLN layer learned to compensate about 41% of the error introduced by the artificial warping. Table 1 shows that the compensating works better in the lower bins, which is the expected behaviour of VTLN. Figure 3 shows how the internally predicted α_t (green) follows the artificial random warping parameters (red, cornered) on a phoneme-basis. This result proves that the proposed model is capable of learning the expected warping parameter in a phoneme-dependent manner, which suggests that it will also perform well for more complex dependencies between warping parameter and phoneme+context+other (global) influences.

4.2. Multi-Speaker System

In this experiment we train multi-speaker systems with 29 out of the total 33 speakers. We randomly exclude samples from training for validation and test set. We do not explicitly set specific utterances aside for testing. This means that the model is tested to produce a known utterance from a known speaker (note that the combination of the two is unseen for the network). This is necessary because the VCTK database consists of only ~400 different utterances and excluding specific utterances from

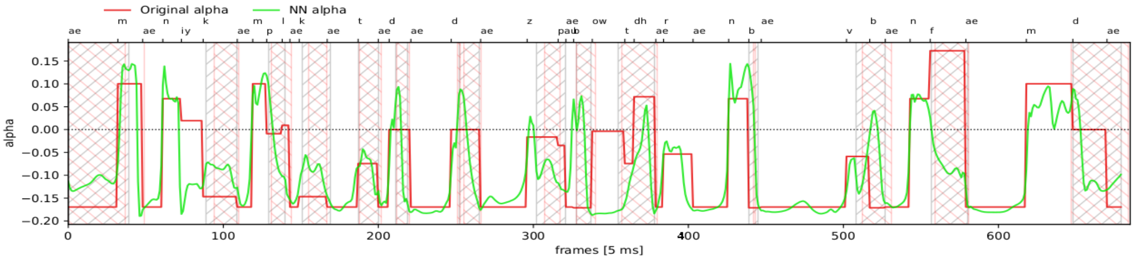

Figure 3: *Internally predicted alpha (green) against artificial alpha (red, cornered) used to create the artificial speaker on a temporal scale of 5 ms per frame. Ground truth V/UV (grey, hatched upwards), predicted V/UV (red, hatched downwards).*

Table 1: *MCD compensation of α predicted by the VTLN layer of original MGCs to artificially warped MGCs for different sets of MGC bins. First column specifies the MGC bins used for MCD computation. Second column shows the MCD of original MGCs compared to artificially warped MGCs. Third column contains the MCD of original MGCs warped with the predicted α compared to the artificially warped MGCs. Last column shows the compensation of error of column three compared to column two.*

Coef	Org [dB]	Org NN α [db]	Compensation [%]
1-10	4.03	2.30	43.0
1-11	4.24	2.43	42.9
1-12	4.41	2.53	42.7
1-13	4.56	2.61	42.7
1-18	4.97	2.92	41.3
all	6.04	3.55	41.1

training results in very low quality because the lexicon coverage is small.

As a baseline system we use a simple yet effective algorithm [19]. It consists of a model which takes an additional speaker embedding as input to all its layers. New speakers can be learned by only learning the embedding vector of the new speaker. In our experiments we use an RNN with two fully-connected layers with ReLU activation and 1024 neurons, three BiLSTM layers with 512 neurons, and a final 97 dimensional output layer [20]. We train the baseline for 15 epochs, 0.05 dropout on all layers, a batch size of 32, and a learning rate of 0.001. Each layer takes a 128 dimensional speaker embedding input. We also tried training from scratch or fine-tuning with a batch size of two, but we did not see improvements in objective scores. The VTLN model uses the baseline architecture for its pre-network and also takes the speaker embedding as input to the VTLN layer (see Figure 2). The VTLN model is trained from scratch for 25 epochs, 0.05 dropout on all layers, a batch size of two, and a learning rate of 0.001. We also train a VTLN model with a pre-trained pre-network. The pre-network is trained in the same way as the baseline system and the VTLN model is trained for another 15 epochs with the same parameters.

Table 2 shows the objective scores of the baseline system, the proposed VTLN system trained from scratch, and the proposed VTLN system trained with the baseline used as pre-network initialization. Our model outperforms the baseline in this general multi-speaker speech synthesis task in objective scores. From the objective scores the initialization does not seem to have a great effect, however, perceptually we notice a higher quality. Therefore we use the initialized system in further experiments.

Table 2: *Objective scores of multi-speaker system trained with 29 speakers.*

Model	MCD [dB]	F_0 RMSE	V/UV	BAP [dB]
Baseline	6.1	17.6	12.2%	21.3
VTLN scratch	5.3	16.3	11.6%	17.9
VTLN	5.3	16.4	11.6%	17.7

4.3. Speaker Adaptation

As a last experiment we test the two systems (baseline and VTLN with pre-network initialization) on a speaker adaptation task. We use the four speakers previously excluded from training (two male and two female). As some samples were excluded from the database after recording we make sure that we use only utterances which are available for all of the four speakers (exactly 400). Even though we randomly split the utterances we use the same utterances for all speakers in training, validation, and test set respectively. Both systems learn only the speaker embedding of the new speakers. We train both systems for 128 epochs with a learning rate of 0.01 and use early stopping to select the best model. For the objective scores we train once with 380 (~14 minutes) and once with only 10 utterances per speaker (~25 seconds). Table 3 shows the average objective scores for the speaker adaptation task. We see that our model also outperforms the baseline in this task. The high VTLN pre-network MCD shows that our model makes heavy use of its warping ability and non-zero warping parameters are learned. F_0 RMSE and BAP are better for male than female speakers revealing a shortcoming of both models for high pitch voices. More adaptation data does not lead to better objective scores. We hypothesise that the model either has learned the concept of speaker well and/or that the new speakers are close the known speakers so that ten utterances are enough to learn a proper embedding.

To evaluate the subjective quality of our model adapted with 10 utterances we conducted a preference test with 46 participants. We gave the original sample as a reference and asked the listeners to select if the baseline or our model is closer to it in terms of speaker similarity. The participant did not know which of the provided audio came from which system. The listeners could also select that they do not prefer any of the two systems. The results in Figure 4 show that our model is also subjectively superior to the baseline system. Half of the listeners preferred our model.

Table 3: *Objective scores of speaker adaptation task with four speakers (two male, two female). The scores are also separated into gender (*a*: all,* f*: female,* m*: male). VTLN pre-network is the score of the VTLN pre-network without warping.*

Model	**MCD [dB]**			**F_0 RMSE**			**V/UV [%]**			**BAP [dB]**		
	a	f	m	a	f	m	a	f	m	a	f	m
Number of adaptation / test utterances per speaker: 380 / 10[1]												
Baseline	**6.4**	6.4	6.4	21.0	26.6	15.4	13.4	13.3	13.5	20.9	22.1	19.7
VTLN	**5.7**	5.6	5.7	20.0	25.8	14.2	12.2	12.6	11.7	17.5	18.5	16.5
VTLN pre-network	12.6	12.7	12.5									
Number of adaptation / test utterances per speaker: 10 / 195												
Baseline	**6.4**	6.4	6.5	19.9	22.8	17.0	13.3	12.8	13.8	21.0	22.4	19.7
VTLN	**5.7**	5.6	5.8	18.8	21.1	16.4	12.9	13.1	12.8	17.7	18.6	16.9
VTLN pre-network	12.7	12.5	12.8									

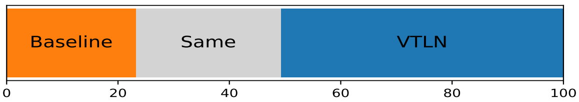

Figure 4: *Preference test: 23.2% prefer the baseline, 26.1% have no preference, 50.8% prefer the proposed VTLN system.*

5. Conclusion

In this work we proposed a neural VTLN implementation which allows efficient training and inference and is less data-hungry due to its small parameter space. On an artificial speaker with known warping parameters compared to a base speaker, we showed that the network can learn the ground truth time-dependent warping parameters. Additionally, we showed that this technique improves objective scores of a multi-speaker model and objective and subjective scores on a speaker adaptation task with few adaptation data. Even though we tested the proposed technique on a classical acoustic model we argue that it is applicable in the same way to state-of-the-art Encoder-Decoder models. In future work we aim to test the proposed technique in emotional speech synthesis where it is known that emotions cause a shift of formants.

Source code: https://github.com/idiap/IdiapTTS.

6. Acknowledgement

This work has been conducted with the support of the Swiss NSF under grant number 165545: Multilingual Affective Speech Synthesis (MASS). http://p3.snf.ch/Project-165545

7. References

[1] J. Cohen, T. Kamm, and A. G. Andreou, "Vocal tract normalization in speech recognition: Compensating for systematic speaker variability," *The Journal of the Acoustical Society of America*, vol. 97, no. 5, pp. 3246–3247, 1995.

[2] A. Zolnay, R. Schluter, and H. Ney, "Acoustic feature combination for robust speech recognition," in *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, vol. 1. IEEE, 2005, pp. I–457.

[3] D. Giuliani and M. Gerosa, "Investigating recognition of children's speech," in *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03).*, vol. 2. IEEE, 2003, pp. II–137.

[4] N. Jaitly and G. E. Hinton, "Vocal tract length perturbation (VTLP) improves speech recognition," in *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language*, vol. 117, 2013.

[5] M. Pitz and H. Ney, "Vocal tract normalization equals linear transformation in cepstral space," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 5, pp. 930–944, 2005.

[6] K. Tokuda, T. Kobayashi, T. Masuko, and S. Imai, "Mel-generalized cepstral analysis-a unified approach to speech spectral estimation," in *Third International Conference on Spoken Language Processing*, 1994.

[7] D. Sundermann and H. Ney, "VTLN-based voice conversion," in *Proceedings of the 3rd IEEE International Symposium on Signal Processing and Information Technology (IEEE Cat. No. 03EX795)*. IEEE, 2003, pp. 556–559.

[8] M. Eichner, M. Wolff, and R. Hoffmann, "Voice characteristics conversion for TTS using reverse VTLN," in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1. IEEE, 2004, pp. I–17.

[9] N. Shah, M. C. Madhavi, and H. Patil, "Unsupervised vocal tract length warped posterior features for non-parallel voice conversion," in *Proceedings of Interspeech*, 2018.

[10] L. Saheer, J. Dines, P. N. Garner, and H. Liang, "Implementation of VTLN for statistical speech synthesis," in *SSW7*, Kyoto, Japan, September 2010.

[11] L. Saheer, J. Dines, and P. N. Garner, "Vocal tract length normalization for statistical parametric speech synthesis," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 7, pp. 2134–2148, 2012.

[12] G. Kotani, D. Saito, and N. Minematsu, "Voice conversion based on deep neural networks for time-variant linear transformations," in *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2017, pp. 1259–1262.

[13] A. V. Oppenheim and D. H. Johnson, "Discrete representation of signals," *Proceedings of the IEEE*, vol. 60, no. 6, pp. 681–691, 1972.

[14] C. Veaux, J. Yamagishi, K. MacDonald *et al.*, "CSTR VCTK corpus: English multi-speaker corpus for CSTR voice cloning toolkit," *University of Edinburgh. The Centre for Speech Technology Research (CSTR)*, 2017.

[15] A. Black, P. Taylor, R. Caley, and R. Clark, "The festival speech synthesis system," 1998. [Online]. Available: http://www.cstr.ed.ac.uk/projects/festival/

[16] P. C. Woodland, J. J. Odell, V. Valtchev, and S. J. Young, "Large vocabulary continuous speech recognition using HTK." in *ICASSP (2)*, 1994, pp. 125–128.

[17] M. Morise, F. Yokomori, and K. Ozawa, "WORLD: a vocoder-based high-quality speech synthesis system for real-time applications," *IEICE TRANSACTIONS on Information and Systems*, vol. 99, no. 7, pp. 1877–1884, 2016.

[1] A reviewer has pointed out that these measurements have high variance because of small test data. We acknowledge this, retaining the table with the caveat that it is work in progress.

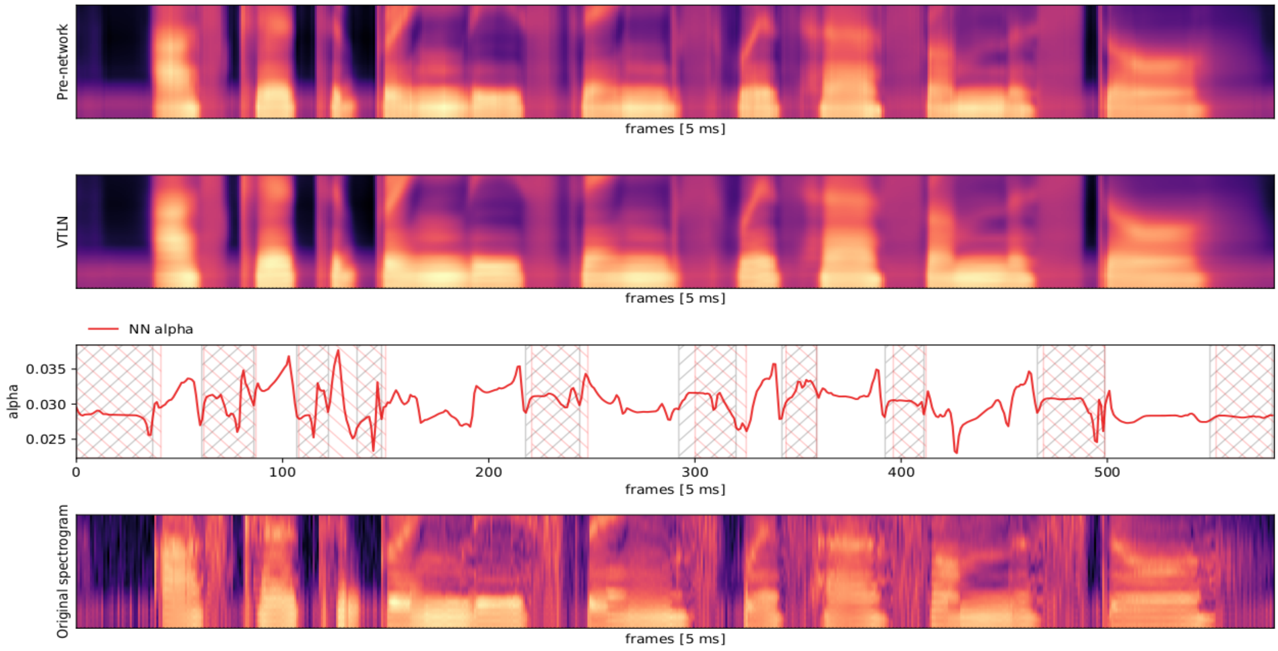

Figure 5: *First 150 bins of spectrogram of pre-network (first), final output after VTLN (second), warping parameter used (third), and original extracted from audio for utterance 002 of speaker p276 (female). The network is using only positive warpings for the female speaker. It is visible that formants are shifted slightly upwards.*

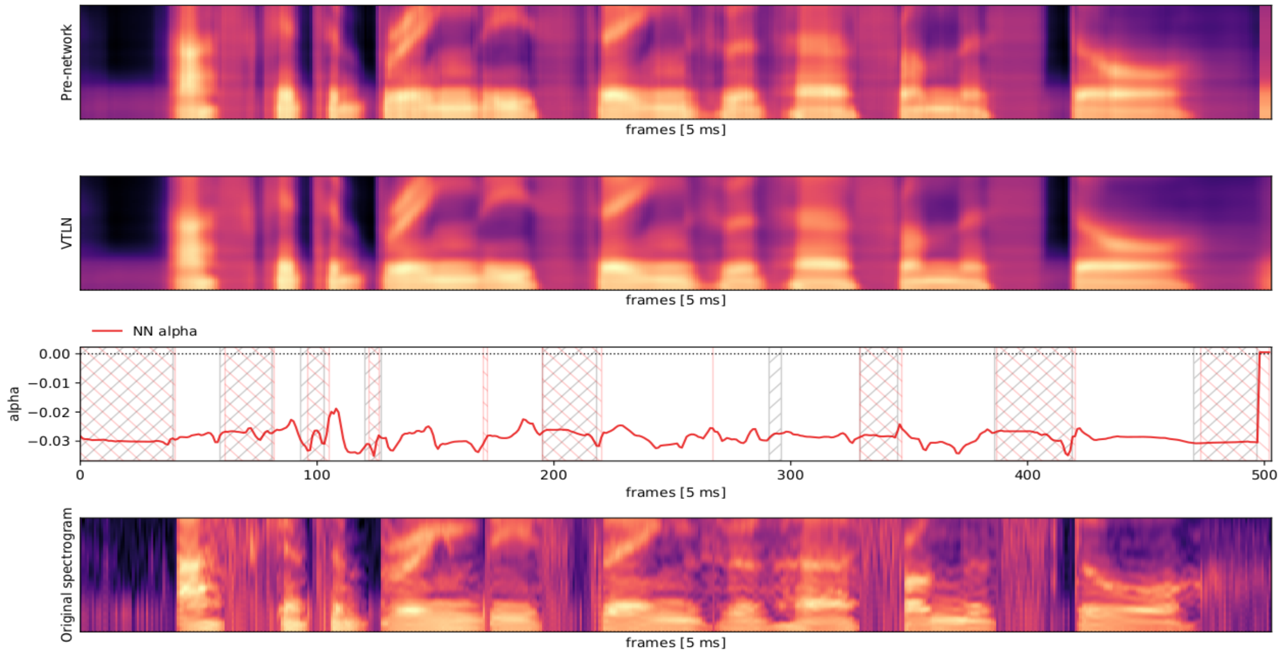

Figure 6: *Same as above but for a male speaker. The network is using only negative warpings for him. Formants are shifted slightly downwards.*

[18] M. Morise, “D4C, a band-aperiodicity estimator for high-quality speech synthesis,” *Speech Communication*, vol. 84, pp. 57–65, 2016.

[19] H. Luong, S. Takaki, G. E. Henter, and J. Yamagishi, “Adapting and controlling DNN-based speech synthesis using input codes,” in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017, pp. 4905–4909.

[20] Y. Fan, Y. Qian, F.-L. Xie, and F. K. Soong, “TTS synthesis with bidirectional LSTM based recurrent neural networks,” in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.