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

Speaker adaptation in DNN-based speech synthesis using d-vectors

Rama Doddipatla[1], *Norbert Braunschweiler*[1], *Ranniery Maia*[2],

[1]Toshiba Research Europe Limited, Cambridge Research Laboratory, Cambridge, UK.
[2]Dept. of Electrical and Electronics Engineering, Federal University of Santa Catarina, Florianópolis, Brazil.

[rama.doddipatla,norbert.braunschweiler]@crl.toshiba.co.uk, rmaia@linse.ufsc.br

Abstract

The paper presents a mechanism to perform speaker adaptation in speech synthesis based on deep neural networks (DNNs). The mechanism extracts speaker identification vectors, so-called *d-vectors*, from the training speakers and uses them jointly with the linguistic features to train a multi-speaker DNN-based text-to-speech synthesizer (DNN-TTS). The *d-vectors* are derived by applying principal component analysis (PCA) on the bottle-neck features of a speaker classifier network. At the adaptation stage, three variants are explored: (1) *d-vectors* calculated using data from the target speaker, or (2) *d-vectors* calculated as a weighted sum of *d-vectors* from training speakers, or (3) *d-vectors* calculated as an average of the above two approaches. The proposed method of unsupervised adaptation using the *d-vector* is compared with the commonly used *i-vector* based approach for speaker adaptation. Listening tests show that: (1) for speech quality, the *d-vector* based approach is significantly preferred over the *i-vector* based approach. All the *d-vector* variants perform similar for speech quality; (2) for speaker similarity, both *d-vector* and *i-vector* based adaptation were found to perform similar, except a small significant preference for the *d-vector* calculated as an average over the *i-vector*.
Index Terms: speech synthesis, speaker adaptation, i-vectors, d-vectors.

1. Introduction

Statistical parametric speech synthesis (SPSS) [1] has advantages when compared with speech synthesis methods based on unit selection and concatenation [2]. One of the key advantages is the ability to adapt to different voices using a small database. Recently, important advances on SPSS have been achieved with the use of deep learning, e.g. [3, 4, 5, 6]. The vast recent literature shows that the utilisation of deep learning not only improves quality on existing SPSS vocoding frameworks but also stimulates the use of different methods to improve speech modelling, based on speech recognition approaches.

Although a significant quality improvement has already been reached with deep learning for SPSS, flexibility at the same level that has been achieved so far using hidden semi-Markov models (HSMMs) [7] is still an open problem. Recently, there has been increased effort to perform adaptation in deep neural network (DNN) based SPSS (DNN-TTS) [8, 9, 10, 11, 12, 13]. In [8], the authors study three different forms of conducting speaker adaptation under a DNN-TTS framework: (1) using *i-vectors* [14] as speaker-id input feature for the DNN; (2) network manipulation; and (3) conversion of the predicted features using Gaussian mixture models (GMMs). They also test a combination of these three methods. The authors conclude that the most effective of the approaches is simply applying voice conversion techniques at the output feature level, while the use of *i-vectors* together with input labels was the less effective. In [12], *i-vectors* have been shown to improve the quality of synthetic speech and enable to control the speaker identity. Following a different concept, in [9] the authors propose an approach which uses speaker dependent output layers on a multi-speaker DNN-TTS system. Later, using a similar philosophy, the same authors proposed a speaker and language factorisation method [11], with the same functionality of the equivalent HSMM-based approach shown in [15], which was eventually applied to a talking head [16].

This paper presents an approach for unsupervised speaker adaptation in DNN-TTS, based on the use of a speaker-id information extracted through a neural network. The speaker-id is considered in the form of a single vector and is referred to as *d-vector* [17]. The motivation for the use of *d-vectors* is based on the results reported in [17], in which the authors verified that *d-vectors* outperformed the popular *i-vectors* in speaker verification tasks. In our proposed adaptive DNN-TTS system, at training time, *d-vectors* are extracted from a multi-speaker database and used as input together with linguistic features to train a multi-speaker DNN-TTS system, following the same concept of the work reported in [8] using *i-vectors*. At adaptation time, *d-vectors* are estimated for the target speaker and presented along with the linguistic features to synthesise speech for the target speaker. The paper primarily investigates speaker representation in the form of *d-vectors* and compares with the commonly used *i-vector* based speaker representation for speaker adaptation in DNN-TTS. Listening tests were conducted to evaluate the performance of both these systems.

This paper is organised as follows. Section 2 describes our method to train, estimate, and calculate *d-vectors*. Section 3 provides a brief overview of *i-vectors*. Section 4 shows how the proposed adaptive DNN-TTS system works in detail. Section 5 presents our experiments, followed by conclusions in Section 7.

2. *d-vector* for adaptive DNN-TTS

For *d-vector* estimation and calculation, a speaker identification task using a neural network is set, as illustrated in Fig. 1, where speech parameters are input and speaker's probabilities are output. This section describes the processes of training and estimation of *d-vectors* and follows the idea presented in [17].

2.1. Training

At training time, speech parameters $\boldsymbol{O}^{(s)}$ are extracted from a multi-speaker training database and used as input, where s represents the speaker's index. Each frame at time t, $\boldsymbol{o}_t^{(s)}$ contains parameters used to train a typical statistical parametric speech synthesis system, i.e., $\boldsymbol{o}_t^{(s)} = \left[\boldsymbol{c}_t^{(s)\top} \quad \boldsymbol{f}_t^{(s)\top} \quad \boldsymbol{b}_t^{(s)\top} \quad \boldsymbol{\phi}_t^{(s)\top} \quad v_t^{(s)}\right]^\top$, where $\boldsymbol{c}_t^{(s)}$, $\boldsymbol{f}_t^{(s)}$,

http://dx.doi.org/10.21437/Interspeech.2017-1038

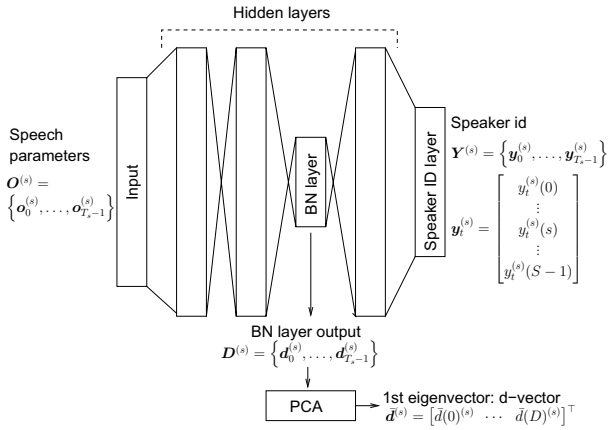

Figure 1: *Method for d-vector extraction based on a speaker identification task.*

$\boldsymbol{b}_t^{(s)}$, $\boldsymbol{\phi}_t^{(s)}$ are vectors that contain respectively the following entities with their respective dynamic parameters: (1) spectral parameters; (2) logarithm of continuous F_0; (3) band-aperiodicity; and (4) phase features for the t-th frame of speaker s. $v_t^{(s)}$ is a corresponding voicing flag, where $v_t^{(s)} = 0$ if the frame is unvoiced, and $v_t^{(s)} = 1$ if the frame is voiced. At the output, the posterior probabilities $\boldsymbol{Y}^{(s)} = \left\{\boldsymbol{y}_0^{(s)}, \ldots, \boldsymbol{y}_{T_s}^{(s)}\right\}$, are set such as $\boldsymbol{y}_t^{(s)} = \left[y_t^{(s)}(0) \quad \cdots \quad y_t^{(s)}(S-1)\right]^\top$, where S is the number of training speakers, and $y_t^{(s)}(k) = 1$, if $k = s$, or $y_t^{(s)}(k) = 0$, if $k \neq s$. T_s is the number of frames for voice s.

2.2. *d-vector* estimation

Once training is finished, the network of Fig. 1 is used for *d-vector* estimation. For that, the layers after the bottle-neck (BN) are dropped and the bottle-neck output is taken into account. It is important to note that the BN layer is linear and does not have a Sigmoid activation layer. For each speaker s, speech parameters $\boldsymbol{O}^{(s)}$ are fed to the trained network and a sequence of frame-based speaker-dependent vectors are considered, $\boldsymbol{D}^{(s)} = \left\{\boldsymbol{d}_0^{(s)}, \ldots, \boldsymbol{d}_{T_s-1}^{(s)}\right\}$ having a dimension D. In [17], a *d-vector* is obtained by taking the average of the bottle-neck output over all the frames from a given speaker. Here we apply principal component analysis on $\boldsymbol{D}^{(s)}$. The final estimated *d-vector* for the corresponding speaker, $\bar{\boldsymbol{d}}^{(s)}$, is then taken as the first eigenvector [18].

3. *i-vector* for adaptive DNN-TTS

The proposed approach using *d-vectors* in this paper is compared with the commonly used *i-vector* based approach for deriving speaker representations. This section presents a brief overview of *i-vectors* to derive speaker identity.

The mean supervector x of a speaker dependent Gaussian mixture model (GMM) can be represented as:

$$x_s \approx y + Ai_s, \quad i \sim \mathcal{N}(0, I) \tag{1}$$

where y is the mean supervector of the speaker independent universal background model (UBM) and A is the total variability matrix estimated on the background data. i is the speaker identity vector, also called the *i-vector*, which is obtained by maximum aposteriori (MAP) estimation given the speech segments

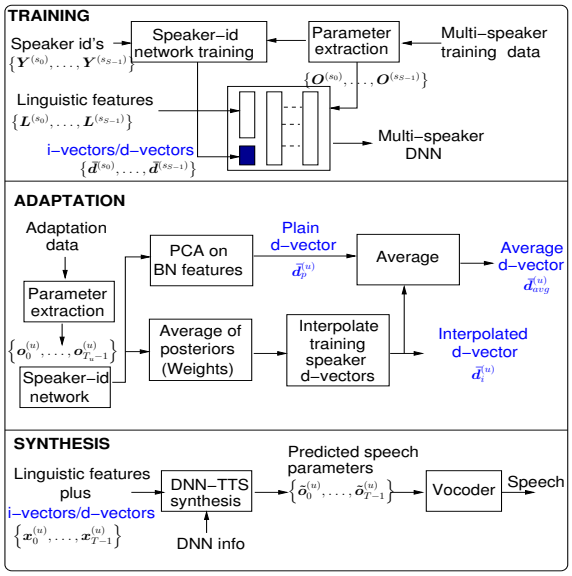

Figure 2: *Adaptive DNN-TTS system.*

from speaker s [19]. More details about *i-vectors* can be found in [14].

4. Adaptive DNN-TTS using speaker representations

The adaptive DNN-TTS system using speaker representations as inputs is composed of three parts, as illustrated in Fig. 2: (1) training; (2) adaptation; (3) synthesis. In the following, each one of them is explained in detail[1].

4.1. Training

DNN-TTS training uses linguistic information as input, $\boldsymbol{L}^{(s)} = \left\{\boldsymbol{l}_0^{(s)}, \ldots, \boldsymbol{l}_{T_s-1}^{(s)}\right\}$, where $\boldsymbol{l}_t^{(s)}$ is a vector of linguistic information from speaker s at frame t. These features are concatenated with the corresponding *d-vector* ($\bar{\boldsymbol{d}}^{(s)}$) or *i-vector* ($\boldsymbol{i}^{(s)}$) at the frame level. Therefore, the DNN input is $\boldsymbol{X}^{(s)} = \left\{\boldsymbol{x}_0^{(s)}, \ldots, \boldsymbol{x}_{T_s-1}^{(s)}\right\}$, where each input vector is given by $\boldsymbol{x}_t^{(s)} = \left[\boldsymbol{l}_t^{(s)\top} \quad \bar{\boldsymbol{d}}^{(s)\top}\right]^\top$ or $\boldsymbol{x}_t^{(s)} = \left[\boldsymbol{l}_t^{(s)\top} \quad \boldsymbol{i}^{(s)\top}\right]^\top$.

4.2. Adaptation

In the proposed DNN-TTS system for speaker adaptation, three variants of *d-vectors* are explored for use as speaker representations at adaptation time: (1) *d-vectors* obtained for the test speakers using the speaker-id network, referred to as *plain d-vector* in our discussion. (2) *d-vectors* for the test speakers obtained by interpolating the *d-vectors* of the speakers in the training set, referred to as *interpolated d-vectors* in our discussion. (3) *d-vectors* for the test speakers obtained by averaging the *d-vectors* obtained in (1) and (2), referred to as *average d-vectors* in our discussion. The performance of the above systems are compared with the *i-vector* based system. All the approaches start by extracting speech parameters $\boldsymbol{O}^{(u)}$, where u is the target speaker. The *d-vector* system uses the speech parameters described in Section 2.1, while the *i-vector* system uses MFCC

[1]Note that in Fig. 2 we use multi-speaker notation: $s = \{s_0, \ldots, s_{S-1}\}$.

features as speech parameters.

4.2.1. Adaptation using plain d-vectors

The *plain d-vectors* $\bar{\boldsymbol{d}}_p^{(u)}$ for the target speaker are estimated in the same manner as estimated for the training speakers. The speech parameters of the target speaker $\boldsymbol{O}^{(u)}$ are propagated through the trained speaker-id network to extract the bottle-neck features $\boldsymbol{D}^{(u)}$. PCA is then applied on the bottle-neck features $\boldsymbol{D}^{(u)}$ and the first eigenvector is considered as the *plain d-vector*.

4.2.2. Adaptation using interpolated d-vectors

In this approach, the *d-vector* for the target speaker is calculated by interpolating the *d-vectors* of the training speakers. This is given by:

$$\bar{\boldsymbol{d}}_{intp}^{(u)} = \sum_{s=0}^{S-1} \lambda^{(u)}(s)\bar{\boldsymbol{d}}^{(s)}, \tag{2}$$

where S is the number of training speakers, and the weights $\lambda^{(u)}(s)$ correspond to the means of the posteriors (output) of the *d-vector* neural network when $\boldsymbol{O}^{(u)}$ is input. This is given by:

$$\lambda^{(u)}(s) = \frac{1}{T_u} \sum_{t=0}^{T_u-1} y_t^{(u)}(s). \tag{3}$$

The calculated *d-vector* for the target speaker is referred to as *interpolated d-vector*.

4.2.3. Adaptation using average d-vectors

In this approach, the *d-vector* for the target speaker is calculated as the average of the *d-vectors* obtained in Sections 4.2.1 and 4.2.2 respectively. This is given by:

$$\bar{\boldsymbol{d}}_{avrg}^{(u)} = (\bar{\boldsymbol{d}}_p^{(u)} + \bar{\boldsymbol{d}}_{intp}^{(u)})/2. \tag{4}$$

All these *d-vector* variants can be used as speaker representations for the target speaker for performing speaker adaptation in DNN-TTS. For the experiments in this paper, we compare the performance of these approaches with the commonly used *i-vector* based approach to represent the target speaker. It is important to note that the *i-vector* and *d-vector* based approaches perform unsupervised speaker adaptation in DNN-TTS and do not require adaptation data labels. Speaker representations are derived by only using the information extracted from the audio data from the target speaker.

4.3. Synthesis

Speech synthesis with the target voice is done by using the multi-speaker DNN-TTS fed by linguistic features from the text to be synthesised together with the estimated *d-vector* or *i-vector* of the target speakers as described in the previous section. Once the speech features are retrieved, they are then un-normalised, smoothed, and applied to the vocoder in order to produce synthetic speech.

5. Experimental Setup

Experiments were conducted on a multi-speaker database. The data is comprised of studio recordings from 20 American English speakers, 10 male and 10 female, sampled at 22.05 kHz. 18 speakers were used for training, while two speakers, one female and one male, were left out as target speakers for adaptation. The total amount of training data was roughly 20 hours, while approximately 10 minutes of each target speaker was used for adaptation.

From the speech material, F_0 was extracted using the algorithm presented in [20], while 40 mel-cepstral coefficients and 39 phase features were derived using complex cepstrum analysis [21]. Aperiodicity features were also extracted using pitch-synchronous warped amplitude spectra of the voiced and unvoiced components of speech. Then, from the aperiodicity features 20 mel band-aperiodicity coefficients were obtained. For complex cepstrum analysis and aperiodicity extraction, glottal closure instants were detected using the algorithm in [22]. All the features had delta parameters aside from $\ln F_0$, which also included second order delta-delta. The voicing flag had no dynamic feature. The $\ln F_0$ values in the unvoiced regions were obtained through cubic interpolation to result into final continuous $\ln F_0$ trajectories. The total dimension of each $\boldsymbol{o}_t^{(s)}$ was 202.

The *d-vector* neural network had 3 hidden layers, with the bottle-neck (BN) layer placed before the final hidden layer. The BN layer has 6 neurons while the hidden layers have 1024 neurons each. The soft-max output layer had 18 neurons corresponding to the number of training speakers. The network was trained using cross-entropy training criterion and convergence was achieved after 18 epochs. The *i-vector* system was trained using MFCC features, where the UBM had 512 Gaussian mixtures. Both *d-vector* neural network and *i-vector* systems were trained using the KALDI toolkit [23] and had 6 dimension representations for each speaker. The adaptive DNN-TTS had 6 hidden layers, each one with 1024 neurons, and the number of inputs was $L = 519$ linguistic features plus $D = 6$ *d-vector* or *i-vector* components (total 525). The input linguistic features were normalised between $[0.01; 0.99]$ while the output was normalised so as to have mean zero and variance one. Alignments to train the DNN-TTS were provided by HSMM monophone models. DNN training was performed using the mean squared error (MSE) criterion, the number of warm-up epochs was set to 10, and convergence was obtained after 25 epochs.

At synthesis time, all the samples were synthesised by the complex cepstrum vocoder [21]. DNN outputs were un-normalised and a parameter generation algorithm [24] applied in order to smooth the trajectories. For un-normalisation and parameter generation, mean and variances of the adaptation data was used. Variance scaling was applied to the final mel-cepstrum trajectories [25].

6. Results and Discussion

Two listening tests were conducted to evaluate the perceptual impact of the proposed adaptation techniques: one for evaluating *speech quality* and another for evaluating *similarity to the target speaker*.

The *speech quality* test contrasted synthesised samples from individual systems in a preference test. The systems under consideration used the adaptation methods presented in Section 2 and are named as follows: (1) D-VEC - adaptation using *plain d-vectors*, (2) I-VEC - adaptation using *i-vectors*, (3) D-VEC-INTP - adaptation using *interpolated d-vectors*, and (4) D-VEC-AVRG - adaptation using *average d-vectors*.

Listeners were asked: "Please choose which sample, A or B, sounds better". Subjects had the option to choose 'neither'. 20 paid subjects took part in the test, all of them English native speakers with no reported hearing impairments. The listening test was conducted at the University of Edinburgh under controlled conditions and listeners did wear headphones. To

Table 1: *Synthetic speech quality. Results are in percentage of choice by all subjects.*

Speaker	D-VEC	I-VEC	Neither	p-value
Male	52.30%	41.12%	6.85%	$p = 0.0434$
Female	62.66%	22.08%	15.26%	$p < 0.001$
ALL	**57.52%**	31.54%	10.95%	$p < 0.001$
	D-VEC	D-VEC-INTP		
Male	51.80%	41.97%	6.23%	$p = 0.0760$
Female	37.87%	50.17%	11.96%	$p = 0.0227$
ALL	44.88%	46.04%	9.08%	$\mathbf{p = 0.7658}$
	D-VEC	D-VEC-AVRG		
Male	34.87%	36.18%	28.95%	$p = 0.7862$
Female	34.12%	39.19%	26.69%	$p = 0.3096$
ALL	34.50%	37.67%	27.83%	$\mathbf{p = 0.3618}$
	I-VEC	D-VEC-AVRG		
Male	33.22%	59.87%	6.91%	$p < 0.001$
Female	16.33%	76.53%	7.14%	$p < 0.001$
ALL	24.92%	**68.06%**	7.02%	$p < 0.001$

Table 2: *Similarity to target speaker. Results are in percentage of choice by all subjects.*

Speaker	D-VEC	I-VEC	Neither	p-value
Male	31.25%	45.94%	22.81%	$p = 0.0026$
Female	41.56%	34.06%	24.38%	$p = 0.1231$
ALL	36.41%	40.00%	23.59%	$\mathbf{p = 0.2987}$
	I-VEC	D-VEC-AVRG		
Male	38.75%	37.81%	23.44%	$p = 0.8484$
Female	33.44%	48.44%	18.12%	$p = 0.0028$
ALL	36.09%	**43.12%**	20.78%	$p = 0.0455$

also test the impact of using different adaptation speakers, each adaptation method was used to synthesise 4 different speakers (2 male, 2 female). These speakers were realised by rotating training and testing speakers in the multi-speaker DNN-TTS.

8 sentences were synthesised from a test set not used in training. Each subject listened to all 128 stimuli pairs (4 systems $\times$ 8 sentences $\times$ 4 speakers). Results are shown per gender and for all speakers combined for each system contrast in Table 1. To measure statistical significance, t-tests were conducted, under the assumption that both systems were expected to achieve 50% of preference. This was regarded as the null hypothesis. Two systems were considered as significantly different when the p-scores were smaller than 0.5.

The results of the speech quality test show, that the *d-vector* based adaptation method is significantly preferred over the *i-vector* based adaptation method. Also the D-VEC-AVRG system is significantly preferred over the I-VEC system. However, there was no significant difference between systems D-VEC and D-VEC-INTP nor between D-VEC and D-VEC-AVRG.

Fig. 3 shows the weights obtained for one female and one male target speaker for calculating the interpolated *d-vectors* discussed in Section 4.2.2. It can be seen that for the female speaker (top), weights are higher in the female speakers zone, while the equivalent happens for the male speaker. It is interesting to note that D-VEC and D-VEC-INTP perform very similar despite the fact that they are calculated in very different ways. D-VEC is estimated using data from the target speaker while D-VEC-INTP interpolates training speakers *d-vectors*.

In the speaker similarity test, for each test sentence, first the natural speech was played followed by the two synthetic versions in randomised order. Subjects were asked which of the two samples did sound more similar to the reference speaker.

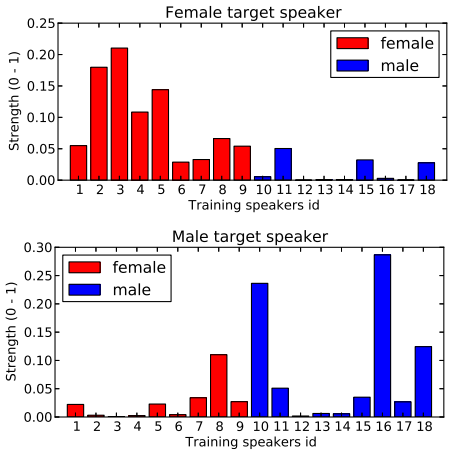

Figure 3: *Weights obtained from the speaker-id network for interpolating the training speaker d-vectors to derive target female (top) and male (bottom) speaker interpolated d-vectors.*

Only a subset of systems was selected based on listening impressions from the first test, including only systems for which a difference in similarity could be expected. Results are shown in Table 2. Across all speakers, speaker similarity was not found to differ significantly between D-VEC and I-VEC systems. However, when separating male and female speakers, a significant preference for the I-VEC system is observed in case of the male speakers, but not for the female speakers. Looking at the individual male speakers results showed one was significantly preferred in the I-VEC version while the other male speaker did not show any significant difference.

In terms of the difference between I-VEC and D-VEC-AVRG systems, across all speakers, a narrow significant preference for system D-VEC-AVRG is observed. However, separating genders shows again differences, i.e. a significant difference was only observed for female speakers while the male speakers showed no significant difference. Similar to the previous contrast, the significant difference for female speakers was a result of one speaker being more strongly preferred in the D-VEC-AVRG version while the other female speaker did not show any significant difference.

Overall, *d-vector* based approaches perform similar or better than the *i-vector* based approaches for speaker similarity. For speech quality there is a clear preference for the *d-vector* based approaches.

7. Conclusions

The paper presented an approach to derive speaker representations in the form of *d-vectors*, that are used as inputs along with linguistic features to perform unsupervised speaker adaptation in DNN-TTS. Three variants of *d-vectors* as speaker representations during synthesis time were explored and their performance was compared with the commonly used *i-vector* approach in listening tests. The results showed that the *d-vector* approach has a significant preference over the *i-vector* approach for speech quality. All the *d-vector* variants performed similar in terms of speech quality. On the other hand, both *d-vectors* and *i-vectors* performed very close in terms of speaker similarity, except a small significant preference for the *d-vector* calculated as average over the *i-vector*.

8. References

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

[2] A. J. Hunt and A. W. Black, "Unit selection in a concatenative speech synthesis system using a large speech database," in *Proc. of ICASSP*, 1996, pp. 373–376.

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

[4] Y. Fan, Y. Qian, F. Xie, and F. K. Soong, "TTS synthesis with bidirectional LSTM based recurrent neural networks," in *Proc. of Interspeech*, 2014, pp. 1964–1968.

[5] H. Zen and H. Sak, "Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis," in *Proc. of ICASSP*, 2015, pp. 4470–4474.

[6] Z. Wu, C. Valentini-Botinhao, O. Watts, and S. King, "Deep neural network employing multi-task learning and stacked bottleneck features for speech synthesis," in *Proc. of ICASSP*, 2015, pp. 4460–4464.

[7] J. Yamagishi, T. Kobayashi, Y. Nakano, K. Ogata, and J. Isogai, "Analysis of speaker adaptation algorithms for HMM-based speech synthesis and a constrained SMAPLR adaptation algorithm," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 17, no. 1, pp. 66–83, Jan. 2009.

[8] Z. Wu, P. Swietojanski, C. Veaux, S. Renals, and S. King, "A study of speaker adaptation for DNN-based speech synthesis," in *Proc. of Interspeech*, 2015, pp. 879–883.

[9] Y. Fan, Y. Qian, F. Soong, and L. He, "Multi-speaker modeling and speaker adaptation for DNN-based TTS synthesis," in *Proc. of ICASSP*, 2015, pp. 4475–4479.

[10] ——, "Unsupervised speaker adaptation for DNN-based TTS synthesis," in *Proc. of ICASSP*, 2016, pp. 5135–5139.

[11] ——, "Speaker and language factorization in DNN-based TTS synthesis," in *Proc. of ICASSP*, 2016, pp. 5540–5544.

[12] Y. Zhao, D. Saito, and N. Minematsu, "Speaker representations for speaker adaptation in multiple speakers' BLSTM-RNN-based speech synthesis," in *Proc. of Interspeech*, 2016, pp. 2268–2272.

[13] N. Hojo, Y. Ijima, and H. Mizuno, "An investigation of DNN-based speech synthesis using speaker codes," in *Proc. of Interspeech*, 2016, pp. 2278–2282.

[14] N. Derak, P. J. Kenny, R. Derak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, May 2011.

[15] H. Zen, N. Braunschweiler, S. Buchholz, M. Gales, K. Knill, S. Krstulovic, and J. Latorre, "Statistical parametric speech synthesis based on speaker and language factorization," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 6, pp. 1713–1724, Aug. 2012.

[16] V. Wan, R. Anderson, A. Blokland, N. Braunschweiler, L. Chen, B. Kolluru, J. Latorre, R. Maia, B. Stenger, K. Yanagisawa, Y. Stylianou, M. Akamine, M. J. F. Gales, and R. Cipolla, "Photo-realistic expressive text to talking head synthesis," in *Proc. of Interspeech*, 2013, pp. 2667–2669.

[17] E. Variani, X. Lei, E. McDermott, I. Lopez Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *Proc. of ICASSP*, 2014, pp. 4052–4056.

[18] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[19] Y. Miao, H. Zhang, and F. Metze, "Towards speaker adaptive training of deep neural network acoustic models," in *Proc. of Interspeech*, 2014, pp. 2189–2193.

[20] A. Camacho, "SWIPE: A sawtooth waveform inspired pitch estimator for speech and music," Ph.D. dissertation, University of Florida, 2007.

[21] R. Maia, M. Akamine, and M. Gales, "Complex cepstrum for statistical parametric speech synthesis," *Speech Communication*, vol. 5, no. 55, pp. 606–618, Jun. 2013.

[22] P. Naylor, A. Kounoudes, J. Gudnason, and M. Brookes, "Estimation of glottal closure instants in voiced speech using the DYPSA algorithm," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 1, pp. 34–43, Jan. 2007.

[23] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011.

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

[25] H. Silén, E. Hel, J. Nurminen, and M. Gabbouj, "Ways to implement global variance in statistical speech synthesis," in *Proc. of Interspeech*, 2012.