Predicting Articulatory Movement from Text Using Deep Architecture with Stacked Bottleneck Features

Zhen Wei* Zhizheng Wu‡ Lei Xie*†
* School of Software and Microelectronics, Northwestern Polytechnical University, Xi'an, China
† School of Computer Science, Northwestern Polytechnical University, Xi'an, China
E-mail: {zhwei, lxie} @nwpu-aslp.org
‡ The Centre for Speech Technology Research, University of Edinburgh, U.K.
E-mail: wuzhizheng@gmail.com

***Abstract*—Using speech or text to predict articulatory movements can have potential benefits for speech related applications. Many approaches have been proposed to solve the acoustic-to-articulatory inversion problem, which is much more than the exploration for predicting articulatory movements from text. In this paper, we investigate the feasibility of using deep neural network (DNN) for articulartory movement prediction from text. We also combine full-context features, state and phone information with stacked bottleneck features which provide wide linguistic context as network input, to improve the performance of articulatory movements' prediction. We show on the MNGU0 data set that our DNN approach achieves a root mean-squared error (RMSE) of 0.7370 mm, the lowest RMSE reported in the literature. We also confirmed the effectiveness of stacked bottleneck features, which could include important contextual information.**

Index Terms: articulatory movement prediction, stacked bottleneck features, deep neural network

I. Introduction

Humans use articulatory movements, involving systematic combinations of motions from tongue, jaw, lips, velum and other articulators, to produce sound. In practice, as an effective description of speech, articulatory movements, or so-called articulatory features, are known to be quite useful in many practical applications. In speech recognition, articulatory features can improve the recognition performance [1], [2] by providing additional speech production information. In speech synthesis, articulatory features can supplement text to improve the voice quality or to retouch the characteristics of synthesized voice [3], [4]. In talking-head animation, articulatory features can be regarded as an intermediate parametrization of speech that has a close link with facial movements [5], [6]. More practically, it can be used to help language learners to correct pronunciation and find the pronunciation defects. Usually, human articulography, e.g., electromagnetic articulography (EMA) [7] can be used to acquire articulatory movements, but with a cumbersome setup and a complicated recording procedure. Due to the complex procedure and professional facilities, using recorded articulatory movements cannot be popularized, which leads to many trials on predicting them from text or speech.

In this study, we aim to predict articulatory movements from text. In this area, various methods have been previously proposed, and here we just review a few of the most influential ones. In [8], a Gaussian distribution model of articulator positions at phone midpoints together with an explicit coarticulation model was adopted to predict articulatory movements from time-aligned phone strings. In [9], each kinematic triphone model was characterised by three kinematic features of a tri-phone and by the intervals between two successive phones in the tri-phone. Similar to statistical parametric speech synthesis (SPSS), hidden Markov model (HMM) - based articulatory prediction usually adopts a rich set of features, which may include both linguistic and prosodic representations. In [10], HMM was proven to be quite useful in articulatory prediction from text (and speech), and the combination of text and acoustic features led to the best result in prediction accuracy.

Recently, due to the tremendous success in speech recognition [11], [12] and synthesis [13], [14], deep neural networks (i.e., neural networks with multiple hidden layers), have been introduced to solve the articulatory prediction problem. As an early attempt, Uria *et al.* [15] investigated a DNN and a deep version of the trajectory mixture density network (TMDN) in articulatory prediction from acoustic input. By using a pre-defined fixed-length context window which covers several frames of acoustic features as the network input, the important speech context information is modelled. Their approach achieves state-of-the-art performance in articulatory prediction from speech with an average RMSE of 0.885 mm on the MNGU0 test data set [7].

Following the success in speech-to-articulatory-movement prediction (i.e., articulatory inversion), in this paper, we investigate the feasibility of using deep neural network in predicting articulartory movements from textual input. We show on the MNGU0 data set that our DNN approach achieves a root mean-squared error (RMSE) of 0.7370 mm. To the best of our knowledge, this is the lowest RMSE reported on this corpus for the text-to-articulatory-movement prediction task.

In the previous study on articulatory prediction from speech input [15], to get around the frame-by-frame independence problem of a DNN, a context window covering several acoustic frames is adopted as network input. But big window might be infeasible when textual features are used as input. This is because the frame-wise full-context features can expand to several hundred dimensions. Hence, motivated by [16], [17],

[18], [19], in this paper, we propose to use *bottleneck feature stacking* to effectively model the context. Note that recurrent neural networks (RNNs) can be applied to model sequential data in the task [20], but sometimes they can be difficult or computationally expensive to optimize [16]. In contrast, stacked bottleneck features are widely used as a compact representation, modeling the context information in a simpler, but highly-effective way. Specially in our approach, we first train a DNN with a narrow bottleneck hidden layer and the activations of the bottleneck units yield a compact representation of linguistic information for each frame independently. Then multiple consecutive frames of bottleneck features are stacked to result in a wide context around the current frame. The stacked bottleneck features are combined with the full-context features, state and phone information, used as input to a second DNN that predicts the articulatory movements. On the MNGU0 data set, we confirmed the effectiveness of stacked bottleneck features with further RMSE reduction.

II. DNN for Text-to-articulatory Prediction

DNN-based text-to-articulatory prediction includes training model and generation of predicted articulatory parameters, which is shown in the Fig. 1. During training, the complex relationship between normalized input linguistic features and corresponding normalized output articulatory features, which consist of static features, and corresponding dynamic features, is learned. The dynamic features are used as a constraint to generate smooth parameter trajectories.

A sequence of linguistic features is given to the trained model during generation, and then the corresponding articulatory features can be generated by performing a forward propagation once per frame. After denormalizing generated output, we use the maximum likelihood parameter generation (MLPG) algorithm [21], taking the dynamic feature constraints into account, to generate the smoothed parameter trajectories.

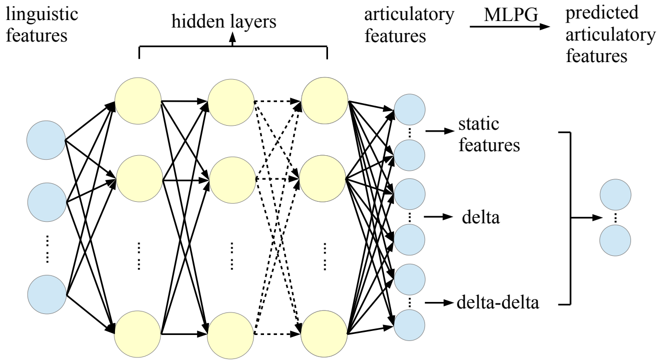

Fig. 1. Deep neural network (DNN) for text-to-articulatory prediction.

III. Stacked Bottleneck Features

Bottleneck features are the activations at a bottleneck layer in a DNN, which have a smaller number of hidden units compared with other hidden layers in the same network. Since there are far fewer hidden units in the bottleneck layer, the DNN training process forces the activation signals in this layer to form a low-dimensional compact representation of the original inputs, and reduces the redundancy of the input features. Bottleneck features have been extensively employed in automatic speech recognition (ASR) as a compact representation of acoustic features [22], [23], [24]. For speech synthesis, text-to-articulatory prediction and any other tasks using linguistic features as input, bottleneck features can be viewed as a compressive transform of linguistic features, extracted at the frame level.

Fig. 2 shows a typical architecture of a DNN that stacks several consecutive frames of bottleneck features around the current frame. A four-hidden-layer bottleneck network on the left has a bottleneck layer on the second layer near the input [16], [25]. In practice, the number of layers, the number of nodes in bottleneck layer, the training data and other settings in specific networks can be different from the example network. After bottleneck features are extracted from the left-hand network, they are then stacked as input to the network on the right, which doesn't increase the input dimensionality and the computational complexity of the right-hand network much since the dimensionality of the bottleneck features is quite small (e.g., 32).

In particular, in text-to-articulatory prediction, we can generate bottleneck features from three sources.

(1) **Linguistic-to-articulatory**: We can use linguistic features as input and articulatory features as output to train a bottleneck network, and generate bottleneck features as a compact representation of linguistic features which are more relevant to articulatory movements.

(2) **Linguistic-to-acoustic**: Given the high correlation between articulatory movements and acoustic features shown in articulatory inversion [15], [20], we use linguistic features as input and acoustic features as output to train the bottleneck network, to investigate the effect of bottleneck features generated from this method.

(3) **Linguistic-to-acoustic-from-multiple-speakers**: Acoustic features can be used as the output of the bottleneck network, and the bottleneck features may be more representative and robust if generated by the bottleneck network trained with a large amount of data. Thus we can use speech data from multiple speakers and extract bottleneck features according to the method introduced in (2).

In this paper, we will stack these three sources of bottleneck features with original linguistic features, respectively, to predict articulatory movements and compare the effectiveness of them.

IV. Experimental Setup

A. Data sets and linguistic features

Our experiments are carried out on MNGU0 database [7] with 1,263 English utterances from a single speaker recorded in a single session. Parallel recordings of acoustic data and

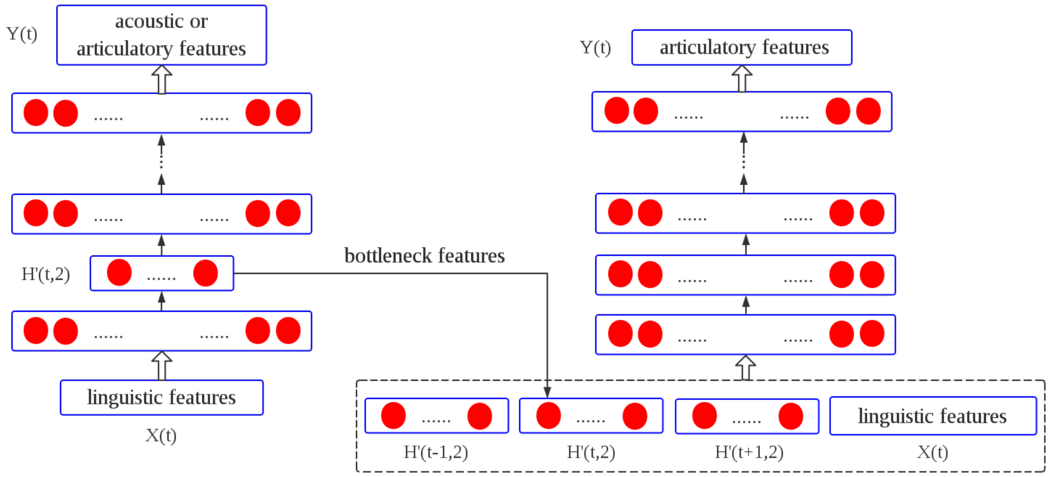

Fig. 2. Deep neural network (DNN) with stacked bottleneck features. In this example, the bottleneck features for three consecutive frames are stacked as input to the second network. In practice, more than three frames can be included. $H'(t, 2)$ is the vector of bottleneck feature for the t^{th} frame.

EMA data are available. EMA data are collected with a sampling frequency of 200Hz from 6 sensors located at the tongue dorsum, tongue body, tongue tip, lower lip, upper lip, and lower incisor. Since the movements in z-axis are very small, we only use x- and y- coordinates of the 6 receivers in the experiments. The waveforms are in 16kHz PCM format, and we use the STRAIGHT vocoder to extract 60-dimensional Mel-Cepstral Coefficients (MCCs), 25-dimensional band aperiodicities (BAPs), and log-scale fundamental frequency ($logF_0$ at a 5ms frame step). We will also use bottleneck features from bottleneck network trained by other large databases, so we use default phone list in Festival to regenerate the full-context labels for MNGU0 database instead of using its own phone list, and finally leave 1, 262 utterances to do the text-to-articulatory prediction task. The MNGU0 database is partitioned into three subsets: a training set with 1,200 utterances, a validation set comprising 32 utterances and a test set consisting of the other 30 utterances.

For all the networks used in our experiments which were trained by an open-source neural network TTS toolkit, Merlin[1] [26], the input features (with silence segments excluded) consist of 492-dimensional binary features and 9-dimensional numerical features (501 dimension in total). Following the standard configuration in DNN-based text-to-speech [16], the 492-dimensional binary features are derived from linguistic context such as quin-phone identities, position information of phoneme, syllable, word and phrase, etc.; the 9-dimensional numerical features are the frame position information, such as frame position in HMM state and phoneme, state position in phoneme, and state and phoneme durations.

B. *Deep neural network*

To verify the effectiveness of deep neural network, we use a feed-forward network trained to minimise frame-by-frame prediction error. The output features are 12-dimensional articulatory features from MNGU0 database and their corresponding delta, delta-delta features (36 dimension in total). The network has empirically set to six hidden layers, each of 256 units. The bottom layers use tangent activation function, while the output layer is a linear regression layer. Learning rate is 0.0015 and momentum is 0.3 in the first 10 epochs, and then after 10 epochs the momentum is set to 0.9. The maximum number of epochs is 25. In the experiment, the network settings including depth of network, number of nodes in hidden layer, learning rate and momentum will be the same if using MNGU0 database to train the network for text-to-articulatory prediction.

C. *DNN with stacked bottleneck features*

DNN with stacked bottleneck features splices consecutive frames of bottleneck features and linguistic features as input. Considering the difference among training data sets used in bottleneck network, as described in Section III, we divide this experiment into three parts, listed as follows:

- **DNN-BN-Arti**: We use linguistic features and articulatory features extracted from MNGU0 database to train the bottleneck network, as described in Section III (a). For the bottleneck network, the input is 501-dimensional binary features, and the corresponding output is 36-dimensional articulatory features including 12-dimensional articulatory movements with their corresponding delta and delta-delta features (36 dimension in total). There are 6 feed-forward hidden layers, each of which has 256 hidden units, except that the second hidden layer is set as the bottleneck layer, with only 32 hidden units. For the network of text-to-articulatory prediction, from only 1 frame to 21 consecutive frames (middle frame +/- 10 frames) of bottleneck features are stacked as input. Hence, the dimension of input layer increases from 533 dimension

[1] https://github.com/CSTR-Edinburgh/merlin

(32-dimensional bottleneck features + 501-dimensional linguistic features) to 1173 dimension (32-dimensional bottleneck features × 21 + 501-dimensional linguistic features).

- **DNN-BN-Acoustic**: We extract acoustic features, instead of articulatory features, from MNGU0 database as the output of the bottleneck network, as described in Section III (b). For bottleneck network, the input is 501-dimensional binary features, and the corresponding output is 259-dimensional acoustic features including 60-dimensional MCCs, 25-dimensional BAPs, 1-dimensional $logF_0$, their corresponding delta and delta-delta features, and 1-dimensional unvoiced/voiced information (259 dimension in total). The settings of the bottleneck network and the network of text-to-articulatory prediction are the same as DNN-BN-Arti.
- **DNN-BN-Acoustic-MS**: We use linguistic features and acoustic features extracted from VCTK database to train the bottleneck network, and here 'MS' means multiple speakers, as described in Section III (c). For the bottleneck network, the VCTK database[2] is used to train the model, which contains speech data from 103 speakers, including 44 male and 59 female speakers. Each speaker has around 400 utterances, and 41,294 sentences in total. We take 40,294 randomly selected sentences for model training, 500 randomly selected sentences as development set and another 500 randomly selected sentences with the 1262 utterances in MNGU0 database as test set. The input and output features are the same as that of bottleneck network in DNN-BN-Acoustic, which are 501-dimensional binary features and 259-dimensional acoustic features respectively. There are 6 feed-forward hidden layers, each of which has 1,536 hidden units, except that the second hidden layer is set as bottleneck layer, with 32 hidden units. The settings of the network for text-to-articulatory prediction are the same as DNN-BN-arti, so the dimension of input layer is from 533 dimension to 1173 dimension.

The hyper-parameters (i.e., the number of hidden layers, the number of hidden units, the learning rate) of all the neural networks are tuned on the development set.

V. Experimental Results

We employed a widely used objective measure, termed root mean-squared error (RMSE), which is the same as [10], [15], [20], to test different DNN systems:

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_i (e_i - t_i)^2}, \qquad (1)$$

where e_i is the network output and t_i is the actual value at time i. Actually, RMSE is calculated dimension by dimension, and then the average RMSE of 12 dimensions are used to assess the performance of the models.

A. Effect of DNN in text-to-articulatory prediction

The prediction results of our DNN is shown in Table I. We can see that using text-only as input in DNN has the average RMSE of 0.7370mm, while in the recent HMM approach [10] the average RMSE is 0.872mm. This means DNN brings a clear RMSE reduction. Please note that the comparison between these two methods is not direct, since the exact allocation of training and test sets used in [10] is not available. But we tried to keep the number of utterances in the training/test sets similar with that in [10].

TABLE I
Average RMSE of HMM, DNN and DNN-BN-Arti on the test set.

	HMM [10]	DNN	DNN-BN-Arti
rmse (mm)	0.872	0.7370	0.7203

TABLE II
Average RMSE of DNN-BN-Arti, DNN-BN-Acoustic and DNN-BN-Acoustic-MS on the test set.

	rmse (mm)	stacked BN features (frames)
DNN-BN-Arti	0.7203	21
DNN-BN-Acoustic	0.7257	17
DNN-BN-Acoustic-MS	0.7243	13

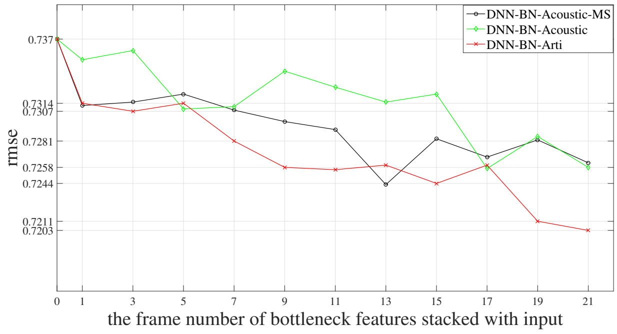

Fig. 3. Comparison of three sources for bottleneck features in test set; RMSE as a function of the number of stacked frames.

B. Effect of stacked bottleneck features

Fig. 3 shows the RMSE of three different stacked bottleneck feature sources, as a function of the number of stacked frames, while the best RMSE achieved are summarized in Tabel II. We notice that no matter how the bottleneck network (the right-hand network in Fig. 2) is trained and how many frames are stacked, stacking contextual frames with text can include richer linguistic information, and achieve better performance than using text-only feature of the current frame. Among these three sources of bottleneck features, DNN-BN-Arti, which uses MNGU0 database and sets articulatory features as output when training the bottleneck network, achieves the best result. Besides, comparing DNN-BN-Acoustic with DNN-BN-Acoustic-MS, which both use acoustic features as output when training the bottleneck network, the average RMSE of

[2] http://homepages.inf.ed.ac.uk/jyamagis/page3/page58/page58.html

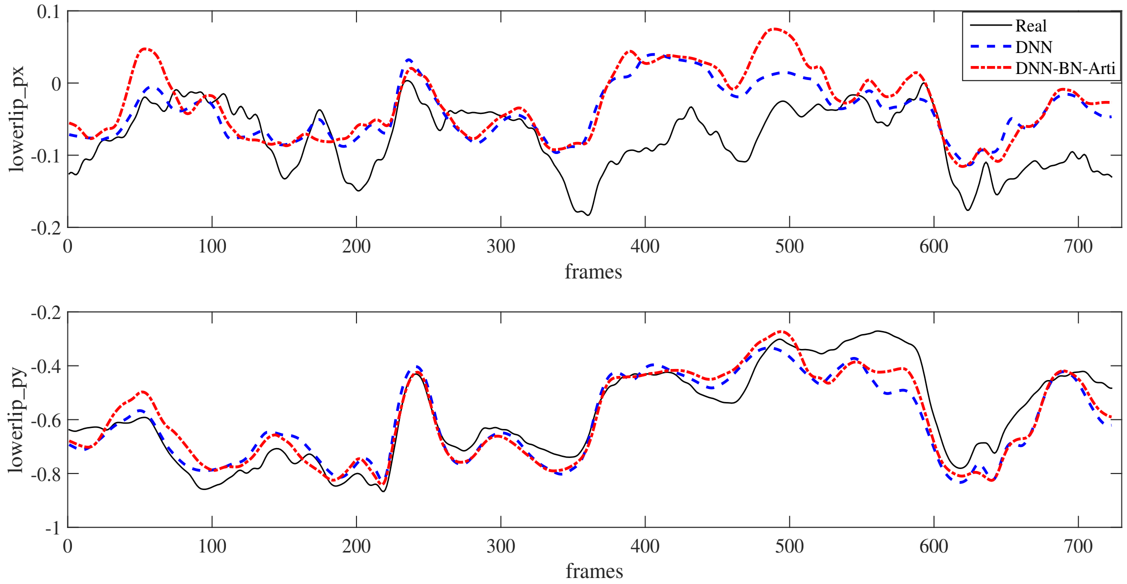

Fig. 4. Comparison of real and predicted articulatory movements by DNN and DNN-BN-Arti. For lack of space, we only show two of the 12-dimensional output in this figure. The top subgraph shows the lower lip movements on the x axis, and the bottom subgraph is on the y axis. The black solid line, the blue dotted line and the red dashed line represent real data, predicted by DNN and predicted by DNN-BN-Arti, respectively.

DNN-BN-Acoustic-MS is lower, which shows that when the acoustic data from the target speaker is limited, borrowing data from another big corpus with multiple speakers helps. This observation is consistent with that in [17], which uses stacked bottleneck features for DNN-based text-to-speech synthesis.

We put the best RMSE achieved by DNN-BN-Arti in Tabel I. We can see that a further 2.27% relative error reduction is achieved by stacking bottleneck features as compared with an ordinary DNN. Fig. 4 shows the real and predicted articulatory movements of two articulatory positions for an utterance in the test set. We can see that the predicted trajectories follow the real ones quite well. The red dashed line, representing the predicted articulatory movements generated by DNN-BN-Arti, seems to be more close to the real trajectory shown in solid black line.

VI. Conclusions

In this paper, we boost the performance of articulatory prediction from text on the MNGU0 corpus to a new level by the use of deep neural network, which decreases RMSE to 0.7370mm. Moreover, we manage to stack bottleneck features as network input to capture the important contextual information for DNN-based text-to-articulatory prediction. We find that stacking bottleneck features can bring further RMSE reduction. There is still a substantial amount of work to do in the future. First, our recent work has shown that recurrent neural networks show superior performance in articulatory inversion [20]. Thus we plan to investigate RNN's effectiveness in text-to-articulatory prediction. Second, bottleneck features can be used as supplement of acoustic features too, to improve the performance of the articulatory inversion task.

Acknowledgment

This paper is partially supported by a grant from National Science Foundation of China (No.61571363).

References

[1] J. Sun and L. Deng, “An overlapping-feature-based phonological model incorporating linguistic constraints: Applications to speech recognition,” *The Journal of the Acoustical Society of America*, vol. 111, no. 2, pp. 1086–1101, 2002.

[2] S. King, J. Frankel, K. Livescu, E. McDermott, K. Richmond, and M. Wester, “Speech production knowledge in automatic speech recognition,” *The Journal of the Acoustical Society of America*, vol. 121, no. 2, pp. 723–742, 2007.

[3] Z.-H. Ling, K. Richmond, J. Yamagishi, and R.-H. Wang, “Integrating articulatory features into hmm-based parametric speech synthesis,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 6, pp. 1171–1185, 2009.

[4] T. Toda, A. W. Black, and K. Tokuda, “Statistical mapping between articulatory movements and acoustic spectrum using a gaussian mixture model,” *Speech Communication*, vol. 50, no. 3, pp. 215–227, 2008.

[5] A. Ben-Youssef, H. Shimodaira, and D. A. Braude, “Speech driven talking head from estimated articulatory features,” in *Proc. ICASSP*. Florence, Italy: IEEE, May 2014, pp. 4573–4577.

[6] K. Zhao, Z.-Y. Wu, and L.-H. Cai, “A real-time speech driven talking avatar based on deep neural network,” in *Proc. APSIPA*. Kaohsiung, Taiwan: IEEE, October 2013, pp. 1–4.

[7] K. Richmond, P. Hoole, and S. King, “Announcing the electromagnetic articulography (day 1) subset of the mngu0 articulatory corpus.” in *Proc. INTERSPEECH*. Florence, Italy: ISCA, August 2011, pp. 1505–1508.

[8] S. Y. C. S. Blackburn, “A self-learning predictive model of articulator movements during speech production,” *The Journal of the Acoustical Society of America*, vol. 107, no. 3, pp. 1659–1670, 2000.

[9] M. H. T. Okadome, “Generation of articulatory movements by using a kinematic triphone model,” *The Journal of the Acoustical Society of America*, vol. 110, pp. 453–463, 2001.

[10] Z.-H. Ling, K. Richmond, and J. Yamagishi, “An analysis of hmm-based prediction of articulatory movements,” *Speech Communication*, vol. 52, no. 10, pp. 834–846, 2010.

[11] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[12] L. Deng, G. Hinton, and B. Kingsbury, “New types of deep neural network learning for speech recognition and related applications: An overview,” in *Proc. ICASSP*. Vancouver, BC, Canada: IEEE, May 2013, pp. 8599–8603.

[13] Y. Qian, Y.-C. Fan, W.-P. Hu, and F. K. Soong, “On the training aspects of deep neural network (dnn) for parametric tts synthesis,” in *Proc. ICASSP*. Florence, Italy: IEEE, May 2014, pp. 3829–3833.

[14] H. Zen, “Deep learning in speech synthesis,” *Proc. ISCA SSW8*, August 2013.

[15] B. Uria, I. Murray, S. Renals, and K. Richmond, “Deep architectures for articulatory inversion,” in *Proc. INTERSPEECH*. Portland, Oregon, USA: ISCA, September 2012, pp. 867–870.

[16] Z. Wu, C. Valentini-Botinhao, O. Watts, and S. King, “Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis,” in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4460–4464.

[17] Z. Wu and S. King, “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*, vol. 24, no. 7, pp. 1255–1265, 2016.

[18] ——, “Minimum trajectory error training for deep neural networks, combined with stacked bottleneck features,” in *Proc. Interspeech*, 2015.

[19] N. T. Vu, J. Weiner, and T. Schultz, “Investigating the learning effect of multilingual bottle-neck features for asr.” in *INTERSPEECH*, 2014, pp. 825–829.

[20] P. Zhu, L. Xie, and Y. Chen, “Articulatory movement prediction using deep bidirectional long short-term memory based recurrent neural networks and word/phone embeddings,” in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[21] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, “Speech parameter generation algorithms for hmm-based speech synthesis,” in *Acoustics, Speech, and Signal Processing, 2000. ICASSP’00. Proceedings. 2000 IEEE International Conference on*, vol. 3. IEEE, 2000, pp. 1315–1318.

[22] F. Grézl and P. Fousek, “Optimizing bottle-neck features for lvcsr,” in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*. IEEE, 2008, pp. 4729–4732.

[23] D. Yu and M. L. Seltzer, “Improved bottleneck features using pretrained deep neural networks.” in *INTERSPEECH*, vol. 237, 2011, p. 240.

[24] T. N. Sainath, B. Kingsbury, and B. Ramabhadran, “Auto-encoder bottleneck features using deep belief networks,” in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4153–4156.

[25] J. Gehring, Y. Miao, F. Metze, and A. Waibel, “Extracting deep bottleneck features using stacked auto-encoders,” in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 3377–3381.

[26] Z. Wu, O. Watts, and S. King, “Merlin: An open source neural network speech synthesis system,” *Proc. 9th ISCA Speech Synthesis Workshop (SSW9), Sunnyvale, CA, USA*, 2016.