THE UNIVERSITY of EDINBURGH

Edinburgh Research Explorer

Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis.

Citation for published version:
Wu, Z, Valentini-Botinhao, C, Watts, O & King, S 2015, 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. Brisbane, Australia, pp. 4460-4464.

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 05. Apr. 2019

DEEP NEURAL NETWORKS EMPLOYING MULTI-TASK LEARNING AND STACKED BOTTLENECK FEATURES FOR SPEECH SYNTHESIS

Zhizheng Wu *Cassia Valentini-Botinhao* *Oliver Watts* *Simon King*

Centre for Speech Technology Research, University of Edinburgh, United Kingdom

ABSTRACT

Deep neural networks (DNNs) use a cascade of hidden representations to enable the learning of complex mappings from input to output features. They are able to learn the complex mapping from text-based linguistic features to speech acoustic features, and so perform text-to-speech synthesis. Recent results suggest that DNNs can produce more natural synthetic speech than conventional HMM-based statistical parametric systems. In this paper, we show that the hidden representation used within a DNN can be improved through the use of Multi-Task Learning, and that stacking multiple frames of hidden layer activations (stacked bottleneck features) also leads to improvements. Experimental results confirmed the effectiveness of the proposed methods, and in listening tests we find that stacked bottleneck features in particular offer a significant improvement over both a baseline DNN and a benchmark HMM system.

Index Terms— Speech synthesis, acoustic model, multi-task learning, deep neural network, bottleneck feature

1. INTRODUCTION

Statistical parametric speech synthesis (SPSS) has made significant advances in naturalness [1] and is generally highly-intelligible [2]. However even though it offers greater flexibility and controllability than unit selection [3], the naturalness of speech generated by SPSS is still below that of human speech, and cannot compete with good unit selection systems. Zen et al. [1] suggest various factors which limit naturalness or quality. One of the key issues they highlight is the core of the system: the acoustic model, which learns the complex relationship between the linguistic representation (derived from text) and acoustic features. In this paper we propose some techniques to improve acoustic modelling, which result in improvements to synthesised speech quality.

1.1. Relation to prior work

Significant efforts have been made to improve the acoustic models for SPSS, targeting the underlying model, the way the parameters are estimated during training, or the method for generating speech parameter trajectories when performing synthesis: minimum generation error training for HMM [4], global variance enhancement [5], and trajectory hidden Markov model [6], just to name a few examples.

More recently, neural networks have re-emerged as a potential acoustic model for SPSS [7, 8, 9, 10, 11] following their success in speech recognition [12]. Two weaknesses in HMM-based SPSS are the density function over the acoustic features (usually a Gaussian) and the decision-tree driven parameterisation of the model, in which parameters must be shared across groups of linguistic contexts.

Deep neural networks (DNNs) have the potential to address both areas. They can been viewed as a replacement for the decision tree in [7, 10, 11] while in [8] a deep belief network (DBN) was employed to jointly model the relationship between linguistic and acoustic features. These approaches map linguistic features directly to the corresponding acoustic features through multiple layers of hidden representations, frame by frame. In [9], restricted Boltzmann machines (RBMs) were used to replace Gaussian mixture models over the acoustic features, allowing more spectral detail to be learned, which resulted in better speech quality. We can identify at least two problems in the way DNNs are currently applied to speech synthesis: *perceptual suboptimality* and *frame-by-frame independence*. These two problems are also common in conventional HMM-based acoustic models.

1.2. The novelty of this work

The first problem – *perceptual suboptimality* – arises because the training criterion typically aims to maximise the likelihood of (or minimise the error to) acoustic features which are a rather poor representation of human speech perception. Unless the error is reduced all the way to zero, the error in the speech feature space is not an accurate reflection of expected perceptual error. The choice of speech features is constrained by the requirements of the vocoder: it must be invertible, i.e. allow for reconstruction. This rules out the use of many interesting and powerful perceptual representations of speech which – from the point of view of maximising the perceived quality of system output – would otherwise be very attractive.

To get around this, we use *multi-task learning* (MTL) [13] in a DNN. The DNN learns to predict a perceptual representation of the target speech as a secondary task, in parallel to learning to predict the usual invertible vocoder parameters as the main task. The predictions of the perceptual representation are discarded at synthesis time; rather, their purpose is to provide additional supervision during training and to 'steer' the hidden layers of the network towards a perceptually salient representation.

The second problem – *frame-by-frame independence* – arises in many SPSS systems where predicted values for consecutive acoustic states or frames are conditionally independent of one another given their linguistic contexts. Although the maximum likelihood parameter generation (MLPG) algorithm uses dynamic features to smooth acoustic feature trajectories, a framewise independence assumption remains in the underlying model. There are DNN architectures that model sequences of data, such as Recurrent Neural Networks which have been applied to speech synthesis [14], but they can be difficult or computationally expensive to optimise. As a much simpler, but still highly-effective alternative, we propose *bottleneck feature stacking*. We train a first DNN with a bottleneck (i.e., relatively small number of units) hidden layer. The activations of the bottleneck units yield a compact representation of both acoustic and linguistic information for each frame independently. Then, we stack multiple consecutive frames of bottleneck features to produce a wide context around the current frame, combine these with the linguistic

features, and use this as input to a second DNN.

2. PROPOSED MULTI-TASK DNN WITH STACKED BOTTLENECK FEATURES

DNN-based speech synthesis involves two stages: offline training and runtime synthesis. During offline training, linguistic features are derived from text and aligned with acoustic features. This alignment is typically obtained by forced-alignment against a HMM-GMM acoustic model and can be at phone or HMM-state level. The DNN then learns the relationship between the linguistic and acoustic features. At runtime, phone or state durations are predicted and the linguistic features for each frame are mapped to vocoder parameters, which are then passed to a synthesis filter to reconstruct the speech.

There are two particular problems in this arrangement that we address here. First, the optimisation criterion is to minimise the difference between predicted and reference vocoder parameters. If these parameters are not monotonically correlated with perception, the synthesised speech will be perceptually suboptimal. Second, although the linguistic features contain wide context, the mapping is still learned frame by frame without access to any *acoustic context*.

2.1. Multi-task learning

Multi-task learning (MTL) is a way to train a 'universal' model for several different but related tasks using a shared representation [13]. Usually, there is one main task and one or more secondary tasks. It is generally believed that the model learned in the multi-task learning fashion can generalise better and make more accurate predictions than a model for a single task, provided that the secondary task(s) are related to the main task and at the same time complementary (i.e., not identical) [13]. MTL has produced good results in automatic speech recognition [15] and natural language processing [16], for example.

When using MTL with a DNN, the main task and the secondary tasks share the same hidden representations, as illustrated in Fig. 1. For speech synthesis, the main task is to predict vocoder parameters, while the secondary task could be to predict a perceptual representation of the same speech. Conveniently, a MTL DNN is learned using exactly the same optimisation techniques as a single-task DNN: implementation is trivial.

2.2. Stacked bottleneck features

In DNN or MTLDNN, the acoustic features for each frame are predicted independently without any acoustic contextual constraints. Although the maximum likelihood parameter generation (MLPG) algorithm [17] can be used to impose dynamic constraints and so obtain smooth parameter trajectories, long-term context is still ignored and is not explicitly modelled by the DNN. We propose the use of bottleneck features as a compact, learned representation, and to stack multiple frames of them to provide wide linguistic-acoustic contextual information.

The bottleneck features are simply vectors consisting of the activations at a bottleneck layer, which has a relatively small number of hidden units compared to the other hidden layers in the network. Bottleneck features have been extensively used in speech recognition [18, 19, 20]. The left network in Fig. 2 is an example of a network with a bottleneck layer. In the example, the left network has four layers, of which the third layer is the bottleneck layer. Note that either a DNN or MTLDNN can have a bottleneck layer and so be used to produce bottleneck features for use by a subsequent network.

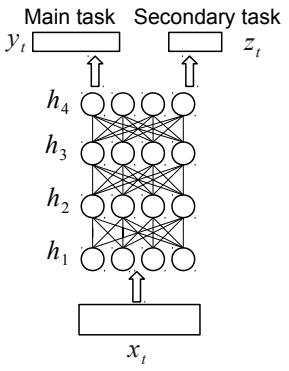

Fig. 1. A multi-task deep neural network (MTLDNN). In a MTLDNN, the main task and the secondary task share the same hidden representations (i.e., hidden layer activations). The optimisation criterion is to minimise the mean square error on both tasks. $\mathbf{x}_t$, $\mathbf{y}_t$ and $\mathbf{z}_t$ are the linguistic input, main task and secondary task, respectively, at frame t. $\mathbf{h}_k$ is the k^{th} hidden layer

We might hope that the MTLDNN will produce superior bottleneck features because of the additional supervision provided by the secondary task.

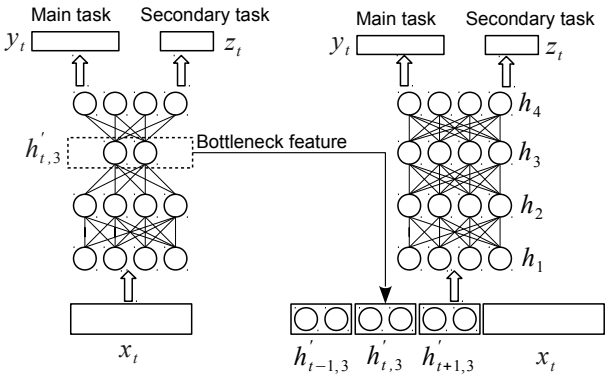

Fig. 2. Multi-task deep neural network (MTLDNN) with stacked bottleneck features. In this example, the bottleneck features for three consecutive frames are stacked as input to the second network. In practice, more than three frames can be included. $\mathbf{h}'_{t,3}$ is the vector of bottleneck feature for the t^{th} frame.

Multiple frames of bottleneck features are stacked together, then used alongside the linguistic features as inputs to another DNN or MTLDNN, this time with no bottleneck layer. Because the dimensionality of the bottleneck features is low, stacking multiple frames does not increase complexity significantly. Figure 2 illustrates the proposed bottleneck feature stacking method, which proceeds as follows:

(a) Train the first DNN or MTLDNN, which contains a bottleneck layer;

(b) Make a forward pass through this network to generate bottleneck features for the training, development and evaluation data, frame by frame;

(c) Stack bottleneck features from several consecutive frames around the current frame alongside the linguistic features;

(d) Train the second DNN or MTLDNN using as input the features constructed in step (c);

(e) Make a forward pass using the development and evaluation data to generate acoustic features (the main task) from the second DNN or MTLDNN

In this work, the last but one layer was the bottleneck layer, which is the usual setting in speech recognition [20].

3. EXPERIMENTS

3.1. Experimental setups

A speech database from a British male speaker was used in the experiments, comprising 2542 utterances: 2400 utterances as training set, 70 utterances as development set, and 72 utterances for evaluation. The sampling rate of the speech data was 48 kHz. STRAIGHT [21] was used to extract 60-dimensional Mel-Cepstral Coefficients (MCCs), 25 band aperiodicities (BAPs) and logarithmic fundamental frequency ($\log F_0$) at 5 msec frame intervals. For the MTLDNNs, we experimented with various secondary tasks: formant centre frequencies (F1-F4)[1], 40-dimensional line spectral frequencies (LSF), 64-dimensional Gammatone spectrum [22], or 55-dimensional spectro-temporal excitation pattern (STEP) representation features were extracted at the same frame rate as MCCs. The STEP feature was derived from the Glimpse Proportion measure for speech intelligibility in noise that was proposed in the context of the Glimpse model for speech perception in noise [23].

For comparison, an HMM system was trained on the same data, employing five-state, left-to-right hidden semi-Markov models (HSMM). The MCCs and BAPs with deltas and delta-deltas appended were modelled by single-component Gaussians, and $\log F_0$ with delta and delta-delta was modelled by a 3-dimension multi-space probability distribution (MSD). Decision tree state clustering used a minimum description length (MDL) factor of 1.0. During parameter generation, global variance (GV) enhancement was applied[2]. The publicly available HTS toolkit[3] was used to implement the baseline HMM system.

In the DNN-based systems, the input features consisted of 592 binary features and 9 numerical features. The binary features were derived from a subset of the questions used by the decision tree clustering in the HMM system, and included linguistic contexts such as quinphone identity, and part-of-speech, positional information within the syllable, word and phrase, and so on. 9 numerical features were appended: the frame position within the HMM state and phoneme, the state position within the phoneme, and state and phoneme durations. Frame-aligned training data for the DNN was created by forced alignment using the HMM system described above. The main task DNN outputs comprised MCCs, BAPs and continuous $\log F_0$ (all with deltas and delta-deltas) plus a voiced/unvoiced binary value. In the MTLDNN systems, a secondary task was added, but dynamic features were not used for the secondary tasks. Input features were normalised to the range of [0.01, 0.99] and output features were normalised to zero mean and unit variance. MLPG using pre-computed variances from the training data was applied to the main task output features, and spectral enhancement post-filtering was applied to the MCCs[4].

In both DNN and MTLDNN, the tangent or tanh function was used as the hidden activation function, and a linear activation function was employed at the output layer. During training, L_2 regularisation was applied on the weights with penalty factor of 0.00001, the mini-batch size was set to 256 and momentum was used. For the first 10 epochs, momentum was 0.3 with a fixed learning rate of 0.002. After 10 epochs, the momentum was increased to 0.9 and from that point on the learning rate was halved at each epoch. The learning rate of the top two layers was half that of other layers. The maximum epochs was set to 25 (early stoping). For software implementation, we used Theano version 0.6 [24] and training was conducted on a GPU.

3.2. Objective results

We first conducted objective evaluations to assess the performance of the proposed methods. We know that these are unlikely to correspond directly to perceived quality, but they are necessary when tuning the systems (e.g., selecting the learning rate, etc). Results are presented in Table 1. Across all acoustic parameters, the DNN makes more accurate predictions that the HMM. In particular, the MCD of HMM is reduced from 4.56 dB to 4.17 dB, and the V/UV error rate is reduced from 5.92 % to 4.24 %. These objective results are consistent with results in the literature [7, 25, 11].

After establishing the DNN baseline, we then assessed the effectiveness of multi-task learning. Across the four different secondary tasks that we tried, only LSF was an invertible feature, while formants, Gammatone spectra and STEP features cannot readily be used to generate speech. All tasks decreased MCD and V/UV error rates, even the formant task, the extraction of which is not accurate. The STEP features, which relate to speech perception, showed the most promise.

Next, we evaluated the performance of DNN or MTLDNN with stacked bottleneck features. The bottleneck layer was always the last layer but one and always had 128 hidden units. The performance of the bottleneck network alone is generally slightly worse than the equivalent standard network, simply due to the small size of the bottleneck layer, and is to be expected. However, stacking these features and training a second network produces improvements in all cases.

As we increase the number of stacked frames, MCD drops from 4.17 dB for 1-frame context to 4.12 dB for 9-frame context. Other objective results also improved: V/VU error dropped from 4.19 % with 1-frame context to 3.91 % with 9-frame context. Stacking bottleneck features is quite effective.

Finally, we combined MTLDNN and bottleneck feature stacking and saw further improvements in the objective results, except that F_0 error increased. Recall that our hope is that multi-task learning leads to more informative bottleneck features and so should provide further improvements when those features are stacked.

3.3. Subjective results

We conducted subjective evaluations to assess the naturalness of the synthesised speech using a MUSHRA (MUltiple Stimuli with Hidden Reference and Anchor) test, which allowed us to evaluate multiple samples in a single trial without breaking the task into many pairwise comparisons. We evaluated nine systems: the benchmark HMM with global variance enhancement (HMM-GV), the baseline DNN, MTLDNN with four different secondary tasks, DNN-DNN and MTLDNN-MTLDNN[5]. Note that 9-frame context was used in DNN-DNN and MTLDNN-MTLDNN.

[1] Praat (http://www.fon.hum.uva.nl/praat/) was used to extract formants.

[2] GV enhancement was not used when computing the objective results.

[3] http://hts.sp.nitech.ac.jp/

[4] The Speech Signal Processing Toolkit (SPTK) was used, which is available at: `http://sp-tk.sourceforge.net/`

[5] Samples are available at: `http://homepages.inf.ed.ac.uk/zwu2/dnn_tts/demo.html`

Table 1. Objective results for the baseline HMM, DNN, proposed MTLDNN and DNN with stacked bottleneck features. Here, DNN-DNN means the first network (used to extract bottleneck features) is a single-task DNN, and the second network is also a normal DNN. Root mean squared error (RMSE) of F_0 was computed in linear frequency. V/UV error means frame-level voiced/unvoiced error. MCD and BAP are the Mel Cepstral Distortion and BAP error, respectively.

	Secondary feature	Architecture	Contextual size of Bottleneck features	MCD (dB)	BAP (dB)	F_0 RMSE (Hz)	V/UV error rate (%)
HMM	n/a	1 mix	n/a	4.56	2.06	9.90	5.92
DNN	n/a	6*1024	n/a	4.17	1.96	9.34	4.24
MTLDNN	Gammatone	6*1024	n/a	4.14	1.96	9.53	4.13
	Formants	6*1024	n/a	4.17	1.96	9.46	4.09
	LSF	6*1024	n/a	4.15	1.96	9.54	4.19
	STEP	6*1024	n/a	4.12	1.96	9.52	4.09
MTLDNN	STEP	5*1024+128	n/a	4.22	1.97	9.89	4.13
DNN	n/a	5*1024+128	n/a	4.27	1.97	9.31	4.24
DNN-DNN	n/a	6*1024	1	4.17	1.96	9.54	4.19
DNN-DNN	n/a	6*1024	3	4.14	1.95	9.27	4.04
DNN-DNN	n/a	6*1024	5	4.15	1.95	9.24	4.05
DNN-DNN	n/a	6*1024	7	4.14	1.95	9.35	4.00
DNN-DNN	n/a	6*1024	9	4.12	1.94	9.23	3.91
DNN-MTLDNN	STEP	6*1024	9	4.08	1.94	9.23	3.89
MTLDNN-DNN	STEP	6*1024	9	4.10	1.94	9.65	3.92
MTLDNN-MTLDNN	STEP	6*1024	9	4.08	1.94	9.66	3.82

15 native English listeners with no reported hearing difficulties participated in the MUSHRA test. Each listener rated 20 sets which were randomly selected from the 72 evaluation utterances. Each set included 10 stimuli of the same sentence generated by each of the nine systems plus the natural speech used as the hidden reference. The listeners were asked to rate each stimulus from 0 (extremely bad) to 100 (perfect: same as the reference natural speech), and they were also instructed to give exactly one of the 10 stimuli in every set a rating of 100. From the complete test, we obtained 300 sets of scores. By excluding one set of scores, in which the hidden reference was not rated at 100, we had 299 sets for further analysis.

The MUSHRA scores for all the systems are presented in Fig. 3. It can be seen that all the DNN-based systems outperform the HMM system significantly. Due to the large variability across listeners and stimuli, the differences between DNN-based systems is not as clear. However, the MUSHRA design allows us to use paired t-tests to identify significant differences between all pairs of systems.

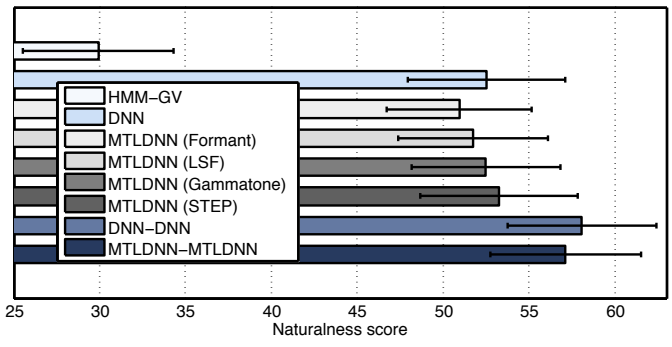

Fig. 3. MUSHRA results with 95 % confidence interval. The score for the natural speech is not included: it is always 100.

We performed paired t-tests (95 % confidence level) on the MUSHRA scores between two different DNN-based systems for further analysis. Although MTLDNN with the STEP feature scores higher than the baseline DNN, the difference is not significant. Similarly, the differences between the baseline DNN and other MTLDNNs are also not significant.

However, both bottleneck feature stacking methods, DNN-DNN and MTLDNN-MTLDNN, are significantly better than all other systems, (although the difference between DNN-DNN and MTLDNN-MTLDNN is not significant).

4. CONCLUSIONS

We have proposed the use of multi-task learning in DNN-based speech synthesis as simple and convenient way to provide additional supervision during training. It also allows the use of perceptually-salient, but unfortunately non-invertible, representations of speech. Objective results suggest that several different secondary tasks are all helpful, and that the perceptual representation (STEP) is the most promising. To provide additional context, we also propose the use of stacked bottleneck features. Again, objective measures suggest that these are quite effective.

Listening test results suggest some improvements yielded by multi-task learning, but significant improvements were not observed in the subjective evaluation. The results also demonstrate that stacking bottleneck features does lead to statistically significant improvements in naturalness.

The selection of the secondary task is important, and it affects the performance of the main task. In follow-up work, we will continue to experiment with other secondary tasks, and to discover whether multi-task learning and bottleneck feature stacking can be effectively combined.

Acknowledgements: this work was partially supported by EPSRC Programme Grant EP/I031022/1 (Natural Speech Technology) and partially supported from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement 287678 (Simple4All).

5. REFERENCES

[1] Heiga Zen, Keiichi Tokuda, and Alan W Black, "Statistical parametric speech synthesis," *Speech Communication*, vol. 51, no. 11, pp. 1039–1064, 2009.

[2] Simon King, "Measuring a decade of progress in text-to-speech," *Loquens*, vol. 1, no. 1, 2014.

[3] Andrew J Hunt and Alan W Black, "Unit selection in a concatenative speech synthesis system using a large speech database," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 1996.

[4] Yi-Jian Wu and Ren-Hua Wang, "Minimum generation error training for HMM-based speech synthesis," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2006.

[5] Toda Tomoki and Keiichi Tokuda, "A speech parameter generation algorithm considering global variance for HMM-based speech synthesis," *IEICE Transactions on Information and Systems*, vol. 90, no. 5, pp. 816–824, 2007.

[6] Heiga Zen, Keiichi Tokuda, and Tadashi Kitamura, "Reformulating the HMM as a trajectory model by imposing explicit relationships between static and dynamic feature vector sequences," *Computer Speech & Language*, vol. 21, no. 1, pp. 153–173, 2007.

[7] Heiga Zen, Andrew Senior, and Mike Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2013.

[8] Shiyin Kang, Xiaojun Qian, and Helen Meng, "Multi-distribution deep belief network for speech synthesis," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2013.

[9] Zhen-Hua Ling, Li Deng, and Dong Yu, "Modeling spectral envelopes using Restricted Boltzmann Machines and Deep Belief Networks for statistical parametric speech synthesis," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 10, pp. 2129–2139, 2013.

[10] Heng Lu, Simon King, and Oliver Watts, "Combining a vector space representation of linguistic context with a deep neural network for text-to-speech synthesis," *Proc. the 8th ISCA Speech Synthesis Workshop (SSW)*, 2013.

[11] Yao Qian, Yuchen Fan, Wenping Hu, and Frank K Soong, "On the training aspects of deep neural network (DNN) for parametric TTS synthesis," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2014.

[12] G. Hinton, Li Deng, Dong Yu, G.E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[13] Rich Caruana, *Multitask learning*, Springer, 1998.

[14] Yuchen Fan, Yao Qian, Fenglong Xie, and Frank K. Soong, "TTS synthesis with bidirectional LSTM based recurrent neural networks," in *Proc. Interspeech*, 2014.

[15] Michael L Seltzer and Jasha Droppo, "Multi-task learning in deep neural networks for improved phoneme recognition," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2013.

[16] Ronan Collobert and Jason Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proc. IEEE Int. Conf. on Machine Learning (ICML)*, 2008.

[17] Keiichi Tokuda, Takayoshi Yoshimura, Takashi Masuko, Takao Kobayashi, and Tadashi Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2000.

[18] Dong Yu and Michael L Seltzer, "Improved bottleneck features using pretrained deep neural networks," in *Proc. Interspeech*, 2011.

[19] Tara N Sainath, Brian Kingsbury, and Bhuvana Ramabhadran, "Auto-encoder bottleneck features using deep belief networks," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2012.

[20] Jonas Gehring, Yajie Miao, Florian Metze, and Alex Waibel, "Extracting deep bottleneck features using stacked auto-encoders," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2013.

[21] Hideki Kawahara, Ikuyo Masuda-Katsuse, and Alain de Cheveigné, "Restructuring speech representations using a pitch-adaptive time–frequency smoothing and an instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sounds," *Speech communication*, vol. 27, no. 3, pp. 187–207, 1999.

[22] D. P. W. Ellis, "Gammatone-like spectrograms," web resource: http://www.ee.columbia.edu/ dpwe/resources/matlab/gammatonegram/, 2009.

[23] Martin Cooke, "Glimpsing speech," *Journal of Phonetics*, vol. 31, pp. 579 – 584, 2003.

[24] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio, "Theano: a CPU and GPU math expression compiler," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010.

[25] Heiga Zen and Andrew Senior, "Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2014.