ON THE TRAINING ASPECTS OF DEEP NEURAL NETWORK (DNN) FOR PARAMETRIC TTS SYNTHESIS

Yao Qian Yuchen Fan Wenping Hu Frank K. Soong

Microsoft Research

{yaoqian, v-yufan, v-wenh, frankkps}@microsoft.com

ABSTRACT

Deep Neural Network (DNN), which can model a long-span, intricate transform compactly with a deep-layered structure, has recently been investigated for parametric TTS synthesis with a fairly large corpus (33,000 utterances) [6]. In this paper, we examine DNN TTS synthesis with a moderate size corpus of 5 hours, which is more commonly used for parametric TTS training. DNN is used to map input text features into output acoustic features (LSP, F0 and V/U). Experimental results show that DNN can outperform the conventional HMM, which is trained in ML first and then refined by MGE. Both objective and subjective measures indicate that DNN can synthesize speech better than HMM-based baseline. The improvement is mainly on the prosody, i.e., the RMSE of natural and generated F0 trajectories by DNN is improved by 2 Hz. This benefit is likely from the key characteristics of DNN, which can exploit feature correlations, e.g., between F0 and spectrum, without using a more restricted, e.g. diagonal Gaussian probability family. Our experimental results also show: the layer-wise BP pre-training can drive weights to a better starting point than random initialization and result in a more effective DNN; state boundary info is important for training DNN to yield better synthesized speech; and a hyperbolic tangent activation function in DNN hidden layers yields faster convergence than a sigmoidal one.

Index Terms— *Speech Synthesis, HMM, DNN, TTS*

1. INTRODUCTION

In recent years, a new machine learning algorithm named DNN has improved speech recognition significantly [1-3]. In the DNN-HMM approach, acoustic features along with contextual-dependent phone sequence are firstly modeled by conventional GMM-HMMs, then the aligned pairs of decision-tree based HMM tied states ("senones") and corresponding acoustic feature vectors (GMM-HMM is used for forced alignment) are modeled by DNN, which has become the state-of-art acoustic modeling in speech recognition. Comparing with the conventional HMM-based approach, DNN based approach can: model long-span (e.g., 11 frames), high dimensional and strongly correlated features as its input feature; find a highly non-linear mapping between input and output features with a deep-layered, hierarchical structure; train model parameters discriminatively through back-propagation with the gradient of a cost function.

Over the past decade, corpus-driven speech synthesis system trained as HMM [4, 5] has gained its popularity steadily in the TTS community. A parametric HMM is effective to model the evolution of speech signals as a stochastic sequence of acoustic feature vectors. Many techniques have been developed for HMM-based speech recognition, e.g. context-dependent modeling, state-tying based on decision tree clustering, and speaker adaptation. They have been applied equally well to HMM-based TTS for speech parameter trajectory generation. The trajectory thus generated with trained HMMs is fairly smooth and very rarely results in concatenation glitches which occur occasionally in unit-selection synthesis. However, overly-smoothed parameter trajectories due to statistical average in HMM training still tend to make synthesized speech sound not as lively as desired.

Contrary to speech recognition, speech synthesis can be viewed as an inverse or a production process. Motivated by the success of DNN in speech recognition, a few DNN research attempts have been tried in order to improve the performance of vocoder-based speech synthesis. Zen, et al. [6] comprehensively listed the limitations of the conventional HMM-based approach, e.g. decision-tree based contextual state clustering, and proposed to use DNN to overcome these limitations for speech synthesis. It shows DNN based approach, which models the relationship between input texts and their corresponding acoustic features, can outperform the HMM-based approach with a similar number of parameters. Phone, letter and Vector Space Model (VSM) based binary or continuous features used as input features, and frame or state used as timescale for output predictions are tested in DNN based speech synthesis [7]. Deep Belief Network (DBN) with stacked, Restricted Boltzmann Machines (RBMs), which can model the structure in the input data as generative "pre-training" and find a region of the weight-space to reduce over-fitting for the discriminative "fine-tuning" phase in speech recognition [3], is also employed to model joint distribution of linguistic and acoustic features for speech synthesis [8]. In addition, RBM is directly used to represent the distribution of the spectral envelopes at each HMM state [9], where the estimated mode of RBM is better than the mean of GMM and results in a better voice quality in speech synthesis.

In this paper, we further investigate various training aspects of DNN as a generation model for TTS synthesis. Instead of a large database of 33,000 utterances used in [6], a moderate size, 5-hour corpus (female speaker) is used. This moderate or even smaller corpus size is commonly used in training parametric TTS. Also, parametric TTS trained with this size corpus, has been shown to outperform unit-selection based TTS in synthesis quality.

2. DNN FOR TTS SYNTHESIS

2.1. DNN for Regression Problem

A DNN is a feed-forward, artificial neural network with multiple hidden layers between its input and output. For each hidden unit j, a nonlinear activation function $f(\cdot)$, is used to map all inputs from the lower layer, x_j, to a scalar state, y_j, which is then fed to the upper layer,

$$y_j = f(x_j) \tag{1}$$

where

$$x_j = b_j + \sum_i y_i w_{ij} \quad (2)$$

and b_j is the bias of unit j; i is the unit index of lower layer; w_{ij} is the weight of the connection between unit j and unit i in the layer below. Generally we choose the activation function $f(\cdot)$ to be a sigmoid function:

$$f(x_j) = \frac{1}{1 + e^{-x_j}} \quad (3)$$

where the input-output mapping is defined by a logistic regression, or a hyperbolic tangent (or tanh) function:

$$f(x_j) = \frac{e^{x_j} - e^{-x_j}}{e^{x_j} + e^{-x_j}} \quad (4)$$

which is a rescaled version of the sigmoid, and its output range is [-1, 1] instead of [0, 1].

All weights and biases are generally initialized in pre-training [10], and then trained by optimizing a cost function which measures the discrepancy between target vectors and the predicted output with a Back-Propagation (BP) procedure [11]. Given a fixed training set $\{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), \ldots, (\mathbf{x}^{(T)}, \mathbf{y}^{(T)})\}$ with T training examples, the cost function to be minimized is defined by

$$\boldsymbol{C} = \frac{1}{2T} \sum_{t=1}^{T} \left\| \boldsymbol{f}(\boldsymbol{x}^{(t)}) - \boldsymbol{y}^{(t)} \right\|^2 \quad (5)$$

To prevent over-fitting, a regularization term (also called a weight decay term [12]) is added into Eq. 5. Also, learning can be simply terminated when the performance on a held-out validation set starts to deteriorate [13]. The DNN is trained by using batch gradient descent. It is optimized by a "mini-batch" based stochastic gradient descent algorithm,

$$(W^l, b^l) \leftarrow (W^l, b^l) + \varepsilon \frac{\partial C}{\partial (W^l, b^l)}, \quad 0 \le l \le L \quad (6)$$

where ε is a preset learning rate.

The cost function in (5) is often used for classification and regression problems. TTS is a regression problem, the outputs are first scaled to ensure that they lie in the range of [0, 1] or [-1, 1] if a tanh activation is used. Nonlinear activation function can allow the neural networks to compute nontrivial problems by using only a small number of nodes. However, for neural network based regression, commonly a nonlinear activation function is adopted for hidden layers while linear activation function is employed only at the final output layer.

2.2. DNN Pre-training

Pre-training is crucial in training deep structured models for speech recognition tasks [14, 15] and it can initialize the weights to a better starting point than random initialization such that BP can have a rapid learning. It is shown that DNNs with deeper layers can outperform traditional shallow networks [1, 17].

DNNs can be pre-trained as a Deep Belief Network (DBN) [3] or "layer-wise BP" [16]. In DBN pre-training, the network is trained in a layer-by-layer manner with stacked, Restricted Boltzmann Machines (RBMs), where each successive pair of layers is treated as an RMB. The weights that connect each pair of layers are trained in an unsupervised fashion by the criterion of contrastive divergence [10]. Alternatively, the network can be initialized using "layer-wise BP" pre-training [16]. This procedure starts by training a Multi-Layered Perceptron (MLP) with one hidden layer using back-propagation. The weights of the first hidden layer are then fixed, and a new randomly initialized hidden layer is added into the network and output layer is introduced to replace the output layer of the initial network. The deeper network is trained again using back-propagation. This procedure is repeated until a desired number of hidden layers is reached.

The cost function in Eq.5 is to minimize the average sum-of-squared errors between target and generated (predicted) vectors. In DBN pre-training, although DNN weights are pre-trained in a generative manner, an approximate maximum likelihood criterion of contrastive divergence is still used to learn the model parameters. While in layer-wise BP, the hidden layers are added to the neural networks one by one to full convergence, it can remedy the modeling inaccuracies in DBN pre-training.

2.3. DNN vs. HMM for TTS Synthesis

Fig. 1 shows the DNN-based TTS synthesis next to the conventional HMM-based TTS synthesis.

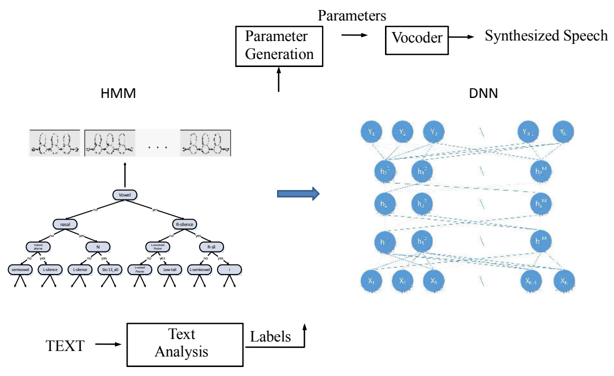

Fig. 1. DNN vs. HMM for TTS synthesis

In HMM-based TTS, the spectral envelope, fundamental frequency, and duration are modeled simultaneously by the corresponding HMMs. Rich contexts are used to capture co-articulation effects in HMM modeling. However, in practice, limited by insufficient training data, we usually have to tie models of long contexts into generalized ones to predict unseen contexts in testing. State tying via a clustered decision tree is commonly used. In synthesis, for a given text sequence, text is first converted into a sequence of contextual labels through the text analysis module. The corresponding contextual label is then used to access the decision tree to get the contextual HMM state sequence. The corresponding means and covariances of HMM states are fed into the parameter generation module to generate smooth speech parameter trajectories, which are synthesized in a maximum probability sense along with the dynamic infomation. Finally, speech waveform is synthesized from the generated spectral and excitation parameters via a vocoder.

In DNN-based TTS, rich contexts are also used as input features, which contain the binary features for categorical contexts, e.g. phone labels, POS labels of the current word, and TOBI labels, and numerical features for the numerical contexts, e.g., the number of words in the phrase or the position of current frame in the current phone. The output features are acoustic features like spectral envelope and fundamental frequency. Input features and output features are time-aligned frame-by-frame by well-trained HMM models. The weights of DNN are trained by using pairs of input and output features extracted from training data to minimize the errors between the mapped output from a given input and the target output. In synthesis, the input text is converted first into input feature vector through the text analysis, then input feature vectors are mapped to output vectors by a trained DNN using forward propagation. By setting the predicted output features from the DNN as mean vectors and pre-computed (global) variances of output features from all training data, the speech feature generation module can generate smooth trajectories of speech parameter features which satisfy the statistics of static and dynamic features. Finally, speech waveform is synthesized with the generated speech parameters.

We conjecture that the strong learning capability of DNN should benefit TTS speech synthesis. DNN simulates human speech production by a layered hierarchical structure to transform linguistic text information into final speech output. Deep-layered architectures can represent long-span, highly-complex function (transformation) compactly [18]. The functions can be compactly represented with a *k* level deep architecture which can outperform a shallow architecture of many more weights. A decision tree, which is generally used in HMM-based TTS for clustering the similar context-dependent state into tied states, is such a shallow architecture. HMM also has a conditional independence assumption that all observations are dependent only upon the states that generate them, but independent of their neighboring observations. HMM states and decision tree decompose the training data into small partitions and the model parameters are updated with the corresponding data in the partitions independently, while the weights of DNN are updated by looping through all training data to utilize full training efficiency. Based upon the above points, DNN-based TTS should have lower model complexity than that of HMM TTS. Additionally, DNN training is not constrained by the limitations of the intrinsic greedy search in a decision tree.

3. EXPERIMENTS AND RESULTS

3.1. Experimental Setup

A phonetically and prosodically rich corpus in American English is used in our experiments. The English corpus consists of 5,000 training utterances (around 5 hours) and 200 extra utterances are used for testing. The corpus is recorded by a female speaker. Speech signals are sampled at 16 kHz, windowed by a 25-ms window shifted every 5-ms. The LPC of 40th order is transformed into static LSPs and their dynamic counterparts. The phonetic and prosodic contexts include quin-phone, the position of phone, syllable and word in phrase and sentence, the length of word and phrase, stress of syllable, TOBI and POS of word.

In the baseline HMM-based TTS, five-state, left-to-right HMM phone models, where each state is modeled by a single Gaussian, diagonal covariance output distribution, are adopted. The phonetic and prosodic contexts are used as a question set in growing decision trees. Minimum description length (MDL) criterion [22] for balancing model complexity and training data size is used as a stopping criterion for state clustering in decision tree growing. HMM parameters are firstly trained in the Maximum Likelihood (ML) sense and then refined by the minimum generation error (MGE) training. It adjusts HMM parameters trained by the conventional EM algorithm to minimize the generation error between synthesized and original parameter trajectories of the training data [19].

In the DNN-based TTS, the input feature vector contains 355 dimensions, where 319 are binary features for categorical linguistic contexts and the rest are numerical linguistic contexts. The output feature vector contains voiced/unvoiced flag, log *F0*, LSP, gain, their dynamic counterparts, totally 127 dimensions. Voiced/unvoiced flag is a binary feature that indicates whether the current frame is voiced or not. An exponential decay function [20] is used to interpolate F0 in unvoiced speech regions. 80% of silence frames are removed from the training data to reduce the computational cost. Removing silence frame in DNN training was found useful for avoiding DNN overlearning silence label in speech recognition task. Both input and output features of training data are normalized to zero mean and unity variance. The weights are trained by back-propagation procedure with a “mini-batch” based stochastic gradient ascent algorithm.

For the testing utterances, both HMM and DNN outputs are firstly fed into a parameter generation module to generate smooth feature parameters with dynamic feature constraints [4], then formant sharpening based on LSP frequencies [21] is used to reduce the over-smoothing problem of statistic parametric modeling and the resultant degraded synthesized speech quality, finally the waveforms are synthesized by a LPC synthesizer by using generated speech parameters.

3.2. Evaluation Results and Analysis

Objective and subjective measures are used to evaluate the performance of two TTS systems on testing data. Synthesis quality is measured objectively in terms of distortions between natural

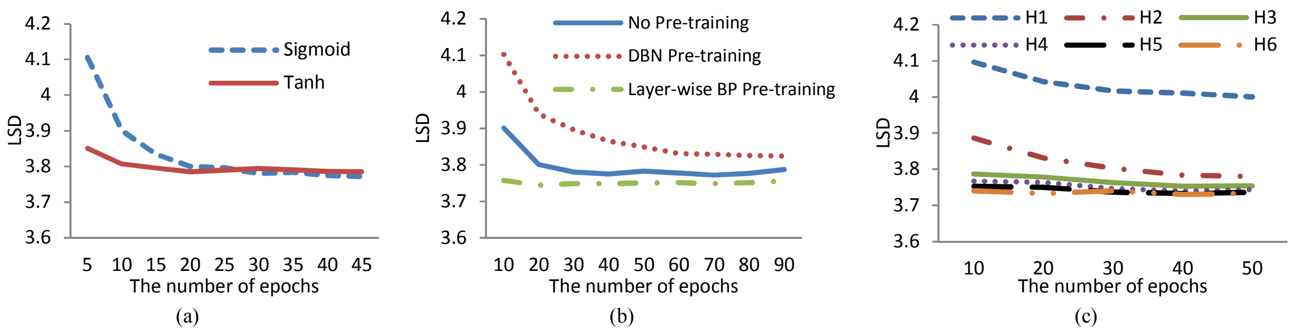

Fig. 2. (a) Sigmoid vs. Tanh for the activation function of hidden layers in DNN training; (b) No pre-training vs. pre-training (DBN or layer-wise BP) in DNN training; (c) LSD of different number of hidden layers.

test utterances of the original speaker and the synthesized speech frame-synchronously where oracle state durations (obtained by forced alignment) of natural speech are used. The objective measures are F0 distortion in the root mean squared error (RMSE), voiced/unvoiced (V/U) swapping errors and normalized log spectrum distance (LSD) in dB. The subjective measure is an AB preference test between speech sentence pairs synthesized by HMM-based TTS and DNN-based TTS.

The sigmoid and hyperbolic tangent activation functions are used for the hidden layers of DNN, while the linear activation function is employed for the output layer. The average LSDs of testing utterances by using sigmoid and tanh activation functions in a 4-layer DNN (with 3 hidden layers) are shown along the number of epochs in Fig.2. (a), where the performances of the two activation functions are almost the same but tanh converges much faster than sigmoid. Fig.2. (b) shows the average LSDs of testing utterances of a 4-layer DNN with/without pre-training. The resultant LSDs of DNN with DBN pre-training is worse than that of without pre-training while that of DNN with layer-wise BP pre-training is the other way around. We think it is due to that the criterion of layer-wise BP pre-training is consistent with that of DNN training. The average LSDs of testing utterances by using different number of hidden layers (from 1 to 6) are illustrated in Fig.2. (c), where the performances of 3~6 hidden layers are close although the deepest DNN is the best on LSDs. The other two objective measures, V/U error rate and the RMSE of F0, show similar trend as LSDs, hence not shown here due to limited space.

The results of objective measures of different structures and different MDL factors in DNN and HMM trainings are shown in Tables 1 and 2, respectively. For DNN training, 3 or 6 hidden layers with 512 or 1024 nodes for each layers yield very similar performances in all three objective measures. For HMM training, larger MDL factors yield worse objective results. By comparing the results of DNN with those of HMM, DNN can achieve a better performance on F0, i.e., the RMSE of natural and generated F0 trajectories by DNN is improved by 2 Hz, with slightly over 1/4 of the system sixe. The LSD of natural and generated spectra by DNN is about the same as HMM. The MGE training, a more consistent criterion than ML between training and synthesis, is used in HMM-based TTS system and generally is more effective in producing better spectral fidelity than F0.

Table 1. The results of objective measures of different structures (the number of system parameters) in DNN training

Measures / DNN Structure	LSD (dB)	V/U Error rate	F0 RMSE (Hz)
512 *3 (0.77 M)	3.76	5.9%	15.8
512 *6 (1. 55M)	3.73	5.8%	15.8
1024*3 (2.59 M)	3.73	5.9%	15.9

Table 2. The results of objective measures of different MDL factors (the number of system parameters) used by decision-tree based clustering in HMM training

Measures / MDL Factor	LSD (dB)	V/U Error rate	F0 RMSE (Hz)
1 (2.89 M)	3.74	5.8%	17.7
1.6 (1.52M)	3.85	6.1%	18.1
3 (0.85M)	3.91	6.2%	18.4

The performance of DNN and HMM systems are further evaluated by perceptual subjective tests. 30 utterances, which are randomly selected from the testing set and synthesized by the best baseline HMM system (MDL=1) and DNN systems (512*3 and 1024*3), are evaluated in two AB preference tests participated by 6 subjects. There are three preference choices: 1) the former is better; 2) the latter is better; 3) no preference or neutral (The difference between the paired sentences cannot be perceived or can be perceived but difficult to judge which one is better). The preference scores are shown in Fig. 3. It shows the speech synthesized by the DNN system with 1024*3 structure is preferred than the best HMM system. Its preference score (67%) is higher than the baseline system (23%). While the perception difference between the best HMM system and DNN system (512*3) is not significant, i.e. the preference scores are 46% vs. 44%. (Some samples of synthesized utterances are given on the web link: http://research.microsoft.com/en-us/projects/dnntts/default.aspx)

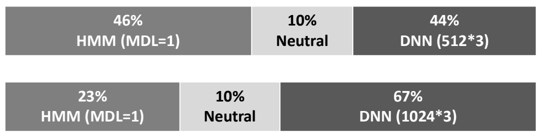

Fig. 3. The preference scores of the best baseline HMM system and DNN systems with different system complexities.

It is difficult to use DNN for sequence modeling while HMM is effective to model the evolution of speech signals as a stochastic sequence of acoustic feature vectors. To use DNN for modeling speech features, generally the input and output features need to be force aligned by HMMs in advance. In our experiments of DNN training, the input text features and output acoustic features are aligned at state level. Each input feature vector has 5 dimensional binary features, which indicate the current frame belonging to the state position of current phone. The state duration of both training and testing sentences are also given by HMM based system. We also try to align input and output features at phone level and use input features to indicate the coarse boundaries (by evenly dividing a phone into five parts) in a given phone. The results of objective measures, shown in Table 3, indicate that aligned state boundaries are effective for generating better synthesized speech than coarse boundaries.

Table 3. The results of objective measures of DNN (512*3) trained by given precise state or coarse boundary

Measures / Boundary	LSD (dB)	V/U Error rate	F0 RMSE (Hz)
Precise	3.76	5.9%	15.8
Coarse	3.92	6.7%	16.2

4. CONCLUSIONS

DNN training is examined in this study for parametric TTS synthesis. The results show that DNN performs better than HMM-based baseline in TTS, very likely due to the distinctive advantages of DNN, such as DNN is efficient and effective in representing high dimensional and correlated features, and modeling highly complex mapping function in a compact manner. In future work, we will refine the training criterion of DNN by incorporating MGE into a sequence training procedure at the sentence level in order to exploit the correlations of static and dynamic features and carry out a sentence level, global optimization.

5. REFERENCES

[1] G. E. Dahl, D. Yu, L. Deng, and A. Acero, “Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition,” *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

[2] F. Seide, G. Li, and D. Yu, “Conversational speech transcription using context-depedent deep neural networks,” in *Proc. InterSpeech*, pp. 437–440, 2011.

[3] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, “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.

[4] K. Tokuda, T. Kobayashi, T. Masuko, T. Kobayashi,, and T. Kitamura,, “Speech Parameter generation algorithms for HMM-based speech synthesis”, In *Proc. ICASSP,* pp. 1315-1318, 2000.

[5] H. Zen， K. Tokuda, and W. Black, Alan, “Statistical parametric speech synthesis”, *Speech Communication,* Volume 51, Issue 11, pp. 1039-1064, 2009.

[6] H. Zen, A. Senior and M. Senior, “Statistical Parametric Speech Synthesis Using Deep Neural Networks”, In *Proc. ICASSP*, pp. 8012-8016, 2013.

[7] H. Lu, S. King, and O. Watts, “Combining a vector space representation of linguistic context with a deep neural network for text-to-speech synthesis”, In *8th ISCA Workshop on Speech Synthesis*, pp. 281-285, 2013.

[8] S. Kang, X. Qian, and H. Meng, “Multi-distribution deep belief network for speech synthesis”, In *Proc. ICASSP*, pp. 7962-7966, 2013.

[9] Z.-H. Ling, L. Deng, and D. Yu, “Modeling spectral envelopes using restricted Boltzmann machines for statistical parametric speech synthesis”, In *Proc. ICASSP*, pp. 7825-7829, 2013.

[10] G.E. Hinton, S. Osindero and Y. W. Teh, “A Fast Learning Algorithm for Deep Belief Nets,” Neural Computation, vol. 18, no. 7, pp. 1527–1554, 2006.

[11] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” Nature, vol. 323, no. 9, pp. 533–536, 1986.

[12] A. Krogh and J. A. Hertz, “A Simple Weight Decay Can Improve Generalization”, in *Advance in Neural Information Processing Systems 4*, J.E. Moody, S.J. Hanson and P.R. Lippmann, eds. Morgan Kauffmann Publishers, San Mateo CA, pp. 950-957, 1992.

[13] H. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*, Kluwer Academic Publishers, Norwell, MA, USA, 1993.

[14] D. Erhan, Y. Bengio, A. Courville, P.A. Manzagol, P. Vincent, and S. Bengio, “Why does unsupervised pre-training help deep learning?,” JMLR, 2010.

[15] D. Yu, L. Deng, and G. Dahl, “Roles of pre-training and fine-tuning in context-dependent DBN-HMMs for real-world speech recognition,” in *NIPS Workshop*, 2010.

[16] F. Seide, G. Li, X. Chen, and D. Yu, “Feature engineering in context-dependent deep neural networks for conversational speech transcription,” in IEEE ASRU, 2011.

[17] T.N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novak, and A.R. Mohamed, “Making deep belief networks effective for large vocabulary continuous speech recognition,” in IEEE ASRU, 2011

[18] Y. Bengio, “Learning deep architectures for AI,” *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[19] Y.-J. Wu and R.H. Wang, “Minimum generation error training for HMM-based speech synthesis”, In Proc. ICASSP, 2006.

[20] C. Julian Chen, Ramesh A. Gopinath, Michael D. Monkowski, Michael A. Picheny, and Katherine Shen, “New methods in continuous Mandarin speech recognition.,” in EUROSPEECH. 1997, ISCA.

[21] Z.-H. Ling, Y.-J. Wu, Y.-P. Wang, L. Qin, and R.-H. Wang, “USTC System for Blizzard Challenge 2006 an Improved HMM-based Speech Synthesis Method,” *Proc. Blizzard Challenge 2006 Workshop*, 2006.

[22] K. Shinoda, and T. Watanable, “MDL-based Context-Dependent Sub-word Modeling for Speech Recognition”, J. Acoust. Soc. Jpn(E), vol.21, no.2, pp.79-86, 2000.