Design and development of a large vocabulary, continuous speech recognition system for Tamil

A. Madhavaraj
Electrical Engineering,
Indian Institute of Science,
Bangalore, India - 560012
madhavaraja@iisc.ac.in

A. G. Ramakrishnan
Electrical Engineering,
Indian Institute of Science,
Bangalore, India - 560012
agr@iisc.ac.in

***Abstract*—This paper presents our work on building a large vocabulary continuous speech recognition system for Tamil using deep neural networks (DNN). Well known techniques, namely, maximum likelihood linear transformation and speaker-adaptive training have been used to build our final deep neural network based speech recognition system. We have used 6.5 hours of Tamil speech recorded from 30 speakers covering a vocabulary of 13,026 words, of which 4.5 hours of data was used for training, 1 hour of data for testing and 1 hour of data for cross-validation. Two independent recognition systems were built, one for phone recognition (PR) and the other for continuous speech recognition (CSR) and they achieve phone error rate of 24.9% and word error rate of 3.5%, respectively. DNN-based triphone acoustic model shows an absolute improvement of about 1% and 23% over the monophone acoustic model for CSR and PR, respectively.**

***Index Terms*—Speech recognition, acoustic model, language model, deep neural networks, hidden Markov model, ASR, Tamil**

I. Introduction

In the last 40 years, we have seen steady progress in speech recognition research [1]. This progress can be attributed to two factors: (i) the use of hidden Markov model (HMM) in modeling the temporal variations in speech [2] and (ii) the increasing computational power of modern computers [3]. In the past ten years alone, we have seen many low-cost, commercial interactive speech recognition applications developed by Apple, Google, Microsoft, Amazon, etc. It is also well known that automatic speech recognition (ASR) research is mainly focused on English and other European languages [4]. It can be said that no substantial progress has been made for Indian languages, especially Dravidian languages such as Tamil. One of the main drawbacks in developing a Tamil speech recognition system is the unavailability of standard speech and text corpora. Our research focuses on overcoming these limitations to build a reasonably good, large vocabulary, continuous speech recognition (LVCSR) system for Tamil.

Speech recognition researchers around the world have acknowledged the efficiency of deep neural networks (DNNs) in building ASR systems. DNNs trained on several thousand hours of speech have reduced the word error rate significantly compared to the traditional methods and achieve word-level accuracies of nearly 90% for vocabulary sizes of about 200,000 for English language [4]. Such ASR systems are now widely used for commercial and entertainment purposes. Due to the lack of standard speech databases, ASR research in Tamil has not progressed at all. This motivated us to take up the work on building a domain and speaker-independent ASR system for a large vocabulary task.

The rest of the paper is organized as follows. Section II describes the building blocks of an ASR system. Section III discusses the tools and databases used in building our Tamil ASR system. In Section IV, we describe the steps in building a Tamil ASR. We provide the evaluation results of our phone and continuous speech recognition systems in Section V. Finally, we conclude and briefly discuss our future research directions in Section VI.

II. Description of our Tamil ASR system

The process of speech recognition involves many modules as indicated in Fig. 1. The first step is to acquire the speech signal through a microphone and convert it to digital format. The next step is pre-processing, which involves noise removal and converting the signal to a sequence of frames using a windowing technique. The next step is to extract relevant features from the frame sequence. The commonly used features are Mel-frequency cepstral coefficients (MFCC) and perceptual linear prediction coefficients. The extracted features are presented to the decoder, which uses an acoustic model (AM), a language model (LM) and a lexicon model (pronunciation dictionary) to decode the best possible word sequence.

A. *Pre-emphasis and framing*

The speech signal, acquired through a microphone, is first pre-processed. This stage involves mean subtraction, noise removal and pre-emphasis to minimize the effects due to channel degradation. Since speech is a quasi-stationary signal, we take overlapping frames, each of size 20 msec with a frame shift of 5 msec, so that the properties of speech for a phone remain invariant within a frame. Each frame is then multiplied by a Hamming window function so as to reduce the ripple effects of the framing process.

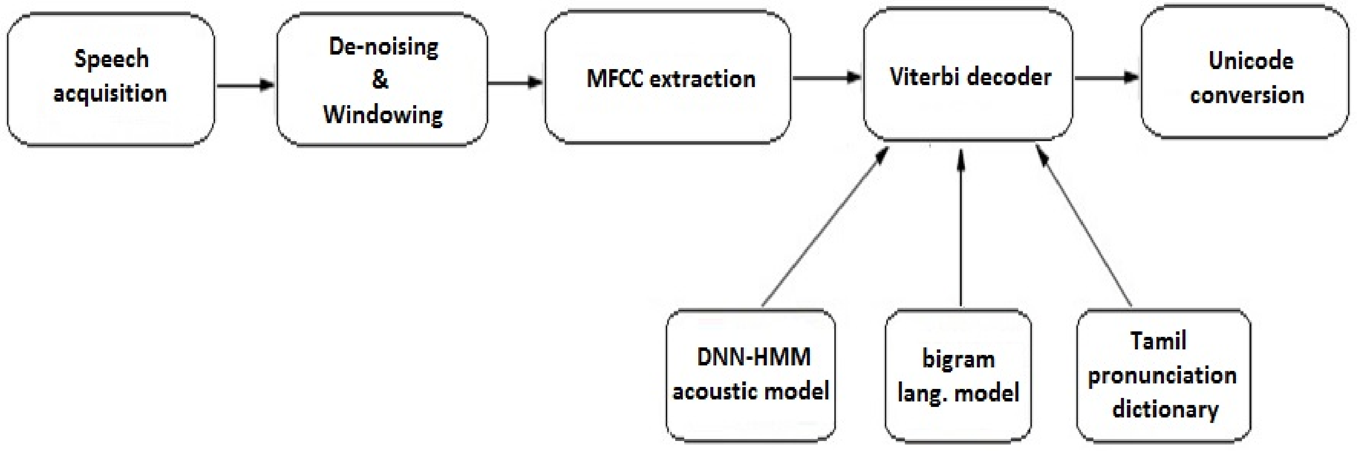


Fig. 1. Block diagram of our Tamil, large vocabulary, automatic speech recognition system.

B. Extraction of MFCC features

The next stage involves extracting relevant features, which can characterize the phonemes in the speech frames. Conventional ASR systems use one of the following features: Mel-frequency cepstral coefficients, log filter-bank energy, perceptual linear prediction cepstral coefficients [5]. The feature extraction block essentially extracts features, which serve as a good acoustic representation of the speech units (or phones), while suppressing other irrelevant variations in the signal due to the other factors such as the speaker, the channel, the speaking style of the speaker and the recording environment.

C. DNN-HMM based acoustic model

The acoustic model statistically represents the relation between the phone labels and the features from the speech signal. Modern ASR systems use left-to-right hidden Markov models to capture and model the temporal variations in each phone. Each state in a HMM models a probability density function specified by a Gaussian mixture model (GMM) [6]. Recently, GMMs have been replaced by deep neural networks in modeling the probability density function of the state (also called as state-posteriors), thus giving rise to DNN-HMM acoustic models [4][7]. The ability of DNNs in being able to accurately learn highly non-linear functions is exploited in modeling the complex posterior density functions.

D. Word bigram language model

The language model is a statistical model, which facilitates the ASR system to distinguish between words or phrases that sound similar. Usually, word N-gram models are used as the LM, which models the conditional density function of a word given the previous $N-1$ words. Thus, the joint probability of any hypothesis word sequence can be approximated as the product of these conditional probabilities as indicated in (1). These N-gram models are learned from a huge corpus of text. Back-off and other estimation techniques can be used to prune the LM for a desired language perplexity [8].

$$p(w_1, w_2, .., w_K) \approx \prod_{i=2}^{K} p(w_i | w_{i-1}, .., w_{i-N+1}) \tag{1}$$

E. Pronunciation model

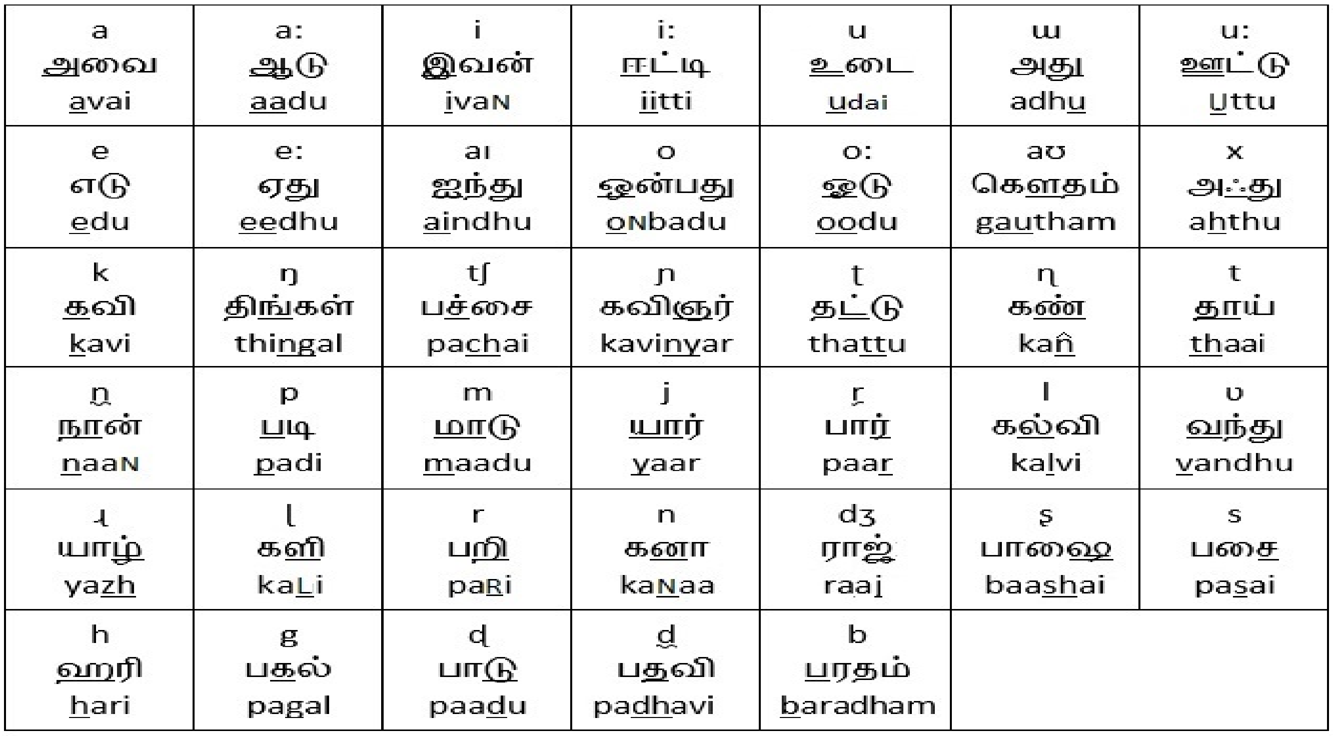
The lexicon model or pronunciation dictionary serves as a link between the acoustic and language models. The lexicon maps the words in the vocabulary to their corresponding phoneme sequences. It also handles multiple pronunciations for a single word, using pronunciation probabilities [9]. Lexicons can be thought of as a grapheme-to-phoneme converter, which are designed by linguists (phonologists). For English ASR, the Carnegie Mellon University (CMU) dictionary is a widely used lexicon [10].

F. Viterbi word sequence decoder

The Viterbi graph search algorithm decodes the best possible word sequence for a given sequence of feature vectors [11]. The decoder uses the AM, LM and lexicon probabilities to predict the best likely word sequence. Mathematically, the decoder solves the optimization problem given in (2). As the search space amongst all the possible word hypotheses becomes exponentially large, beam search can be employed to prune the low probability paths during the search [12].

$$\{w_1^*, .., w_M^*\} = \underset{w_1, .., w_M}{\text{argmax}} \; p(x_1, .., x_N | w_1, .., w_M) p(w_1, .., w_M) \tag{2}$$

Finally, the post-processing stage converts the recognized best word sequence to human/machine readable format, namely a sequence of Unicode strings. For an ASR system to perform efficiently, we need to train the AM on several hundred hours of transcribed speech corpus. Similarly, the LM needs to be trained on a text corpus of huge size.



a அவை avai	a: ஆடு aadu	i இவன் ivaɴ	i: ஈட்டி iitti	u உடை udai	ɯ அது adhu	u: ஊட்டு Uttu
e எடு edu	e: ஏது eedhu	aɪ ஐந்து aindhu	o ஒன்பது oɴbadu	o: ஓடு oodu	aʊ கௌதம் gautham	x அஃது ahthu
k கவி kavi	ŋ திங்கள் thingal	tʃ பச்சை pachai	ɲ கவிஞர் kavinyar	ʈ தட்டு thattu	ɳ கண் kaɴ̂	t தாய் thaai
n̠ நான் naaɴ	p படி padi	m மாடு maadu	j யார் yaar	r̠ பார் paar	l கல்வி kalvi	ʋ வந்து vandhu
ɻ யாழ் yazh	ɭ களி kaLi	r பறி paRi	n கனா kaNaa	dʒ ராஜ் raaj	ʂ பாஷை baashai	s பசை pasai
h ஹரி hari	g பகல் pagal	ɖ பாடு paadu	d̪ பதவி padhavi	b பரதம் baradham		

Fig. 2. List of phonemes in Tamil language and one example word each for their occurrences. For each entry, the Tamil phoneme label (as per International Phonetic Alphabet notation) is given in the first line, an example word containing the phoneme in the second line, and the transcription of the word using Roman alphabet in the last line.

III. Tools and dataset used for Tamil ASR

To develop our Tamil ASR system, we have used the state-of-the-art open source toolkit named Kaldi [13]. It has numerous functionalities, which can be used to build the AM of our desired choice. To build the LM, we have used the toolkit named IRSTLM to compute the bigram word probabilities [14]. We have built our own grapheme to phoneme converter tool [15] to build the lexicon model (in order to convert words to the corresponding phone sequences). We have identified a total of 40 phonemes in Tamil language and our ASR system is built based on this phoneme set. Figure 2 shows the set of Tamil phones used and gives one example word for the occurrence of each phoneme. Along with the 40 phonemes, we have an additional silence phoneme.

Learning an acoustic model and a language model requires transcribed speech corpus and text corpus, respectively. We have obtained 6.5 hours of transcribed speech recordings from the Central Institute of Indian Languages, Mysore [16]. The recordings are single-channel, close-talk, PCM data sampled at 16 kHz with a resolution of 16 bits per sample. This corpus is a newspaper read-speech covering a vocabulary of 13,026 words recorded from 30 speakers (18 male and 12 female). The entire corpus is divided into three chunks: 4.5 hrs (training), 1 hour (development) and 1 hour (test). The training set is used to learn the AM parameters and the development set is used for the purpose of validation and the test set is used for reporting the recognition performance of our ASR system.

We have used two NVIDIA GeForce GTX TITAN X graphics processing units (GPUs) on a 40-core Intel Xeon workstation to run all our training and testing modules.

IV. Training the acoustic model

Our aim is to build a DNN-HMM based acoustic model for our speech recognition task. For this, we need the speech corpus to have time-markings at the phone level (also called as alignments) but we have transcription only at the sentence level. So, the first step is to convert the words in the sentences to phoneme sequences using the lexicon and assign equal intervals for the phonemes corresponding to the speech file under consideration, as shown in Fig. 3. These alignments are called "equal alignments" and each step mentioned below aims to refine the alignments and pass them on to the next stage. At each stage, we do acoustic modeling by increasing the complexity of the system.

A. *Monophone training*

We first build a simple 3-state, monophone HMM model for each of the 41 phonemes and model each HMM state as a GMM. We have used a total of 1000 diagonal covariance Gaussians to be shared among all the 123 states (instead of

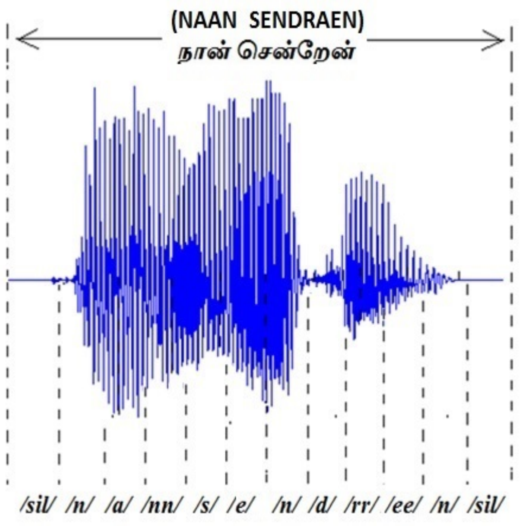


Fig. 3. Illustration of initial alignment - initial assignment of equal duration to each phone of a sample speech file.

having a fixed number of Gaussians per state). The model parameters of GMM-HMM (means, covariances, mixture weights, transition probabilities) are learnt using maximum likelihood estimation procedure (by expectation-maximization algorithm), starting from the equal-alignments. The training is performed for 40 iterations and after each iteration, we do re-alignment and pass on the new alignments to the next iteration.

B. Context-dependent triphone training

In this stage, we perform context-dependency modeling of each phone, considering its left and right context phones (triphone context). Thus we have 41^3 HMMs, one for each triphone. Not all the triphones are likely to occur in the training set and so we share the states and parameters of the HMMs by having a fixed number of diagonal-covariance Gaussians (10,000) and a fixed number of states (2,000). The alignment obtained at the end of monophone training is used to train the triphone model iteratively and after each iteration, the alignments are refined.

C. Maximum likelihood linear transformation (MLLT) training

The modeling power of AM can be further increased by having full-covariance Gaussians instead of diagonal ones. This involves the estimation of a transform, which can be applied on the mean vectors of the trained GMM and simultaneously applying the transformation on the input feature vector. More details about this technique can be found in [17]. Such an MLLT triphone system is trained using the alignments from the previous step.

D. Speaker-adaptive training (SAT)

This stage involves complicating the acoustic model by learning parameters which are specific to the speakers. SAT [18] involves estimating a linear transformation (for each speaker) to be applied on the means of GMM-HMM of a global model. This SAT model is learned using the alignments from the previous stage. The intuition is that the likelihood of the utterance given the speaker-specific model will be more than the likelihood with respect to the global model, thereby increasing the accuracy of the alignments.

E. DNN training

Using the alignments obtained from SAT, we train a DNN to predict the state posterior distribution. We have used back-propagation with momentum to train our 7-layer DNN (with 256 tanh non-linear nodes in each hidden layer) for 50 epochs. There are several issues to be handled during the training of a DNN, such as learning rate adjustment and gradient-explosion/vanishing problem. This can be avoided by initializing the network using a pre-trained restricted Boltzmann machine-based deep belief network (RBM-DBN). This RBM-DBN is learnt in an unsupervised manner from the validation data. Finally, the trained DNN is combined with the bigram LM and lexicon model to arrive at the complete ASR system.

V. Experiments and results

We have built two independent ASR systems: continuous speech and phoneme recognizers and evaluated the performance of both the systems on the 1-hour test set. For the continuous speech recognition (CSR) system, word-level bigram LM has been used and for the phoneme recognition system (PR), phone-level bigram LM has been used. The error rates obtained for the two systems for different acoustic model types are listed in Table I. It can be seen that the error rate gradually reduces as the complexity of the acoustic model increases, though there is an anomaly with respect to the speaker adaptive trained model. This is due to the fact that we have only 18 (out of 30) speakers in the training set and the training procedure fails to estimate the speaker transformation matrix efficiently due to the limited speaker-space. The performance can be improved if we train speech obtained from a large set of speakers.

From Table I, we can see that the best performance is attained by DNN-based acoustic model in both the cases. DNN models show an absolute improvement of about 1% and 23% over monophone models for CSR and PR, respectively.

VI. Conclusion

We have elaborated the steps involved in building an end-to-end, DNN-based Tamil ASR system with the state-of-the-art tools. With 6.5 hours of data (with a vocabulary of 13,026 words), our DNN-triphone AM for continuous speech

TABLE I
PERFORMANCE EVALUATION OF OUR PHONEME AND CONTINUOUS SPEECH RECOGNITION SYSTEMS ON TAMIL TEST DATA OF ONE HOUR DURATION.

Acoustic model type	**Phone error rate (%)**	**Word error rate (%)**
Monophone	47.48	4.42
CD-triphone	37.90	4.25
CD-Triphone+MLLT	27.74	4.20
CD-Triphone + SAT	29.25	6.46
DNN	24.52	3.48

recognizer is able to achieve a word error rate of 3.5% with an absolute improvement of 1% over monophone AM. Similarly, the DNN-triphone AM for phone recognizer achieves a phone error rate of 24.9% with an absolute improvement of 23% over monophone AM. We plan to collect about 150 hours more of speech data for training and scale the vocabulary size to about 100,000 words and use cross-lingual training to further reduce the word error rate.

VII. ACKNOWLEDGEMENT

The authors would like to thank Dr. L. Ramamoorthy, Head, Linguistic Data Consortium for Indian Languages, CIIL Mysore for providing us the transcribed speech corpus, which has been used in this work.

REFERENCES

[1] Lawrence Rabiner and Biing-Hwang Juang. 1993. "Fundamentals of Speech Recognition," Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[2] Lawrence R. Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition," Readings in speech recognition, Alex Waibel and Kai-Fu Lee (Eds.). Morgan Kaufmann Publishers Inc., 1990, San Francisco, CA, USA 267-296.

[3] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos, E. Elsen, J. Engel, L. Fan, C. Fougner, T. Han, A. Hannun, B. Jun, P. LeGresley, L. Lin, S. Narang, A. Ng, S. Ozair, R. Prenger, J. Raiman, S. Satheesh, D. Seetapun, S. Sengupta, Y. Wang, Z. Wang, C. Wang, B. Xiao, D. Yogatama, J. Zhan, Z. Zhu, "Deep speech 2: End-to-end speech recognition in English and Mandarin," Proceedings of the 33rd Intl. Conference on Machine Learning, New York, NY, USA, 2016

[4] G. Hinton et al., "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, Nov. 2012.

[5] Beth Logan, "Mel Frequency Cepstral Coefficients for Music Modeling," International Symposium on Music Information Retrieval, 2000.

[6] Mark Gales and Steve Young. "The application of hidden Markov models in speech recognition," Found. Trends Signal Process. 1, 3 (January 2007), 195-304.

[7] Fred Richardson, Douglas Reynolds and Najim Dehak, "Deep neural network approaches to speaker and language recognition," Signal Processing Letters IEEE, vol. 22, pp. 1671-1675, 2015.

[8] M Mohri, F Pereira and M Riley, "Weighted finite-state transducers in speech recognition," Computer Speech & Language, 2002.

[9] Chen, Guoguo, Hainan Xu, Minhua Wu, Daniel Povey and Sanjeev Khudanpur. "Pronunciation and silence probability modeling for ASR." INTERSPEECH 2015.

[10] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

[11] Zhu Xuan, Chen Yining, Liu Jia and Liu Runsheng, "A novel efficient decoding algorithm for CDHMM-based speech recognizer on chip," Acoustics, Speech, and Signal Processing, Proceedings. (ICASSP '03). IEEE International Conference on, 2003, pp. II-293-6 vol.2.

[12] H. Ney, D. Mergel, A. Noll, A. Paeseler, "A data-driven organization of the dynamic programming beam search for continuous speech recognition," Proc. IEEE Int. Conf. on Acoustics Speech and Signal Proc. pp. 833-836 1987.

[13] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Nagendra Goel, Mirko Hannemann, Yanmin Qian, Petr Schwarz and Georg Stemmer, "The Kaldi speech recognition toolkit," In IEEE 2011 workshop.

[14] Marcello Federico, Nicola Bertoldi and Mauro Cettolo, "IRSTLM: an open source toolkit for handling large scale language models," INTERSPEECH, 2008.

[15] A. G. Ramakrishnan and Laxmi Narayana M, "Grapheme to phoneme conversion for Tamil speech synthesis," Proc. Workshop in Image and Signal Processing (WISP-2007), IIT Guwahati, Dec 28-29 2007, pp. 96-99.

[16] http://www.ldcil.org/resourcesSpeechCorpTamil.aspx

[17] M. J. F. Gales, "Semi-tied covariance matrices for hidden Markov models," IEEE Transactions on Speech and Audio Processing, vol. 7, no. 3, pp. 272-281, May 1999.

[18] M.J.F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," Computer Speech & Language, Volume 12, Issue 2, 1998, Pages 75-98, ISSN 0885-2308.