ISSN 2320 - 2602
Volume 5 No.1, January 2016
International Journal of Advances in Computer Science and Technology
Available Online at http://www.warse.org/IJACST/static/pdf/file/ijacst01512016.pdf

A Combination of Deep Neural Networks for Acoustic modeling of Vietnamese LVCSR

Quoc Bao Nguyen[1], Tat Thang Vu[2], Chi Mai Luong[2]
[1]University of Information and Communication Technology, Vietnam, nqbao@ictu.edu.vn
[2]Institute of Information Technology, Vietnam Academy of Science and Technology, Vietnam, {vtthang,lcmai}@ioit.ac.vn

ABSTRACT

In this work, we propose a deep neural network architecture with the combination of two popular applications of deep neural networks for Vietnamese large vocabulary continuous speech recognition. First, a deep neural network is trained to extract bottleneck features from frames of a combination of Mel frequency cepstral coefficient (MFCC) and tonal feature. This network is then applied as a nonlinear discriminative feature-space transformation for hybrid network training where acoustic modeling is performed by denoising auto-encoder pre-training and back-propagation algorithms. The experiments are carried out on the dataset containing speeches on Voice of Vietnam channel (VOV). The results show that the performance of the system using combined deep neural network architecture obtained relative improvements over the best hybrid HMM/DNN system by 4.1% and over baseline system by 51.4%. Adding tonal feature as input feature of the network reached around 18% relative recognition performance.

Key words : Bottleneck feature, Deep neural network, Vietnamese LVCSR.

1. INTRODUCTION

Recently, deep neural network (DNN) has achieved great success on Large Vocabulary Continuous Speech Recognition (LVCSR) tasks [1][2]. There are two popular approaches today including hybrid and bottleneck feature. In the hybrid approach, a neural network is trained to estimate the emission probabilities for HMM [3]. In contrast, the bottleneck feature approach [4] uses the activations of a small hidden layer of neural network as input values for the common combination of GMM and HMMs.

However, applying DNN in Vietnamese LVCSR is still in early stage. Previous study [5] has demonstrated that performance of Vietnamese LVCSR can be improved by using deep neural network with denoising auto-encoder pre-training method. They are still done in determining which speech features and network architecture are most useful when training deep neural network.

Regarding tonal feature extraction for Vietnamese recognition, previous works [6][7][8] showed efforts toward Vietnamese large vocabulary continuous speech recognition by combining tonal features with a standard acoustic feature like MFCC for acoustic modeling. However their approaches for extracting tonal feature were based on Getf0 algorithm [9] which makes a hard decision whether any given frame is voiced or unvoiced.

In this work, we show the way to extract the pitch feature using modification of the Getf0 algorithm [9] which all frames are treated as voiced and allow the Viterbi search to naturally interpolate across unvoiced regions and it can be combined with acoustic feature for using as input of DNN. Furthermore, we also proposed neural network architecture by combining of two popular deep neural network to see whether bottleneck features are useful input feature for DNN training. The motivation for the proposed architecture is that, combination of separately trained sub-networks into bigger networks is a well-established design principles when building large classifier. Bottleneck features can be regarded as a discriminative dimensionality reduction technique which is known to work well with GMM/HMM system.

2. ACOUSTIC MODEL IN SPEECH RECOGNITION

The acoustic model is used to model the statistics of speech features for each phone or sub phone. Hidden Markov Model (HMM) [10] is the standard used in the state-of-the-art acoustic models. It is a very powerful statistical method to model the observed data in a discrete-time series.

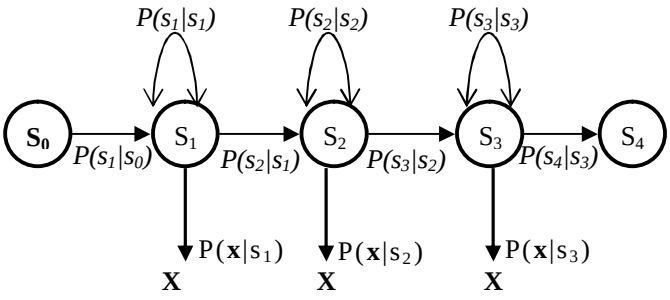

Figure 1: A left-to-right HMM model

An HMM is a structure that contains a group of states connected by transitions. Each transition is specified by its transition probability. In speech recognition, state transitions are usually connected from left to right or self repetition (the left-to-right model) as shown in Figure 1.

Each state of HMMs is usually represented by a Gaussian Mixture Model (GMM) to model the distribution of feature vectors for the given state. A GMM is a weighted sum of M component Gaussian densities and is described by Eq. 1.

$$P(x \mid \lambda) = \sum_{i=1}^{M} w_i g(x \mid \mu_i, \Sigma_i) \quad (1)$$

Where $P(x \mid \lambda)$ is the likelihood of a D dimensional continuous-valued feature vector x, given the model parameters $\lambda = w_i, \mu_i, \Sigma_i$, where w_i is the mixture weight which satisfies the constraint $\Sigma_i^M w_i = 1, \mu_i$ is the mean vector, and Σ_i is the co-variance matrix of the i_{th} Gaussian function which is defined by

$$g(x \mid \mu_i, \Sigma_i) = \frac{1}{(2\Pi)^{D/2} \mid \Sigma_i \mid^{1/2}} exp(-\frac{1}{2}(x-\mu_i)'\Sigma_i^{-1}(x-\mu_i)) \quad (2)$$

3. TONAL FEATURE EXTRACTION

Vietnamese is a syllabic tonal language with six lexical tones, which are very important to decide on the word meanings. A change in tone can lead to the change in word meaning, which can be vastly different from the original word. Therefore, tonal feature extraction task in Vietnamese recognition is very important. In tonal languages, e.g. Vietnamese, Cantonese, Chinese (phonetic) pitch carries phonological (tone) information and needs to be modeled explicitly. To detect tones, one needs to detect rising, falling, or otherwise marked pitch contours. By themselves, pitch features are insufficient to distinguish all the phonemes of a language, but pitch (absolute height or contour) can be the most distinguishing feature between two sounds.

There are various off-the-shelf pitch extractors, namely Yin [11], Getf0 [9], SAcC [12]. The pitch features of SAcC, Yin and getf0 were compared in an ASR task by Pegah et al. [13]. The Getf0 seemed to perform slightly better than Yin, and it is a relatively simple algorithm to implement (SAcC gave better performance but it is a complex method). Based on that comparison Pegah et al. proposed a method which is called the Kaldi pitch tracker (because they have added it to the Kaldi ASR toolkit [14]), it is a highly modified version of the getf0 algorithm [9] which is based on the Normalized Cross Correlation Function (NCCF) as defined in (3):

$$NCCF(\tau) = \frac{1}{\sqrt{e_n e_\tau}} \sum_{n=0}^{N-\tau-1} x(n)x(n+\tau), \quad (3)$$

where $e_i = \sum_{n=i}^{i+N-1} x^2(n)$, x(n) is the input speech sample, N is the length of the speech analysis window, τ is the lag number in range between 0 and N-1.

In this work, we also follow the Kaldi pitch tracker for tonal feature extraction because it improves pitch tracking as measured by Gross Pitch Error, versus the off-the-shelf methods they tested. A Kaldi pitch tracker algorithm is based on finding lag values that maximize the NCCF. The most important change from getf0 is that rather than making hard decisions about voicing on each frame, all frames are treated as voiced to be continuous and allow the Viterbi search to naturally interpolate across unvoiced regions.

The output of the algorithm is 2-dimensional features consisting of pitch in Hz and NCCF on each frame, and then the output is post-processed for use as features for ASR, produces 3-dimensional features consisting of pov-feature, pitch-feature and delta-pitch-feature:

1. pov-feature is warped NCCF. This method was designed to give the feature a reasonably Gaussian distribution (although there are still noticeably separate peaks for voiced and unvoiced frames). Let the NCCF on a given frame be written c. If $-1 \le c \le 1$ is the raw NCCF, the output feature be $f = 2((1.0001-c)^{0.15} - 1)$.

2. pitch-feature is feature that on each time t, subtraction of a weighted average pitch value, computed over a window of width 151 frames centered at t and weighted by the probability of voicing value p. Where p is obtained by plotting the log of *count(voiced)/count(unvoiced)* on the Keele database [15] as a function of the NCCF and manually creating a function to approximate it.

3. delta-pitch-feature is delta feature computed on raw log pitch.

4. DEEP NEURAL NETWORK FOR SPEECH RECOGNITION

4.1 Deep Bottleneck Features versus Hybrid HMM and DNN

The deep neural network architecture for hybrid HMM/DNN is briefly described as in Figure 2. The network consists of a variable number of moderately large, fully connected hidden layers which is followed by the final classification layer. Another deep neural network architecture for bottleneck feature extraction proposed in [16] that is depicted in Figure 3. This architecture is similar to the one for hybrid HMM/DNN, the difference is that it has an additional small bottleneck layer.

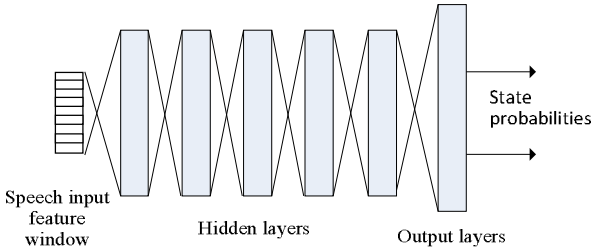

Figure 2: Deep Network Architecture for Hybrid HMM/DNN

The hidden layers are initialized using unsupervised, layer-wise pre-training. Thanks to their success in the deep learning community, restricted Boltzmann machines have become the default choice for pre-training the individual layers of deep neural networks used in speech recognition. Gehring et al. [16] demonstrated that denoising auto-encoders which are straight-forward models that have been successfully used for pre-training neural architectures for computer vision and sentiment classification [17] are suitable to speech data as well.

We follow their training scheme and initialize the hidden layers as denoising auto-encoders, too. Like regular auto-encoders, these models consist of one hidden layer and two identically-sized layers representing the input and output values. The network is usually trained to reconstruct its input at the output layer with the goal to generate a useful intermediate representation in the hidden layer. In denoising auto-encoders, the network is trained to reconstruct a randomly corrupted version of its input, which can be interpreted as a regularizing mechanism that facilitates the learning of large and over complete hidden representations [18].

For denoising auto-encoders working on binary data (i.e. gray scale images or sigmoid activation of a previous hidden layer), Vincent et al. proposed the use of masking noise for corrupting each input vector [18] in order to extract useful features, the network is forced to reconstruct the original input from a corrupted version, generated by adding random noise to the data. This corruption of the input data can be formalized as applying a stochastic process q_D to an input vector x:

$$\tilde{x} \sim q_D(\tilde{x} \mid x) \quad (4)$$

This approach also used in our work is to apply masking noise to the data by setting every element of the input vector to zero with a fixed probability. Then the corrupted input first maps (with an encoder) to the hidden representation y using the weight matrix W of the hidden layer, the bias vector b of the hidden units and a non-linear activation function σ_y as follows:

$$y = \sigma_y(W\tilde{x} + b) \quad (5)$$

The latent representation y or code is then mapped back with a decoder into reconstruction z using the transposed weight matrix and the visible bias vector c. Because in a model using tied weights, the weight values are used for both encoding and decoding, again through a similar transformation σ_z:

The parameters of this model (namely W, W^T, b, c) are optimized such that the average reconstruction error is minimized. The reconstruction error can be measured by the cross-entropy error objective as defined in (7) in order to obtain the gradients necessary for adjusting the network weights.

$$L_H(x, z) = \sum_i x_i \log z_i + (1 - x_i) \log z_i \quad (7)$$

When training a network on speech features like MFCCs, the first layer models real valued rather than binary data, so the mean squared error $L_2(x, z) = \sum_i (x_i - z_i)^2$ is selected as the training criterion. In this work, we also apply masking noise to the first layer, although other types of noise could be used as well [18].

After a stack of auto-encoders has been pre-trained in this fashion, a deep neural network can be constructed. The other remained layers are initialized with random weights and connected to the hidden representation of the top-most auto-encoder, and the resulting network is fine-tuned with standard back propagation.

In the trained network for hybrid DNN/HMM, they are used to compute a posteriori emission probabilities of phone states. If the network is trained to estimate probabilities $p(s_t \mid x_t)$ of state s_t given observations as input feature vector x_t using a cross-entropy criterion, the emission probabilities can be obtained with Bayes' rule:

$$p(x_t \mid s_t) = \frac{p(s_t \mid x_t)\, p(x_t)}{p(s_t)} \quad (8)$$

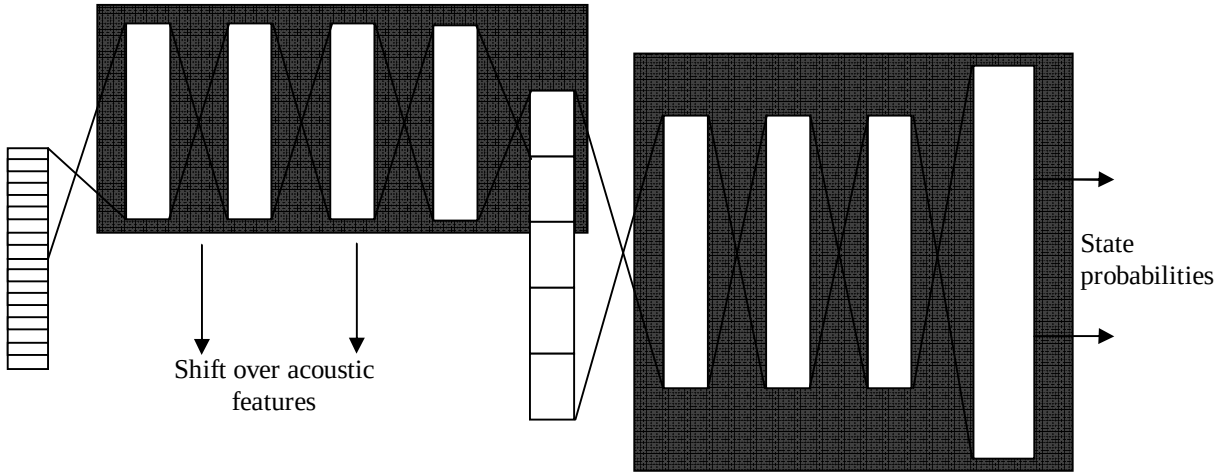

Figure 4: Combined architecture for acoustic modeling

where $p(s_t)$ denotes the prior probability of a phone state, which is estimated using the available training data. During decoding, the most likely sequence of states is computed by the HMM. Since the observation x is independent of the state sequence, therefore its probability $p(x_t)$ can be ignored.

In the network for bottleneck features, the last two layers of the network can be discarded as the units in the bottleneck layer provide the final features used for training standard HMM/Gaussian mixture (GMM) acoustic models.

4.2 Combination of Deep Neural Networks for Acoustic Modeling

In this work, we combined two deep neural network architectures from above section to become a deeper architecture as illustrated in Figure 4. We also used denoising auto-encoder to model a window of bottleneck features which extracted by applying respective neural network to adjacent windows of acoustic features. This scheme is related to the frame shifting done with individual layers in time-delay neural networks [19][20] and forms the basis of many hierarchical or convolutional architectures. The weights of the bottleneck network are fixed during hybrid network training so that each network is trained in isolation. In practice, we can first train the bottleneck feature network, and then compute bottleneck features for all available training data for inputting of hybrid network.

5. EXPERIMENTAL SETUP

5.1 Corpora Description

The voice of Vietnam (VOV) corpus was used in our experiments, which is a collection of story reading, mailbag, new reports, and colloquy from the radio program the Voice of Vietnam. There are 23424 utterances in this corpus including about 30 male and female broadcasters and visitors. The number of distinct syllables with tone is 4923 and the number of distinct syllables without tone is 2101. The total capacity of the corpus in WAV format with 16 kHz sampling rate and analog/digital conversion precision of 16 bits is about 2.5 GB. The total time of the corpus is about 19hours which was separated into training set of 17 hours and 2 hours test set. All of transcriptions in the training data were used to train tri-gram language model.

5.2 Baseline Systems

Baseline HMM/GMM systems were performed with the Kaldi developed at Johns Hopkins University [16]. We extracted two sets of acoustic features to build baseline acoustic models. Those are MFCCs and PLP features, which are popular in speech recognition applications. In both feature extraction, 16-KHz speech input is coded with 13-dimensional MFCCs with a 25ms window and a 10ms frame-shift. Each frame of the speech data is represented by a 39-dimensional feature vector that consists of 13 MFCCs with their deltas and double-deltas. Nine consecutive feature frames are spliced to 40 dimensions using linear discriminant analysis (LDA) and maximum likelihood linear transformation (MLLT) [21] that is a feature orthogonalizing transform, is applied to make the features more accurately modeled by diagonal-covariance Gaussians.

All models used 4,600 context-dependent state and 96,000 Gaussian mixture components. The baseline systems were built, follow a typical maximum likelihood acoustic training recipe, beginning with a flat-start initialization of context-independent phonetic HMMs, followed by tri-phone system with 13-dimensional MFCCs or PLP plus their deltas and double-deltas and ending with tri-phone system using LDA+MLLT.

5.4 Network training

In our experiments, we extracted deep bottleneck features from MFCCs and PLP. The network input for these features was pre-processed using the splicing speaker-adapted features approach as in [22].

Table 1: Recognition performance for the Vietnamese system with MFCC and PLP features

Systems	Baseline	BNF	DBNF (DAE layer)s						
			1	2	3	4	5	6	7
PLP	22.08	14.7	17.19	14.38	13.93	13.88	13.86	**13.77**	13.99
MFCC	21.25	15.5	16.11	13.99	13.76	13.68	13.40	**13.39**	13.48
			No pre-training						
			15.52	14.84	**14.33**	14.35	14.41	14.51	14.69

During supervised fine-tuning, the neural network was trained to predict context-dependent HMM states (there were about 4600 states in our experiments). For pre-training the stack of auto-encoders in the architecture at section 4. Mini-batch gradient descent with a batch size of 128 and a learning rate of 0.01 was used. Input vectors were corrupted by applying masking noise to set a random 20% of their elements to zero. Each auto-encoder contained 1024 hidden units and after 20 epochs the weights were fixed and the next one was trained on top of it.

The remaining layers were then added to the network, with the bottleneck consisting of 39 units. Again, gradients were computed by averaging across a mini-batch of training examples; for fine tuning, we used a larger batch size of 256. The learning rate was decided by the "newbob" schedule: for the first epoch, we used 0.008 as the learning rate, and this was kept fixed as long as the increment in cross-validation frame accuracy in a single epoch was higher than 0.5%. For the subsequent epochs, the learning rate was halved; this was repeated until the increase in cross-validation accuracy per epoch is less than a stopping threshold, of 0.1%. After each epoch, the current model was evaluated on a separate validation set, and the model performing best on this set was used in the speech recognition system afterwards. For these experiments we used GPUs for training of auto-encoder layers and neural networks using the Theano toolkit [23].The training time for each DNN architecture as describe in Table 2.

6. EXPERIMENTAL RESULTS

In previous work [20], we compared different network architectures and inputs for extracting bottleneck feature on the Vietnamese language in terms of the word error rate (WER) as shown in Table 1. For the input data, we found that extracting deep bottleneck features from MFCC instead of PLP data resulted in consistently better recognition performance of about 0.4% WER absolute. Third column of Table 1 lists the WER of BNF systems which were reported previously in [24] and trained on Multilayer Perceptron (MLP) network using 3 layers (1000 units each) without using any pre-training technique. We also evaluated the experiments to extract bottleneck features from the networks of different depth, with and without unsupervised pre-training. We found the best configuration for DBNFs using pre-training of a stacked with 6 denoising auto-encoders. We decided to use MFCC acoustic features and pre-training for further experiments.

Table 2 lists the recognition performance in terms of WER for the Vietnamese system with different type of acoustic models namely baseline HMM/GMM, HMM/GMM using DBNFs (DBNFsHMM/GMM), hybrid HMM/DNN (HMM/DNN) and hybrid HMM/DNN with DNN was trained using the combination of Deep neural network architecture as in section 4.2.(HMM/DBNF+DNN). Regarding the performance of the baseline system, the MFCC number is 21.25% WER and combination of MFCC and tonal feature (MFCC+Pitch) results in systems which gives much gain in term of WER (about 5% absolute).

Table 2: Recognition performance for the Vietnamese system with different type of acoustic models in terms of WER

Acoustic Model	**Features**	**Layer size**	**WER (%)**
Baseline HMM/GMM	MFCC	-	21.25
DBNFs-HMM/GMM		1000	13.39
HMM/DNN		1000	13.20
		2000	13.03
Baseline HMM/GMM	MFCC+ Pitch	-	16.77
HMM/DNN		1000	10.96
		2000	10.71
DBNFs-HMM/GMM		1000	10.58
HMM/DBNF+DNN		1000	10.43
		2000	**10.27**

The hybrid HMM/DNN system outperforms baseline HMM/GMM system, resulting in relative improvements of up to 37.8% (13.20% WER) over the MFCC baseline. We further examined how the size of the hidden layers (excluding the fixed-size bottleneck layer and the classification layer) impacts the final recognition performance. It can be seen that increasing the layer size of neural network to 2000 is slightly better than the one with 1000 hidden units. Specially, when adding tonal feature as input of neural network give relatively improvement up to 49.6% (10.71% WER) over MFCC baseline and 17.8% over HMM/DNN without tonal feature in neural network input. Regarding the performance of DBNFs-HMM/GMM system, it is slightly worse than HMM/DNN system's when use MFCC feature but it is slightly better with MFCC+Pitch feature.

7. CONCLUSION

In this work, we have demonstrated that bottleneck feature are useful input feature for hybrid HMM/DNN aproach. It could be shown that proposed architecture for acoustic modeling obtained relative improvements over the best HMM/DNN system by 4.1%. The systems were tuned on a small-sized VOV speech corpus, which increased the relative improvement in word error rate over the MFCC baseline to 51.4%.

We showed the way to extract tonal feature using modified of the Getf0 algorithm [13] which could increase the recognition performance by 20% relative on baseline system and by 17.8% on DNN/HMM systems. We have also evaluated the layer size in the DNN architectures for acoustic model. The gains were achieved by increasing the layer size to 2000 hidden units.

In the future we intend to build a strong language model using additional training text and deep neural network as in [25] as well as improve acoustic model for the systems using multi-lingual network training approaches [20].

REFERENCES

1. G. Dahl, D. Yu, L. Deng, and A. Acero, "**Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition,"** *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 30-42, jan. 2012.
2. F. Seide, G. Li, and D. Yu, "**Conversational speech transcription using context-dependent deep neural networks,"** *in Interspeech 2011. International Speech Communication Association*, August 2011.
3. H. A. Bourlard and N. Morgan, "**Connectionist Speech Recognition: A Hybrid Approach,"** *Norwell, MA, USA: Kluwer Academic Publishers*, 1993.
4. F. Grezl, M. Kara at, S. Kontair, and J. Cernocky, "**Probabilistic and bottle-neck features for lvcsr of meetings,"** *in Acoustics, Speech and Signal Processing (ICASSP)*, 2007 IEEE International Conference on. IEEE, 2007, pp. V-757- IV-760.
5. Q. B. Nguyen, T. T. Vu, and C. M. Luong, "**Improving bottleneck features for vietnamese large vocabulary continuous speech recognition system using deep neural network,"** *National Journal of Computer Science and Cybernetics*, 2015.
6. T. T. Vu, D. T. Nguyen, M. C. Luong, and J.-P. Hosom, "**Vietnamese large vocabulary continuous speech recognition,"** *in INTERSPEECH*, 2005.
7. N. H. Quang, P. Nocera, E. Castelli, and T. V. Loan, **"A novel approach in continuous speech recognition for vietnamese,"** in SLTU, 2008.
8. N. T. Vu and T. Schultz, **"Vietnamese large vocabulary continuous speech recognition,"** *in Proc. Automatic Speech Recognition and Understanding (ASRU).* Merano, Italy: IEEE, Dec. 2009.
9. D. Talkin, **"A robust algorithm for pitch tracking (RAPT),"** *in Speech Coding and Synthesis*, W. B. Klein and K. K. Palival, Eds. Elsevier, 1995.
10. L. R. Rabiner, **"A tutorial on hidden markov models and selected applications in speech recognition,"** *in PROCEEDINGS OF THE IEEE*, 1989, pp. 257-286.
11. A. de Cheveigné and H. Kawahara, **"Yin, a fundamental frequency estimator for speech and music,"** *J Acoust Soc Am*, vol. 111, pp. 1917-1930, 2002.
12. B. S. Lee and D. P. W. Ellis, **"Noise robust pitch tracking by subband autocorrelation classification."** *in INTERSPEECH. ISCA*, 2012, pp. 707-710.
13. P. Ghahremani, B. BabaAli, D. Povey, K. Riedhammer, J. Trmal, and S. Khudanpur, **"A pitch extraction algorithm tuned for automatic speech recognition,"** *in Acoustics, Speech and Signal Processing (ICASSP)*, 2014 IEEE International Conference on. IEEE Signal Processing Society, May 2014.
14. D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, **"The kaldi speech recognition toolkit,"** *in IEEE 2011 Workshop on Automatic Speech Recognition and Understanding. IEEE Signal Processing Society*, Dec. 2011, iEEE Catalog No.: CFP11SRW-USB.
15. F. Plante, G. F. Meyer, and W. A. Ainsworth, **"A pitch extraction reference database."** *in EUROSPEECH. ISCA*, 1995.
16. J. Gehring, Y. Miao, F. Metze, and A. Waibel, **"Extracting deep bottleneck features using stacked auto-encoders,"** *in ICASSP2013*, Vancouver, CA, 2013, pp. 3377-3381.
17. X. Glorot, A. Bordes, and Y. Bengio, **"Domain adaptation for large-scale sentiment classification:A deep learning approach,"** in Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011, pp. 513-520.

18. P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, **"Extracting and composing robust features with denoising autoencoders,"** *in ICML08*, 2008, pp. 1096-1103.
19. A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, **"Phoneme recognition using time-delay neural networks,"** Acoustics, Speech and Signal Processing, IEEE Transactions on, vol. 37, no. 3, pp. 328-339, 1989.
20. Q. B. Nguyen, J. Gehring, M. Muller, S. Stuker, and A. Waibel, **"Multilingual shifting deep bottleneck features for low-resource asr,"** *in Acoustics, Speech and Signal Processing (ICASSP)*, 2014 IEEE International Conference on, May 2014, pp. 5607-5611.
21. M. Gales, **"Maximum likelihood linear transformations for hmm-based speech recognition,"** *Computer Speech and Language*, vol. 12, no. 2, pp. 75 - 98, 1998.
22. S. P. Rath, D. Povey, K. Vesely, and J. Cernocky, **"Improved feature processing for deep neural networks,"** *in INTERSPEECH*. ISCA, 2013, pp. 109-113.
23. J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, **"Theano: A cpu and gpu math compiler in python,"** *in Proceedings of the 9th Python in Science Conference*, S. van der Walt and J. Millman, Eds., 2010,pp. 3 - 10.
24. V. H. Nguyen, C. M. Luong, and T. T. Vu, **"Applying bottle neck feature for vietnamese speech recognition,"** *National Journal of Computer Science and Cybernetics*, pp. 379-388, 2013
25. N. Q. Pham, H. S. Le, T. T. Vu, and C. M. Luong, **"The speech recognition and machine translation system of ioit for iwslt 2013,"** in *Proceedings of the International Workshop for Spoken Language Translation (IWSLT)*, 2013.