*Journal of Computer Science and Cybernetics, V.31, N.4 (2015), 267–276*
DOI: 10.15625/1813-9663/31/4/5944

# IMPROVING BOTTLENECK FEATURES FOR VIETNAMESE LARGE VOCABULARY CONTINUOUS SPEECH RECOGNITION SYSTEM USING DEEP NEURAL NETWORKS

QUOC BAO NGUYEN[1], TAT THANG VU[2], AND CHI MAI LUONG[2]

[1] *University of Information and Communication Technology, Thai Nguyen University; nqbao@ictu.edu.vn*

[2] *Institute of Information Technology, Vietnam Academy of Science and Technology; vtthang@ioit.ac.vn, lcmai@ioit.ac.vn*

**Abstract.** In this paper, the pre-training method based on denoising auto-encoder is investigated and proved to be good models for initializing bottleneck networks of Vietnamese speech recognition system that result in better recognition performance compared to base bottleneck features reported previously. The experiments are carried out on the dataset containing speeches on Voice of Vietnam channel (VOV). The results show that the DBNF extraction for Vietnamese speech recognition decreases relative word error rate by 14% and 39% compared to the base bottleneck features and MFCC baseline, respectively.

**Keywords.** Deep bottleneck features, neural network, Vietnamese speech recognition.

## 1. INTRODUCTION

In automatic speech recognition systems, features extraction task is an important part of achieving a good recognition performance. Previous works [1,2] have shown that artificial neural networks can be used to extract good, discriminative features that yield better recognition performance than standard feature extraction algorithms like Mel Frequency Cepstral Coefficient (MFCC) and Perceptual Linear Prediction (PLP). One possible approach for this is to train a network with a small bottleneck layer, and then use the activations of the units in this layer to produce feature vectors ("bottleneck features", BNF [1]) for the remaining parts of the system.

Recently, deep learning has gained a lot of attention in the machine learning community. The general objective of this field is the training of large neural networks with many hidden layers, so-called deep neural networks (DNNs). While most frequently used in computer vision, multiple recent works [3–6] have demonstrated the ability of deep networks to achieve superior performance on speech recognition tasks as well.

In this study, deep neural networks are also applied to improve bottleneck features for Vietnamese speech recognition which were reported previously [7]. We show that a pre-training algorithm produce better than neural network models using standard methods, and that deeper neural networks result in better performance of the resulting Vietnamese speech recognition system. We also compare and combine DBNFs with other state-of-the-art techniques in order to determine the important of DBNFs in Vietnamese recognition.

## 2. DEEP LEARNING

In recent years, deep learning has gained a lot of attention in the machine learning community. The general objective of this field is the training of large neural networks with many hidden layers, so-called deep neural networks (DNNs). There are multiple reasons why deep learning is attractive. From a theoretical point of view, they are more efficient than shallow ones in the sense that they are able to represent complex functions with exponentially fewer computational elements [8]. In theory, this makes them more suitable for high-dimensional classifications problems with complicated decision.

Another motivation for deep architectures is the automatic discovery of feature hierarchies where high-level features are composed of low-level features. For example, in image processing task a feature representing a face might be composed of features for eyes, a nose and more, which are represented as combinations of simple edge detectors.

There are some deep learning algorithms that are also unsupervised in that they do not use labels when training the network. While accurately labeling training data takes a time and is labor-intense task, this is a very attractive property. Especially in speech recognition, where recordings first have to be transcribed by humans, systems can benefit from leveraging unlabeled speech data.

### 2.1. Denoising Auto-encoders

Auto-encoders are a special kind of ANNs that are not trained for classification task but are trained to reconstruct the network input as well as possible after they have been transformed by a hidden layer. The hidden layer of a trained auto-encoder provides an hidden representation (encoding) of the input data. If the number of units in the hidden layer is smaller than the number of input features, it is forced to learn a compact and invertible encoding of its input [9, 10]. This corresponds to a non-linear dimensionality reduction.

The Figure 1 shows the architecture of an auto-encoder with a small number of hidden units. The model is consisting of an input layer, a single hidden layer and an output layer with the same dimensionality as the input layer. Again, the backpropagation algorithm can be used to train the auto-encoder, using the values of the input vector as targets for the output layer.

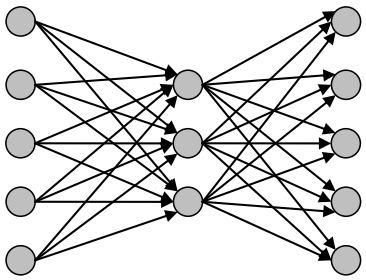

Figure 1: Auto-encoder architecture.

Auto-encoder architecture can be used for building deep neural networks, which was explored by Bengio et al. as an alternative to restricted Boltzmann machines. The main idea of this approach is training of each additional hidden layer as an auto-encoder for the hidden representation of the

previous layer. The resulting stack of auto-encoders can be transformed into a standard, feed-forward neural network.

Denoising Auto-encoders (DAE) is an alternative approach to extend classic auto-encoders for deep learning purposes proposed by Vincent et al. [11] the input data is first corrupted by applying random noise to the individual features. Afterwards, the model is trained to reconstruct the uncorrupted input from the corrupted input in an auto-encoder-like fashion. In their work, Vincent et al. showed that hidden representations learned from randomly corrupted input differ from the results achieved with standard sparsity constraints and may provide more useful features when adding further layers.

The individual steps of the computation are performed by a denoising auto-encoder which are illustrated in Figure 2. The main difference to normal auto-encoders is the corruption of an input vector $x$. This can be formalized as the application of a stochastic process $q_D$

$$\tilde{x} \sim q_D(\tilde{x}|x) \tag{1}$$

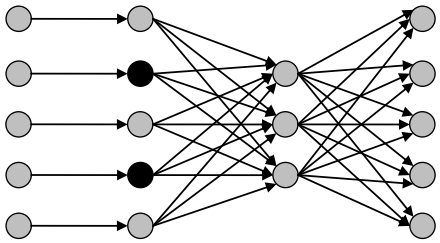

Figure 2: Architecture of a denoising auto-encoder with masking noise. The black color indicates that units are set to zero (masked).

This process is performed by adding random noise to individual training examples. Vincent et al. proposed multiple possible noise models [12]. The most common one consists of randomly setting a fraction of the elements of the input vector to zero. Another one is Gaussian noise which replaces every element of a random sample drawn from a normal distribution with mean and a fixed variance. The last one is salt-and-pepper noise which sets random elements of to their minimum or maximum value. While some types of noise can be regards as more natural choices for a given task, Vincent et al. demonstrated that all types result in learning useful hidden representations.

In this work, the masking noise is also applied to the data by setting every element of the input vector to zero with a fixed probability. Then the corrupted input $\tilde{x}$ 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 $\sigma_y$ as follows:

$$y = \sigma_y(W\tilde{x} + b) \tag{2}$$

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 $\sigma_z$:

$$z = \sigma_z(W^T y + c) \tag{3}$$

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 (4) in order to obtain the gradients necessary for adjusting the network weights.

$$L_H(x, z) = \sum_i x_i \log z_i + (1 - x_i) \log z_i. \tag{4}$$

However, 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.

### 2.2. Building Deep Neural Networks from Denoising Auto-encoders

Denoising auto-encoders can be stacked into deep learning architectures [12] like standard auto-encoders or restricted Boltzmann machines. This process starts with training a single DAE that is trained to reconstruct corrupted versions of the input data. Afterwards, another DAE is trained on the hidden representation $y$ of the first model, leaving the weights of the first model fixed as describe in Figure 3 . This scheme can be continued for the desired number of layers. Each time, auto-encoders computing the input representation for the model that is being trained to perform their computation on uncorrupted input.

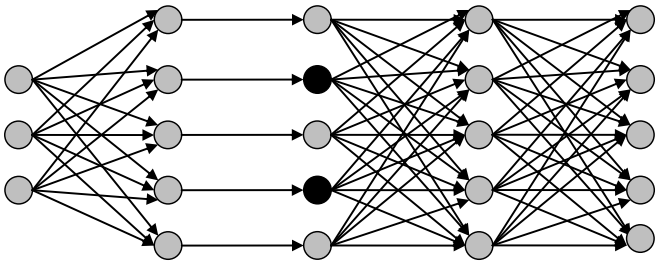

Figure 3: Training a stack of denoising auto-encoders. The hidden representation $y$ of the first DAE is now used as the input $x$ of the second DAE, which is being corrupted, encoded to $y$ and reconstructed.

When each layer has been pre-trained, the hidden representation of the whole system can be transformed into a deep neural network. This involves replacing the decoding part of the top-most DAE with a neural network output layer, and using the encoder parts of the remaining auto-encoders to initialize the hidden layers. The resulting network now can be treated like a multi-layer perceptron and fine-tuned with standard backpropagation.

## 3. BOTTLENECK FEATURES USING DEEP NEURAL NETWORKS

This section is mentioned as in our previous works [13, 14], the researchers briefly describe the deep neural network architecture for bottleneck feature extraction proposed in [4] and depicted in Figure 4. The network consists of a variable number of moderately large, fully connected hidden layers and a small bottleneck layer which is followed by an additional hidden layer and the final classification layer. The architecture differs from setups previously described, where the bottleneck layer has been

placed in the middle of a deep network [6], [15] or added as a second model trained on the output values of the original network [16].

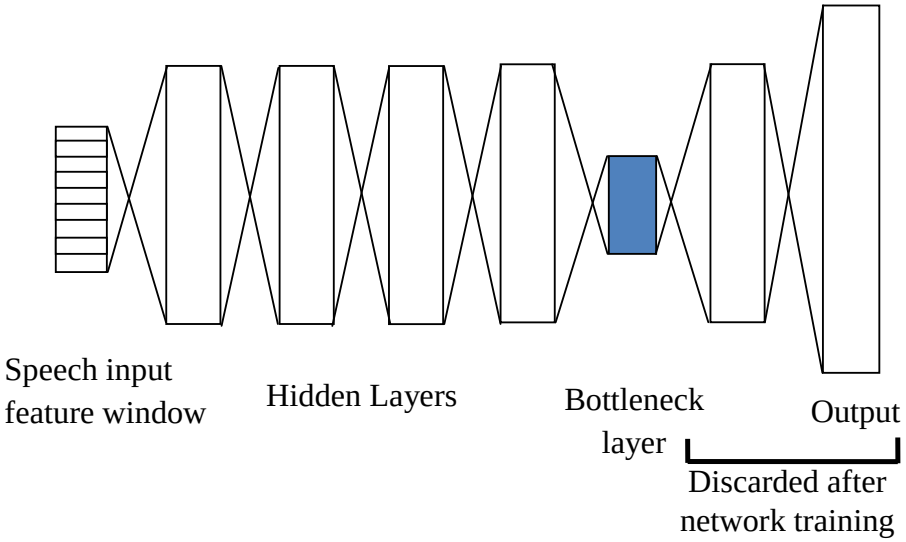


Figure 4: Deep Bottleneck Features Architecture.

### 3.1. Neural Network Input

The MFCCs were used as input of deep neural network, which contain 39 coefficients including 12 cepstral coefficients, 1 energy coefficient added with delta and double-delta features were extracted after windowing with the window size of 25 milliseconds and frame shift of 10 milliseconds. Then they were pre-processed using the splicing speaker-adapted features approach as in [17] with 40 dimensions. This features for each frame were stacked with 9 adjacent samples, resulting in a total of 360 dimensions.

### 3.2. Layer-wise Pre-training

The hidden layers in front of the bottleneck 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. [4] demonstrated that denoising auto-encoders [11] which are straight-forward models that have been successfully used for pre-training neural architectures for computer vision and sentiment classification [18] are applicable to speech data as well.

The researchers follow their training scheme and initialize the hidden layers as denoising autoencoders, 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 [11].

### 3.3. Adding the Bottleneck and Fine-tuning

After a stack of auto-encoders has been pre-trained in this fashion, a deep neural network can be constructed. The bottleneck layer, an additional hidden layer and the classification layer are initialized with random weights and connected to the hidden representation of the top-most auto-encoder. While all output hidden units use sigmoid active function, the classification layer output is obtained with the softmax activation function. The resulting network is then trained with supervision to estimate either context-independent or context-dependent HMM tri-phone states. For this last training step, errors are obtained with the cross-entropy function. Finally, the last two layers of the network can be discarded and the units in the bottleneck layer provide the final features used for training standard Gaussian mixture (GMM) acoustic models.

## 4. EXPERIMENTAL SETUP

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.

### 4.1. Baseline Systems

Baseline HMM/GMM systems were performed with the Kaldi developed at Johns Hopkins University [19]. The researchers 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) [20] 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.

### 4.2. Network training

The details of network training was mentioned in our previous works [3, 14]. The network input for these features was pre-processed using the splicing speaker-adapted features approach as in [17].

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 3. 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, a larger batch size of 256 was used. The learning rate was decided by the "newbob" schedule: for the first epoch, 0.008 was used 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 [21]. Training on 17 hours of VOV data took about a 18 hours for the architecture of network, around 9 million parameters were used.

## 5. EXPERIMENTAL RESULTS

In first experiments, different network architectures and inputs were compared. For the input data, it was found that extracting deep bottleneck features from MFCC instead of PLP data resulted in consistently better recognition performance of about 0.4% WER absolute. Then it was decided to use MFCC features for further experiments.

Third column of Table 1 lists the WER of BNF systems which were reported previously in [7] and trained on Multilayer Perceptron (MLP) network using 3 layers (1000 units each) without using any pre-training technique. The WER of BNF system using MFCC as input feature of neural network is 15.5% and 14.7% with PLP feature. While in this experiments, the DBNF proposed for Vietnamese speech recognition is described in section 3.. As it is seen that DBNF number is about 2% absolute (about 12% relative) better than BNF number.

| 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 |

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

### 5.1. Importance of Pre-training

The experiments were evaluated to extract bottleneck features from the networks of different depth and without unsupervised pre-training in order to check the importance of pre-training the stack of denoising auto-encoder in front of the bottleneck layer.

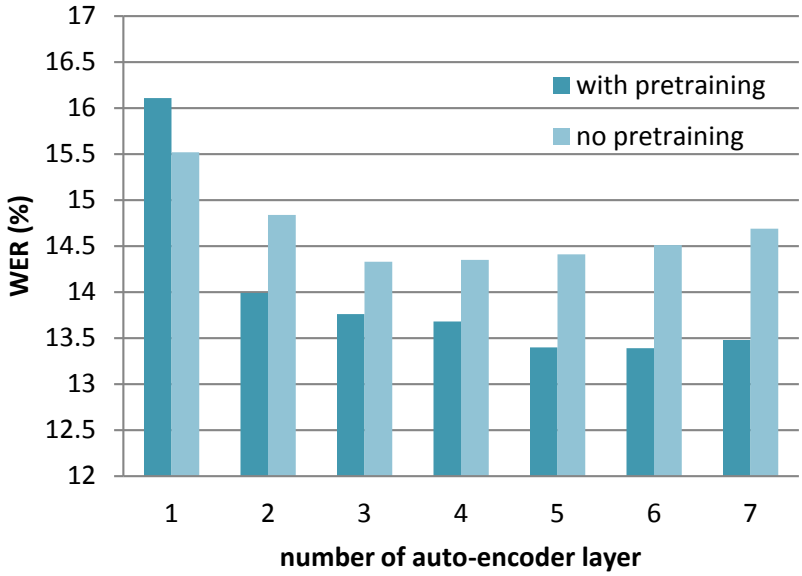


Figure 5: Comparison of recognition performance for pretrained and purely supervised trained neural networks.

Table 1 and Figure 5 list experiments to determine that if pre-training is applied, recognition performance improves when adding additional auto-encoders. If the neural network just trained supervised starting from the random initialization, recognition performance went down with deeper models.

### 5.2. Compare and Combine with state-ofthe-art Techniques

Since DBNFs was demonstrated in achieving good Vietnamese recognition performance, some experiments were evaluated using the other state-of-the-art techniques as follows: Speak Adaptive Training (SAT) [22], discriminative training using maximum mutual information (MMI) objective function [23] and subspace gaussian mixture models (SGMM) [24] in order to compare the recognition performance with DBNFs and determine the important of DBNFs in Vietnamese recognition. According to the results listed in Table 2, the state-of-the-art techniques still can be applied after DBNFs system and the best full-range of techniques DBNFs number is 12% relative better than the best full-range of techniques applied at baseline number (12.88% vs 14.68%).

| System | MLLT | SAT | MMI | SAT+MMI | SGMM |
|---|---|---|---|---|---|
| Baseline | 21.25 | 17.08 | 17.66 | 16.89 | **14.68** |
| DBNF | 13.39 | 12.90 | 13.19 | **12.88** | 13.04 |

Table 2: Combine DBNFs and Baseline systems with Different State-of-the-art Techniques.

## 6. CONCLUSIONS

In this work, the bottleneck features of Vietnamese speech recognition system improved by using deep neural network and DBNFs have shown the ability to achieve significant improvements from a 27% relative word error rate reduction reported previously to 39%, compared to MFCC baseline system. It is shown that denoising auto-encoders proved to be good models for initializing bottleneck networks of Vietnamese speech recognition system and the others state-ofthe-art techniques still can be applied after DBNFs system. The experiment setups used in this paper do not employ tonal feature as input feature of neural network and just use a weak language model. Therefore, in the future the researchers intend to investigate the tonal feature extraction to be used as apart of neural network input as well as build a stronger language model using deep neural network as in [25].

## REFERENCES

[1] F. Grezl, M. Karafiat, 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.

[2] K. Kilgour, I. Tseyzer, Q. B. Nguyen, and A. Waibel, "Warped minimum variance distortionless response based bottle neck features for lvcsr." in *ICASSP*, 2013, pp. 6990–6994.

[3] Q. B. Nguyen, J. Gehring, K. Kilgour, and A. Waibel, "Optimizing deep bottleneck feature extraction," in *Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF), 2013 IEEE RIVF International Conference on*, Nov 2013, pp. 152–156.

[4] 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.

[5] 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.

[6] D. Yu and M. L. Seltzer, "Improved bottleneck features using pretrained deep neural networks," in *INTERSPEECH*, 2011, pp. 237–240.

[7] V. H. Nguyen, C. M. Luong, and T. T. Vu, "Applying bottle neck feature for vietnamese speech recognition," pp. 379–388, 2013.

[8] Y. Bengio, "Learning deep architectures for ai," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, Jan. 2009. [Online]. Available: http://dx.doi.org/10.1561/2200000006

[9] H. Ackley, E. Hinton, and J. Sejnowski, "A learning algorithm for boltzmann machines," *Cognitive Science*, pp. 147–169, 1985.

[10] G. E. Hinton, "Connectionist learning procedures," *Artif. Intell.*, vol. 40, no. 1-3, pp. 185–234, 1989.

[11] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *ICML08*, 2008, pp. 1096–1103.

[12] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, Dec. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1756006.1953039

[13] Q. B. Nguyen, T. T. Vu, and C. M. Luong, "Improving acoustic model for english asr system using deep neural network," in *Computing & Communication Technologies-Research, Innovation, and Vision for the Future (RIVF), 2015 IEEE RIVF International Conference on.* IEEE, 2015, pp. 25–29.

[14] ——, "Improving acoustic model for vietnamese large vocabulary continuous speech recognition system using deep bottleneck features," in *Knowledge and Systems Engineering.* Springer, 2015, pp. 49–60.

[15] Z. Tüske, R. Schlüter, and H. Ney, "Deep hierarchical bottleneck mrasta features for lvcsr," in *ICASSP*, 2013, pp. 6970–6974.

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

[17] S. P. Rath, D. Povey, K. Vesely, and J. Cernocky, "Improved feature processing for deep neural networks." in *INTERSPEECH.* ISCA, 2013, pp. 109–113.

[18] 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.

[19] 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.

[20] M. Gales, "Maximum likelihood linear transformations for hmm-based speech recognition," *Computer Speech and Language*, vol. 12, no. 2, pp. 75 – 98, 1998.

[21] 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.

[22] T. Anastasakos, J. W. McDonough, R. M. Schwartz, and J. Makhoul, "A compact model for speaker-adaptive training." in *ICSLP.* ISCA, 1996.

[23] D. Povey, "Discriminative training for large vocabulary speech recognition," Ph.D. dissertation, Ph. D. thesis, Cambridge University, 2004.

[24] D. Povey, L. Burget, M. Agarwal, P. Akyazi, F. Kai, A. Ghoshal, O. Glembek, N. Goel, M. Karafiát, A. Rastrow *et al.*, "The subspace gaussian mixture modela structured model for speech recognition," *Computer Speech & Language*, vol. 25, no. 2, pp. 404–439, 2011.

[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.

*Received on March 08 - 2015*
*Revised on October 19 - 2015*