Improving Language-Universal Feature Extraction with Deep Maxout and Convolutional Neural Networks

Yajie Miao, Florian Metze

Language Technologies Institute, School of Computer Science, Carnegie Mellon University
{ymiao,fmetze}@cs.cmu.edu

Abstract

When deployed in automated speech recognition (ASR), deep neural networks (DNNs) can be treated as a complex feature extractor plus a simple linear classifier. Previous work has investigated the utility of multilingual DNNs acting as language-universal feature extractors (LUFEs). In this paper, we explore different strategies to further improve LUFEs. First, we replace the standard sigmoid nonlinearity with the recently proposed maxout units. The resulting maxout LUFEs have the nice property of generating sparse feature representations. Second, the convolutional neural network (CNN) architecture is applied to obtain more invariant feature space. We evaluate the performance of LUFEs on a cross-language ASR task. Each of the proposed techniques results in word error rate reduction compared with the existing DNN-based LUFEs. Combining the two methods together brings additional improvement on the target language.

Index Terms: language-universal feature extraction, deep maxout networks, deep convolutional networks

1. Introduction

On large vocabulary continuous speech recognition (LVCSR), deep neural networks (DNNs) have shown significant gains over the traditional Gaussian mixture model/hidden Markov model (GMM/HMM) systems [1, 2]. With multiple hidden layers, a DNN acoustic model is trained to estimate the posterior probabilities of HMM context-dependent states. The hidden layers of DNNs can be treated as a series of nonlinear transformations that convert the original input features to high-level representations. The final softmax layer is added to classify speech frames into HMM states. It is revealed in [3] that the effectiveness of DNNs comes largely from the invariance of the representations to variability such as speakers, environments and channels.

Following this feature learning formulation, [4] trains DNNs over multiple languages, with the hidden layers shared across languages. These shared layers are taken as a language-universal feature extractor (LUFE) [4]. Given a new language, acoustic models can be built over the outputs from the LUFE, instead of the raw features (e.g., MFCCs). Porting LUFEs to a new language provides an elegant way for cross-language knowledge transfer. This approach has shown nice improvement in the literature [4, 5], especially when the new language has limited transcribed speech (e.g., only several hours). The goal of this paper is to investigate various strategies to improve LUFE-based feature extraction and thus boost the ASR performance on the new language.

First, sparsity is introduced into the learned deep feature representations. Sparse feature learning [6, 7, 8] is an active research topic in the machine learning area. On the complex image and speech signals, sparse features (e.g., sparse coding [9]) tend to give better classification accuracy compared with non-sparse feature types [9, 10, 11, 12]. We implement sparse LUFEs by taking advantage of the deep maxout networks (DMNs) [13, 5, 14, 15]. In the DMN, units at each hidden layer are divided into groups, and each group generates a single output with max-pooling. When the DMN is applied, the *DMN-LUFE* feature extractor can generate representations with truly-zero sparsity (many of the features become zeros). Second, we propose to train LUFEs based on the convolutional neural network (CNN) architecture. The advantage of CNNs has been confirmed experimentally, with CNNs outperforming the state-of-the-art DNNs on both phone recognition [16] and LVCSR [17, 18, 19] tasks. Due to the local filters and max-pooling layers, CNNs are able to reduce spectral variation in the speech data. Therefore, more robust and invariant representations are expected to be obtained from the *CNN-LUFE* feature extractor.

Our experiments are on the BABEL multilingual corpus [5, 20, 21] where we establish a cross-language acoustic modeling task. LUFEs are trained over a group of source languages and evaluated by WERs on the target language. We compute the *population sparsity* [8] metric to quantitatively measure the sparsity level of various LUFEs. In comparison with the DNN-based LUFE, the DMN feature extractor corresponds to sparser features and better recognition results. Extensive experiments are conducted to determine CNN configurations (number of convolution layers, learning rate, etc) suitable for feature extraction. We observe that for the CNN-LUFE, optimal features are hidden outputs from the fully-connected layer which follows the convolution stages. Finally, our proposed techniques are combined together to give further gains on the target language. With the resulting *CNN-DMN-LUFE*, we are able to obtain a 3.7% absolute WER reduction with the DNN-based LUFE as the baseline.

2. Feature Extraction with DNNs

A multilingual DNN feature extractor can be learned over a group of languages as depicted in Figure 1(a). The hidden layers of the multilingual DNNs are tied across all the languages, while each language has its own softmax output layer to classify context-dependent states specific to that language. Fine-tuning of the DNNs can be carried out using stochastic gradient descent (SGD) based on mini-batches. Each epoch traverses data from all the languages instead of one single language. When picking up one mini-batch, the SGD estimator switches to the corresponding softmax layer. Parameters of the shared layers are updated with gradients gathered from all the source languages.

When the multilingual DNNs are trained, a consecutive subset of the shared layers act as the LUFE. On the target language, as shown in Figure 1 (b), a hybrid system is built using the features generated from this extractor. The DNN in this hybrid system is trained to estimate posterior probabilities of the target language senones. This cross-language acoustic

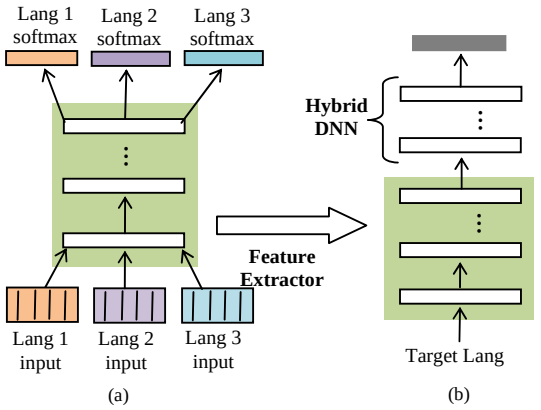

Figure 1. (a) *Learning a language-universal feature extractor (LUFE) over 3 languages*; (b) *applying this extractor to a target language for cross-language hybrid system building*.

modeling improves speech recognition on the target language, especially when the target language has highly limited transcribed speech [5, 22].

Our approach described above differs from the method proposed by [4] on two aspects. First, the extractor used in [4] consists of all the shared layers, while we only take a subset of the layers as the extractor. The intuition is that the higher layers (e.g., the last hidden layer) are closer to the language-specific softmax layers, which may introduce undesirable language bias to their activations. Second, on the target language, we are building a DNN model, instead of a single softmax layer, over the extracted features. In our experiments, these two modifications have resulted in notable improvement over the method in [4].

3. LUFEs with Convolutional Networks

Instead of using fully-connected parameter matrices, CNNs are characterized by parameter sharing and local feature filtering. The local filters help to capture locality along the frequency bands. On top of the convolution layer, a max-pooling layer is usually added to normalize spectral variations. As a result, the CNN hidden activations become invariant to different types of speech variability and provide better feature representations.

Figure 2 shows our CNN architecture which works slightly different from the existing proposals [16, 17, 18]. In the convolution layer, we only consider filters along frequency, assuming that the time variability can be modeled by HMM. Inputs into CNNs are N neighbouring frames of acoustic features (e.g., log filterbanks), where each frame $\mathbf{v}_i$ is a 1D feature map. The hidden outputs from this layer contain J vectors ($[\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3 \ldots]$). The trainable 1D filter $\mathbf{r}_{ji}$ connects input feature map $\mathbf{v}_i$ and output feature map $\mathbf{h}_j$, and is shared across the frequency axis along $\mathbf{v}_i$. Outputs from this convolution layer can be computed as

$$\mathbf{h}_j = \sigma\left(\sum_{i=1}^{N} \mathbf{r}_{ji} * \mathbf{v}_i + b_j\right) \quad (1)$$

where * represents the 1D discrete convolution operator, and b_j is the trainable bias attached to $\mathbf{h}_j$. In this paper, we use the logistic sigmoid activation function σ.

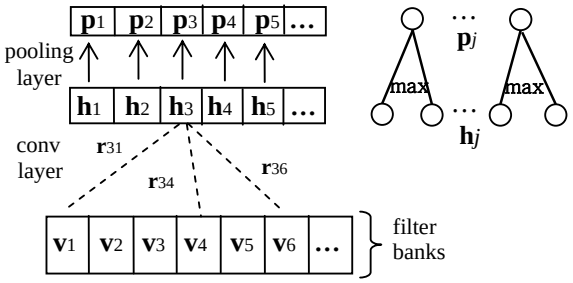

Figure 2. *One stage of CNNs consisting of the convolution and pooling layers. We are also showing the details regarding* 1D *max-pooling from* $\mathbf{h}_j$ *to* $\mathbf{p}_j$ *when the pooling size is 2.*

Then, a max-pooling layer is added on top of the convolution layer. Max-pooling is carried out in a vector-wise mode. More formally, for each vector $\mathbf{h}_j$, we divide its units into non-overlapping groups and output the maximum activation within each group. When the pooling size is k, the size of each after-pooling feature map $\mathbf{p}_j$ is $1/k$ of the size of the before-pooling $\mathbf{h}_j$. The convolution and pooling layers together are called a *convolution stage*. In our setups, CNNs stack two such stages where outputs from the lower pooling layer are propagated to the higher convolution layer. Multiple fully-connected DNN layers and finally the softmax layer are added over these two stages. From the feature learning perspective, the convolution and pooling layers in this structure are trained to extract invariant features, while the fully-connected layers use these high-level features to better classify HMM states.

4. Sparse Feature Extraction

Our past study [5] presents the deep maxout networks (DMNs) for speech recognition. An example of a maxout layer is shown in Figure 3(a) where the group size (the number of units in each group) is 3. The units at each hidden layer are partitioned into groups, and we impose max-pooling on each group. Compared with standard DNNs, DMNs perform better on both hybrid systems and bottleneck-feature (BNF) tandem systems [5]. In this paper, we focus on the application of DMNs as sparse feature extractors. Sparse representations can be generated from any of the maxout layers via a *non-maximum masking* operation, as exemplified by Figure 3(b). Specifically, given one input frame, all the units within each group have their individual outputs, instead of being pooled together into one output. However, only the maximum value in this group is retained, while the other outputs are rounded to 0. It's worth noting that non-maximum masking only happens

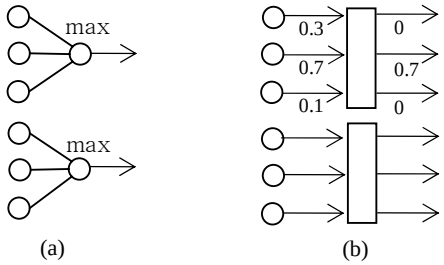

Figure 3. *An example for (a) maxout layer and (b) sparse feature generation with non-maximum masking.*

during the feature extraction stage. The training stage always applies max-pooling. We extend our previous idea from the following three aspects.

First, we compute *population sparsity* [8] for each feature type as a quantitative indicator. If the *m*-th frame has the feature vector $\mathbf{f}_m$, then population sparsity

$$pSparsity = \left|\left| \frac{\mathbf{f}_m}{||\mathbf{f}_m||_2} \right|\right|_1 \quad (2)$$

measures how sparse the features are on this example. We report the value of this metric as an average over the entire target-language training set. Unless otherwise stated, population sparsity is shortened as *pSparsity* throughout this paper. A lower pSparsity means higher sparsity of the features.

Second, to better understand the impact of sparsity, we compare DMNs against the rectifier networks [23]. DNNs consisting of rectifier units, referred to as *deep rectifier networks* (DRNs), have shown competitive accuracy on speech recognition [24, 25, 26]. The rectifier units have the activation function $\max(0, x)$. As a result, high-sparsity features can be naturally produced from a DRN-based extractor, because many of the hidden outputs are forced to 0.

Third, DMNs are combined with CNNs to obtain both sparse and invariant feature representations. The CNN-DMN-LUFE extractor is structured in the similar way as CNN-LUFE (Section 3). The only difference is that the fully-connected layers in CNNs are replaced by maxout layers. In Section 5.5, we will experimentally prove that this combined feature extractor ends up to be the best LUFE in this work.

5. Experiments

5.1. Experimental Setup

On the BABEL corpus, we evaluate feature extractors in the context of cross-language hybrid system building. Our experiments follow the setup in [5], using languages from the IARPA BABEL research program. We aim at improving speech recognition on the Tagalog (TG, IARPA-babel106-v0.2f) limited language pack (LimitedLP). This is a low-resource condition because only 10 hours of telephone conversation speech are available for system building. Moreover, the data collection covers a variety of acoustic conditions, speaking styles and dialects. A large portion of the audio data are either non-speech events (e.g., breath, laugh, cough) or non-lexical speech (e.g., hesitations and fragments). All these factors make speech recognition under this condition a very difficult task. Therefore, we get higher WERs [5, 20, 21] than on other benchmark datasets such as Switchboard.

On the target language Tagalog LimitedLP, WERs are reported on a 2-hour testing set. The training and testing sets have no over-lapping speakers. During decoding, we use a trigram language model built from training transcriptions. The source languages, on which feature extractors are trained, include the LimitedLP sets of Cantonese (CN, IARPA-babel101-v0.4c), Turkish (TU, IARPA-babel105b-v0.4) and Pashto (PS, IARPA-babel104b-v0.4aY). LimitedLP sets of the four languages have statistics summarized in Table 1.

On each language, we build the GMM/HMM system with the same recipe. An initial maximum likelihood model is first trained using 39-dimensional PLPs (plus deltas and double deltas) with per-speaker mean normalization. Then 9 frames are spliced together and projected down to 40 dimensions with linear discriminant analysis (LDA). A maximum likelihood linear transform (MLLT) is applied on the LDA features and generates the LDA+MLLT model. Speaker adaptive training (SAT) is performed using feature-space maximum likelihood linear regression (fMLLR). The number of triphone states for each language is shown in Table 1.

Table 1. *Statistics of the datasets used in the experiments. The last row shows the number of classes on each language.*

Statistics	Target	Source		
	TG	CN	TU	PS
#training speakers	132	120	121	121
training (hours)	10.7	17.8	9.8	9.8
dictionary size	8k	7k	12k	8k
# of classes	1920	1867	1854	1985

5.2. Monolingual DNNs and CNNs

We experiment with DNN and CNN acoustic models on the Tagalog LimitedLP training set, without applying any LUFEs. Inputs of DNNs and CNNs include 11 consecutive frames of 30-dimensional log-scale filterbanks with per-speaker mean and variance normalization. During network fine-tuning, we start from a learning rate of 0.08 which keeps unchanged for 15 epochs. Then the learning rate is halved at each epoch until the cross-validation accuracy on a held-out set stops to improve. A momentum of 0.5 is used for fast convergence and the mini-batch size is 256.

The DNN model has 6 hidden layers and 1024 units at each layer. DNN parameters are initialized with stacked denoising autoencoders (SDAs) [27] using masking noise and the denoising factor of 0.2. Each layer in the DNN corresponds to a denoising autoencoder (DA) which minimizes the difference between the reconstruction of corrupted input and the clean version of it. Pre-training of each layer has the learning rate of 0.01 and runs for 10 epochs. Previous work [20, 21] has applied SDAs on LVCSR and shown that SDAs perform comparably as restricted Boltzmann machines (RBMs) [28] for DNN pre-training.

For CNNs, the size of the filter vectors $\mathbf{r}_{ji}$ is constantly set to 5. We use a pooling size of 2, meaning that the pooling layer shrinks convolution outputs by half. After tuning the CNN architecture, we observe that the best setting has two convolution stages and three fully-connected layers. The first and second convolution layers contain 100 and 200 feature maps respectively. Continuing to augment the convolution and fully-connected layers brings no further gains. Table 2 shows that the CNN achieves 2.6% absolute WER improvement (68.2% vs. 70.8%) over the DNN model, which demonstrates the advantage of CNNs for acoustic modeling. In this paper, no pre-training is performed for CNNs. We leave SDAs

Table 2. *WERs(%) of DNN and CNN models on the target language. The last three rows are cross-language DNN models based on LUFEs. "Feature Dim" means the dimension of the features from each LUFE.*

Models	Feature Dim	WER%
Monolingual DNN	---	70.8
Monolingual CNN	---	68.2
DNN-LUFE	1024	69.6
CNN-LUFE - FeatType 1	1000	67.8
CNN-LUFE - FeatType 2	1024	**67.1**

initialization of CNN parameters to our future work.

5.3. Results of DNN- and CNN-LUFEs

LUFEs are trained over the three source languages. When using DNNs, we share their hidden layers across the languages. After DNNs are trained, the *lower four layers* are employed as the feature extractor, which gives better results than generating features from the last hidden layer [4]. Multilingual training of CNNs also exploits the parameter sharing idea. The convolution and fully-connected layers are shared over the source languages. Then we can take the *two convolution stages* as the LUFE, and features are generated from the second max-pooling layer. Alternatively, we can also extract features from the lowest fully-connected layer which is on top of the convolution stages. These two manners of CNN feature extraction are called *FeatType1* and *FeatType2*.

On Tagalog LimitedLP, DNN-based hybrid systems are built on these extracted feature representations. Table 2 (the last three rows) shows the results of the hybrid systems when various LUFEs are applied. For fair comparison, the identical DNN topology, i.e., 4 hidden layers each with 1024 units, is used for hybrid systems over different feature extractors. We can see that cross-language models based on LUFEs give consistently better results than the monolingual DNN. The FeatType2 CNN extractor outperforms the DNN extractor by 2.5% absolute WER (67.1% vs 69.6%). FeatType1 performs worse than FeatType2 partly because the layout of convolution outputs differs a lot from the DNN activations. This incompatibility may hurt the performance if we build DNN hybrid systems directly on top of convolution outputs.

5.4. Results of Sparse LUFEs

As with [5, 24], the dropout technique [29] is applied in both DMN and DRN raining, in order to prevent overfitting and improve model robustness. We impose dropout on each hidden layer by following the implementation described in [22]. The drop factor, which governs the binomial distribution for hidden activation masking, is constantly set to 0.2. Learning of multilingual DRNs and DMNs has the similar configuration as multilingual DNNs (Section 5.3). Due to the introduction of dropout, we use a larger learning rate for both DRNs and DMNs, by increasing the starting learning rate to 0.1. For DMNs, we keep the number of units at each hidden layer approximately to be 1024, and vary the combination of group number and group size.

Table 3 compares DRNs and DMNs based LUFEs in terms of pSparsity and target-language WERs. DRNs generate sparser features with lower pSparsity values. However, features produced by DMNs achieve better WERs than features from DRNs. Increasing the group size (from 2 to 3 and finally 4) in DMNs gives sparser features but degrades the recognition results. This reveals that although sparsity is a desirable property, over-sparsification may hurt speech recognition performance. The best sparse features in Table 3 are generated by DMNs with 512 unit groups and the group size of 2 at each hidden layer.

Table 3. *WERs(%) and pSparsity of DRNs and DMNs feature extractors. "DMN-LUFE, m×n" means that each layer of the DMN extractor has m unit groups each of which has n units.*

LUFE Models	WER%	pSparsity
DRN-LUFE	68.2	10.7
DMN-LUFE, 512×2	**67.5**	17.7
DMN-LUFE, 342×3	67.8	14.6
DMN-LUFE, 256×4	67.9	12.8

5.5. Combining CNNs and DMNs

Finally, we examine the effectiveness of combining CNNs and DMNs for better feature extraction. The structure of the convolution layers as defined in Section 5.2 remains the same. We replace the sigmoid fully-connected layers with maxout layers. During multilingual training, the convolution layers and maxout layers use the starting learning rates of 0.08 and 0.1 respectively. Features are generated from the lowest maxout layer on top of the convolution stages (i.e., FeatType2 defined in Table 2). We can see from Table 4 that compared with the CNN-LUFE, the CNN-DMN-LUFE generates sparse features as well as target-language WER reduction. This combined extractor is the best LUFE presented in this paper. It obtains 3.7% absolute WER improvement (65.9% vs 69.6%) over the baseline DNN-based LUFE.

Table 4. *Comparison of CNN-LUFE and CNN-DMN-LUFE in terms of pSparsity and target-language WERs(%).*

LUFE Models	WER%	pSparsity
CNN-LUFE FeatType2	67.1	20.4
CNN-DMN-LUFE FeatType2	**65.9**	16.6

6. Acknowledgments

This work was supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense U.S. Army Research Laboratory (DoD / ARL) contract number W911NF-12-C-0015. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government. This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number OCI-1053575.

7. Conclusions and Future Work

This paper has investigated the effectiveness of deep maxout and convolutional networks in performing language-universal feature extraction. Following experiments on the BABEL corpus, we are able to draw the following principal conclusions: 1) in comparison with DNNs, CNNs generate better feature representations and more WER improvement on cross-language hybrid systems; 2) maxout networks have the property of generating sparse hidden outputs, which makes the learned representations more interpretable and explanatory; 3) CNN-DMN-LUFE, a hybrid of CNNs and maxout networks, results in the best recognition performance on the target language. For our future work, we are interested to examine the effects of pre-training on CNNs as feature extractors. Also, we would like to extend various LUFE models to more languages and study how the LUFE approach performs under larger datasets.

8. References

[1] G. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large vocabulary speech recognition," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 20(1), pp. 30-42, 2012.

[2] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *Proc. ASRU*, pp. 24–29, 2011.

[3] D. Yu, M. L. Seltzer, J. Li, J. Huang, and F. Seide, "Feature learning in deep neural networks – studies on speech recognition tasks," in *International Conference on Learning Representations 2013*.

[4] J. Huang, J. Li, D. Yu, L. Deng, and Y. Gong, "Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers," in *Proc. ICASSP*, pp. 7304-7308, 2013.

[5] Y. Miao, F. Metze, and S. Rawat, "Deep maxout networks for low-resource speech recognition," in *Proc. ASRU*, 2013.

[6] H. Lee, C. Ekanadham, and A. Y. Ng, "Sparse deep belief net model for visual area V2," in *Proc. NIPS*, 2008.

[7] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proc. ICML*, pp. 609-616, 2009.

[8] J. Ngiam, P. Koh, Z. Chen, S. Bhaskar, and A. Y. Ng, "Sparse filtering," in *Proc. NIPS*, 2013.

[9] G. Sivaram, S.K. Nemala, M. Elhilali, T.D. Tran, and H. Hermansky, "Sparse coding for speech recognition," in *Proc. ICASSP*, pp. 4346–4349, 2010.

[10] H. Lee, Y. Largman, P. Pham, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Proc. NIPS*, 2009.

[11] G. Sivaram, and H. Hermansky, "Multilayer perceptron with sparse hidden outputs for phoneme recognition," in *Proc. ICASSP*, pp. 5336–5339, 2011.

[12] O. Vinyals, and L. Deng, "Are sparse representations rich enough for acoustic modeling?," in *Proc. Interspeech*, 2012.

[13] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," in *Proc. ICML*, 2013.

[14] M. Cai, Y. Shi, and J. Liu, "Deep maxout neural networks for speech recognition," in *Proc. ASRU*, 2013.

[15] X. Zhang, J. Trmal, D. Povey, and S. Khudanpur, "Improving deep neural network acoustic models using generalized maxout networks," in *Proc. ICASSP*, 2014.

[16] O. Abdel-Hamid, A. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," in *Proc. ICASSP*, pp. 4277-4280, 2012.

[17] T. N. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *Proc. ICASSP*, pp. 8614-8618, 2013.

[18] O. Abdel-Hamid, L. Deng, and D. Yu, "Exploring convolutional neural network structures and optimization techniques for speech recognition," in *Proc. Interspeech*, pp. 3366-3370, 2013.

[19] T. N. Sainath, B. Kingsbury, A. Mohamed, G. Dahl, G. Saon, H. Soltau, T. Beran, A. Aravkin, and B. Ramabhadran, "Improvements to deep convolutional neural networks for LVCSR," in *Proc. ASRU*, 2013.

[20] J. Gehring, W. Lee, K. Kilgour, I. Lane, Y. Miao, and A. Waibel, "Modular combination of deep neural networks for acoustic modeling," in *Proc. Interspeech*, pp. 94-98, 2013.

[21] J. Gehring, Y. Miao, F. Metze, and A. Waibel, "Extracting deep bottleneck features using stacked auto-encoders," in *Proc. ICASSP*, pp. 3377-3381, 2013.

[22] Y. Miao, and F. Metze, "Improving low-resource CD-DNN-HMM using dropout and multilingual DNN training," in *Proc. Interspeech*, pp. 2237-2241, 2013.

[23] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. AISTATS*, 2011.

[24] G. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *Proc. ICASSP*, pp. 8609-8613, 2013.

[25] L. Toth, "Phone recognition with deep sparse rectifier neural networks," in *Proc. ICASSP*, pp. 6985-6989, 2013.

[26] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *ICML Workshop on Deep Learning for Audio, Speech, and Language Processing* (WDLASL), 2013.

[27] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. Manzagol, "Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, 2010.

[28] G. E. Hinton, "A practical guide to training restricted Boltzmann machines," UTML TR., 2010.

[29] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv:1207.0580*, 2012.