A Multi-Region Deep Neural Network Model in Speech Recognition

Jia Cui, George Saon,Bhuvana Ramabhadran,Brian Kingsbury

IBM T. J. Watson Research Center, Yorktown Heights, NY, USA

{jiacui, bhuvana, gsaon, bedk}@us.ibm.com

Abstract

This work proposes a new architecture for deep neural network training. Instead of having one cascade of fully connected hidden layers between the input features and the target output, the new architecture organizes hidden layers into several *regions* with each region having its own target. Regions communicate with each other during the training process by connections among intermediate hidden layers to share learned internal representations from their respective targets. They do not have to share the same input features. This paper presents the performance of acoustic models built using this architecture with speaker independent and dependent features. Experimental results are compared with not only the baseline DNN model, but also the ensemble DNN, unfolded RNN and stacked DNN. Experiments on the IARPA sponsored Babel tasks demonstrate improvements ranging from 0.8% to 2.7% absolute reduction in WER.

Index Terms: speech recognition, deep neural network, multi-task training

10.21437/Interspeech.2015-653

1. Introduction

Deep Neural Network (DNN) has been shown to be very successful in speech recognition[1, 2, 3, 4, 5]. A conventional DNN model usually contains a cascade of hidden layers of processing units. Each layer uses the output of previous/lower layer as input. The assumption for that architecture is that higher layers represents more abstract concepts, which can be learned from less abstract concepts. In this paper, we propose a Multi-Region DNN (MR-DNN) architecture where hidden layers are organized into different regions. Each region is composed of a cascade of hidden layers and has its own target layer. Regions communicate with each other during both training and decoding through connections among intermediate layers.

There have been DNN variations which require modifications of the conventional feedforward cascade structure. Convolutional deep neural networks (CNN) [6, 7] include local or global pooling layers to combine the outputs of neuron clusters. Recurrent NN, including long-short term memory [8, 9, 10] models, allow connections from upper levels to lower levels to form a directed cycle. This enables the model to use internal memory to store and process arbitrary sequences of inputs. [11] proposes multi-column DNN for image classification where an arbitrary number of columns can be trained on inputs preprocessed in different ways. The final prediction is obtained by averaging individual predictions of all DNNs. [12] uses different columns of layers to model long temporal acoustic contexts, with the last hidden layer activations combined to jointly predict a single softmax output layer. Multi-task DNNs [13, 14, 15, 16] assign both the primary classification task and several auxiliary tasks to a shared DNN representation to improve the model's generalization performance. Joint training [17] takes both CNN and DNN input features and combines them with shared hidden layers, targeting to achieve system combination performance without extra training time.

MR-DNN integrates the techniques of both joint training and multi-tasking in terms of accepting multiple inputs features and modeling different targets simultaneously. Furthermore, MR-DNN encourages partitioning of hidden layers targeted to different targets. To be more specific, instead of having r big hidden layers with each one having $m * n$ hidden units, we can have m regions with each region containing r layers of n units. This partition can greatly decrease the number of parameters and save training time. Multi-column DNNs[11] [12] also enables multiple paths from input to output, but all columns are targeted to one output layer instead of different ones.In [16], each DNN predicts multiple target layers instead of one and there is no cooperation among different DNNs during training.

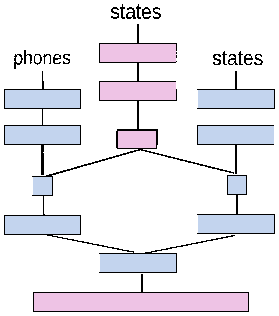
In the rest part of this paper, Section 2 describes in detail the MR-DNN architecture and several interesting variations. Section 3 presents experimental results on the languages used in the iARPA-sponsored BABEL program, namely, Tamil (Limited Language Pack) LLP, Telegu (Very Limited Language Pack) VLLP and Kurmanji VLLP. Each of these packs correspond to different amounts of training data. Section 4 makes conclusions and suggests possible future work for MR-DNN.

2. Multi-Region DNN

2.1. MR-DNN for Low-level Abstract Sharing in Multi-task Learning

The proposal of MR-DNN is triggered by investigations of multi-task learning for acoustic models. In many DNN acoustic models for speech recognition, the target layers are states from a baseline Hidden Markov Model (HMM). When multi-task learning is involved, a natural choice of auxiliary target is phones or phone sequences. After futile attempts to improve model performance by simply adding extra output layers to the top, we borrow the idea of bottleneck features. The assumption is that the highest abstract representation for phone classification might be very different from that for state classification. Instead, the lower abstract representation which contributes for phone classification might be useful for state classification after being further refined. Figure 1 presents the idea of taking lower abstractions from both tasks and combining them for training the final state classification model. Two blue cascades of layers form two sub-networks(regions) with shared input features and the first layer abstraction. The left sub-network learns phone classification and the right one learns HMM state classification. The bottleneck layers from both sub-networks are fused together to train the primary sub-network with HMM state targets which is colored in magenta.

Figure 1: An example of MR-DNN acoustic model fusing lower-level abstractions trained with different targets



2.2. MR-DNN For Long-Context Input Features

MR-DNN provides a united style of architecture for modeling. Prior knowledge can be integrated though specific target selection and layer connection. For sequence training, longer context input features always provide more useful information. However, the influence of a feature decreases when it is farther away from the central frame. We could use MR-DNN architecture to model this prior knowledge. The long context input features are segmented and sent to different regions for different targets. The abstract of these features are extracted from auxiliary regions and send to support the primary region.

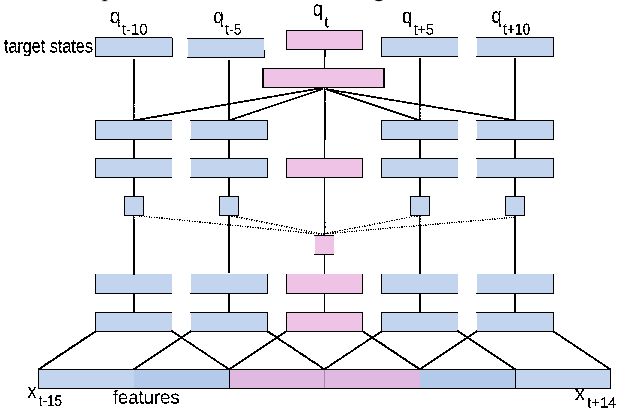
Figure 2 shows a specific implementation of MR-DNN acoustic model that uses a 30-frame long input feature. Instead of having a large hidden layer to accommodate the oversized input feature, MR-DNN uses 5 regions(sub-networks). The primary region(sub-network) is colored magenta, while the auxiliary regions are colored blue. Each region takes a 10-frame context window as input and predicts frames in order at time t-10, t-5, t, t+5 and t+10. The primary region predicts the context dependent state of the central frame at time t. The assumption underlying the architecture is that the far-away frames affect the central frame classification in an indirect way. The information from those frames can be extracted as an abstract representation and transmitted by connections between hidden layers of different regions. In Figure 2, the *continuous* cross-network lines connect the output of hidden layers from auxiliary region to the input of hidden layers in the primary region. The *dotted* lines represent reversed connections sending output of primary layers to the input of auxiliary layers.

2.3. Variations of MR-DNN

MR-DNN architecture is a non-loop lattice of hidden layers so it can be trained by regular back propagation algorithms. The only difference is that during back propagation, the gradients from multiple output (or upper) layers are interpolated. A larger interpolation weight means more control from that output layer. It is possible for an interpolation weight to be 0, which enables the output layer to accept features from the current layer yet without interfering with its learning process. By constructing different connections and configuring different interpolation weights, we can build various types MR-DNN models.

1. *supportive DNN*: A DNN that uses the architecture of Figure 2 without dotted line, with no gradient from the primary region either. This means that the auxiliary regions are trained completely independently of one another, therefore only used as feature extractors for the primary region. Comparing the performance of this model to that of the *multi-task centralized* architecture tells us if tuning the auxiliary regions on the primary task is useful.

Figure 2: An example of MR-DNN acoustic model with large context input features and different targets



2. *single-task DNN*: A DNN that uses the architecture of Figure 2 without dotted line, with no auxiliary target layers either. In other words, all regions are trained only on the primary task. Comparing the performance of this model to that of the full multi-task architecture tells us if multi-task training is helpful.

3. *multi-task centralized*: A DNN that uses the architecture of Figure 2 without the dotted connections. The primary region accepts features from sub-regions as well as feeding back gradients.

4. *multi-task broadcast*: A DNN that uses the architecture of Figure 2 with the dotted connections to broadcast BN features of the primary region to auxiliary regions.

3. Experiments

3.1. Experimental Setup

We investigate MR-DNN on speech recognition with data from IARPA Babel Program. The Babel program simulates a low resource language by collecting data in a limited time from a new language, comprising of scripted and conversational speech. The Limited Language Pack (LLP) scenario is one where only one tenth of the total data (referred to as Full Language Pack (FLP)) is used for training. The Very Limited Language Pack (VLLP) scenario contains only 3 hours of transcribed data.

Our speech recognition system is trained using the IBM Attila speech recognition toolkit. The Input features are GMM/HMM feature trained with the recipe described in [18]. The feature processing pipeline computes 13-dimensional PLP features with speaker-based mean and variance normalization and Vocal Tract Length Normalization (VTLN), a context of 9 frames projected down to a 40-dimensional feature space using linear discriminant analysis (LDA), and further de-correlated with a global, semi-tied covariance (STC) transform. These features comprise our *baseline feature set* for both GMM and DNN acoustic models in any language.

Most of our experiments are conducted with Tamil LLP data. The transcribed data of Babel LLP contains 20 hours of

conversational data and 3 hours of scripted data. Large portion of this data is silence, leaving approximately 15 hours of real speech for acoustic model training. The transcription contains 12K sentences with 78K word tokens and results in a 16K lexicon which contains words in the training data only. The development set includes 20 hours data, 11k sentences with 71K words. The Out-of-Vocabulary (OOV) rate is 15% and a modified Kneser-Ney smoothed tri-gram language model results in a perplexity of 336 on this test set.

3.2. Baseline Models and MR-DNN with Phone Targets

The baseline DNN model contains 4 hidden layers with 1K sigmoid units per layer except the third one which has dimension 80 It has a final softmax output with 1,000 targets corresponding to 1000 context dependent states. The input features for the baseline DNN training are DT trained SAT features from the GMM baseline, concatenated with log-mel features (fMPE.SA+logmel) with 9 frames of context, plush the domain awareness features which is a 13 dimensional PLP mean vector.

Table 1 presents two GMM/HMM baselines for Tamil LLP data. The speaker adapted maximum likelihood model (SAT) gives 86.6% WER and the discriminatively trained FMPE model gives 81.2%. After that, we first build an initial DNN with alignment from the GMM model, conduct both cross-entropy(XENT) training and distributed Hessian-free (HF) training using the state-level minimum Bayes risk criterion [19, 20]. This model is then used to re-generate state and phone alignment used for training the baseline DNN and all other DNN models.

The baseline DNN model has exactly the same structure as the initial DNN model and takes the same GMM features as input. It yields 80.6% WER with random initialization and 80.0% with layer-wise discriminative pre-training. A simple improvement of the baseline model is to incorporate the bottleneck (BN) layers from the initial DNN model (after HF training) as part of the input features, that gives 0.5% improvement after XENT training.

Table 1: Baseline GMM/HMM and DNN results on Tamil LLP

Model	WER
GMM FSA	86.6
GMM FMPE	81.2
DNN XENT w/o pretraining	80.6
DNN XENT w/ pretraining	80.0
DNN with BN features XENT	79.5
Multi-task with phone targets XENT	81.0
MR-DNN with phone targets XENT	79.5

The initial multi-task training by adding phones as an auxiliary target layer fails with 1% worse in terms of WER. Inspired by the BN experiment, we redesign the DNN strcture as showed in Figure 1. This model achieves the same performance as that from the BN model. It means that the BN layers learned from the auxiliary phone classification task does help in predicting the state classification. Compared to the previous two-stage modeling, the MR-DNN model does not need an initial DNN to provide BN features and it is randomly initialized without pretraining.

3.3. MR-DNN With Multiple Temporal Targets

We first use MR-DNN to simulate the idea of an unfolded RNN. RNNs allows the network to contain feed-back connections to improve sequence recognition by forming a directed cycle between different layers. The unfolded RNN architecture proposed in [21] shows that a limited history expansion for speech recognition works as well as unlimited recursion. We use a MR-DNN structure as showed in 2, with five regions predicting frames at time t-4,t-3,t,t-2,t-1 from left to right. Each region takes 10 frames centering at each position correspondingly. For fair comparison, we also built five independent DNN models. Each DNN has the baseline structure, taking the same input feature which has a total of 20 frames and predicting frames at time t-4,t-3,t-2,t-1,t respectively. During decoding, the scores at each position are interpolated equally. This gives 0.5% improvement over the baseline. Table 2 shows that simulation successfully improves the the WER by 1.1%. It is better than than just building 5 DNNs separately with 5 five different targets and then interpolating the results. An interesting experimental result shows that with the same MR-DNN structure, if we replace all targets as the primary targets, the performance is almost the same 79.1%. It indicates that most gains come from the independent region training and feature sharing.

Table 2: Using MR-DNN to simulate unfolded RNN on Tamil LLP

Model	XENT
Baseline	80.0
Interpolation of 5 DNNs	79.5
MR-DNN with targets of history frames	78.9
MR-DNN with targets of only the central frame	79.1

We can also use MR-DNN architecture to simulate the idea of a stacked DNN [22, 23]. In [22], two DNNs are built consecutively. The first DNN is trained with GMM features with 5 frames in each side. The second DNN takes the BN layer of the first DNN as input with 10 frames context. This input actually include information from a total of 31 frames of GMM features. Rather than using the complete input features, they sample the input feature every 5 frames to actually take only a length of 5 frames of BN features as the input of the second DNN. We can use the joint training property and multitasking in MR-DNN to simulate the whole process. We use the structure showed in Figure 2. The input features are 30 frames of GMM features but each each region takes only 10 frames centering at position $t-10, t-5, t, t+5, t+10$ respectively. Each region predicts HMM state of a frame at the above positions respectively.

Compared to the current baseline with only 1024 units in each hidden layer, a MR-DNN with 1024 in each hidden layer takes much more input information and much larger hidden capacity. For fair comparison, we also build a baseline DNN model (baselineB) with the same input features and expand the hidden unit is each layer to 5120. The new baseline gives 79.1% WER as shown in Table 3. The stacked DNN gives 78.6%. MR-DNN gets even better result as low as 77.7% which is almost the same as the baseline sequence training result 77.4%. Even though MR-DNN gains a lot from the long context information, it can still be further improved by sequence training and lower the WER to 76.2%. which is similar to the stacked DNN sequence training results 76.3%. Note that this MR-DNN does not completely simulate the stacked DNN. It is an enriched version because the primary region affects the auxiliary region during training which in stacked DNN, the second DNN has no influence on the first DNN training. A better similation is the supportive version of MR-DNN discussed in Section 2.3 and it

will be discussed in next subsection.

Table 3: Use MR-DNN to simulate stacked DNN on Tamil LLP

Model	WER
Baseline	80.0
Baseline B	79.1
stacked DNN	78.6
MR-DNN	77.7
baseline with sequence training	77.4
stacked DN with sequence training	76.3
MR-DNN with sequence training	76.2

3.4. Analysis and Variations of MR-DNN

As we have discussed in Section 2, there could be many variations of MR-DNN depending not only on the position of the cross region connections, but also on the back propagation weight control. Building models of these variations helps us to better understand which factor benefits MR-DNN modeling. Table 4 shows the WER results on Tamil LLP using four variations of MR-DNN, all using the same input and output targets as in Section 3.3.

The first row in Table 4 shows that if no auxiliary target layers are used and all gradients comes only from the primary target layer, it is still better 79.2% than the the interpolation of five DNNs (79.5%)as showed in Table 2. It achieves almost the same results as the large baseline with 5012 hidden units in each layer (79.1% in Table 3), with much fewer parameters. When compared with multi-task centralized 78.0%, it shows that gradients from the auxiliary targets are helpful.

The second row in Table 4 shows the case that all auxiliary regions are trained completely independently and they all just contribute their last hidden layer features to the primary task without getting any gradient feedback. It gives much more gains than the single-task model with 78.5%. The supportive model is a better simulation of stacked DNN (78.6%) and it does give almost the same WER performance. Actually, if the supportive layers are lowered, that performance could be even better and reaching 78.1%.

The third row in Table 4 presents the MR-DNN when the supportive layers of auxiliary regions communicate with the primary region in both directions. The forth row enhance the communication between auxiliary regions and the primary region in that the auxiliary regions also accept BN features from the primary region. These two factor adds up gives another decent gain for MR-DNN and the WER drops to 77.7%.

Other than the connections, the positions of the connections affect the performance. If the dotted lines from the primary regions go to higher hidden layers, the WER performance can drop 0.4%. In the supportive case, if the supportive layers shift down in the auxiliary region, the performance can actually improve a little bit around 0.4%. Yet this gain disappears in the multi-task broadcast setup. The split of input features are also neccesary. If the same structure is used as in multi-task broadcast, but the input is not split, but all 30 frames are used, the WER is 78.3%. It indicates that expanding input features does not always help.

3.5. Other MR-DNN Experiments

We have also conducted some preliminary MR-DNN experiments with Telegu VLLP dataset and Swahili FLP dataset. Table 4: MR-DNN Variations on Tamil LLP

Model	WER
single-task	79.2
supportive	78.5
multi-task centralized	78.0
multi-task broadcast	77.7

ble 5 presents the experimental results of only XENT models. For Telegu, the input features are GMM/HMM fmllr features with 9 frames in context. The GMM models are graphemic models trained using the same pipeline as mentioned in Section 3.1. DNN models are built with sigmoid functions for hidden layers. There are 3295 sentences and 7079 unique words in the training data and 11541 tokens in the dev data. For Swahili, we use speaker-independent features with alignment from an initial DNN model. There are 52K sentences and 24K uniq words in the training data, and 76K word tokens in the dev data. In both cases, the MR-DNN models have given decent improvement over the baseline models.

Table 5: MR-DNN With Telegu VLLP and Kurmanji SSL

Model	WER
Telegu VLLP baseline	84.8
Telegu VLLP MR-DNN	84.0
Swahili FLP baseline	55.1
Swahili FLP MR-DNN	52.3

4. Conclusions and Future Work

We have presented a Multi-Region DNN architecture which integrates joint training, multi-task learning and ensemble learning. We demonstrate that the MR-DNN can be used to simulate unfolded RNN and stack DNN modeling with the same or better performance on low-resource speech data sets. MR-DNN also provides a new way to use a large input context: instead of using fully connected large hidden layers, MR-DNN splits each layer into several smaller layers and cascades them independently. By careful structure designing with prior knowledge, the MR-DNN can improve the performance significantly. The current implementation of cross region connection of MR-DNN is only a feature fusion, but a complicated connection such as multiplication or summation may be more beneficial.

5. Acknowledgments

This work is 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-0012. 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 effort uses the IARPA Babel Program language collection release IARPA-babel204b-v1.1b limited language pack and and IARPA-babel303b-v1.0a very limited language packs and IARPA-babel202b-v1.0d.

6. References

[1] Y. Bengio, "Learning deep architectures for ai," *Foundations and Trends in Machine Learning*, 2009.

[2] G. E. Hinton, L. Deng, D. Yu, G. E. Dahl, A. rahman Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, p. 829, 2012.

[3] L. Deng and D. Yu, "Deep learning: Methods and applications," *tech report of Microsoft*, 2014.

[4] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. PAMI, special issue Learning Deep Architectures*, 2013.

[5] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, pp. 85–117, 2015.

[6] M. Matusugu, M. Katsuhiko, M. Yusuke, and K. Yuji, "Subject independent facial expression recognition with robust face detection using a convolutional neural network," *Neural Networks*, pp. 555–565, 2003.

[7] Y. LeCun, "Lenet-5, convolutional neural networks," *Retrieved*, 2013.

[8] F. A. Gers and J. Schmidhube, "Lstm recurrent networks learn simple context free and context sensitive languages," *IEEE Transactions on Neural Networks*, pp. 1333–1340, 2001.

[9] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for improved unconstrained handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009.

[10] C. Weng, D. Yu, S. Watanabe, and B.-H. Juang, "Recurrent deep neural networks for robus speech recognition," in *ICCASP*, 2014.

[11] D. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," *Computer Vision and Pattern Recognition (cs.CV)*, 2012.

[12] B. Li and K. C. Sim, "Modeling long temporal contexts for robust dnn-based speech recognition," in *INTERSPEECH*, 2014.

[13] D. Chen, B. Mak, and S. Sivadas, "Joint sequence training of phone and grapheme acoustic model based on multi-task learning deep neural networks," in *Interspeech*, 2014.

[14] Y. Huang, W. Wang, L. Wang, and T. Tan, "Multi-task deep neural network for multi-label learning," in *Proceedings of the IEEE International Conference on Image Processing*, 2013, p. 2897290.

[15] D. Chen, B. Mak, C.-C. Leung, and S. Sivadas, "Joint acoustic modeling of triphones and trigraphemes by multi-task learning deep neural networks for low-resource speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, May 2014, pp. 5592–5596.

[16] G. H. Navdeep Jaitly, Vincent Vanhoucke, "Autoregressive product of multi-frame predictions can improve the accuracy of hybrid models," in *Interspeech*, 2014.

[17] H. Soltau, G. Saon, and T. N. Sainath, "Joint training of convolutional and non-convolutional neural networks," in *ICASSP*, 2014.

[18] H. Soltau, G. Saon, and B. Kingsbury, "The IBM Attila speech recognition toolkit," in *Spoken Language Technology Workshop (SLT), 2010 IEEE*. IEEE, 2010, pp. 97–102.

[19] B. Kingsbury, T. N. Sainath, and H. Soltau, "Scalable minimum bayes risk training of deep neural network acoustic models using distributed hessian-free optimization," in *INTERSPEECH*, 2012.

[20] T. N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novk, and A. rahman Mohamed, "Making deep belief networks effective for large vocabulary continuous speech recognition," in *ASRU*, 2011, pp. 30–35.

[21] G. Saon, H. Soltau, A. Emami, and M. Picheny, "Unfolded recurrent neural networks for speech recognition," in *Interspeech*, 2014.

[22] F. GREZL, M. KARAFIAT, and L. Burget, "Investigation into bottle-neck features for meeting speech recognition," in *Proc. Interspeech 2009*, 2009, pp. 2947–2950.

[23] M. KARAFIAT, F. GREZL, K. VESELY, M. HANNEMANN, I. SZOKE, and J. CERNOCKY, "But 2014 babel system: Analysis of adaptation in nn based systems," in *Interspeech*, 2014, pp. 3002–3006.