Deep Convolutional Neural Networks with Layer-wise Context Expansion and Attention

Dong Yu[1], *Wayne Xiong*[1], *Jasha Droppo*[1], *Andreas Stolcke*[1], *Guoli Ye*[2], *Jinyu Li*[2], *Geoffrey Zweig*[1]

[1]Microsoft Research, Microsoft Corporation
[2]Information Platform Group, Microsoft Corporation
{dongyu, weixi, jdroppo, anstolck, guoye, jinyli, gzweig}@microsoft.com

Abstract

In this paper, we propose a deep convolutional neural network (CNN) with layer-wise context expansion and location-based attention, for large vocabulary speech recognition. In our model each higher layer uses information from broader contexts, along both the time and frequency dimensions, than its immediate lower layer. We show that both the layer-wise context expansion and the location-based attention can be implemented using the element-wise matrix product and the convolution operation. For this reason, contrary to other CNNs, no pooling operation is used in our model. Experiments on the 309hr Switchboard task and the 375hr short message dictation task indicates that our model outperforms both the DNN and LSTM significantly.
Index Terms: speech recognition, convolutional neural network, attention, very deep networks.

1. Introduction

Since 2010, the year in which deep neural networks (DNNs) were successfully applied to the large vocabulary speech recognition (LVSR) [1, 2, 3] tasks and led to significant recognition accuracy improvement over the then state of the art, various deep learning models have been developed to further improve the performance of speech recognition systems. The majority of these new models are variations and/or combinations of the recurrent neural networks (RNNs) and convolution neural networks (CNNs) [4].

In this paper, we propose a new CNN architecture for LVSR. CNNs were designed to exploit the translational invariance in signals. A typical CNN system consists of one or more pairs of convolution and pooling layers. In the convolution layer, a set of filters are shifted and repeatedly applied to the input[1]. A max-pooling or average-pooling layer is used to generate a lower resolution version of the convolution layer activations. The pooling layer is often important to tolerate translational variances.

CNNs are critical to image recognition tasks. All ImageNet winning systems use CNNs. Recently, the 152-layer ResNet (residual network [5]) drove the ImageNet classification error rate down to 3.52% and became the new state of the art.

For speech recognition, CNNs have also been demonstrated as promising [6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24]. Originally, CNNs were applied along the frequency dimension [6, 7, 8, 10] to reduce variations caused by vocal tract length differences between speakers. Later, they were applied to the time dimension [17, 22] to account for speaking rate variation. In these early works, only one to two CNN layers were typically used and they were used to model either the spectral or the temporal variations.

However, spectrograms are images with special patterns. Experienced people can tell what has been said by inspecting only the spectrogram. For this reason, we believe CNNs, which can model temporal and spectral local correlations and gain translational invariance, should be applied to both the frequency and time dimensions. In fact, successful attempts along this line has been reported very recently [23, 24].

In this paper, we propose a deep CNN that operates in both frequency and time dimensions. Unlike prior art, our model incorporates layer-wise context expansion with location-based attention mechanism and employs a jump connection between layers, similar to the linearly augmented DNN (LADNN) [25] and ResNet [5]. We show that the layer-wise context expansion and attention can be implemented using the element-wise matrix product and convolution operations. For this reason, neither max-pooling nor average-pooling is used in our model. Experiments show that our model can reduce word error rate from 14.9% (DNN) and 12.6% (LSTM) to 11.0% on the 309hr SWB task, and from 16.1% (DNN) and 14.4% (LSTM) to 13.0% on the 375hr short message dictation task, all without model adaptation and sequence discriminative training.

The rest of the paper is organized as follows. In Section 2 we first revisit the convolution operation and point out that the convolution operation essentially computes weighted sums and as such average pooling can be implemented using the convolution operation directly. We then describe the key concepts and detailed components in our proposed model. We discuss related works in Section 3 and report experimental results on both the Switchboard (SWB) and the Short Message Dictation (SMD) tasks in Section 4. We conclude the paper in Section 5.

2. Deep CNNs with Layer-wise Context Expansion and Attention

2.1. Revisit Convolution Operation

An image (or spectrogram) can be represented as a three-dimensional tensor *(row, column, channel)*. Each channel is a view of the same data. All channels have the same size *(height, width)*. The convolution operation applies kernels of a four-dimensional tensor *(kernel height, kernel width, input channel, output channel)* to local regions (receptive fields) of multiple channels in an input image. For each output channel ℓ and input slice (i, j) (the i-th step along the vertical direction and j-th step along the horizontal direction), the value after the convolution

[1]If we consider the whole utterance as a single image, the feed-forward DNN can also be considered as a special CNN architecture that applies the same filter repeatedly to patches of the input.

operation is

$$\upsilon_{ij\ell}\left(\mathbf{K},\mathbf{X}\right) \quad = \quad \sum_{n} \mathrm{vec}\left(\mathbf{K}_{n\ell}\right) \cdot \mathrm{vec}\left(\mathbf{X}_{ijn}\right), \qquad (1)$$

where $\mathbf{K}_{n\ell}$ of size (H_k, W_k) is a kernel matrix associated with input channel n and output channel ℓ and has the same size as the input image patch $\mathbf{X}_{ijn}$ of channel n, $\mathrm{vec}(\dot{})$ is the vector formed by stacking all the columns of the matrix, and $\cdot$ is the inner product of two vectors. The total number of kernels equals the product of the number of input channels C_x and that of output channels C_v. The kernel moves along (and thus shared across) the input image with strides (or subsampling rate) S_r and S_c at the vertical and horizontal direction, respectively.

By default, the kernel stops moving when the size of the kernel is larger than that of the patch left. The kernel operation can still be valid under this condition if we allow for zero padding in the image. Since either the kernel height, the kernel width, or both are greater than 1, the resulting image always shrinks in size without padding. If the strides are 1 in both directions and zero padding is applied, the resulting image has the same size as the input image.

We want to point out two important properties of the convolution operation. First, each pixel in the output image is a weighted sum of all pixels (along all channels) within a patch. Both the patch size and the weights are determined by the kernel. This property indicates that it is possible to convert weighted sum of pixels to a convolution operation. For example, average pooling can be represented as a convolution operation with a kernel whose values are all $1/(H_k \times W_k)$. Second, the image can be down-sampled by using a stride that is larger than 1. These two properties indicates that a pooling operation is not necessary to make CNNs work effectively.

2.2. Layer-wise Context Expansion and Attention

It is well known that exploiting contextual frames is important for deep learning based models to achieve good speech recognition accuracy. In feed-forward DNNs, this is achieved by using a window of frames as inputs, where the size of the contextual window is determined a priori. RNNs, on the other hand, can exploit the contextual information automatically by remembering them in the recurrent states. While the uni-directional RNNs only use information prior to the current frame, the bi-directional RNNs can exploit contextual information in both directions. Unlike feed-forward DNNs, RNNs can learn the effective length of the contextual window.

In this work, we propose the LAyer-wise Context Expansion and Attention (LACEA) model as shown in Figure 1 after noticing that the length of useful context is limited for phoneme-state recognition. In this architecture, each higher layer is a weighted sum of nonlinear transformation of a window of lower layers. For example, in Figure 1(a), layer 2 operates on a window of 5 frames of layer 1, which in turn operates on a window of 5 frames of input feature. It is clear that each frame at higher layer spans more frames and covers longer contexts. For example, each frame in layer 1 and layer 2 covers 5 and 9 frames of input feature, respectively. Since each higher layer operates on a window of frames in their immediate lower layer, it potentially operates on more abstract patterns. Different from using the contextual window in the input feature only, as in the feed-forward DNNs, in LACEA the lower layers can focus on extracting simple local patterns and higher layers can extract complex patterns that cover broader contexts.

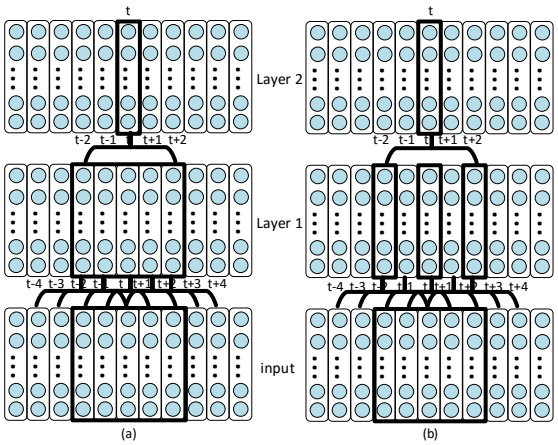

Figure 1: *Illustration of the layer-wise context expansion.*

The LACEA model can be more efficiently trained if each utterance is a training unit. This is because each frame at the lower layer will be used by multiple frames in the immediate higher layer. If each utterance is a training unit, each frame at any layer only needs to be computed once and reused multiple times when computing the higher layer units. In practice, however, we found that doing so has three problems: First, we can only use utterance randomization which often converges worse than the frame randomization. Second, it prevents us from building deeper models due to GPU memory limitation. This is because the length of utterances is in principle unlimited. Third, using utterance as the training unit may actually slow down, instead of speed up, the training process due to less degree of parallelization in spite of the elimination of duplicated computation. This is, again, tied to the fact that GPU memory is limited and you can only process very small number of utterances in parallel even if the model is not deep. For these reasons, in this study we employed the frame-level training strategy which provides higher flexibility with regard to the number of samples in each minibatch and configurations with deeper and wider networks at the cost of duplicated computations.

To further reduce the computation cost, we can down-sample the image after each layer. As shown in Figure 1(b), since the adjacent frames are very similar at each higher layer, we only need to keep, for example, odd frames, and drop even frames. Since the main operation in LACEA is weighted sum over the time (and optionally frequency) dimensions, the whole mechanism can be implemented using the convolution operation by specifying a stride that is larger than 1.

In the convolution operation, however, the same kernel is used across the whole spectrogram, while the contribution from each frame (and each frequency bin) often varies with its distance to the current frame. This is different from the image classification task. To incorporate this observation, we introduce the attention mechanism into the system. Basically, an importance (or static attention) weight matrix is multiplied, element-wise, with each layer before the convolution operation is applied. This attention matrix is the same for all channels and covers the whole down-sampled image. In other words, it spans wider range than what can be covered by each kernel. Combining all these treatments, we can significantly reduce computation cost without sacrificing accuracy.

2.3. The Complete Model

Figure 2 illustrates the detailed structure of the whole model. As shown in Figure 2(a), the model is composed of N jump blocks, a convolution layer whose goal is to compute the weighted sum (over width and height) of channels as we explained in Section 2.1, and a softmax layer which is essentially a log-linear model. Each element of the kernel in the last convolution layer is initialized to be $1/(H_k \times W_k)$.

Figure 2(b) describes the jump block. Each jump block starts with a convolution layer, whose goal is to reduce the resolution of the image but increase the number of channels. In our study, we kept using kernels with size $(3, 3)$ and strides of $(2, 2)$ and always double the number of channels for this layer in all jump blocks. Note that kernels with different sizes are often used in image classification models. However, we have found that using a small kernel is important for speech data mainly because the resolution of the speech feature is already low compared to images. It is obvious that this layer is the combination of layer-wise context expansion and frame dropping. After this critical convolution layer are the M jump nets, and an element-wise matrix product operation that implements the location-based attention mechanism. As mentioned in Section 2.2 the attention matrix covers the whole image at that layer and is the same for all channels. Unlike kernels, which are initialized randomly, each element in the attention matrix is initialized to 1. Note that this weighted image will be the input to the next jump block, which involves another LACEA operation. We did not apply the attention mechanism to the input feature and it is possible that doing so would be beneficial.

Each jump net is a complicated nonlinear transformation as shown in Figure 2(c). Basically, the input image is first processed with a conventional convolution ReLU layer with batch normalization. Another convolution operation is then applied to $y1$, the output of this first convolution layer. The output of this second convolution operation $c2$ is summed with the input of the jump net and processed by another batch normalization and ReLU layer. To be able to sum with the input image, the dimensions of the image tensor are kept the same across all the layers in the same jump net. In our model, this is implemented by using stride one across both width and height and by enabling zero padding. Note that convolution and sum operations are both linear operations. Thus it is critical to have nonlinear components such as the ReLU units used in our model.

The jump connection from x to the plus operation allows for a direct gradient backpropagation and thus alleviates the gradient vanishing problem. It helps to train very deep models more effectively although it may not be critical to achieve our specific experimental results reported in Section 4 since the total number of layers used in our model is similar to the VGG net.

3. Related Work

Various CNN structures [6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24] have been evaluated for speech recognition. The ones most similar to our work are [20, 24, 26] which employed the famous very deep VGG network structure. These works, however, do not include the attention mechanism, the jump connection, and the weighted sum at the top layer.

The layer-wise context expansion has been studied in [27, 21, 22] under different names and setups. However, as pointed out by Amodei et al. [21], the basic block of the computation can be implemented as a convolution operation and is in fact called "row convolution" in their work. Note that while in [21] the context is expanded for each higher layer, in [27, 22] the contextual information is only exploited at the input layer and optionally the first hidden layer. Even in [21] only the right context information is expanded, similar to [28], since their goal is to incorporate information from future frames into RNNs without looking to the end of the utterance. In all these works, deep CNNs are not part of their models and the context expansion only happens along the time dimension.

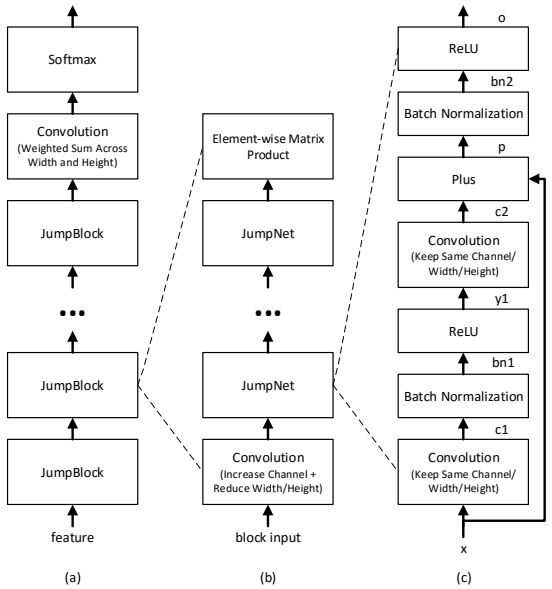

Figure 2: *The detailed diagram of the final model.*

The attention mechanism was firstly proposed in the CNN-based speech recognition [7] as an automatic way to decide the pooling range and weights. It became popular after it is shown to be important in the sequence-to-sequence translation model [29]. The attention mechanism has also been used for speech recognition [30] under the sequence-to-sequence framework.

The concept of using jump connection to improve the convergence property has been studied in the linearly augmented deep neural network (LADNN) [25], the highway LSTM [28] and the residual network (ResNet) [5]. In fact, the basic jump net architecture in this work is very similar to that in ResNet with differences in the ordering of some operations.

4. Experiments

Our models were built using the computational network toolkit (CNTK) [31]. The experiments were carried out on a GPU cluster that is optimized for CNTK for rapid no-hassle deep learning model training and evaluation. The cluster, which has high-throughput distributed storage, virtual file systems, and fault tolerance, is managed by specially designed automated cluster management and job/container scheduling software. Each GPU machine in the cluster contains four K40 GPU cards. Infiniband is used to connect nearby GPU machines for high-speed data transmission between GPUs across machines. To speed up the experiments, we have exploited the 1-bit quantized SGD algorithm [32] built into CNTK and run all the experiments on 8 GPUs across two computers.

We evaluated our proposed model on the Switchboard and short message dictation tasks, both of which have about 300

hours of training data. In our experiments, the kernel sizes are always set to $(3, 3)$ and we always use $M = 4$ jump blocks. The channel sizes for these blocks are 128, 256, 512, and 1024. We choose these numbers mainly because the final channel number 1024 is a popular number for hidden layers in DNN-based ASR systems. We did not attempt to tune these values. Given the input context of 30-1-30 frames, which is the typical value used in our experiments, and the feature dimension of 40, the image tensors in each jump block are $(20, 31, 128)$, $(10, 16, 256)$, $(5, 8, 512)$, and $(3, 4, 1024)$. The models used the CD-DNN-HMM [2] framework.

We mainly compare against the DNN and LSTM baselines. The DNN model has 5 hidden layers, each with 2048 sigmoid units. The LSTM model has 4 layers, each with 1024 LSTM cells and a 512-unit projection layer. The output HMM state label for the LSTM model is delayed by 5 frames. The mean-variance feature normalization was used for the DNN and LSTM models but not for the CNN-LACEA model. All models were trained using the frame cross-entropy criterion.

4.1. Switchboard

Switchboard (SWB) is a conversational speech transcription task. In our experiments, we used the 309hr training set. We reserved 80 conversational sides selected randomly from the training set as the validation set for controling the learning rate during model training.

The feature used in this task is the 40-dim log Mel-scale filter bank. The speaker-independent crossword triphones use the common 3-state topology and share 9000 CART-tied states across all models. The GMM-HMM baseline system was trained with the maximum likelihood (ML) criterion and refined with the boosted maximum-mutual-information (BMMI) sequence-discriminative training criterion. This baseline GMM-HMM system was used to generate the label alignment to train, with the frame cross-entropy criterion, a six-layer ReLU-DNN. This DNN is then used to generate a refined alignment to train all other models, including a six-layer ReLU-DNN baseline. Each layer in this DNN baseline contains 2048 units.

The language model (LM) used in this task incorporates 3M words of SWB transcripts, 21M words from the Fisher corpus, and 191M words of conversational Web data compiled by U. Washington, pruned to 13.6M bigrams, trigrams and fourgrams. The vocabulary size is 226k. Decoding used Microsoft's WFST-based Argon recognizer. The AM weight was set to 0.0667 (corresponding to an LM weight of 15) and not tuned. No insertion penalty was used. The evaluation was conducted on the 1831-segment SWB part of the NIST 2000 Hub5 eval set. In all experiments, recognition was done in a single-pass without any speaker adaptation or using any auxiliary information.

Table 1 compares our proposed model with the baseline systems[2]. We can observe that the proposed approach performs significantly better than DNNs and LSTMs. The context window of 30-1-30 seems to be sufficient. Increasing it to 40-1-40 does not improve the accuracy.

The CNN-LACEA models converges very fast. After sweeping through the 309hr data twice, the training already converges to close-to-optimal parameters. The CNN-LACEA results reported in the table were achieved after sweeping the training set three times. CNN-LACEA takes 5x more time than LSTM to sweep data once if both are implemented using cuDNN but requires 1/5 of sweeps.

[2]Recently WERs of 11.6% (LSTM) and 10.3% (BLSTM) were reported using CE training [33]. However, these results were achieved using speaker-adaptive features, speed-perturbation for 3-fold data augmentation, and iVector for instantaneous adaptation.

Table 1: *Word error rate on the SWB task*

Model	Window	WER	WER (Full LM)
DNN	-	15.1%	14.9%
LSTM	-	12.8%	12.6%
BLSTM	-	12.0%	11.8%
CNN-LACEA	30-1-30	11.3%	11.0%
CNN-LACEA	40-1-40	11.3%	11.0%

4.2. Short Message Dictation

The Short Message Dictation (SMD) task contains 375 hours of transcribed US-English data, of which 12 hours are reserved as the validation set to determine the learning rate schedule. The test set contains 17.8k word tokens.

The 80-dim log Mel-scale filter bank feature was used in the study for the LSTM and CNN-LACEA models. For the DNN baseline model, an 87-dimensional feature that consists of the 29-dimensional static log-filter-bank outputs and their first- and second-order derivatives, is used. This 87-dim feature outperformed the 80-dim log filter bank feature for the DNN baseline. All models evaluated in this study use 5976 tied-triphone states (senones), determined by a CD-GMM-HMM system, and were trained to minimize the frame-level cross-entropy criterion.

Table 2 summarizes the WER on the SMD task. In this study, the context window used in the CNN-LACEA model is 30-1-30 per the discussion in Section 4.1. We can observe that on this task the proposed system also outperformed both the DNN and LSTM baselines. We want to point out that we only swept the training set three times to achieve the reported CNN-LACEA results while the LSTM model was optimized with 16 sweeps of the data.

Table 2: *Word error rate on the SMD task*

Model	WER
DNN	16.1%
LSTM	14.4%
CNN-LACEA	13.0%

5. Conclusions

In this paper, we proposed a novel deep CNN for LVSR. We employed layer-wise context expansion and attention for more powerful modeling and jump connection for better convergence. No pooling is used in our model since pooling can be better implemented using a convolution operation as well.

Our work indicates that the full potential of CNNs may yet to be explored. This is our first attempt to build very deep CNNs. There are many dimensions to explore. By combining our model with existing techniques such as sequence discriminative training, further improvements should be achievable.

6. Acknowledgments

The authors would like to thank Xuedong Huang, Frank Seide and Alexey Kamenev for valuable discussions and comments, and Frank Seide, Alexey Kamenev, Amit Agarwal, and Chris Basoglu for CNTK and GPU cluster support.

7. References

[1] D. Yu, L. Deng, and G. E. Dahl, "Roles of pre-training and fine-tuning in context-dependent dbn-hmms for real-world speech recognition."

[2] G. E. 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, no. 1, pp. 30–42, 2012.

[3] F. Seide, G. Li, and D. Yu, "Conversational speech transcription using context-dependent deep neural networks." in *Proc. Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2011, pp. 437–440.

[4] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[6] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 4277–4280.

[7] O. Abdel-Hamid, L. Deng, and D. Yu, "Exploring convolutional neural network structures and optimization techniques for speech recognition." in *Proc. Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2013, pp. 3366–3370.

[8] T. N. Sainath, A.-r. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 8614–8618.

[9] P. Swietojanski, A. Ghoshal, and S. Renals, "Convolutional neural networks for distant speech recognition," *IEEE Signal Processing Letters*, vol. 21, no. 9, pp. 1120–1124, 2014.

[10] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 22, no. 10, pp. 1533–1545, 2014.

[11] T. N. Sainath, R. J. Weiss, K. W. Wilson, A. Narayanan, M. Bacchiani, and A. Senior, "Speaker location and microphone spacing invariant acoustic modeling from raw multichannel waveforms," in *Proc. IEEE Workshop on Automfatic Speech Recognition and Understanding (ASRU)*, 2015.

[12] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4580–4584.

[13] T. N. Sainath, R. J. Weiss, A. Senior, K. W. Wilson, and O. Vinyals, "Learning the speech front-end with raw waveform CLDNNs," in *Proc. Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2015.

[14] T. Yoshioka, S. Karita, and T. Nakatani, "Far-field speech recognition using CNN-DNN-HMM with convolution in time," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4360–4364.

[15] J.-T. Huang, J. Li, and Y. Gong, "An analysis of convolutional neural networks for speech recognition," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4989–4993.

[16] W. Chan and I. Lane, "Deep convolutional neural networks for acoustic modeling in low resource languages," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 2056–2060.

[17] L. Toth, "Modeling long temporal contexts in convolutional neural network-based phone recognition," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4575–4579.

[18] P. Golik, Z. Tüske, R. Schlüter, and H. Ney, "Convolutional neural networks for acoustic modeling of raw time signal in LVCSR," in *Proc. Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2015.

[19] D. Palaz, R. Collobert *et al.*, "Analysis of CNN-based speech recognition system using raw speech as input," in *Proc. Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2015.

[20] M. Bi, Y. Qian, and K. Yu, "Very deep convolutional neural networks for LVCSR," in *Proc. Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2015.

[21] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos *et al.*, "Deep speech 2: End-to-end speech recognition in English and Mandarin," *arXiv preprint arXiv:1512.02595*, 2015.

[22] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proc. Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2015.

[23] T. Zhao, Y. Zhao, and X. Chen, "Time-frequency kernel-based CNN for speech recognition," in *Proc. Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2015.

[24] T. Sercu, C. Puhrsch, B. Kingsbury, and Y. LeCun, "Very deep multilingual convolutional neural networks for LVCSR," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[25] P. Ghahremani, J. Droppo, and M. L. Seltzer, "Linearly augmented deep neural network," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[26] V. Mitra and H. Franco, "Time-frequency convolutional networks for robust speech recognition," in *Proc. IEEE Workshop on Automfatic Speech Recognition and Understanding (ASRU)*, 2015, pp. 317–323.

[27] S. Zhang, C. Liu, H. Jiang, S. Wei, L. Dai, and Y. Hu, "Feed-forward sequential memory networks: A new structure to learn long-term dependency," *arXiv preprint arXiv:1512.08301*, 2015.

[28] Y. Zhang, G. Chen, D. Yu, K. Yao, S. Khudanpur, and J. Glass, "Highway long short-term memory rnns for distant speech recognition," *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[29] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[30] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Proc. Neural Information Processing Systems (NIPS)*, 2015, pp. 577–585.

[31] A. Agarwal, E. Akchurin, C. Basoglu, G. Chen, S. Cyphers, J. Droppo, A. Eversole, B. Guenter, M. Hillebrand, R. Hoens, X. Huang, Z. Huang, V. Ivanov, A. Kamenev, P. Kranen, O. Kuchaiev, W. Manousek, A. May, B. Mitra, O. Nano, G. Navarro, A. Orlov, M. Padmilac, H. Parthasarathi, B. Peng, A. Reznichenko, F. Seide, M. L. Seltzer, M. Slaney, A. Stolcke, H. Wang, Y. Wang, K. Yao, D. Yu, Y. Zhang, and G. Zweig, "An introduction to computational networks and the computational network toolkit," Microsoft Technical Report MSR-TR-2014-112, Tech. Rep., 2014.

[32] F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu, "1-bit stochastic gradient descent and its application to data-parallel distributed training of speech DNNs." in *Proc. Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2014, pp. 1058–1062.

[33] D. Povey, V. Peddinti, D. Galvez, P. Ghahrmani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," *Submitted to Interspeech*, 2016.