INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Acoustic Modeling using Bidirectional Gated Recurrent Convolutional Units

Markus Nussbaum-Thom, Jia Cui, Bhuvana Ramabhadran, Vaibhava Goel

IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, U.S.A.
{nussbaum, jiacui, bhuvana, vgoel}@us.ibm.com

Abstract

Convolutional and bidirectional recurrent neural networks have achieved considerable performance gains as acoustic models in automatic speech recognition in recent years. Latest architectures unify long short-term memory, gated recurrent unit and convolutional neural networks by stacking these different neural network types on each other, and providing short and long-term features to different depth levels of the network.

For the first time, we propose a unified layer for acoustic modeling which is simultaneously recurrent and convolutional, and which operates only on short-term features. Our unified model introduces a bidirectional gated recurrent unit that uses convolutional operations for the gating units. We analyze the performance behavior of the proposed layer, compare and combine it with bidirectional gated recurrent units, deep neural networks and frequency-domain convolutional neural networks on a 50 hour English broadcast news task. The analysis indicates that the proposed layer in combination with stacked bidirectional gated recurrent units outperforms other architectures.

Index Terms: gated recurrent units, convolutional neural networks, automatic speech recognition

1. Introduction

In the past few years, Deep Neural Networks (DNNs) have achieved overwhelming performance improvements for large vocabulary continuous speech recognition (LVCSR) tasks [1]. Further improvements have been achieved using Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) [2, 3].

Recently simple RNNs are replaced by the more complex Long Short-Term Memory (LSTM) RNNs which outperform DNNs and simple RNNs on a number of tasks [4, 5]. However in our experiments we decided to use Gated Recurrent Units (GRUs) instead of LSTMs for multiple reasons.

The GRU recently introduced in [6] has a structure similar to the LSTM. Interestingly, the studies in [7, 8] indicate that the GRU matches the LSTM performance. GRUs also consume a smaller number of parameters compared to LSTMs for the same hidden layer size due to a smaller gating mechanism and missing peepholes. In initial experiments, we verified that bidirectional GRUs (BGRUs) match the performance of bidirectional LSTMs for automatic speech recognition (ASR) and decided to use BGRUs in the remaining experiments.

Current architectures incorporationg CNNs and RNNs use different input features. CNNs [9] typically use a context window of short-term features surrounding the current frame while RNNs prefer the current short-term feature only as input. In detail, common short-term features are Vocal Tract Length Normalized (VTLN) Log-Mel features augmented with deltas and double deltas. The context window refers to the concatenation of the 20 feature frames surrounding the current frame. In fact, the original LSTM work in [4] looked at modeling a sequence of 20 consecutive short-term features. The more recent work in [10] unrolls stacked bidirectional LSTMs on a context window of short-term features and reports that this technique is superior to using a longer utterance-like contexts to unroll the LSTM. We could confirm the recipe from [10] and used it in all our experiments. We also observed that using long-term features for RNNs turned out to be worse than using short-term features.

Latest architectures try to unify CNNs and RNNs by stacking or fusing these layers. The architecture in [3] first feeds long-term features separately into stacked CNNs, RNNs, and a DNN and fuses the scores of the different layers for classification. A different approach is chosen in [11], where first long-term features are provided to stacked CNNs. Then the output of the stacked CNNs is augmented with short-term features and fed into stacked LSTMs. Finally the output of the stacked LSTMs is augmented with short-term features and fed into a DNN.

Here we choose a different approach to unify CNNs and RNNs by introducing a novel layer type which is simultaneously recurrent and convolutional but uses short-term features only. In more detail, we introduce a Bidirectional Gated Recurrent Convolutional Unit (BGRCU) which is a BGRU with gates using convolutional matrix operations instead of fully connected matrix multiplications. Our proposed modification makes the CNN maps recurrent in time.

The PyraMiD-LSTM and ConvLSTM models introduced in [12, 13] define (multidimensional) LSTMs using convolutional gates to perform image segmentation and weather forecasting. Our model differs from these approaches in multiple points. First, we apply our novel model for the first time to ASR while the PyramMiD-LSTM is used on a medical segmentation and the ConvLSTM is used on a weather forecasting task. Second, the structure of our model is different. We construct our model from the BGRU structure. That is different from the PyraMiD-LSTM and ConvLSTM which construct their model from the LSTM structure. The gating units of our BGRCU are composed of convolutions and max-pooling operations whereas the PyraMiD-LSTM and ConvLSTM use only convolutional operations. Our BGRCU model is bidirectional but the PyramidMiD-LSTM and ConvLSTM are unidirectional.

We find that after cross-entropy training our proposed BGRCU layers in combination with BGRU layers provide a 16-19% relative WER improvement over the DNN and a 14% relative improvement over a state-of-the-art CNN architecture on the DEV04F development and the RT04 test set. Also the combination of BGRCU with BGRU layers yields a 3% relative improvement over BGRU standalone architecture.

http://dx.doi.org/10.21437/Interspeech.2016-212

The rest of this paper is organized as follows. In Section 2 we describe the CNN, GRU, BGRU layers and introduce our BGRCU layer which unifies these models. The experimental setup is described in Section 3 while the result are shown in Section 4. Section 5 concludes the paper and discusses future work.

2. Different Layer Types

In this section we review convolutional neural networks, gated recurrent units and bidirectional gated recurrent units. We show how these different layer types can be unified into the bidirectional gated recurrent convolutional unit which is convolutional, bidirectional recurrent.

2.1. Fully Connected Layers

The input to a fully connected layer is a D dimensional observation $x \in \mathbb{R}^D$. The hidden vector y is calculated by the equation:

$$y = \sigma(Wx + b)$$

The symbol W denotes a matrix, b is a bias and σ is a non-linear activation, such as sigmoid or a rectified linear unit.

2.2. Convolutional Layers

The convolutional neural networks (CNNs) [9] have become popular in ASR. CNNs have been applied mostly to VTLN Log-Mel features augmented with deltas and double deltas.

The input to a one dimensional CNN operating in the frequency scale with C channels and input dimension D is a vector $x \in \mathbb{R}^{C \times D}$. Free parameters of the CNN consist of a weight matrix $W \in \mathbb{R}^{F \times C \times L}$ with F features maps of length L and a bias $b \in \mathbb{R}^F$. For the convolutional operation used in the CNN assume that the input has been zero padded, so that all overlaps between the feature maps and input can be considered in the convolution.

$$x_{c,j-\lfloor L/2 \rfloor} = \begin{cases} x_{c,j-\lfloor L/2 \rfloor}, & 1 \leq j - \lfloor L/2 \rfloor \leq D \\ 0, & \text{otherwise} \end{cases}$$

The main operation of the CNN is based on the local convolution $W * x + b \in \mathbb{R}^{F \times D + L - 1}$ which is calculated by equation:

$$W * x + b = \left(\sum_{c=1}^{C} \sum_{j=l}^{l+L} W_{f,c,j-l} x_{c,j-\lfloor L/2 \rfloor} + b_f \right)_{\substack{f=1,\ldots,F \\ l=1,\ldots,D+L-1}}$$

The hidden CNN vector $y \in \mathbb{R}^{F \times D + L - 1}$ is calculated by:

$$y = \sigma(W * x + b)$$

The symbol σ denotes the non-linearity of the CNN.

2.3. Max-pooling Layers

A max-pooling layer helps to remove variability from the convolutional layers, that exist due to speaking styles, channel distortions, etc. Specifically, each max-pooling unit receives activations from C channels, and outputs the maximum of the activations from these channels.

Maxpooling layers assume the same type of input vector $x \in \mathbb{R}^{C \times D}$ as the convolutional layers. A one dimensional max-pooling operation $\text{pool}(x) \in \mathbb{R}^{C \times D - L + 1}$ of length L is calculated by equation:

$$\text{pool}(x) = \left(\max_{j=l,\ldots,l+L-1} \{x_{c,j}\} \right)_{\substack{c=1,\ldots,C \\ l=1,\ldots,D-L+1}}$$

Notice, a convolution of length L grows the output filter dimension from D to $D + L - 1$ while the max-pooling operation of length L decreases the filter dimension from D to $D - L + 1$. If both operations are executed one after another with the same filter length the output filter length remains unchanged. This is an important detail to understand that also the dimension of all BGRCU gates remain unchanged.

2.4. Gated Recurrent Units

A Gated Recurrent Unit (GRU) was proposed by [6] to make each recurrent unit to adaptively capture dependencies of different time scales. Similarly to the LSTM, the GRU has gating units that modulate the flow of information inside the memory, however, without having a separate memory cells.

An input to a GRU in step t is a D-dimensional vector $x_t \in \mathbb{R}^D$. The hidden vector sequence $h_1^T := h_1, \ldots, h_T$ is calculated by iterating the following equations from $t = 1, \ldots, T$:

$$\begin{aligned} z_t &= \sigma(W_z x_t + U_z h_{t-1} + b_z) \\ r_t &= \sigma(W_r x_t + U_r h_{t-1} + b_r) \\ \overline{h}_t &= \sigma(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \\ h_t &= (1 - z_t) h_{t-1} + z_t \overline{h}_t \end{aligned}$$

The activation is composed of *update gate* z_t, *reset gate* r_t, *candidate gate* $\overline{h}_t$ and *output activation* h_t. $W_{(\bullet)}, U_{(\bullet)}$ and $b_{(\bullet)}$ denote appropriate sized matrices and biases. The symbol σ is the sigmoid activation, and $\odot$ is an elementwise multiplication.

2.5. Bidirectional Gated Recurrent Units

Bidirectional GRUs [5] process data in both directions with forward and backward hidden layers. Compared to the unidirectional case the number of free parameters doubles. The result of both directions is then concatenated in the output.

Let $\overrightarrow{h}_1^T$ be the *forward output* of the BGRU by processing the input sequence x_1^T through $t = 1, \ldots, T$ and let $\overleftarrow{h}_1^T$ be the corresponding *backward output* by processing the input sequence in reverse direction through $t = T, \ldots, 1$. The output h_1^T of the BGRU is the step-wise concatenation of the forward and backward output $h_t := (\overrightarrow{h}_t, \overleftarrow{h}_t)$.

2.6. Bidirectional Gated Recurrent Convolutional Units

The input to a Gated Recurrent Convolutional Unit (GRCU) with C channels and input dimension D is a sequence x_1^T of vectors $x_t \in \mathbb{R}^{C \times D}$. The hidden vector sequence h_1^T is computed by iterating the following equations from $t = 1, \ldots, T$:

$$\begin{aligned} z_t &= \sigma(W_z * x_t + \text{pool}(U_z * h_{t-1}) + b_z) \\ r_t &= \sigma(W_r * x_t + \text{pool}(U_r * h_{t-1}) + b_r) \\ \overline{h}_t &= \sigma(W_h * x_t + \text{pool}(U_h * (r_t \odot h_{t-1})) + b_h) \\ h_t &= (1 - z_t) h_{t-1} + z_t \overline{h}_t \end{aligned}$$

$W_{(\bullet)} \in \mathbb{R}^{F \times C \times L}$ and $U_{(\bullet)} \in \mathbb{R}^{F \times F \times L}$ are F features maps of length L. The max-pooling operations also have length L

and $b_{(\bullet)} \in \mathbb{R}^F$ are the biases. A GRCU uses the same gating structure as the BGRU but all fully-connected matrix multiplications are substituted with a convolutional matrix operation followed by a max-pooling operation with the same filter length. Thereby as pointed out in Section 2.2 all output gates will have the same dimensionality. Combining GRCU and bidirectional RNNs results in the bidirectional GRCU (BGRCU), which is defined analogous to the BGRU.

3. Experimental Setup

We perform experiments to learn the behavior of our novel BGRCU model in combination also with other models, like the BGRU, DNN and CNN layers. All neural network models are evaluated as hybrid acoustic model for ASR. The acoustic models are trained on 50 hours of data from the 1996 and 1997 English Broadcast News Speech Corpora. Results are reported on the DEV04F development and RT04 test sets. Unless otherwise indicated, we use a 40 dimensional vocal tract length normalized warped VTLN Log-Mel features augmented with delta + double-delta.

3.1. The Baseline System

The baseline GMM system was trained using the recipe from [14], which is briefly described below. The raw acoustic features are 19-dimensional PLP features with speaker-based mean, variance, and vocal tract length normalization. Temporal context is included by splicing 9 successive frames of PLP features into supervectors, then projecting to 40 dimensions using linear discriminant analysis (LDA). The feature space is further diagonalized using a global semi-tied covariance (STC) transform. The GMMs are speaker-adaptively trained, with a feature-space maximum likelihood linear (fMLLR) transform estimated per speaker in training and testing. Following maximum-likelihood training of the GMMs, featurespace discriminative training (fBMMI) and model-space discriminative training are done using the boosted maximum mutual information (BMMI) criterion. At test time, unsupervised adaptation using regression tree MLLR is performed. The GMM system with 5k quinphone states and 150k diagonal covariance Gaussians has been used to generate the alignment for the hybrid neural network cross entropy training.

3.2. Architectures

In pilot experiments we determined suitable hyperparameter sizes for the different layer types. The resulting stacked CNN, BGRCU, BGRU and DNN layer configurations can be seen in Figure 1. For the configuration of the stacked 1D-CNN and BGRCU layers we chose the common configuration in two parts [9]. The first CNN layer has 128 and the second 256 feature maps with a filter length of 9 and 4. The corresponding max-pooling layers have a pooling length of 3. The BGRU block consists of four layers with each 512 forward and backward hidden nodes. The dropout layers in all experiments have a dropout-rate of 0.1.

The dimension of the last layer of a CNN and BGRCU is large. Therefore we add a linear layer to reduce the output dimension to 256 before passing it to a DNN or BGRU. In our experiments we found that this does not affect the error rate.

The stacked two dimensional CNN combined with a DNN is denoted by (2D-CNN+DNN). The 2D-CNN+DNN has a configuration similar to [9] only here we use more output targets.

3.3. Neural Network Training

For neural network training we use the Theano wrapper Keras [15, 16] connected to the decoding and feature extraction pipeline of the IBM speech recognition system [14]. In the training procedure, the truncated back-propagation through time (BPTT) learning algorithm is used. Before the next epoch the utterances are sorted by length and uniformly sampled. Each utterance is split into subsequences of 21 frames with an overlap of 10 frames. For each epoch the starting point of the utterance sampling is randomly shifted by an offset of 0 to 9. In experiments we found this procedure to give an additional performance boost to the BRNNs. The recurrent models are trained on minibatches of size 250, which are shuffled randomly after each epoch. Each position in the minibatch is composed of a subsequence of 21 frames coming from a different utterance. The bidirectional neural networks are unrolled on the sequence of 21 frames with all alignment targets presented in training. In decoding on the other hand similar to [10] the bidirectional neural network is unrolled on the context window of 20 frames surrounding the current frame. The last layer of a stack of bidirectional layers in decoding only returns the center frame output to the remaining non-recurrent network. Non-recurrent neural networks are trained on a minibatch of 250 composed of the features corresponding to the model, but otherwise follow the same recipe.

All architectures are cross-entropy trained on the GMM alignment using the gradient descent optimizer ADAM [17] with a minibatch size of 250 and learning rate schedule as follows. After one pass through the data, the cross entropy loss is measured on a held-out set. If the cross entropy loss does not improve sufficiently the learning rate is reduced by a factor of 0.85 and the clock of ADAM is reinitialized for the next epoch. Training stops after 50 epochs. The choice of the overall best model is determined in recognition steps of 5 epochs on the development set and in after. After determining the best model on the development set this model is evaluated on the test set.

4. Experimental Results

In this section, we explore the proposed BGRCU from Section 2 in combination with other layer types described in Figure 1. We also examine what happens when the BGRCU is exchanged with a non-recurrent one dimensional CNN. All neural network models are evaluated as hybrid acoustic models in the HMM system. Results are reported on both the DEV04F development and the RT04 test set. In the following the notation A+B means that architecutre B is stacked on top of architecture A with respect to the architectures defined in Figure 1.

Table 1 shows the performance of the different neural network architectures which we compare to DNN, 1D-CNN, 2D-CNN+DNN and BGRU HMM systems. The table indicates that the hybrid 2D-CNN+DNN system offers a 2-5% relative improvement over the DNN. However, the BGRU is far better than the all DNN and CNN based architectures. The BGRU offers between a 14-16% relative improvement over the DNN, and a 11-12% relative improvement over the 2D-CNN+DNN.

The 1D-CNN performs worse than all other combinations resulting in WERs beyond 20%.

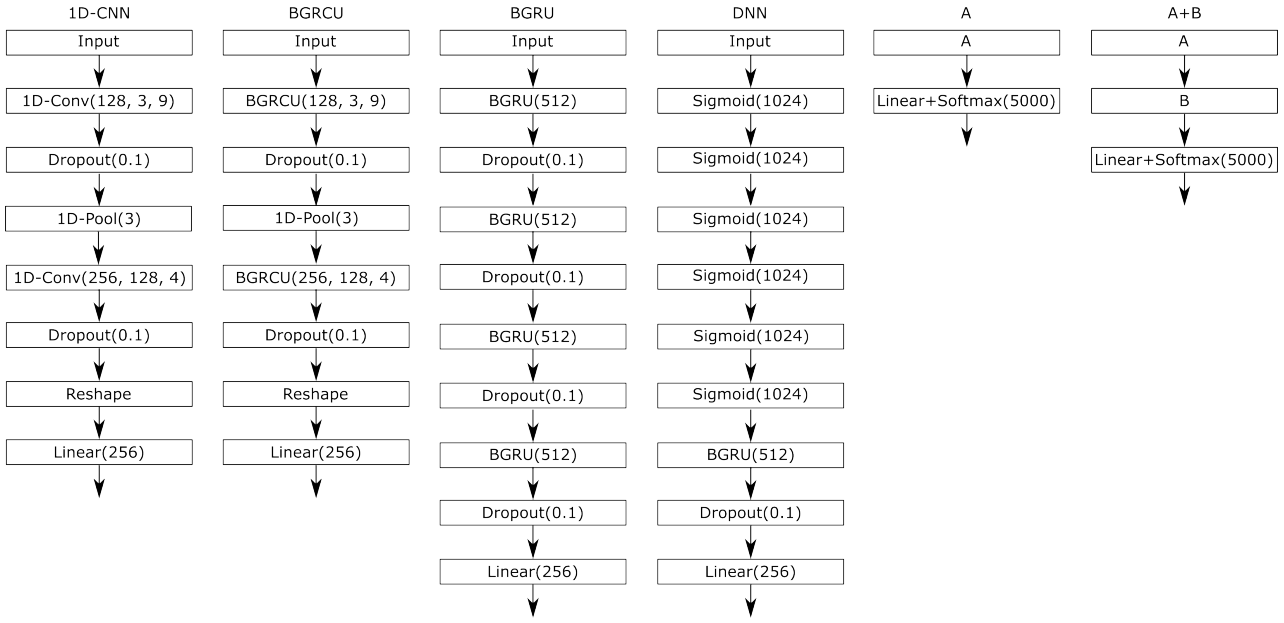

Figure 1: *Architecture of the stacked CNN, BGRCU, BGRU and DNN layers used in the experiments. The corresponding hyperparameters of the layers in this figure have the following meaning: 1D-CONV(#feature maps, #input channels, filter length) is a one dimensional CNN, BGRCU(#feature maps, #input channels, filter length) is a one dimensional BGRCU, 1D-POOL(pooling length) one dimensional max-pooling layer, BGRU(#hidden nodes), DROPOUT(dropout rate).*

Table 1: *The performance of layer combinations measured as a function of the WER[%] on the* BN50H *dataset. The parameter size of the corresponding models is also given in Millions (M).*

	WER[%]		
model	model size	DEV04F	RT04
DNN	10 M	16.2	15.8
2D-CNN+DNN	66 M	15.8	15.0
BGRU	24 M	14.2	13.6
1D-CNN	4 M	26.8	25.4
1D-CNN+DNN	14 M	24.3	22.1
1D-CNN+BGRU	32 M	14.1	13.6
BGRCU	11 M	15.4	14.7
BGRCU+DNN	14 M	15.6	14.5
BGRCU+BGRU	37 M	**13.8**	**13.2**

Also the 1D-CNN+DNN combination does not significantly improve the performance over the 1D-CNN. Surprisingly, the 1DCNN+BGRU combination achieves a big improvement over the 1D-CNN standalone model. In this combination, it performs slightly better than the BGRU architecture on the development set, but achieves the same performance on the test set. It seeems, BGRUs are able to connect the one dimensional frequency CNNs with the temporal context which 2D-CNN+DNNs have by definition.

Also BGRCUs have this temporal context by definition due to their recurrent bidirectional nature. In our experiments, BGRCUs provide a 4-7% relative improvement over DNNs and about 2% relative improvement over 2D-CNNs. The BGRCU+DNN combination performs similar compared to the BGRCU standalone architecture. The biggest improvement is achieved by the BGRCU+BGRU combination which provides a 16-19% relative improvement over the DNN, about 14% relative improvement over the 2D-CNN+DNNs, and also outperforms the BGRU and 1D-CNN+BGRU architectures by a 3% relative WER improvement.

5. Conclusions

In this paper, we introduced the Bidirectional Gated Recurrent Convolutional Unit (BGRCU) which unifies convolutional neural networks and bidirectional gated recurrent units into one model. We showed that our novel layer outperforms the corresponding one and two dimensional convolutional neural networks. In addition when stacking bidirectional gated recurrent units on top of the BCGRU this combination outperforms all other architectures. On a 50 hour broadcast news task this combination shows a 16-19% relative WER improvement over DNNs, a 14% relative WER improvement over two dimensional CNNs and a 3% relative WER improvement over bidirectional gated recurrent units. For future research we will investigate sequence training of our proposed model and use two dimensional convolutions.

6. Acknowledgment

The Research leading to these results has received funding from the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense U.S. Army Research Labotory (DoD/ARL) contract no. W911NF-12-C-0012. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes not withstanding 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.

7. References

[1] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep Neural Networks for Acoustic Modeling in Speech Recognition," *Signal Processing Magazine*, 2012.

[2] T. N. Sainath, B. Kingsbury, G. Saon, H. Soltau, A. Mohamed, G. E. Dahl, and B. Ramabhadran, "Deep Convolutional Neural Networks for Large-scale Speech Tasks," *Neural Networks*, vol. 64, pp. 39–48, 2015.

[3] G. Saon, H. Soltau, A. Emami, and M. Picheny, "Unfolded Recurrent Neural Networks for Speech Recognition," in *INTERSPEECH*. ISCA, 2014, pp. 343–347.

[4] H. Sak, A. W. Senior, and F. Beaufays, "Long Short-term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling," in *INTERSPEECH*. ISCA, 2014, pp. 338–342.

[5] A. Graves, N. Jaitly, and A. Mohamed, "Hybrid Speech Recognition with Deep Bidirectional LSTM," in *ASRU*. IEEE, 2013, pp. 273–278.

[6] K. Cho, B. Van Merriënboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734.

[7] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014.

[8] R. Józefowicz, W. Zaremba, and I. Sutskever, "An Empirical Exploration of Recurrent Network Architectures," in *ICML*, ser. JMLR Proceedings, vol. 37. JMLR.org, 2015, pp. 2342–2350.

[9] T. N. Sainath, B. Kingsbury, A. Mohamed, G. E. Dahl, G. Saon, H. Soltau, T. Beran, A. Y. Aravkin, and B. Ramabhadran, "Improvements to Deep Convolutional Neural Networks for LVCSR," in *ASRU*. IEEE, 2013, pp. 315–320.

[10] A. Rahman Mohamed, F. Seide, D. Yu, J. Droppo, A. Stolcke, G. Zweig, and G. Penn, "Deep Bi-directional Recurrent Networks over Spectral Windows," in *Proc. IEEE Automatic Speech Recognition and Understanding Workshop*. IEEE Institute of Electrical and Electronics Engineers, December 2015, pp. 78–83.

[11] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, Long Short-term Memory, fully connected Deep Neural Networks." IEEE, 2015, pp. 4580–4584.

[12] M. F. Stollenga, W. Byeon, M. Liwicki, and J. Schmidhuber, "Parallel Multi-Dimensional LSTM, with Application to Fast Biomedical Volumetric Image Segmentation," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 2998–3006.

[13] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. WOO, "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 802–810.

[14] H. Soltau, G. Saon, and B. Kingsbury, "The IBM Attila Speech Recognition Toolkit," in *SLT*. IEEE, 2010, pp. 97–102.

[15] 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 Expression Compiler," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, Jun. 2010, oral Presentation.

[16] F. Chollet, "Keras," https://github.com/fchollet/keras, 2015.

[17] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *CoRR*, vol. abs/1412.6980, 2014.