LSTM TIME AND FREQUENCY RECURRENCE FOR AUTOMATIC SPEECH RECOGNITION

Jinyu Li, Abdelrahman Mohamed, Geoffrey Zweig, and Yifan Gong

Microsoft Corporation, One Microsoft Way, Redmond, WA 98052
{ jinyli, asamir, gzweig, ygong}@microsoft.com

ABSTRACT

Long short-term memory (LSTM) recurrent neural networks (RNNs) have recently shown significant performance improvements over deep feed-forward neural networks (DNNs). A key aspect of these models is the use of time recurrence, combined with a gating architecture that ameliorates the vanishing gradient problem. Inspired by human spectrogram reading, in this paper we propose an extension to LSTMs that performs the recurrence in frequency as well as in time. This model first scans the frequency bands to generate a summary of the spectral information, and then uses the output layer activations as the input to a traditional time LSTM (T-LSTM). Evaluated on a Microsoft short message dictation task, the proposed model obtained a 3.6% relative word error rate reduction over the T-LSTM.

Index Terms— *LSTM, RNN, time and frequency*

1. INTRODUCTION

Recently, significant progress has been made in automatic speech recognition (ASR) thanks to the application of deep neural networks (DNNs) [1][2][3][4][5][6]. Unlike in the 1990s, today's DNN systems often contain tens of millions of parameters and are more powerful than their counterparts two decades ago [7][8] in modeling speech signals. DNNs, however, only consider information in a fixed-length sliding window of frames and thus cannot exploit long-range correlations in the signal. Recurrent neural networks (RNNs), on the other hand, can encode sequence history in their internal state, and thus have the potential to predict phonemes based on all the speech features observed up to the current frame. Unfortunately, simple RNNs, depending on the largest eigenvalue of the state-update matrix, may have gradients which either increase or decrease exponentially over time. Thus, the basic RNN is difficult to train, and in practice can only model short-range effects. Long short-term memory (LSTM) RNNs [9][10] were developed to overcome these problems. LSTM-RNNs use input, output and forget gates to achieve a network that can maintain state and propagate gradients in a stable fashion over long spans of time. These networks have been shown to outperform DNNs on a variety of ASR tasks [11][12][13] [14][15][16]. All previously proposed LSTMs use a recurrence along the time axis to model the temporal patterns of speech signals, and we call them T-LSTMs in this paper. The main contribution of this paper is the proposal of a two-level network where the first level performs recurrence along the frequency axis, and the second performs time recurrence. We term this the frequency-time LSTM or F-T-LSTM.

Our model is inspired by the way people read spectrograms. Note that in common practice, log-filter-bank features are often used as the input to the neural-network-based acoustic model [19][20]. In standard systems, the log-filter-bank features are independent of one-another, i.e. switching the positions of two filter-banks won't affect the performance of the DNN or LSTM. However, this is not the case when a human reads a spectrogram: a human relies on both patterns that evolve on time, and frequency, to predict phonemes. Switching the positions of two filter-banks will destroy the frequency-wise patterns. Our model addresses this phenomenon by explicitly modeling the frequency-wise evolution of spectral patterns. Evaluated on a Microsoft internal short message dictation task, the proposed F-T-LSTM obtained 3.6% relative word error rate (WER) reduction from the T-LSTM.

The rest of the paper is organized as follows. In Section 2, we briefly introduce LSTMs and then we present the proposed model which combines frequency LSTM and time LSTM in Section 3. We differentiate the proposed method from the convolutional LSTM DNN (CLDNN) [16] and multi-dimensional RNN [17][18] in Section 4. Experimental evaluation of the algorithm is provided in Section 5. We summarize our study and draw conclusions in Section 6.

2. THE LSTM-RNN

An RNN is fundamentally different from the feed-forward DNN in that the RNN does not operate on a fixed window of frames; instead, it maintains a hidden state vector, which is recursively updated after seeing each time frame. The internal state encodes the history all the way from the beginning of an utterance up to the last input, and can thus potentially model much longer span effects than a fixed-window DNN. In other words, an RNN is a dynamic system and is more general than the DNN which performs a static input-output transformation. The inclusion of internal states enables RNNs to represent and learn long-range sequential dependencies.

However, the simple RNN suffers from the vanishing/exploding gradient problem [21] when the error signal is back-propagated through time. This problem is well handled in the LSTM-RNNs through the use of the following four components:

- *Memory units*: these store the temporal state of the network;
- *Input gates*: these modulate the input activations into the cells;
- *Output gates*: these modulate the output activations of the cells ;
- *Forget gates*: these adaptively reset the cell's memory.

Taken together as in Figure 1 below, these four components are termed a LSTM cell.

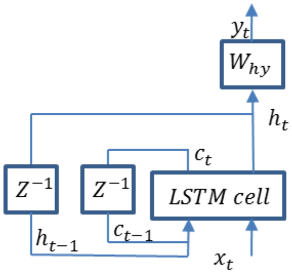

Figure 1. Architecture of LSTM-RNNs with one recurrent layer. Z^{-1} is a time-delay node.

Figure 1 depicts the architecture of an LSTM-RNN with one recurrent layer. In LSTM-RNNs, in addition to the past hidden-layer output $\boldsymbol{h}_{t-1}$, the past memory activation $\boldsymbol{c}_{t-1}$ is also an input to the LSTM cell.

This model can be described as:

$$\boldsymbol{i}_j^l = \sigma\left(\boldsymbol{W}_{xi}^l \boldsymbol{x}_j^l + \boldsymbol{W}_{hi}^l \boldsymbol{h}_{j-1}^l + \boldsymbol{W}_{ci}^l \boldsymbol{c}_{j-1}^l + \boldsymbol{b}_i^l\right), \quad (1)$$

$$\boldsymbol{f}_j^l = \sigma\left(\boldsymbol{W}_{xf}^l \boldsymbol{x}_j^l + \boldsymbol{W}_{hf}^l \boldsymbol{h}_{j-1}^l + \boldsymbol{W}_{cf}^l \boldsymbol{c}_{j-1}^l + \boldsymbol{b}_f^l\right), \quad (2)$$

$$\boldsymbol{c}_j^l = \boldsymbol{f}_j^l .* \boldsymbol{c}_{j-1}^l + \boldsymbol{i}_j^l .* \tanh\left(\boldsymbol{W}_{xc}^l \boldsymbol{x}_j^l + \boldsymbol{W}_{hc}^l \boldsymbol{h}_{j-1}^l + \boldsymbol{b}_c^l\right), \quad (3)$$

$$\boldsymbol{o}_j^l = \sigma\left(\boldsymbol{W}_{xo}^l \boldsymbol{x}_j^l + \boldsymbol{W}_{ho}^l \boldsymbol{h}_{j-1}^l + \boldsymbol{W}_{co}^l \boldsymbol{c}_j^l + \boldsymbol{b}_o^l\right), \quad (4)$$

$$\boldsymbol{h}_j^l = \boldsymbol{o}_j^l .* \tanh\left(\boldsymbol{c}_j^l\right), \quad (5)$$

where $\boldsymbol{i}_j^l$, $\boldsymbol{o}_j^l$, $\boldsymbol{f}_j^l$, and $\boldsymbol{c}_j^l$ denote the activation vectors of input gate, output gate, forget gate, and memory cell at the l-th layer and time j, respectively. $\boldsymbol{h}_j^l$ is the output of the LSTM cells at layer l and time j. $\boldsymbol{W}$ terms denote different weight matrices. For example, $\boldsymbol{W}_{xi}^l$ is the weight matrix from the cell input to the input gate at the l-th layer. $\boldsymbol{b}$ terms are the bias terms (e.g., $\boldsymbol{b}_i^l$ is the bias of input gate at layer l). ".*" denotes element wise multiplication.

In [13], a LSTM with an additional projection layer prior to the output (termed LSTMP) was proposed to reduce the computational complexity of LSTM. A projection layer is applied to $\boldsymbol{h}_j^l$ as

$$\boldsymbol{r}_j^l = \boldsymbol{W}_{hr}^l \boldsymbol{h}_j^l$$

And then $\boldsymbol{h}_{j-1}^l$ in Eqs (1)--(4) is replaced by $\boldsymbol{r}_{j-1}^l$.

3. FREQUENCY-TIME LSTM-RNN

In this section, we propose a frequency-time LSTM (F-T-LSTM) which combines frequency LSTM with time LSTM as shown in Figure 2. We first use a frequency LSTM (F-LSTM) to scan the frequency bands so that frequency-evolving information is summarized by the output of the F-LSTM. The formulation of the F-LSTM is the same as that of the T-LSTM except that the index j now stands for frequency steps instead of time steps. Then we can take the outputs from all F-LSTM steps and use them as the input to T-LSTM to do time analysis in the traditional way.

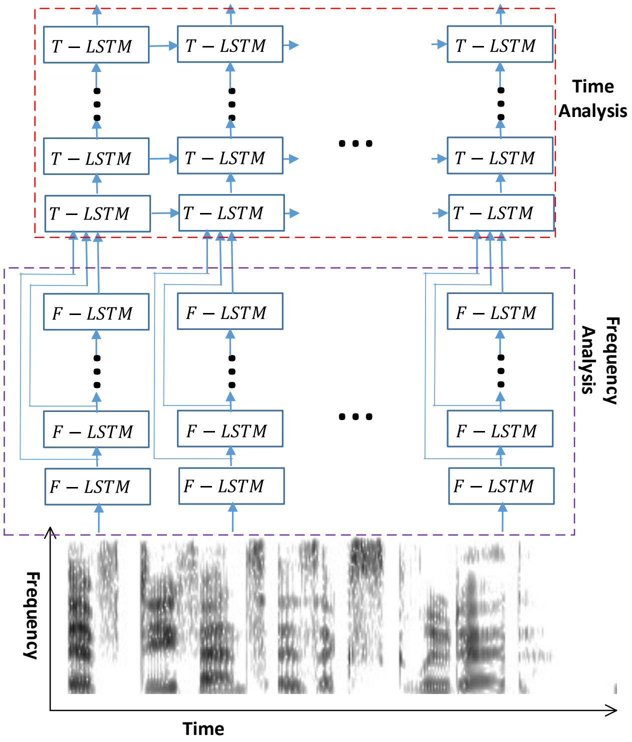

Figure 2: A frequency- time LSTM-RNN which scans the frequency axis first for frequency analysis and then scans the time axis for time analysis.

The detailed F-LSTM processing for *each* time step is described as follows.

- Divide total N log-filter-banks at current time into M overlapped chunks and each chunk contains B log-filter-banks. There are C log-filter-banks overlapped between adjacent chunks. Here we have the relationship $M = \frac{N-C}{B-C}$. An extreme case is $C=0$ where there is no overlapped log-filter-bank. In such a case, $M = \frac{N}{B}$.
- Use the M overlapped chunks as the frequency steps of F-LSTM and generate the output of $\boldsymbol{h}_m, m = 0 \ldots M-1$.

- Merge $\boldsymbol{h}_m, m = 0 \ldots M-1$. into a super-vector $\boldsymbol{h}$ which can be considered as a trajectory of frequency patterns at current time. Then use $\boldsymbol{h}$ as the input to a T-LSTM with multiple layers.

Figure 3 shows an example setup of the F-LSTM used in our experiments. The input at each frame consists of a 40 dimensional vector of log-filter-bank values at the current time *t*. We divide the 40 log-filter-bank channels into 33 overlapped chunks with each chunk containing 8 log-filter-banks. This results in 7 log-filter-banks of overlap between adjacent chunks (C=7). Therefore, the first F-LSTM cell takes eight inputs: the log-filter-banks from 0 to 7, and the second F-LSTM cell takes the log-filter-banks from 1 to 8, and so on. The *m*-th F-LSTM cell generates outputs $\boldsymbol{h}_m$, which will be passed into the *m+1*-th F-LSTM cell. Finally, $\boldsymbol{h}_m, m = 0 \ldots M-1$ (*M*=33 in this example) will be concatenated as the input to a T-LSTM.

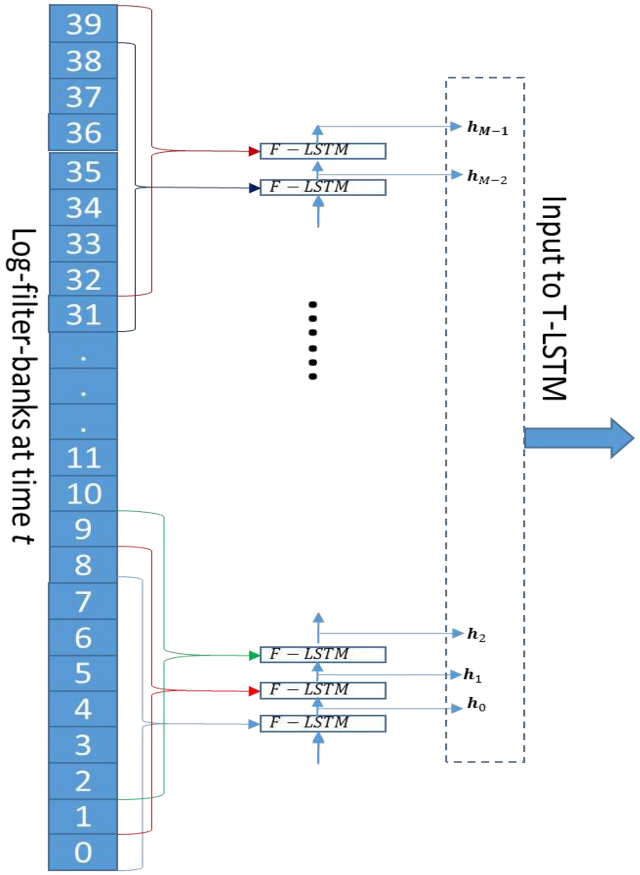

Figure 3: An example setup of F-LSTM.

4. RELATION TO PRIOR WORK

In this section, we first discuss the difference between our proposed F-T-LSTM and the convolutional LSTM DNN (CLDNN) [16] which combines CNNs, LSTMs, and DNNs together. The CLDNN first uses a CNN [22][23] to reduce the spectral variation, and then the output of the CNN layer is fed into a multi-layer LSTM to learn the temporal patterns. Finally, the output of the last LSTM layer is fed into several fully connected DNN layers for the purpose of classification.

The key difference between the proposed F-T-LSTM and the CLDNN is that the F-T-LSTM uses frequency recurrence with the F-LSTM, whereas the CLDNN uses a sliding convolutional window for pattern detection with the CNN. While the sliding window achieves some invariance through shifting, it is not the same as a fully recurrent network. The two approaches both aim to achieve invariance to input distortions, but the pattern detectors in the CNN maintain a constant dimensionality, while the F-LSTM can perform a general frequency warping.

The proposed F-T-LSTM performs 1-D recurrence over the frequency axis and then performs 1-D recurrence over the time axis. This is different from the concept of multidimensional processing which has been proved very successful in the handwriting recognition tasks [17][18] and outperformed the traditional handwriting systems that use CNNs [22][23] as the feature extractor. To summarize, the T-F-LSTM works on multidimensional space separately with simplicity while the multidimensional RNN [17][18] works jointly on multidimensional space with more powerful modeling.

5. EXPERIMENTS AND DISCISSIONS

In this section, we use a Windows Phone short message dictation task to evaluate the proposed method. The training data consists of 60 hours of transcribed US-English audio. The test set consists of 3 hours of data from the same Windows Phone task. The audio data is 16k HZ sampled, recorded in mobile environments using Windows phones. The vocabulary has around 130k words and the LM has around 6.6M ngrams (up to trigram). All experiments were conducted using the computational network toolkit (CNTK) [24], which allows us to build and evaluate various network structures efficiently without deriving and implementing complicated training algorithms. All the models were trained to minimize the frame-level cross-entropy criterion.

The input to the baseline CD-DNN-HMM system consists of 40-dimensional log-filter-bank features. We augment these feature vectors with 5 frames of context on either side (5-1-5). The DNN has 5 hidden layers, each with 2048 sigmoid units. Both the baseline and LSTM systems use 1812 tied-triphone states or senones.

The baseline T-LSTMP is modeled after that in [13]. It has four T-LSTMP layers: each has 1024 hidden units and the output size of each T-LSTM layer is reduced to 512 using a linear projection layer. There is no frame stacking, and the output HMM state label is delayed by 5 frames as in [13]. When training T-LSTMP, the backpropagation through time (BPTT) [25] step is 20.

We built the F-T-LSTM with a single F-LSTM that scans the log-filter-banks and three T-LSTMP layers. The number of parameters of the F-T-LSTM is between the numbers of parameters of the three- and four- layer LSTMPs. To generate the input to the F-LSTM, we use the example setup in Section 3 by dividing the 40 log-filter-bank channels into 33 overlapped chunks with each chunk containing 8 log-filter-banks. The F-LSTM has 24 memory cells.

In Table 1, we compare the WERs of a DNN, T-LSTM, and F-T-LSTM. The T-LSTM is clearly better than the DNN due to its temporal modeling power. With both the frequency and temporal modeling, the F-T-LSTM is better than the 4-layer T-LSTM, with 3.6% relative WER reduction.

Table 1: WER comparison of DNN, T-LSTM, and F-T-LSTM

Model	WER (%)
DNN	21.84
3-layer T-LSTMP	20.79
4-layer T-LSTMP	20.38
F-LSTM (24 cells)+3-layer T-LSTMP	19.64

We investigate the impact of different cell numbers in the F-LSTM in Table 2. When the number of cells is very small, e.g., 8, the power of F-LSTM is very limited with only a slight improvement over the T-LSTM. However, when the number of cells becomes 24, the F-LSTM shows its advantage because the memory cells are powerful enough to store the frequency patterns. When we increase the number of cells to 48, there is no further improvement.

Table 2: Impact of cell numbers in F-LSTM

Model	WER (%)
F-LSTM (8 cells)+3-layer T-LSTMP	20.19
F-LSTM (24 cells)+3-layer T-LSTMP	19.64
F-LSTM (48 cells)+3-layer T-LSTMP	19.81

In all the aforementioned experiments, we have not stacked multiple frames of log-filter-banks as the input to F-T-LSTM. This decision is made based on our previous experience with T-LSTMs, where we found that stacking multiple frame inputs doesn't have any benefit, and [13] also doesn't have the frame stacking. In Table 3, we compare the setup with and without multiple-frame stacking. Stacking *N* frames means that every chunk now has *8*N* log-filter-banks. When stacking 11 frames, we predict the center frame's label. As shown in Table 3, it doesn't provide any benefit to WER by stacking 11 frames as the input to F-LSTM.

Table 3: Comparison of F-T-LSTM with and without stacking frame inputs

Model	Number of Input Frames	WER (%)
F-LSTM (24 cells)+3-layer T-LSTMP	1	19.64
F-LSTM (24 cells)+3-layer T-LSTMP	11	20.08
F-LSTM (48 cells)+3-layer T-LSTMP	1	19.81
F-LSTM (48 cells)+3-layer T-LSTMP	11	20.01

6. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a FT--LSTM architecture that scans both the time and frequency axis to model the evolving patterns of the spectrogram. The F-T-LSTM first uses an F-LSTM to performs a frequency recurrence that summarizes frequency-wise patterns. This is then fed into a T-LSTM. The proposed F-T-LSTM obtained a 3.6% relative WER reduction from the traditional T-LSTM on a short message dictation task. We have shown that as long as the number of memory cells in the F-LSTM is reasonable, the F-T-LSTM can achieve good performance. We also evaluated the impact of stacking multiple frames as the input to F-LSTM, and found that it is best to simply present the frames one at a time

Several research issues will be addressed in the future to further increase the effectiveness of the algorithm presented in this paper. First, we will compare the performance of F-T-LSTMs with CLDNNs to better understand their relative advantages. Second, we want to explore architectural variants of the F-T-LSTM. For example, we will examine whether frequency overlapping of the input to F-LSTM is necessary. Third, we will move the input of F-LSTM from log-filter-banks directly to log-spectrum. There are studies showing that directly working of log-spectrum can be beneficial to DNN [26]. By applying the F-LSTM directly on log-spectrum, we can naturally remove the hand-crafted filter-banks, and automatically learn the frequency patterns that benefit the recognizer. Fourth, in [27] it is shown that CNNs can consistently provide advantages over DNNs in mismatched training-test conditions. It is interesting to see whether the frequency recurrence brought by the F-LSTM can be more helpful in the mismatched conditions. Last and most importantly, we will advance our study by proposing a multidimensional LSTM with a simplified structure which performs recurrence over the time and frequency axes jointly [28]. We term it the time-frequency LSTM (TF-LSTM). We will compare TF-LSTM and F-T-LSTM in [28] by using a much larger ASR task. It will be shown that F-T-LSTM is still effective on that larger ASR task.

REFERENCES

[1] F. Seide, G. Li, and D. Yu, “Conversational speech transcription using context-dependent deep neural networks,” in *Proc. Interspeech*, pp. 437-440, 2011.

[2] N. Jaitly, P. Nguyen, A. Senior, and V. Vanhoucke, “An application of pretrained deep neural networks to large vocabulary conversational speech recognition,” in *Proc. Interspeech*, 2012.

[3] T. N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novak, A.-R. Mohamed, “Making deep belief networks effective for large vocabulary continuous speech recognition,” in *Proc. ASRU*, pp. 30-35, 2011.

[4] G. E. Dahl, D. Yu, L. Deng, and A. Acero, “Large vocabulary continuous speech recognition with context-dependent DBN-HMMs,” in *Proc. ICASSP*, pp. 4688-4691, 2011.

[5] A. Mohamed, G. E. Dahl, and G. Hinton, “Acoustic modeling using deep belief networks,” *IEEE Trans. Audio Speech and Language Process.*, vol. 20, no. 1, pp. 14-22, Jan. 2012.

[6] L. Deng, J. Li, J.-T. Huang et. al. “Recent advances in deep learning for speech research at Microsoft,” in *Proc. ICASSP*, 2013.

[7] H. Bourlard and N. Morgan. *Connectionist speech recognition - A Hybrid approach*, Kluwer Academic Press, 1994.

[8] S. Renals, N. Morgan, H. Boulard, M. Cohen, and H. Franco. “Connectionist probability estimators in HMM speech recognition,” *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 1, pp. 161–174, 1994.

[9] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[10] A. Gers, J. Schmidhuber, and F. Cummins. “Learning to forget: Continual prediction with LSTM,” *Neural Computation*, vol. 12, no. 10, pp. 2451-2471, 2000.

[11] A. Graves, A. Mohamed, G. Hinton. “Speech recognition with deep recurrent neural networks,” in *Proc. ICASSP*, 2013.

[12] A. Graves, N. Jaitly, A. Mohamed. “Hybrid speech recognition with deep bidirectional LSTM,” in *Proc. ASRU*, 2013.

[13] H. Sak, A. Senior, F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proc. Interspeech*, 2014.

[14] H. Sak, O. Vinyals, G. Heigold, A. Senior, E. McDermott, R. Monga, M. Mao, "Sequence discriminative distributed training of long short-term memory recurrent neural networks," in *Proc. Interspeech*, 2014.

[15] X. Li and X. Wu, “Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition,” in *Proc. ICASSP*, 2015.

[16] T. N. Sainath, O. Vinyals, A. Senior and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Proc. ICASSP*, 2015.

[17] A. Graves, S. Fernández, J. Schmidhuber, “Multi-dimensional recurrent neural networks,” in *ICANN*, pp. 549-558, 2007.

[18] A. Graves and J. Schmidhuber, “Offline handwriting recognition with multidimensional recurrent neural networks,” *Advances in Neural Information Processing Systems*, pp. 545-552, 2009.

[19] A. Mohamed, G. Hinton, and G. Penn, “Understanding how deep belief networks perform acoustic modelling,” in *Proc. ICASSP*, pp. 4273–4276, 2012.

[20] J. Li, D. Yu, J. T. Huang, and Y. Gong. "Improving wideband speech recognition using mixed-bandwidth training data in CD-DNN-HMM," *Proc. IEEE Spoken Language Technology Workshop*, pages 131–136, 2012.

[21] Y. Bengio, P. Simard, and P. Frasconi. “Learning long-term dependencies with gradient descent is difficult,” *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157-166, 1994.

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

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

[24] D. Yu, A. Eversole, M. Seltzer, et. al., "An introduction to computational networks and the computational network toolkit," *Microsoft Technical Report MSR-TR-2014-112*, 2014.

[25] Jaeger, H. “Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the echo state network approach,” *GMD Report 159*, GMD—German National Research Institute for Computer Science, 2002.

[26] T. N. Sainath, B. Kingsbury, A. Mohamed and B. Ramabhadran, "Learning filter banks within a deep neural network framework," in *Proc. ASRU*, 2013.

[27] J.-T. Huang, J. Li, and Y. Gong, An analysis of convolutional neural networks for speech recognition, in *Proc. ICASSP*, 2015.

[28] J. Li, A. Mohamed, G. Zweig, and Yifan Gong, “Exploring multidimensional LSTMs for large vocabulary ASR,” submitted to *Proc. ICASSP*, 2016.