Training Deep Bidirectional LSTM Acoustic Model for LVCSR by a Context-Sensitive-Chunk BPTT Approach

Kai Chen[1,2], *Zhi-Jie Yan*[2], *Qiang Huo* [2]

[1]University of Science and Technology of China, Hefei, China
[2]Microsoft Research Asia, Beijing, China
{v-kachen, zhijiey, qianghuo}@microsoft.com

Abstract

This paper presents a study of using deep bidirectional long short-term memory (DBLSTM) as acoustic model for DBLSTM-HMM based large vocabulary continuous speech recognition (LVCSR), where a context-sensitive-chunk (CSC) backpropagation through time (BPTT) approach is used to train DBLSTM by splitting each training sequence into chunks with appended contextual observations, and a (possibly overlapped) CSCs based decoding method is used for recognition. Our approach makes mini-batch based training on GPU more efficient and reduces the latency of DBLSTM-based LVCSR from a whole utterance to a short chunk. Evaluations have been made on Switchboard-I benchmark task. In comparison with epochwise BPTT training, our method can achieve about three times speed-up on a single GPU card. In comparison with a highly optimized DNN-HMM system trained by a frame-level cross entropy (CE) criterion, our CE-trained DBLSTM-HMM system achieves relative word error rate reductions of 9% and 5% on Eval2000 and RT03S testing sets, respectively.

Index Terms: Long short-term memory, DBLSTM, DNN, LVCSR, Context sensitive chunk, BPTT.

1. Introduction

As a special type of artificial neural networks, recurrent neural networks (RNNs) can model contextual information of a sequence (e.g., [1, 2]), and its bidirectional version (BRNN) [3] provides a framework to utilize future and past information simultaneously at each time instance. These advantages led to its early success in small-scale automatic speech recognition (ASR) tasks (e.g., [1, 2]), but it was difficult to scale up for larger ASR tasks due to the intricacy in training. Recently, deep neural networks (DNNs), which stack multiple feed-forward layers on top of each other, have been successfully used as acoustic models for large vocabulary continuous speech recognition (LVCSR) (e.g., [4, 5] and the references therein). Because DNNs can only provide limited temporal modeling power by feeding a fixed-size sliding window of feature vectors, more powerful model for sequence signal such as RNNs, especially a long short-term memory (LSTM) version (e.g., [6–8]), have attracted the attention of many speech research groups again.

LSTM replaces the neurons in recurrent layers of standard RNN with carefully designed memory blocks to ease training [6–8]. LSTM and its bidirectional version (BLSTM) [9] have been applied to ASR (e.g., [9, 10]) and handwriting recognition (HWR) (e.g., [11]). Combined with connectionist temporal classification (CTC) output layer and trained from unsegmented sequence data by using CTC training [12], it was demonstrated in [11] that BLSTM-based system outperforms a state-of-the-art hidden Markov model (HMM) based system for both offline and online HWR. The same technique has also been applied to ASR and achieved promising results on TIMIT task [12]. Following the success of DNNs for acoustic modeling, (B)LSTM layers can also be stacked on top of each other to build deep (B)LSTMs for ASR [13]. More recently, DBLSTM with CTC training has been used to build an end-to-end ASR system without leveraging any Gaussian mixture model HMM (GMM-HMM) system [14]. Another way to use D(B)LSTM for ASR is to combine it with HMM in a hybrid mode [15]. In [16], a hybrid DBLSTM-HMM system gives state-of-the-art results on TIMIT task and outperforms a DNN-HMM system on Wall Street Journal (WSJ) task. However, both TIMIT and WSJ are small to medium scale ASR tasks. In [17, 18], hybrid DLSTM-HMM systems achieve state-of-the-art results with both frame-level cross entropy (CE) training and sequence-level discriminative training on the LVCSR tasks of voice search and short message dictation. In this paper, we study how DBLSTM-HMM system works for large-scale LVCSR tasks.

Applying DBLSTM to LVCSR faces several challenges. Firstly, DBLSTMs are often trained with an epochwise backpropagation through time (BPTT) algorithm (e.g., [9, 19]), where network states of all time steps of a sequence need be stored. Nowadays, GPUs are widely used in deep learning by leveraging massive parallel computations via mini-batch based training. When applied to DBLSTM (e.g., [20]), GPU's limited memory restricts the number of sequences that can be used in a mini-batch, especially for LVCSR tasks with long training sequences and large model sizes. Secondly, full sequence dependence at each time step makes DBLSTM unsuitable for low-latency recognition, because a delay of a whole utterance will be incurred. In [21], a context-sensitive-chunk (CSC) BPTT algorithm, which splits each sequence into chunks with appended contextual observations, is proposed to deal with similar challenges for offline HWR. Since CSC-BPTT can parallelize more chunks and recognition delay is only a short chunk, this technique can also be applied to LVCSR. Another motivation to use CSC-BPTT to train speech DBLSTM is similar to HWR, namely speech feature vectors of a phone are mostly influenced by several phones before and after it. Inspired by ensemble method, we also propose a decoding method with overlapped CSCs to improve recognition accuracy further.

The remainder of this paper is organized as follows. In Section 2, we present our approach. In Section 3, we report experimental results and analyze the effect of different factors of our approach. Finally, we conclude the paper in Section 4.

Kai Chen contributed to this work when he worked as an intern in the Speech Group of Microsoft Research Asia, Beijing, China.

2. Our Approach

2.1. Hybrid DBLSTM-HMM LVCSR System

Our LVCSR system is based on a DBLSTM-HMM framework as in a neural-network/HMM hybrid system [15], where DBLSTM acts as acoustic model. Frame-level state targets are provided by a forced alignment given by a GMM-HMM system. The activation function of DBLSTM's output layer is softmax, whose unit number is the total number of HMM states. Different from a DNN-HMM hybrid system, which appends an acoustic context window of frames to either side of the one being classified at each time step, DBLSTM only feeds a single frame at a time since it is able to store past and future states internally. In training, a frame-level cross-entropy objective function is minimized. In recognition, the state-dependent scores derived from the DBLSTM are combined with HMM state transition probabilities and language model (LM) scores to determine the recognition result by using an in-house decoder for LVCSR.

2.2. DBLSTM Training

2.2.1. *Context-sensitive-chunk BPTT*

Due to its bidirectional interdependence among frames in a sequence, DBLSTM is often trained with epochwise BPTT (e.g., [9,19,20]). Given a sequence, this method must accumulate the history of activations in the network over the entire sequence, along with the history of errors, then back-propagation is carried out to calculate gradients. After that, weights are updated accordingly. In order to accelerate training by GPU, mini-batch technique can be used as in DNN training, but the mini-batch here has to be defined over sequences (e.g., [20]). Therefore, for long sequences and large networks, the memory size of GPU restricts the number of parallel sequences in a mini-batch so the acceleration is quite limited.

A simple but effective solution to the limited acceleration problem is to use chunk BPTT [22], which splits sequences into chunks of particular length, and treats these chunks as isolated sequences. However, the lost interdependence among chunks results in performance degradation when the chunk size is small. In [21], a CSC-BPTT algorithm was proposed to address this issue. As shown in Fig. 1, given a sequence, it is firstly split into (possibly overlapped) chunks of fixed length N_c, then N_l previous frames are appended before each chunk as left context and N_r future frames after it as right context. For the first/last chunk of each sequence, no left/right contextual frames are appended. This kind of chunk is called context-sensitive-chunk (CSC). The appended frames only act as context and gives no output, so no error signals will be generated during training, as illustrated in Fig. 2. CSCs from all the sequences are pooled together and randomized before every sweep of training. Because CSCs are treated as isolated sequences, this technique increases the number of parallel chunks in a mini-batch, leading to faster training. Moreover, if the length of CSC is short enough, such trained DBLSTM can be applied to low-latency decoding, while it is impossible for DBLSTM trained by traditional epochwise BPTT because of full sequence dependence at each time step.

A CSC with N_c chunk frames, N_l left and N_r right contextual frames is denoted as "N_l-N_c+N_r" for simplicity. So, for chunk BPTT, its chunk of length N_c can be represented as "0-N_c+0", and for epochwise BPTT, the chunk configuration is denoted as "0-Full+0" because a chunk here is the full sequence. It is noted that CSC-BPTT can also be used to train DLSTM as we demonstrated in [21]. In our implementation, the per-time-step input could be a single frame or a concatenation of a local

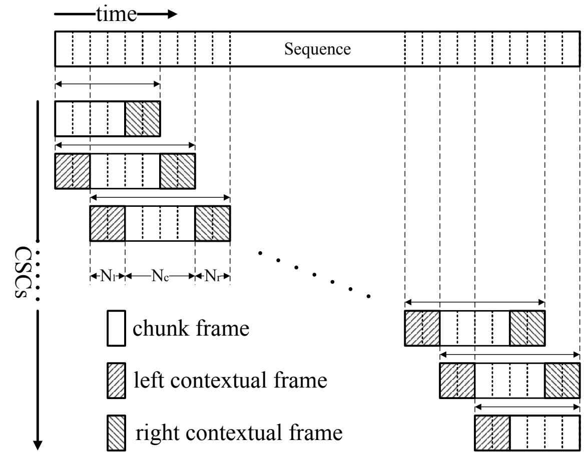

Figure 1: *Illustration of context-sensitive-chunk (CSC).*

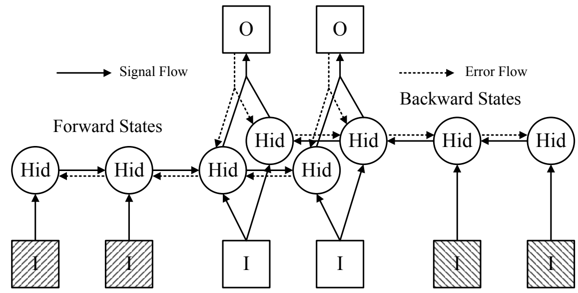

Figure 2: *Illustration of forward and backward procedure of CSC-BPTT to train BLSTM.*

window of feature vector sequence. The unfolded RNN in [23] is a special case of CSC, whose configuration is "N_l-1+0" and per-time-step input consists of current frame and several future frames, but is much more expensive for training and decoding than other configurations with $N_c >> 1$.

2.2.2. *Learning rate scheduling*

Tuning learning rate, especially the scheduling of reducing the learning rates during training, is critical to the performance of deep learning (e.g., [24]). In this study, we have used a semi-automatic learning rate scheduling mechanism similar to the "Newbob+/Train" method in [25]. Our learning rate scheduling approach works as follows: Firstly, a sub-set of the full training data set is reserved as a validation set. It differs from the conventional definition of validation because those data would still be used in training. Secondly, DBLSTM training starts with an empirically chosen initial learning rate. The frame error rate (FER) on the validation set is evaluated once after the model has been updated a fixed number of times. Lastly, the learning rate is adjusted according to the results of adjacent evaluations. If the FER improvement between current and previous evaluations is above a threshold τ, the learning rate stays unchanged; else if the error rate improvement between previous and antepenultimate evaluations is not above τ, the learning rate stays unchanged; otherwise, the learning rate will be multiplied by a fixed factor ρ ($0 < \rho < 1$). The above second rule makes our learning rate scheduling different from that in [25], which is confirmed in our experiments to be helpful in the final stage of learning when shrinking the learning rate does not lead to enough FER improvement. In this stage, conventional scheduling rules would continuously cut the learning rate and force the learning to converge prematurely. Our heuristic essentially prevents the learning rate from shrinking so the network is still able to learn something in the final training stage.

2.3. CSC-based Decoding

Given a CSC-BPTT trained DBLSTM, an unknown utterance for recognition should also be split into CSCs of the same configuration as used in training. In a preliminary study, we found that no difference was made for recognition accuracy whether the training sequences are split into CSCs with or without overlap, therefore we use no-overlap scheme in this paper. For a testing utterance, we could have options. If computational cost is a concern, then the testing utterance can be split into non-overlapping CSCs as in [21]; otherwise, the testing utterance can be split into overlapped CSCs to achieve higher recognition accuracy. If overlapped CSCs are used, the t-th frame of observation may be evaluated N_t times, whose scores are denoted as $\{y_t^{(1)}, \cdots, y_t^{(N_t)}\}$. The final score could be obtained by taking an arithmetic mean $y_t = \frac{1}{N_t}\sum_{i=1}^{N_t} y_t^{(i)}$ or a geometric mean $y_t = \sqrt[N_t]{\prod_{i=1}^{N_t} y_t^{(i)}}$. The effect of these two averaging methods will be compared in Section 3.2.3. Because different scores of a given frame are predicted from several neighboring chunks, the averaged score leverages more frames as context, which leads to better recognition accuracy as we will show in next section.

3. Experiments

3.1. Experimental Setup

Switchboard-I conversational telephone speech transcription task [26] is used for evaluation. About 300 hours of speech from 520 speakers are used in training. About 2 hours of speech from 2000 Hub5 evaluation (Eval2000) and about 6.3 hours of speech from Spring 2003 NIST rich transcription set (RT03S) are used in testing. For front-end spectral feature extraction, 13-dimensional PLP features along with their time derivatives up to third order are extracted every 10 ms to form a 52-dimensional raw feature vector. Windowed mean and variance normalization is then performed, and a 39×52 HLDA transform is estimated afterwards to reduce the feature dimension to 39 such that GMM-HMM acoustic models are trained with 39-dimensional feature vectors. A set of speaker independent GMM-HMMs which contain 9,304 decision-tree tied triphone HMM states is estimated using maximum likelihood criterion. This GMM-HMM set is used to perform forced alignment on both training and testing sets to get the frame-level state labels. In DNN and DBLSTM training, these labels will be used as ground truth in training and state classification targets in testing.

A 7-hidden-layer DNN is chosen as the baseline system. Each hidden layer contains 2,048 rectified linear units (ReLUs), resulting in approximately 45 million parameters. Input dimension of this network is 572, and the input is a concatenation of 11 frames of 52-dimensional raw feature vectors. The frame-level cross-entropy objective function is minimized via mini-batch SGD with L2 constraint regularization [27], and 7 sweeps of training data are conducted with a carefully tuned learning rate schedule.

All DBLSTMs have 5 hidden layers. Each hidden layer has 512 memory cells (256 for forward and 256 for backward states) with forget gate and peephole connections [16], resulting in approximately 11 million parameters. These models will be optimized by epochwise, chunk or CSC BPTT. Considering the memory size of a single Nvidia Tesla K20Xm GPU card, mini-batch sizes are set to be 8 sequences for epochwise BPTT and 64 chunks for chunk and CSC BPTT. We implement CSC-BPTT for DBLSTM training based on open-source Currennt toolkit [20]. During training process, 30hr data is sampled from training set as validation set. The FER improvement threshold τ is set to be 2% and learning rate tuning factor ρ is set to be 0.5. Evaluation on validation set will be conducted after a sweep of data and 7 sweeps are used for all the models.

Table 1: Performance (in %) comparison on testing sets of DBLSTM-HMM systems trained by epochwise and CSC BPTT methods with DNN-HMM system.

Config.	Eval2000		RT03S			
	FER	WER	FER	WER		
				TOTAL	FSH	SWB
0-Full+0	29.7	14.8	44.7	22.9	18.6	27.0
21-64+21	29.6	**14.7**	44.5	**22.8**	**18.6**	**26.7**
DNN-HMM	39.9	16.2	55.0	24.0	19.5	28.2

Experiments are conducted for various CSC configurations. For each configuration, initial learning rate is carefully tuned and the one leading to the best validation set FER is selected to decode the test sets. Both FER and Word Error Rate (WER) are used to evaluate different models. The vocabulary, pronunciation lexicon, trigram language models are the same as that in [28, 29].

3.2. Experimental Results

3.2.1. Comparison with DNN-HMM

We train two DBLSTMs with epochwise and CSC BPTT, respectively. The CSC configuration is "21-64+21", the number of overlapped frames in CSC-based decoding is 48, and the score averaging method is arithmetic mean. According to the feature extraction method we used, 64 frames is equivalent to about 6 phones. Table 1 shows a performance comparison on Eval2000 and RT03S testing sets. DBLSTMs trained with epochwise and CSC BPTT achieve similar recognition performance. Compared with CE-trained DNN, DBLSTM trained with CSC BPTT improves FER significantly, which leads to a relative WER reduction (WERR) of 9.3% and 5.0% on Eval2000 and RT03S testing sets, respectively. It is noted that the DNN-HMM system here achieves a WER of 16.2% on Eval2000, which is better than the WER of 16.4% achieved by a DNN-HMM system with sigmoid hidden units as reported in [28, 29] on the same task. Since the number of DBLSTM parameters is only a quarter of DNN's, DBLSTM is apparently a better acoustic model for LVCSR than DNN.

3.2.2. Effect of CSC configurations

Experiments are conducted to identify suitable CSC configurations by varying the chunk size and the number of contextual frames. In recognition, CSC-based decoding without overlap is used. The effect of CSC configurations on testing performance (in %) of DBLSTM-HMM systems is compared in Table 2. It is observed that for CSCs with $N_c = 64$, a size of contextual frames ranging from 16 to 32 works well, while chunk size N_c has a relatively big influence. All the DBLSTM-HMM systems trained by CSC-BPTT perform better than the DNN-HMM system. When no contextual frames are appended to chunks (i.e.,

Table 2: Effect of CSC configurations on testing performance (in %) of DBLSTM-HMM systems. CSC-based decoding without overlap is used.

Config.	Eval2000		RT03S			
	FER	WER	FER	WER		
				TOTAL	FSH	SWB
21-42+21	30.3	15.1	45.5	23.8	19.4	27.9
21-64+21	30.1	15.0	45.2	23.0	18.7	27.0
32-64+32	29.7	14.9	44.9	23.2	18.9	27.2
16-64+16	30.2	15.2	45.3	23.3	19.0	27.4
0-64+0	33.5	16.2	48.6	24.6	19.9	29.0
16-32+16	30.9	15.4	46.3	23.9	19.5	28.0

Table 3: Effect of the number of overlapped frames in CSC-based decoding on testing performance (in %) of DBLSTM-HMM systems trained by CSC-BPTT with the configuration of "21-64+21". Arithmetic mean is used for score averaging.

# of overlap frames	Eval2000		RT03S			
	FER	WER	FER	WER		
				TOTAL	FSH	SWB
0	30.1	15.0	45.2	23.0	18.7	27.0
16	29.9	14.9	44.8	23.0	18.8	26.9
32	29.7	14.8	44.7	22.9	18.7	26.8
48	29.6	14.7	44.5	22.8	18.6	26.7

"0-64+0"), CSC-BPTT becomes chunk-BPTT. It is observed that the DBLSTM-HMM system trained by chunk-BPTT performs worse than the DNN-HMM system. Another interesting observation is that "16-32+16" configuration contains the same number of frames with "0-64+0" per chunk, but performs much better, which demonstrates clearly that appended frames are important to compensate for the loss of contextual information.

3.2.3. *Effect of CSC-based decoding*

Table 3 shows the effect of the number of overlapped frames on testing performance (in %) of DBLSTM-HMM systems trained by CSC-BPTT with the configuration of "21-64+21", where arithmetic mean is used for score averaging in CSC-based decoding. When the number of overlapped frames is 48, most frames are evaluated 4 times, while for cases of 16 and 32 overlapped frames, each frame is evaluated at most twice. It is observed that overlap decoding improves the performance slightly because much contextual information has been contained in CSCs. Table 4 compares two averaging methods for CSC-based decoding with overlapped CSCs. No difference is observed, therefore we recommend to use the simpler arithmetic mean method for score averaging. For comparison, Table 5 shows the effect of the number of overlapped frames in chunk-based decoding on testing performance (in %) of DBLSTM-HMM systems trained by chunk-BPTT with the configuration of "0-64+0". It is observed that overlap decoding improves the performance significantly, but overall, DBLSTM trained by chunk-BPTT performs worse than its CSC-BPTT counterpart.

3.2.4. *Training speed-up*

In our experiments, the elapsed time per sweep for epochwise BPTT is about 59.9 hr, while for CSC-BPTT with "21-64+21" configuration, the time is shortened to 21.4 hr, namely a 2.8x speed-up is achieved. If "16-64+16" configuration is used, the elapsed time will become 18.9 hr, resulting in a 3.2x speed-up.

Table 4: Testing performance (in %) comparison of averaging methods for CSC-based decoding with overlapped CSCs. The number of overlapped frames is 48. The DBLSTM is trained by CSC-BPTT with the configuration of "21-64+21".

Averaging methods	Eval2000		RT03S			
	FER	WER	FER	WER		
				TOTAL	FSH	SWB
Arithmetic	29.6	14.7	44.5	22.8	18.6	26.7
Geometric	29.6	14.7	44.5	22.8	18.7	26.7

Table 5: Effect of the number of overlapped frames in chunk-based decoding on testing performance (in %) of DBLSTM-HMM systems trained by chunk-BPTT with the configuration of "0-64+0".

#overlap frames	Eval2000		RT03S			
	FER	WER	FER	WER		
				TOTAL	FSH	SWB
0	33.5	16.2	48.6	24.6	19.9	29.0
16	31.1	15.3	46.5	23.8	19.1	28.1
32	30.7	15.1	45.9	23.0	18.5	27.3
48	29.8	15.0	45.1	22.7	18.2	27.0

4. Conclusion and Discussion

We have demonstrated that DBLSTM-HMM can outperform DNN-HMM on the Switchboard-I benchmark task if both DBLSTM and DNN are trained by minimizing a frame-level CE criterion. It is well-known that bidirectional contextual information plays an important role in acoustic modeling [30], and DBLSTM offers an elegant way of modeling and leveraging the bidirectional contextual information. Given the co-articulation effect in a speech utterance, speech features of each phone are mostly influenced by several phones before and after it, therefore there is no need to model the whole utterance by a DBLSTM directly. This insight motivates us to propose to use a DBLSTM to model a short chunk, a CSC-BPTT method to train the DBLSTM, and a CSC-based decoding method for DBLSTM-HMM based LVCSR. Compared with DNN with a fixed-size window of a feature vector sequence as input, DBLSTM can learn automatically the effective length of input feature vector sequence for acoustic modeling. Compared with standard epochwise BPTT method, our CSC-BPTT method makes mini-batch based training on GPU more efficient, which can achieve about three times speed-up on a single GPU card in our experiments. Furthermore, CSC-based decoding makes low-latency DBLSTM-based LVCSR possible by incurring only a delay of a short chunk rather than a whole utterance. Taking a CSC configuration of "21-64+21" for example, if no frame overlap is used in CSC-based decoding, the delay will be 85 frames or 850 ms, while the computational complexity will only be about 1.27 times of that using the DBLSTM trained by epochwise BPTT, which can only be used for decoding until the whole utterance is observed. In comparison with a highly optimized DNN-HMM system trained by a frame-level CE criterion, our CE-trained DBLSTM-HMM system achieves relative WERRs of 7.4% and 4.2% on Eval2000 and RT03S testing sets, respectively. If a CSC-based decoding with 48 overlapped frames is used, the above relative WERRs will become 9.3% and 5.0% respectively with a cost of increased computational complexity. As future work, a comparison of DBLSTM-HMM and DNN-HMM with sequence-level discriminative training must be made.

5. References

[1] A. J. Robinson, "An application of recurrent nets to phone probability estimation," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp.298-305, 1994.

[2] T. Robinson, M. Hochberg, and S. Renals, "The use of recurrent neural networks in continuous speech recognition," in C.-H. Lee, F. Soong, and K. Paliwal (Eds.), *Automatic Speech and Speaker Recognition*, pp.233-258, 1996.

[3] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 9, no. 11, pp.2673-2681, 1997.

[4] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. 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 Processing Magazine*, vol. 29, no. 16, pp.82-97, 2012.

[5] D. Yu and L. Deng, *Automatic Speech Recognition: A Deep Learning Approach*, Springer, 2015.

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

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

[8] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks," *Journal of Machine Learning Research*, vol. 3, pp.115-143, 2003.

[9] A. Graves, and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks* vol. 18, no. 5, pp.602-610, 2005.

[10] A. Graves, S. Fernández and J. Schmidhuber, "Bidirectional LSTM networks for improved phoneme classification and recognition," *Proc. ICANN-2005*, Springer LNCS 3697, pp.799-804.

[11] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Transactions on PAMI*, vol. 31, no. 5, pp.855-868, 2009.

[12] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," *Proc. ICML-2006*, pp.369-376.

[13] A. Graves, A. R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," *Proc. ICASSP-2013*, pp.6645-6649.

[14] A. Graves, and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," *Proc. ICML-2014*, pp.1764-1772.

[15] H. Bourlard and N. Morgan, *Connectionist Speech Recognition: a Hybrid Approach*, Kluwer, Norwell, MA, 1993.

[16] A. Graves, N. Jaitly, and A. R. Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," *Proc. ASRU-2013*, pp.273-278.

[17] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," *Proc. INTERSPEECH-2014*, pp.338-342.

[18] H. Sak, O. Vinyals, G. Heigold, A. Senior, E. McDermott, R. Monga, and M. Mao, "Sequence discriminative distributed training of long short-term memory recurrent neural networks," *Proc. INTERSPEECH-2014*, pp.1209-1213.

[19] R. J. Williams, and J. Peng, "An efficient gradient-based algorithm for on-line training of recurrent network trajectories," *Neural Computation*, vol. 2, no. 4, pp.490-501, 1990.

[20] F. Eyben, J. Bergmann, and F. Weninger, *CURRENNT: CUDA-enabled Machine Learning Library For Recurrent Neural Networks*, http://sourceforge.net/projects/currennt/.

[21] K. Chen, Z.-J. Yan, and Q. Huo, "A context-sensitive-chunk BPTT approach to training deep LSTM/BLSTM recurrent neural networks for offline handwriting recognition," *Proc. ICDAR-2015*.

[22] P. Doetsch, M. Kozielski, and H. Ney, "Fast and robust training of recurrent neural networks for offline handwriting recognition," *Proc. ICFHR-2014*, pp.279-284.

[23] G. Saon, H. Soltau, A. Emami, and M. Picheny, "Unfolded recurrent neural network for speech recognition," *Proc. INTERSPEECH-2014*, pp.343-347.

[24] A. Senior, G. Heigold, M. A. Ranzato, and K. Yang, "An empirical study of learning rates in deep neural networks for speech recognition," *Proc. ICASSP-2013*, pp.6724-6728.

[25] S. Wiesler, A. Richard, R. Schlüter, and H. Ney, "Mean-normalized stochastic gradient for large-scale deep learning," *Proc. ICASSP-2014*, pp.180-184.

[26] J. J. Godfrey, Edward. C. Holliman, and J. McDaniel, "Switchboard: telephone speech corpus for research and development," *Proc. ICASSP-1992*, pp.I-517-520.

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

[28] F. Seide, G. Li, and D. Yu, "Conversational speech transcription using context-dependent deep neural networks," *Proc. INTERSPEECH-2011*, pp.437-440.

[29] Z.-J. Yan, Q. Huo, and J. Xu, "A scalable approach to using DNN-derived features in GMM-HMM based acoustic modeling for LVCSR," *Proc. INTERSPEECH-2013*, pp.104-108.

[30] Q. Huo and C. Chan, "A study on the use of bi-directional contextual dependence in Markov random field-based acoustic modelling for speech recognition," *Computer Speech and Language*, vol. 10, pp.95-105, 1996.