Interspeech 2018
2-6 September 2018, Hyderabad

Improving CTC-based Acoustic Model with Very Deep Residual Time-delay Neural Networks

Sheng Li[1], *Xugang Lu*[1], *Ryoichi Takashima*[1], *Peng Shen*[1], *Tatsuya Kawahara*[1,2], *and Hisashi Kawai*[1]

[1]National Institute of Information and Communications Technology, Kyoto, Japan
[2]Kyoto University, Kyoto, Japan
sheng.li@nict.go.jp

Abstract

Connectionist temporal classification (CTC) has shown great potential in end-to-end (E2E) acoustic modeling. The current state-of-the-art architecture for a CTC-based E2E model is based on a deep bidirectional long short-term memory (BLSTM) network that provides frame-wise outputs estimated from both forward and backward directions (BLSTM-CTC). Since this architecture can lead to a serious time latency problem in decoding, it cannot be applied to real-time speech recognition tasks. Considering that the CTC label of one current frame can only be affected by a few neighboring frames, we argue that using BLSTM traversing on a whole utterance from both directions is not necessary. In this paper, we use a very deep residual time-delay (VResTD) network for CTC-based E2E acoustic modeling (VResTD-CTC). The VResTD network provides frame-wise outputs with local bidirectional information without needing to wait for the whole utterance. Speech recognition experiments on Corpus of Spontaneous Japanese were carried out to test our proposed VResTD-CTC and the state-of-the-art BLSTM-CTC model. Comparable performance was obtained while the proposed VResTD-CTC does not suffer from the decoding time latency problem.

Index Terms: Speech recognition, acoustic model, connectionist temporal classification (CTC), very deep residual network

1. Introduction

The connectionist temporal classification (CTC) framework [1] is an effective end-to-end (E2E) framework [2] for speech recognition. The CTC modeling technique greatly simplifies the acoustic modeling pipelines. No frame-level labels or initial GMM-HMM systems are needed anymore. Built on top of the deep bidirectional long short-term memory (BLSTM) recurrent neural networks, CTC models achieve state-of-the-art performance on speech recognition tasks [3]. However, BLSTM traverses a whole utterance from both directions to estimate the frame-wise outputs. This leads to serious time latency and makes the model impractical for online speech recognition (especially on our SprinTra WFST decoder [4]).

The BLSTM model can be improved using a context-sensitive-chunk BLSTM (CSC-BLSTM) [5]. This method chops a whole sentence into several overlapping chunks and controls the delay time by appending several contextual chunks (both past and future). A latency-controlled BLSTM (LC-BLSTM) [6] accelerates the training and decoding speed by optimizing the calculation of each chunk. For the same purpose, Xue et al. [7] proposed other structures: forward approximation and a backward simple recurrent neural network BLSTM (FABSR-BLSTM).

More direct methods focus on replacing the BLSTM network with other no-latency structures. Using a unidirectional LSTM model with a similar parameter size [8] or a state-of-the-art Grid LSTM [9] instead is a very effective strategy. Feed-forward sequential memory neural networks (FSMN) [10], a time-delay neural network (TDNN) [11, 12], a residual memory network (RMN) [13], and a layer-wise context expansion and attention (LACEA) network [14] can learn bidirectional long-term dependency without recurrent feedback and thus have very good potential for integration with CTC training.

Other methods predict the future context before recognition using recurrent neural networks [15] or encoder-decoder neural networks enhanced with attention mechanism [16, 17]. These methods show a promising future research direction.

When calculating the frame-level CTC label output, the predicted label of a frame is mostly affected by its neighboring short frame trunks (coarticulation effect [18]). For this reason, processing the whole utterance using the BLSTM-CTC is not necessary. In this paper, we propose a very deep residual time-delay CTC neural network (VResTD-CTC). In our method, a very deep residual structure is used to enhance the conventional TDNN during CTC training.

The rest of this paper is organized as follows. Section 2 briefly reviews related work. Section 3 describes our proposed model's structure. Section 4 presents experiments. Conclusions and future works are given in Section 5.

2. Related Work

2.1. Connectionist Temporal Classification (CTC)

The CTC training criterion is intensively used in E2E [2] acoustic modeling, which focuses on solving the sequence labeling problem between variable-length speech frame inputs and label outputs (phone character, syllable, etc.).

Output based on a CTC-trained network is a frame-level token sequence called a path (denoted as p). With multiple hypotheses, many paths are generated where a blank symbol (ϕ as no label output) is included and inserted between CTC labels. These paths are mapped into label sequences by removing the massive blank symbols detected during the decoding and the duplication of identical labels. One label sequence $\boldsymbol{Z}$ is mapped to multiple CTC label paths (denoted as $Path(\boldsymbol{Z})$). The likelihood of $\boldsymbol{Z}$ can be evaluated as a sum of the probabilities of its corresponding CTC paths:

$$P(\boldsymbol{Z}|\boldsymbol{X}) = \sum_{p \in Path(\boldsymbol{Z})} P(p|\boldsymbol{X}), \tag{1}$$

where $\boldsymbol{X}$ is the speech frame sequence in the whole utterance and p is a CTC path corresponding to label sequence $\boldsymbol{Z}$. By differentiating Eq. 1, we can backpropagate training errors and further update the network parameters based on a forward-backward procedure [1]. The weighted finite state transducer (WFST) [19, 3] is used for decoding with CTC models.

LF-MMI training [20] has been applied on various acoustic models, and it has shown promising results [21]. However, these models need frame-level labels for training, which are often estimated using GMM-HMM system, and thus they are not E2E systems.

2.2. Modeling Temporal Context Information in Feedforward Neural Network without Recurrent Feedback

As mentioned in Section 1, TDNN [22] and its various forms [12, 20] can learn long-term dependency without using recurrent feedback and thus might be integrated with CTC training as an alternative to BLSTM.

A feedforward sequential memory network (FSMN) [10, 16], which can also learn long-term dependency in sequential data without recurrent structure, has memory blocks (usually global vectors) to encode the activations for each hidden layer. Since it also uses a time-delay operation to capture the signal at a certain time step, we can consider it a special TDNN.

2.3. From Deep to Very Deep Structure

In the speech recognition field, the very deep residual networks also achieve significant performance [23, 24, 25, 26, 13, 14] over conventional DNNs. The residual structure [27] is a simplified version of the highway structure [28, 6]. The signals can bypass the transformation across multiple layers from residual connections. We can train much deeper time-delay networks equipped with residual connections in a similar way.

3. Proposed CTC Model based on Very Deep Residual Time-Delay Neural Network

In this section, we introduce our proposed **VResTD-CTC**, which is a very deep residual neural network that enhances the conventional TDNN in CTC training.

Figure 1 is the general architecture of the proposed **VResTD-CTC**. The network has two kinds of residual blocks stacked together:

1. Conventional residual blocks (**ResBlock**): Each ResBlock has several layers connected with activations and a residual-skip [1]. A time-delay operation is not used. The stacked ResBlocks are located close to the input layer and transform the speech feature to a higher level representation (Table 2).
2. Time-delay residual blocks (**TDResBlock**): We integrate a time-delay operation (bidirectional)[2] to the TDResBlock. Two global memory blocks (past and future) encode the output signal after every time-delay operation. The residual connection passes the frame of the current time-step across five layers (Table 2).

In the VResTD-CTC network, the entire input sequence of the l-th time-delay hidden layer can be represented as $\boldsymbol{H^l}$ = (h_1^l, h_2^l, h_3^l, ... , h_T^l). Any h_t^l has the following processing

[1]The residual-skips use linear projections for dimensional matching (Table 2).

[2]Triplet $\{-t_i, 0, +t_i\}$ denotes the frames at three time-steps ($-t_i$, 0 and $+t_i$), which have been subsampled from time-steps ranging from $-t_i$ to $+t_i$, following [12]. $-$, $+$, and 0 indicate the past, future, and current time-steps.

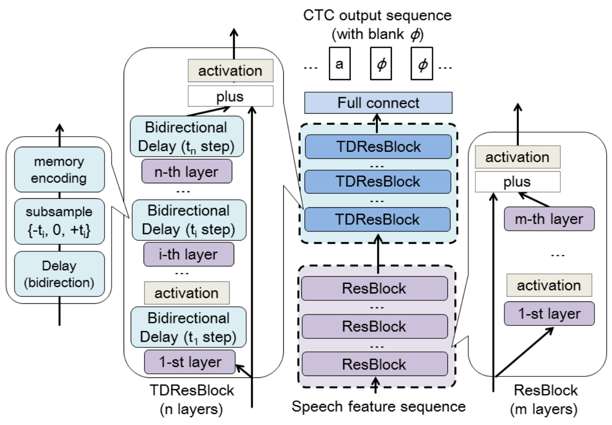

Figure 1: *General network architecture of **VResTD-CTC**: Memory block encodes output signal after every time-delay and subsampling operation by default*

pipeline:

$$h_t^l \xrightarrow{[W^l:b^l]} \tilde{h}_t^l \xrightarrow[subsampling]{time-delay} \tilde{\boldsymbol{H}}_t^l \xrightarrow[encoding]{memory} e_t^l \xrightarrow{ReLU} h_t^{l+1}.$$

First, it is linearly transformed by standard weight matrix W^l and bias b^l for layer l:

$$\tilde{h}_t^l = W^l(h_t^l) + b^l, \tag{2}$$

Then a time-delay operation captures a (N_1^l+1+N_2^l)-length context $\tilde{\boldsymbol{H}}_t^l$ at the l-th time-delay hidden layer for $\tilde{h}_t^l$. $\tilde{\boldsymbol{H}}_t^l = (\tilde{h}_{t-N_1^l}^l, ..., \tilde{h}_t^l, ..., \tilde{h}_{t+N_2^l}^l)$. With subsampling, $\tilde{\boldsymbol{H}}_t^l \approx (\tilde{h}_{t-N_1^l}^l, \tilde{h}_t^l, \tilde{h}_{t+N_2^l}^l)$. N_1^l and N_2^l are the window sizes of the past and future context (N_1^l+1+$N_2^l \leq T$). Following [13], we use symmetric context window ($N_1^l = N_2^l$). We also slightly tuned the past and future contexts. No benefit was identified (Sec. 4.3.2).

The past context ($\tilde{h}_{t-N_1^l}^l$) and future context ($\tilde{h}_{t+N_2^l}^l$) are encoded with transformations individually and summed up with current feature ($\tilde{h}_t^l$) before activation function.

$$e_t^l = \underbrace{a_{N_1^l}^l \odot \tilde{h}_{t-N_1^l}^l}_{subsampled-past} + \tilde{h}_t^l + \underbrace{c_{N_2^l}^l \odot \tilde{h}_{t+N_2^l}^l}_{subsampled-future}, \tag{3}$$

where the encoding weight for the past context is denoted as $a_{N_1^l}^l$ and the encoding weight for the future context is denoted as $c_{N_2^l}^l$. Following vFSMN [10, 16], we use vectors instead of scalers here. $\odot$ is element-wise multiplication. Our experiments (Sec. 4.3.2) show that the performance would drop dramatically without memory vectors. We also tuned the number of the vectors and used two vectors.

$$h_t^{l+1} = \begin{cases} \text{the } l\text{-th layer locates inside of a TDResBlock:} \\ \quad ReLU(e_t^l) \\ \text{the } l\text{-th layer is the output layer of a TDResBlock:} \\ \quad ReLU(e_t^l + h_t^{residual}), \end{cases} \tag{4}$$

where $h_t^{residual}$ is the t-th input frame to the current TDResBlock. Suppose the block has 5 layers and the output layer is l-th layer, $h_t^{residual}$ is actually h_t^{l-4}.

4. Experiments

4.1. Task and Data Descriptions

In this paper, we tested our proposed method on the Corpus of Spontaneous Japanese (CSJ) [29] and used 240-hour lecture recordings as the training set (**CSJ-Train**) [30, 31]. We used three official evaluation sets (**CSJ-Eval01**, **CSJ-Eval02**, and **CSJ-Eval03**), each of which contained ten lecture recordings [31], to evaluate the speech recognition results. Ten lecture recordings were chosen for validation (**CSJ-Dev**) during training. We also selected 27.6-hour training data in CSJ (**CSJ-Train$_{\text{small}}$**) to train the seed models for warm-start initialization [32] and tuning the parameters.

Table 1: *Data Sets of CSJ*

		#Lectures	Hours
Training set	**CSJ-Train$_{\text{small}}$**	155	27.6
	CSJ-Train	957	240
Development set (**CSJ-Dev**)		10	2.0
Testing set	**CSJ-Eval01**	10	2.0
	CSJ-Eval02	10	2.1
	CSJ-Eval03	10	1.4

We trained the baseline models using CSJ-Train. To start the first baseline model (**DNN-HMM-CE**), we first trained a GMM-HMM model and then a DNN model with five hidden layers, each of which is composed of 2048 hidden nodes. The output layer has about 8500 nodes that correspond to the tied-triphone states of the GMM-HMM model. We used 72-dim filter-bank features (24-dim static +Δ +$\Delta\Delta$), which are mean and variance normalized per speaker, and 11 frames were spliced (5 left, current, 5 right). The DNN model was trained using a standard stochastic gradient descent (SGD) based on the cross-entropy (CE) loss criterion. All were implemented using the Kaldi toolkit (nnet1) [33, 30].

We trained the baseline CTC models using the EESEN toolkit [3]. The BLSTM-CTC baseline model (**BLSTM-CTC**) was trained with the same 72-dimensional filter-bank features (24-dim static +Δ +$\Delta\Delta$) without splicing. The BLSTM network has five hidden layers, each of which is composed of 320 nodes. We used 28 Japanese context-independent phones (CI-phones) + blank (ϕ) as acoustic units for the CTC training. The third baseline model is a unidirectional LSTM CTC model (**ULSTM-CTC**) with the same parameter size as the BLSTM-CTC (five hidden layers, each composed of 640 nodes). The right context of the feature is eight frames, and the left context is a zero frame. The subsampling number was set to three.

For decoding, we trained a 4-gram word language model (WLM) from the transcription of 591-hours of CSJ training data. The WLM's vocabulary size was 98K. We compiled WFST-based decoding graphs for these models. The CTC-based models were decoded on EESEN decoder [3, 34] with a prior estimated from the training data. The performances are shown in Section 4.4.

4.2. Settings for Training Proposed Network

We trained the proposed CTC models with CNTK [35]. Similar to BLSTM-CTC, its features are a 72-dim non-spliced filter-bank (24-dim static +Δ +$\Delta\Delta$). The labels are the same CI-phone label of the two CTC baselines.

We first trained seed models with 27.6-hours of CSJ-Train$_{\text{small}}$ with the CTC loss criterion using a single GPU. Training the seed model costs two days and the WER% on CSJ-Eval01 is 19.3%. Then we used this seed model to train the CTC model using 240-hour training data. The FsAdaGrad algorithm (an implementation of Adam [36]) was used during CTC training. To quicken training with the 240-hour training data (CSJ-Train), we used the block-wise model update filtering (BMUF) distributed training algorithm [37] on 4 Tesla K40m GPUs. The initial learning rate for each frame was 0.00001 and automatically adjusted using validation on CSJ-Dev. The minibatch size was 2048. The number of parallel sequences in the same minibatch was 16. The maximum epoch number was 25. The training takes around a week and finished at the 13th epoch. We used the model from the 5th epoch for evaluation.

4.3. Model Structure Tuning

4.3.1. *Structure of proposed model*

We built a set of prototype networks and selected a best 26-layer structure using an evaluation set (CSJ-Dev) shown in Table 2.

Table 2: *Network structure: all layers are connected with ReLU activations, and past and future memories are stored in two global [1024×1] vectors.*

Component (ordered in seq.)	Structure (26 FC layers)		#Para. (35.5M)
Input	72-dim filter-bank features (24-dim static+Δ+$\Delta\Delta$)		
ResBlock 1	3 FC layers	Residual-skip	8.3M
	[72×2048] [2048×2048] [2048×2048]	[72×2048]	
ResBlock 2	3 FC layers	Residual-skip	4.5M
	[2048×128] [128×128] [128×2048]	[2048×2048]	
ResBlock 3	3 FC layers	Residual-skip	2.6M
	[2048×128] [128×128] [128×1024]	[2048×1024]	
TDResBlock 1	5 Time-delay layers	Residual-skip	6.0M
	[1024×1024]×5	[1024×1024]	
TDResBlock 2	5 Time-delay layers	Residual-skip	6.0M
	[1024×1024]×5	[1024×1024]	
TDResBlock 3	5 Time-delay layers	Residual-skip	6.0M
	[1024×1024]×5	[1024×1024]	
Fully-connect	2 FC layers		2.1M
	[1024×2048] [2048×29]		
Output	softmax (28 CI-phones+ϕ, see Sec.4.1)		

The model, which is only constructed with FC layers connected with ReLU activations, has three ResBlocks (each with three FC layers) and three TDResBlocks (each with five FC layers and five time-delay operations). The time-delay operations are all enhanced with two global memory blocks.

4.3.2. *Number of time-delay layers*

The local bidirectional contexts are captured by stacking the TDResBlocks. Following [13], the context-windows of the stacking time-delay operations are shown in Table 3. The local bidirectional context structure is implicitly encoded in the network architecture by stacking the time-delay network structure. The context's length can be exponentially extended by the depth of the stacking layers. The width of the captured context is $\frac{l\times(l+1)}{2}+1$, where l is the total number (15) of the FC layers in these TDResBlocks.

We trained VResTD-CTC models where the number of stacked time-delay layers ranged from 10 to 20. We also slightly tuned the past and future contexts. No benefit was identified (Table 3). Exhaustively searching for the optimized structure is not the focus of this paper.

Table 3: *ASR performance (WER% on CSJ-Dev) by tuning stacked time-delay operations*

Symmetric			Asymmetric	
layer 1: $-1,0,+1$			$-4,0,+1$	$-1,0,+4$
layer 2: $-2,0,+2$			$-5,0,+2$	$-2,0,+5$
layer 3: $-3,0,+3$			$-6,0,+3$	$-3,0,+6$
...			...	...
layer l: $-l,0,+l$			$-(l+3),0,+l$	$-l,0,+(l+3)$
(l=10)	(l=15)	(l=20)	(l=15)	(l=15)
13.1	**11.7**	12.7	12.7	13.7

4.3.3. Number of vectors for memory encoding

We tuned the number of vectors for memory encoding purposes, as shown in Table 4.

Table 4: *ASR performance (WER% on CSJ-Dev) of VResTD-CTC models with global or local memory encoding*

Network	Num. MemVecs				
	w/o	2	6	10	30
WER%	17.6	**11.7**	13.2	12.0	13.8

Our experiments showed that the performance dropped dramatically without memory vectors. We trained VResTD-CTC models where the number of memory encoding vectors ranged from 6 to 30, but they cannot outperform the model with two vectors. In our experiment with full training set, two vectors and ten vectors can achieve the same performance. We choose the simplier structure.

4.4. Performance of Speech Recognition

In this subsection, we compared the performance of the proposed acoustic models (**VResTD-CTC**) on the three CSJ evaluation sets with the baseline systems: **DNN-HMM-CE**, **BLSTM-CTC**, and **ULSTM-CTC**, see Section 4.1. We decoded the CTC model by feeding the network's scaled log-likelihood output to the EESEN decoder [3]. The decoding setting is the same with BLSTM-CTC and ULSTM-CTC.

Table 5: *ASR performance (WER%) of acoustic models*

Network	WER%			
(CSJ 240 hours)	Eval 01	Eval 02	Eval 03	Ave.
DNN-HMM-CE	**14.4**	**11.8**	15.6	**13.9**
BLSTM-CTC [34]	**14.4**	**11.9**	**15.1**	**13.8**
ULSTM-CTC	15.9	13.1	16.6	15.2
VResTD-CTC	15.1	**12.0**	**14.5**	**13.9**

As shown in Table 5, DNN-HMM-CE and BLSTM-CTC have similar performances. Compared to these two baselines, the word error rate (WER%) of VResTD-CTC is about 0.7% higher on CSJ-Eval01. However its performance can still improve on CSJ-Eval02 and CSJ-Eval03 and has comparable averaged performances on these three evaluation sets. Moreover, it outperformed ULSTM-CTC on all three evaluation sets. Its structure difference may explain such performance gaps between TDNN- (including our model) and BLSTM-based models. Every TDNN layer is only fed with the past and future contexts from its lower layers, while BLSTM can also get feedback from its current layer. Our model still has room for improvement.

Concerning the decoding speed, we tested the real-time factors (RTFs) of the above models. The decoding is applied on the same workstation without sharing with other jobs and the feedforward pass and decoding are all calculated into RTFs. Since BLSTM-CTC cannot be used in the Sprintra WFST decoder [4], we didn't test its speed. Compared with the DNN-HMM-CE model (RTF=0.70, HCLG decoding graph), the speed of proposed VResTD-CTC is 0.20 on RTF with TLG decoding graph, which matches the 3x faster results reported in [3]. It is slightly slower than the ULSTM-CTC model (RTF=0.15) due to the much deeper layers.

Parameter size is the another problem of our proposed model. Although compared to DNN-HMM-CE (38M), the proposed 26-layer model (36M) is not very large. But it remains over three times larger than BLSTM-CTC (11M) and ULSTM-CTC (10M). To compress the model, we applied singular value decomposition (SVD)-based parameter compress to all of the FC-layers [38, 39] and then fine-tuned with CSJ-Train$_{\text{small}}$. Empirical evaluations are shown in Table 6. By reducing half of the size of the original very deep model, we have the least performance loss: (1.0% of WER%). This loss can be compensated by using [34, 40] or retraining with full training set.

Table 6: *ASR performance (Average WER% on CSJ-Eval01/Eval02/Eval03) of acoustic models with different neural network structures*

	threshold of SVD			
	1.0 (Full model)	0.3	0.5	0.7
Para. Size (approx.)	36M	12M	18M	25M
WER% (Ave.)	**13.9**	15.7	14.9	15.9

5. Conclusion and Future Work

In this paper, we propose a method that focuses on replacing the BLSTM network with a very deep residual time-delay CTC neural network (VResTD-CTC). It can learn bidirectional long-term dependency without using recurrent feedback and be integrated with CTC training to achieve comparable performance with DNN-HMM-CE and BLSTM-CTC models. In the future, we will further improve this deep structure.

6. Acknowledgements

The authors thank Dr. Naoyuki Kanda for building concrete baseline systems and Mr. Mitsuyoshi Tachimori for his discussions about CTC-based decoding.

7. References

[1] A. Graves, S. Fernandez, F. Gomez, and J. Shmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proc. ICML*, 2006.

[2] A. Graves and N. Jaitly, "Towards End-to-End speech recognition with recurrent neural networks," in *Proc. ICML*, 2014.

[3] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-End speech recognition using deep RNN models and WFST-based decoding," in *Proc. IEEE-ASRU*, 2015, pp. 167–174.

[4] P. Dixon, C. Hori, and H. Kashioka, "Development of the SprinTra WFST speech decoder," *NICT Research Journal*, pp. 15–20, 2012.

[5] K. Chen, Z. Yan, and Q. Huo, "Training deep bidirectional LSTM acoustic model for LVCSR by a context-sensitive-chunk BPTT approach," in *Proc. INTERSPEECH*, 2015.

[6] Y. Zhang, G. Chen, D. Yu, K. Yao, S. Khudanpur, and J. Glass, "Highway long short-term memory RNNs for distant speech recognition," in *Proc. IEEE-ICASSP*, 2016.

[7] S. Xue and Z. Yan, "Improving latency-controlled BLSTM acoustic models for online speech recognition," in *Proc. IEEE-ICASSP*, 2017.

[8] Y. Miao, M. Gowayyed, X. Na, T. Ko, F. Metze, and A. Waibel, "An emprical exploration of ctc acoustic models," in *Proc. IEEE-ICASSP*, 2016.

[9] B. Li and et al., "Acoustic modeling for google home," in *Proc. INTERSPEECH*, 2017.

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

[11] A. Waibel, "Modular construction of time-delay neural networks for speech recognition," *Neural computation*, vol. 1, no. 1, pp. 39–46, 1989.

[12] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proc. INTERSPEECH*, 2015.

[13] M. Baskar and et al., "Residual memory networks: Feed-forward approach to learn long-term temporal dependencies," in *Proc. IEEE-ICASSP*, 2017.

[14] D. Yu, W. Xiong, J. Droppo, A. Stolcke, G. Ye, J. Li, and G. Zweig, "Deep convolutional neural networks with layer-wise context expansion and attention," in *Proc. INTERSPEECH*, 2016.

[15] Y. Zhang, D. Yu, M. Seltzer, and J. Droppo, "Speech recognition with prediction-adaptation-correction recurrent neural networks," in *Proc. IEEE-ICASSP*, 2015.

[16] J. Tang, S. Zhang, S. Wei, and L. Dai, "Future context attention for unidirectional LSTM based acoustic model," in *Proc. INTERSPEECH*, 2016.

[17] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. IEEE-ICASSP*, 2016.

[18] F. Bell-Berti, R. A. Krakow, C. E. Gelfer, and S. E. Boyce, "Anticipatory and carryover effects: Implication for models of speech production," *Producing Speech: A Festschrift in honor of Katherine Safford Harris, edited by F. Bell-Berti and L. Raphael (American Institute of Physics, Woodbury, NY)*, pp. 77–79, 1995.

[19] M. Mohri, F. Pereira, and M. Riley, "Weighted finite-state transducers in speech recognition," *Computer Speech & Language*, vol. 16, no. 1, pp. 69–88, 2002.

[20] 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," in *Proc. INTERSPEECH*, 2016.

[21] W. X. et al., "Toward human parity in conversational speech recognition," *IEEE/ACM Trans. ASLP*, vol. 25, no. 12, pp. 2410–2423, 2017.

[22] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time-delay neural networks," *IEEE/ACM Trans. ASLP*, vol. 37, no. 3, pp. 328–339, 1989.

[23] Y. Wang, X. Deng, S. Pu, and Z. Huang, "Residual convolutional CTC networks for automatic speech recognition," in *arXiv preprint arxiv:1702.07793*, 2016.

[24] M. Bi, Y. Qian, and K. Yu, "Very deep convolutional neural networks for LVCSR," in *Proc. INTERSPEECH*, 2015.

[25] Y. Qian and et al., "Very deep convolutional neural networks for noise robust speech recognition," *IEEE/ACM Trans. ASLP*, vol. 24, no. 12, pp. 2263–2276, 2016.

[26] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *Proc. IEEE-ICASSP*, 2017.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[28] R.Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Proc. NIPS*, 2015.

[29] K. Maekawa, "Corpus of spontaneous japanese: Its design and evaluation," in *Proc. ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition*, 2003.

[30] T. Moriya, T. Shinozaki, and S. Watanabe, "Kaldi recipe for Japanese spontaneous speech recognition and its evaluation," in *Autumn Meeting of ASJ*, no. 3-Q-7, 2015.

[31] T. Kawahara, H. Nanjo, T. Shinozaki, and S. Furui, "Benchmark test for speech recognition using the corpus of spontaneous japanese," in *Proc. ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition*, 2003.

[32] J. Dean and et al., "Large scale distributed deep networks," in *Proc. NIPS*, 2012.

[33] D. Povey and et al., "The Kaldi speech recognition toolkit," in *Proc. IEEE-ASRU*, 2011.

[34] N. Kanda, X. Lu, and H. Kawai, "Maximum a posteriori based decoding for CTC acoustic models," in *Proc. INTERSPEECH*, 2016, pp. 1868–1872.

[35] A. Agarwal and et al., "An introduction to computational networks and the computational network toolkit," in *Microsoft Technical Report MSR-TR-2014-112*, 2014.

[36] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.

[37] K. Chen and Q. Huo, "Scalable training of deep learning machines by incremental block training with intra-block parallel optimization and blockwise model-update filtering," in *Proc. IEEE-ICASSP*, 2016.

[38] J. Xue, J. Li, and Y. Gong, "Restructuring of deep neural network acoustic models with singular value decomposition," in *Proc. INTERSPEECH*, 2013.

[39] M. Sun and et al., "Compressed time delay neural network for small-footprint keyword spotting," in *Proc. INTERSPEECH*, 2017.

[40] N. Kanda, X. Lu, and H. Kawai, "Bayes risk training of CTC acoustic models in maximum a posteriori based decoding framework," in *Proc. IEEE-ICASSP*, 2017.