Interspeech 2018
2-6 September 2018, Hyderabad

Output-Gate Projected Gated Recurrent Unit for Speech Recognition

Gaofeng Cheng[1,2], *Daniel Povey* [4,5], *Lu Huang* [3], *Ji Xu* [1], *Sanjeev Khudanpur*[4,5], *Yonghong Yan*[1,2,6]

[1]Key Laboratory of Speech Acoustics and Content Understanding, IOA, CAS, China
[2]University of Chinese Academy of Sciences, China; [3]Tsinghua University, China
[4]Center of Language and Speech Processing
[5]Human Language Technology Center Of Excellence,
The Johns Hopkins University, Baltimore, MD, USA,
[6]Xinjiang Laboratory of Minority Speech and Language Information Processing,
Xinjiang Technical Institute of Physics and Chemistry, CAS, China
gfcheng.cn@gmail.com

Abstract

In this paper, we describe the work on accelerating decoding speed while improving the decoding accuracy. Firstly, we propose an architecture which we call Projected Gated Recurrent Unit (PGRU) for automatic speech recognition (ASR) tasks, and show that the PGRU could outperform the standard GRU consistently. Secondly, in order to improve the PGRU's generalization, especially for large-scale ASR task, the Output-gate PGRU (OPGRU) is proposed. Finally, time delay neural network (TDNN) and normalization skills are found to be beneficial to the proposed projected-based GRU. The finally proposed unidirectional TDNN-OPGRU acoustic model achieves 3.3% / 4.5% relative reduction in word error rate (WER) compared with bidirectional projected LSTM (BLSTMP) on Eval2000 / RT03 test sets. Meanwhile, TDNN-OPGRU acoustic model speeds up the decoding speed by around 2.6 times compared with BLSTMP.

Index Terms: GRU, LSTM, Lattice-free MMI, Recurrent Neural Network, Speech Recognition

1. Introduction

Recurrent neural network achieves state-of-the-art results in automatic speech recognition (ASR) tasks [1] [2] [3] [4] [5]. Unlike feed-forward neural network, recurrent neural network will feed activations from both the previous time steps and previous layers as input to the network to make decision for the current time step. Therefore, recurrent neural network could model a dynamic contextual window of all sequence history rather than a static fixed window over the input sequence which makes recurrent neural network more suitable for sequence modeling.

However, it is difficult to train the vanilla recurrent neural network due to the vanishing gradient and exploding gradient problems [6]. To address these problems, many sophisticated recurrent units are proposed, among those recurrent neural network variants, long short-term memory (LSTM) [7] and gated recurrent unit (GRU) [3] are two related variants which have been successfully used in speech recognition field [1] [8]. For LSTM, input gate, output gate and forget gate are used to control the information flow. Simpler than LSTM, the GRU uses only reset gate and update gate. In [4], the authors evaluated GRU and LSTM on polyphonic music data and raw speech signal data, but they did not make concrete conclusion on whether LSTM or GRU was better. For ASR task, in [8], the authors has applied GRU for ASR, and showed that GRU based acoustic model outperformed the simple recurrent neural network, but they did not show the recognition results on LSTM.

In this paper, we focus on GRU based recurrent neural network architectures. The contributions of this paper could be described as:

- Proposing PGRU for speech recognition tasks, and comparing it with standard GRU in bidirectional variant.
- Proposing OPGRU and interleaving (O)PGRU with TDNN [9] into one unified unidirectional model.
- Normalizing the projected output of (O)PGRU, and achieving better performance.

The paper is organized as follows: Section 2 presents the prior work, Section 3 presents the proposed models, Section 4 shows the experimental setup, Section 5 presents the results, and conclusion is presented in Section 6.

2. Prior work

In this paper, we propose projected-based GRU architectures for ASR tasks, and compare the PGRU with the standard GRU and Projected LSTM (LSTMP) [1]. Before the evaluation of PGRU, we first describe the LSTMP and GRU in this section.

2.1. Projected LSTM

The LSTM unit was initially proposed by Hochreiter and Schmidhuber [7]. For vanilla recurrent neural network, it is difficult to learn long time dependency due to the vanishing or exploding gradient. But LSTM can learn long time dependency by enforcing constant error flow through 'constant error carrousels' (CEC) [7]. Since then, a number of modifications to the original LSTM have been made.

We follow the implementation of LSTM in the 'projected' variant proposed by Sak [1]. Each LSTMP unit contains an input gate which controls the flow of input activations into the memory cell and an output gate which controls the output flow of LSTMP unit. Also the forget gate is adopted to allow the LSTMP unit to adaptively forgetting or resetting the memory cell [10]. Unlike LSTM, every LSTMP unit contains one recurrent projection layer and one non-recurrent projection layer. Note that we use one equivalent single projection layer to take place of two separate projection layers.

2.2. GRU

GRU was firstly proposed by Cho et al. [3] to make the recurrent unit capture the long time dependency. Similar to the LSTM

10.21437/Interspeech.2018-1403

unit, the GRU also has gating mechanism to modulate the flow of information through the unit.

In our experiments, we implement the standard GRU which is similar to [4] and formulation is:

$$
\begin{aligned}
r_t &= \sigma(W_{rx}x_t + W_{rh}h_{t-1} + b_r) && (1)\\
z_t &= \sigma(W_{zx}x_t + W_{zh}h_{t-1} + b_z) && (2)\\
\tilde{h}_t &= \tanh(W_{\tilde{h}x}x_t + W_{\tilde{h}h}(r_t \odot h_{t-1}) + b_{\tilde{h}}) && (3)\\
h_t &= (1 - z_t) \odot \tilde{h}_t + z_t \odot h_{t-1} && (4)\\
y_t &= h_t && (5)
\end{aligned}
$$

where the memory cell activation h_t at time t is a linear interpolation of the previous activation h_{t-1} and the activation candidate $\tilde{h}_t$ at time t, r_t is the reset gate and z_t is the update gate. y_t is the output of GRU. $\odot$ is element-wise multiplication and σ stands for Sigmoid non-linear activation. The W terms denote weight matrices (e.g. W_{rx} is the weight matrix from the reset gate to the input).

For the standard GRU [4], the activation candidate is computed similarly to the traditional recurrent neural network, z_t decides how much the GRU update its memory cell activation and r_t is used to forget the previously computed state. Different from LSTM, GRU does not have a separate memory cell, and its memory cell is exposed to the next step calculation directly.

3. Proposed model

3.1. Projected GRU

Different from the standard GRU architecture, the PGRU has one projection layer, formulation is:

$$
\begin{aligned}
r_t &= \sigma(W_{rx}x_t + W_{rs}s_{t-1} + b_r) && (6)\\
z_t &= \sigma(W_{zx}x_t + W_{zs}s_{t-1} + b_z) && (7)\\
\tilde{h}_t &= \tanh(W_{\tilde{h}x}x_t + W_{\tilde{h}s}(r_t \odot s_{t-1}) + b_{\tilde{h}}) && (8)\\
h_t &= (1 - z_t) \odot \tilde{h}_t + z_t \odot h_{t-1} && (9)\\
y_t &= W_{yh}h_t && (10)\\
s_t &= y_t[0 : r - 1] && (11)
\end{aligned}
$$

where W_{yh} is the projection matrix, which projects the h_t into y_t with a lower dimension. s_t is the projected recurrence. And r is the projected recurrence dimension. y_t is the output of PGRU. The dimension of r_t is the same as the projected recurrence dimension, and the dimension of z_t is the same as memory cell dimension.

Under our setup, the projected memory cell contains recurrent and non-recurrent part, the recurrent part will be used as the recurrence of PGRU, and all the projected memory cell will be fed into next layer as the output of PGRU. The s_t mismatches the dimension of h_t, so we still use h_{t-1} for the calculation of h_t. With the projection layer, we can preserve a memory cell with higher dimension while keeping the model size small. The existence of non-recurrent part in projected memory cell can keep a higher output dimension without increasing the model size obviously.

3.2. Output-gate Projected GRU

The reset gate r_t allows the model to delete past memory by forgetting the previously computed states. From Fig.1 we can see, the PGRU's reset gates (Sigmoid function) tend to get saturated. This means reset gates from different layers remain or

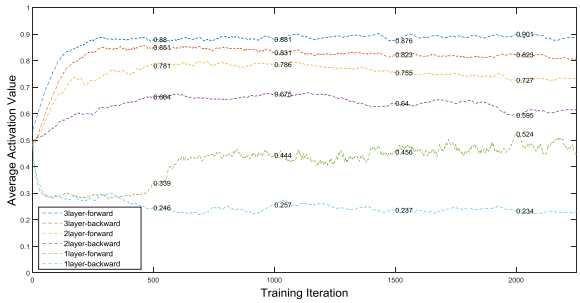

Figure 1: *Average activation values of different reset gates of one 3-layer Bidirectional PGRU on Fisher + Switchboard task.*

delete most of the previously computed states. We believe that computational redundancy exists in the PGRU reset gate. Also the saturated reset gate will hamper the back propagation during model training. But different from [11], we remove the reset gate and replace it with an output gate. The existence of projection layer breaks the inherent value bound of the GRU, the output gate can help regulate how much the cell information is exposed to the projected output. Another modification of PGRU is using h_{t-1} to replace s_{t-1} in Eq.(8). Our experiments show that the proposed OPGRU outperforms PGRU on large-scale speech recognition task. The equations of proposed Output-gate Projected GRU (OPGRU) are as follow:

$$
\begin{aligned}
o_t &= \sigma(W_{ox}x_t + W_{os}s_{t-1} + b_o) && (12)\\
z_t &= \sigma(W_{zx}x_t + W_{zs}s_{t-1} + b_z) && (13)\\
\tilde{h}_t &= \tanh(W_{\tilde{h}x}x_t + U_{\tilde{h}h} \odot h_{t-1} + b_{\tilde{h}}) && (14)\\
h_t &= (1 - z_t) \odot \tilde{h}_t + z_t \odot h_{t-1} && (15)\\
\tilde{y}_t &= o_t \odot h_t && (16)\\
y_t &= W_{y\tilde{y}}\tilde{y}_t && (17)\\
s_t &= y_t[0 : r - 1] && (18)
\end{aligned}
$$

where $U_{\tilde{h}h}$ is learnable vector, we use vector instead of matrix to reduce the model parameter. The dimension of o_t and z_t is the same as memory cell dimension.

4. Experimental setup

All our experiments were conducted under the Kaldi speech recognition toolkit [12]. We focus on 2000 Hr Fisher + Switchboard large vocabulary continuous speech recognition (LVCSR) task, also report some results on 80 Hr AMI single distant microphone (SDM) [13] [14], 200 Hr Ted-lium [15] and 300 Hr Switchboard (SWBD) LVCSR tasks. The training criterion is phone-level sequence training, using the lattice-free MMI objective [16] on outputs of frame rate 33 Hz. 40-dimensional Mel-frequency cepstral coefficients (MFCCs) without cepstral truncation are used as the input into the neural network [17].

The experimental setups for AMI SDM, Ted-lium and Switchboard are the same as the ones described in [18], the AMI SDM LVCSR system is trained with numerator lattices generated from the parallel AMI individual headset microphone (IHM) data [19]. Fisher + Switchboard LVCSR system is the same as the one in [16]. We use speed-perturbation technique [20] for 3-fold data augmentation; and iVector to perform instantaneous adaption of the neural network [21]. The i-Vector is used to provide information about the mean offset of the

Table 1: *WER for BGRU and BPGRU.*

Model	AMI SDM					300 Hr SWBD					
	Parameter Size	Cell Dimension [1]	Layer Number	WER(%) Dev	WER(%) Eval	Parameter Size	Cell Dimension	Layer Number	WER(%) on Eval2000 SWBD	WER(%) on Eval2000 Callhome	WER(%) on Eval2000 Total
BGRU	11.3M	360	3	40.8	44.1	14.9M	360	3	10.3	20.0	15.2
BPGRU	12.5M	1024-128-128	3	**37.8**	**41.1**	15.3M	1024-128-128	3	**9.5**	**18.6**	**14.2**
BGRU	24.9M	600	3	41.6	44.5	30.8M	600	3	10.3	19.9	15.1
BPGRU	24.8M	1024-256-256	3	**37.8**	**41.1**	30.0M	1024-256-256	3	**9.4**	**18.4**	**14.0**
Model	Ted-Lium					2000 Hr Fisher+SWBD					
	Parameter Size	Cell Dimension	Layer Number	WER(%) Dev	WER(%) Eval	Parameter Size	Cell Dimension	Layer Number	WER(%) on Eval2000 SWBD	WER(%) on Eval2000 Callhome	WER(%) on Eval2000 Total
BGRU	14.4M	360	3	9.3	9.3	11.9M	360	2	11.0	19.7	15.5
BPGRU	14.9M	1024-128-128	3	**7.9**	**8.4**	12.0M	1024-128-128	2	**10.6**	**19.0**	**14.8**
BGRU	30.1M	600	3	9.0	9.2	31.0M	600	3	10.2	17.8	14.2
BPGRU	29.4M	1024-256-256	3	**8.1**	**8.4**	30.1M	1024-256-256	3	**9.9**	**17.6**	**13.9**

[1.] For Cell Dimension 1024-256-256: cell size - 1024, recurrent and non-recurrent dimension - 256.

Table 2: *Configurations of various models.*

Model	Architecture [1]	Latency [2]
BLSTMP	$[L_f,L_b],[L_f,L_b],[L_f,L_b]$	2020
BPGRU	$[P_f,P_b],[P_f,P_b],[P_f,P_b]$	2020
TDNN-LSTMP	$T^{100}\ T^{100}\ T^{100}\ L_f\ T\ T\ L_f\ T\ T\ L_f$	200
TDNN-PGRU	$T^{100}\ T^{100}\ T^{100}\ P_f\ T\ T\ P_f\ T\ T\ P_f$	200
TDNN-OPGRU	$T^{100}\ T^{100}\ T^{100}\ O_f\ T\ T\ O_f\ T\ T\ O_f$	200

[1.] Forward LSTMP - L_f, backward LSTMP - L_b, forward PGRU - P_f, backward PGRU - P_b, TDNN - T, forward OPGRU - O_f, default layer frame-rate - 33 Hz and other frame rates are specified in the super-script.

[2.] See [22] for the definition of latency. We use the same training and decoding configuration for BLSTMP as that in [22]. Also TDNN-PGRU and TDNN-OPGRU have the same configuration as TDNN-LSTM-C in [22], just with LSTMP layers replaced, so they have the same latency.

speaker's data, so cepstral normalization is not necessary.

4.1. Neural network configuration

The baseline BLSTMP and TDNN-LSTMP neural networks are the same as the models described in [18] [22]. Regarding the projection layers in LSTMP, PGRU and OPGRU in this paper, the dimensions of projected recurrence and projected non-recurrence are always one quarter the cell dimension. For instance, if the memory cell dimension is 1024, the recurrence projection would be of dimension 256 and the output dimension would be 512. The model details are shown in Table 2. The default cell dimension is 1024 unless specified.

Additional temporal context was found to be beneficial for TDNN-LSTMP used in AMI tasks. So additional TDNN layers between successive recurrent layers were used to provide additional context for TDNN-{LSTMP, PGRU, OPGRU} in AMI SDM task. This leads to larger model parameter size and additional latency for TDNN-{LSTMP, PGRU, OPGRU} in AMI SDM task.

5. Results

5.1. BPGRU vs BGRU

We began by comparing the standard GRU based recurrent neural network architectures with the proposed PGRU based recurrent neural network architectures, specially in the bidirectional variant. For fair comparison, we tuned the parameter size of both bidirectional PGRU (BPGRU) and bidirectional GRU (BGRU) to keep them with the same model size. On the AMI SDM LVCSR task, BPGRU achieves an average relative word error rate (WER) reduction of 7.7% over the BGRU, and the figure is 11.0% on Ted-lium, 7.0% on Switchboard, 3.4% on Fisher + Switchboard. For the Eval2000 test set, we care more about the WER on the total test set, but for convenience, we also show the Switchboard (SWBD) and Callhome subset results in the table. The BPGRU gains consistent improvement compared with BGRU on all tested LVCSR tasks. As for the reason why PGRU outperforms standard GRU, we believe that, the PGRU could preserve a larger memory cell dimension compared with the standard GRU with the same model size.

Table 3: *Interleaving PGRU and OPGRU with TDNN.*

Model on AMI SDM	Parameter Size	WER(%) Dev	WER(%) Eval
TDNN-PGRU	36.5M	35.9	39.0
TDNN-OPGRU	38.7M	36.3	39.6
Model on Ted-lium	**Parameter Size**	**WER(%) Dev**	**WER(%) Test**
TDNN-PGRU	30.2M	8.0	7.7
TDNN-OPGRU	32.4M	7.9	8.0
Model on SWBD	**Parameter Size**	**WER(%) on Eval2000 SWBD**	**WER(%) on Eval2000 Total**
TDNN-PGRU	32.7M	9.0	13.3
TDNN-OPGRU	34.9M	9.1	13.3
Model on Fisher+SWBD	**Parameter Size**	**WER(%) on Eval2000 SWBD**	**WER(%) on Eval2000 Total**
TDNN-PGRU	32.8M	9.1	12.9
TDNN-OPGRU	34.9M	8.6	12.0

Table 4: *WER after applying normalization.*

Model on 2000 Hr Fisher+SWBD	Eval2000 SWBD	Eval2000 Total	RT03 Fsh	RT03 Total
TDNN-PGRU	9.1	12.9	9.9	12.4
TDNN-NormPGRU	8.5	12.0	9.1	11.3
TDNN-OPGRU	8.6	12.0	9.3	11.7
TDNN-NormOPGRU	8.1	11.8	8.9	11.0
TDNN-LSTMP	8.2	12.0	9.4	11.4
TDNN-NormLSTMP	8.1	12.3	9.6	11.6

5.2. TDNN-PGRU vs TDNN-OPGRU

In this section, we firstly explored the effect of interleaving PGRU with TDNN. From Table 1 and Table 3 we can see, compared with 3-layer 1024-256-256 BPGRU in Table 1, TDNN-PGRU gains a 5.1% relative reduction in WER on AMI SDM LVCSR task and the figure is 3.6% on Ted-lium LVCSR task, 5.0% on Switchboard LVCSR task, 7.2% on Fisher + Switchboard LVCSR task. Overall, the improvement indicates that interleaving TDNN is beneficial for the proposed Projected-based GRU.

Secondly, we proposed TDNN-OPGRU by replacing PGRU with OPGRU. From Table 3 we can see, for the three relative small-scale ASR tasks, TDNN-OPGRU performs the same as or slightly worse than the TDNN-PGRU, but for the 2000 Hr Fisher+Switchboard ASR task, TDNN-OPGRU gains

Table 5: *WER comparison between LSTMP and OPGRU.*

Model on 2000 Hr Fisher+SWBD	Model Size	Eval2000			RT03		
		SWBD	Callhome	Total	Fsh	SWBD	Total
Xiong et al. [23] BLSTM, spatial smoothing		8.6	15.4	-	-	-	-
Xiong et al. [23] BLSTM, spatial smoothing, 27k senones		8.3	15.3	-	-	-	-
BLSTMP	41.3M	8.5	15.3	12.0	9.7	13.3	11.6
BLSTMP+dropout		8.4	15.4	12.0	9.2	13.0	11.2
TDNN-LSTMP	39.7M	8.2	15.4	12.0	9.4	13.3	11.4
BatchnormTDNN-LSTMP		8.3	16.4	12.4	9.8	13.9	11.9
BatchnormTDNN-LSTMP+dropout		8.2	15.5	12.0	9.2	13.0	11.2
TDNN-NormOPGRU	34.9M	8.1	15.3	11.8	8.9	12.9	11.0
BatchnormTDNN-NormOPGRU		8.2	15.5	11.9	8.4	12.6	10.6
BatchnormTDNN-NormOPGRU+dropout[1]		8.3	14.7	11.6	8.5	12.6	10.7

[1.] Dropout is applied to the OPGRU's output gate and update gate, similar to dropout Location 4 in paper [18].

obvious improvement on total Eval2000 test set. From Table 3, we could get the conclusion that OPGRU generalizes better than PGRU across different data sets.

5.3. Normalization in OPGRU and PGRU

There have been prior works on applying batch normalization on recurrent neural networks [11] [24]. The author in [11], used Rectified Linear Unit (ReLU) for the GRU, so in order to reduce the activation value of ReLU, they use batch normalization. To stabilize the output of projected-based GRU, after the projection layer, we apply batch normalization in the forward direction. For the projected recurrence (s_t in Eq.11 and Eq.18), we normalize them in the way same as Hinton's layer normalization [25], except we do no mean subtraction, only variance normalization. We call PGRU and OPGRU with normalization NormPGRU and NormOPGRU.

From Table 4 we can see, the TDNN-NormPGRU gains 7.0% / 8.9% relative reduction in WER compared with TDNN-PGRU on total Eval2000 / RT03 test sets. The relative reduction in WER for TDNN-NormOPGRU over TDNN-OPGRU is 1.7% / 6.0% on total Eval2000 / total RT03 test sets. However the proposed normalization method does not benefit the TDNN-LSTMP in decoding accuracy.

5.4. Comparing NormOPGRU with LSTMP

In previous section, the proposed normalization is proved to be trivial to the LSTMP unit, so in the rest of the paper, normal LSTMP is adopted. For comparison, we show BLSTM results from Xiong et al. [23]. It should be noted that we can get better accuracy with feature-fusion [26]. For decoding setup, we only use fisher and switchboard training transcripts to build the 4-gram language model. In general, our BLSTMP baseline is comparable to those of [23]. From Table 5 we can see, applying both batch normalization in TDNN layers and per-frame dropout [18] in the LSTMP/NormOPGRU can slightly improve the decoding results. From the acoustic models in Table 5 we can see, compared with the BLSTMP or TDNN-LSTMP, the proposed TDNN-NormOPGRU can achieve 3.3% / 4.5% relative reduction in WER on Eval2000 / RT03 test sets.

Decoding speed, commonly measured by real time factor (RTF), is important for online acoustic model deploy. Fixed chunk and additional chunk context are used for our recurrent neural network during training and decoding. For decoding setup in Table 6, the acoustic models are decoded with chunk-width 1500ms[1] with 500ms additional chunk context[2]. For the unidirectional recurrent neural networks, we adopted state-saving decoding strategy [22], which copy the state across chunks to reduce latency. From Table 6 we can see, state-saving TDNN-NormOPGRU can speed up decoding by 2.6 times compared with BLSTMP. The state-saving decoding LSTM acoustic models always degrade the performance slightly compared with their non-state-saving counterparts [22]. So under the state-saving decoding setup, the finally proposed TDNN-NormOPGRU is 4.1% / 6.1% relative better than TDNN-LSTMP in WER, and speeds up the decoding by 1.5 times.

Table 6: *Real time factor for various models.*

Model on Fisher+SWBD[1]	Eval2000		RT03		RTF
	SWBD	Total	Fsh	Total	
BLSTMP+dropout	8.4	12.0	9.2	11.2	1.78
TDNN-LSTMP+dropout	8.2	12.0	9.2	11.2	1.19
+state saving decode [22]	8.1	12.1	9.4	11.4	0.99
TDNN-NormOPGRU+dropout	8.3	11.6	8.5	10.7	0.96
+state saving decode [22]	8.2	**11.6**	8.6	**10.7**	**0.68**

[1.] All the TDNN layers are equipped with batch normalization, TDNN-LSTMP is the same model as BatchnormTDNN-LSTMP in Table 5.

6. Conclusions

This paper proposed the OPGRU as an alternative for LSTMP. Compared with LSTMP , the simple OPGRU has only two gates and smaller model size. Also, in order to further improve the performance of projected-based GRUs, we combined them with TDNNs and applied batch normalization as well as a variant of layer normalization to them. With all theses techniques adopted, our finally proposed TDNN-NormOPGRU acoustic model enjoyed faster decoding speed and higher decoding accuracy compared with our old LSTMP system.

7. Acknowledgements

This work is partially supported by the National Key Research and Development Plan (Nos.2016YFB0801203,2016YFB0801200), the National Natural Science Foundation of China (Nos.11590770-4,U1536117,11504406,11461141004),the Key Science and Technology Project of the Xinjiang Uygur Autonomous Region (No.2016A03007-1),the Pre-research Project for Equipment of General Information System (No.JZX2017-0994/Y306). We'd like to thank the cooperation project with the Sony (China) Limited.

8. References

[1] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary

[1]Equivalent to 150 frames.

[2]BLSTMP needs both left and right additional chunk context, unidirectional ones only need left additional chunk context.

speech recognition," *CoRR*, vol. abs/1402.1128, pp. 157–166, 2014. [Online]. Available: http://arxiv.org/abs/1402.1128

[2] K. Chen and Q. Huo, "Training deep bidirectional lstm acoustic model for lvcsr by a context-sensitive-chunk bptt approach," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 7, pp. 1185–1193, July 2016.

[3] K. Cho, B. Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," in *Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST-8)*, 2014.

[4] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[5] M. Ravanelli, P. Brakel, M. Omologo, and Y. Bengio, "Light gated recurrent units for speech recognition," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, pp. 92–102, 2018.

[6] 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, Mar 1994.

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[8] D. Amodei *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," *CoRR*, vol. abs/1512.02595, 2015. [Online]. Available: http://arxiv.org/abs/1512.02595

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

[10] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with lstm," in *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*, vol. 2, 1999, pp. 850–855 vol.2.

[11] M. Ravanelli, P. Brakel, M. Omologo, and Y. Bengio, "Improving speech recognition by revising gated recurrent units," in *Interspeech*, 2017.

[12] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. EPFL-CONF-192584. IEEE Signal Processing Society, 2011.

[13] J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal *et al.*, "The ami meeting corpus: A pre-announcement," in *International Workshop on Machine Learning for Multimodal Interaction*. Springer, 2005, pp. 28–39.

[14] S. Renals, T. Hain, and H. Bourlard, "Recognition and understanding of meetings the ami and amida projects," in *Automatic Speech Recognition & Understanding, 2007. ASRU. IEEE Workshop on*. IEEE, 2007, pp. 238–247.

[15] A. Rousseau, P. Delglise, and Y. Estve, "Ted-lium: an automatic speech recognition dedicated corpus," in *Eight International Conference on Language Resources and Evaluation*, 2012, pp. 125–129.

[16] D. Povey, V. Peddinti, D. Galvez, G. Pegah, V. Manohar, Y. Wang, X. Na, and S. Khudanpur, "Purely sequence-trained neural networks for asr based on lattice-free mmi," in *Interspeech*, 2016.

[17] D. Povey, X. Zhang, and S. Khudanpur, "Parallel training of deep neural networks with natural gradient and parameter averaging," *CoRR*, vol. abs/1410.7455, 2014. [Online]. Available: http://arxiv.org/abs/1410.7455

[18] G. Cheng, V. Peddinti, D. Povey, V. Manohar, S. Khudanpur, and Y. Yan, "An exploration of dropout with lstms," in *Interspeech*, 2017.

[19] V. Peddinti, V. Manohar, Y. Wang, D. Povey, and S. Khudanpur, "Far-field asr without parallel data." in *Interspeech*, 2016, pp. 1996–2000.

[20] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition." in *Interspeech*, 2015, pp. 3586–3589.

[21] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors." in *ASRU*, 2013, pp. 55–59.

[22] V. Peddinti, Y. Wang, D. Povey, and S. Khudanpur, "Low latency acoustic modeling using temporal convolution and lstms," *IEEE Signal Processing Letters*, vol. PP, no. 99, pp. 1–1, 2017.

[23] W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig, "Achieving human parity in conversational speech recognition," *CoRR*, vol. abs/1610.05256, 2016. [Online]. Available: http://arxiv.org/abs/1610.05256

[24] T. Cooijmans, N. Ballas, C. Laurent, and A. C. Courville, "Recurrent batch normalization," *CoRR*, vol. abs/1603.09025, 2016. [Online]. Available: http://arxiv.org/abs/1603.09025

[25] J. Ba, R. Kiros, and G. E. Hinton, "Layer normalization," *CoRR*, vol. abs/1607.06450, 2016.

[26] G. Saon, G. Kurata, T. Sercu, K. Audhkhasi, S. Thomas, D. Dimitriadis, X. Cui, B. Ramabhadran, M. Picheny, L.-L. Lim *et al.*, "English conversational telephone speech recognition by humans and machines," *arXiv preprint arXiv:1703.02136*, 2017.