PAPER

Automatic Speech Recognition System with Output-Gate Projected Gated Recurrent Unit

Gaofeng CHENG[†,††a)], ***Nonmember***, **Pengyuan ZHANG**[†,††b)], ***Member***, ***and*** **Ji XU**[††c)], ***Nonmember***

SUMMARY The long short-term memory recurrent neural network (LSTM) has achieved tremendous success for automatic speech recognition (ASR). However, the complicated gating mechanism of LSTM introduces a massive computational cost and limits the application of LSTM in some scenarios. In this paper, we describe our work on accelerating the decoding speed and improving the decoding accuracy. First, we propose an architecture, which is called Projected Gated Recurrent Unit (PGRU), for ASR tasks, and show that the PGRU can consistently outperform the standard GRU. Second, to improve the PGRU generalization, particularly on large-scale ASR tasks, we propose the Output-gate PGRU (OPGRU). In addition, the time delay neural network (TDNN) and normalization methods are found beneficial for OPGRU. In this paper, we apply the OPGRU for both the acoustic model and recurrent neural network language model (RNN-LM). Finally, we evaluate the PGRU on the total Eval2000 / RT03 test sets, and the proposed OPGRU single ASR system achieves 0.9% / 0.9% absolute (8.2% / 8.6% relative) reduction in word error rate (WER) compared to our previous best LSTM single ASR system. Furthermore, the OPGRU ASR system achieves significant speed-up on both acoustic model and language model rescoring.

key words: *GRU, LSTM, neural network language model, speech recognition*

1. Introduction

The application of deep neural network (DNN), particularly recurrent neural network (RNN), has achieved great success in automatic speech recognition (ASR) [1]–[3]. However, suffering from the gradient explosion or vanishing problem [4], vanilla recurrent neural networks gain very limited success for ASR tasks. To address this problem, the long short-term memory (LSTM) units [5] were proposed and achieved tremendous success [6], [7]. The LSTM unit is designed with a sophisticated gating mechanism and 'constant error carrousels' (CEC) [5] to enforce the constant error flow through the memory cell. Because of its long-term temporal dependency modeling ability, LSTM is widely used in many sequence modeling tasks such as the acoustic model for ASR tasks [7], [8] and the recurrent neural network language model [1].

Manuscript received May 4, 2018.
Manuscript revised September 22, 2018.
Manuscript publicized November 19, 2018.

[†]The authors are with the University of Chinese Academy of Sciences, Beijing, China.
[††]The authors are with the Key Laboratory of Speech Acoustics and Content Understanding, Institute of Acoustics, Chinese Academy of Sciences, Beijing, China.
a) E-mail: chenggaofeng@hccl.ioa.ac.cn
b) E-mail: zhangpengyuan@hccl.ioa.ac.cn (Corresponding author)
c) E-mail: xuji@hccl.ioa.ac.cn

DOI: 10.1587/transinf.2018EDP7155

Although LSTM acoustic models, especially the bidirectional LSTM (BLSTM) [8], [9], have achieved higher recognition accuracy than vanilla RNN and simple DNN, decoding efficiency still remains a challenge for their application. Decoding speed is a realistic and critical consideration when we deploy an ASR system. To reduce the computational cost, an alternative to LSTM — GRU [10] is proposed. Simpler than LSTM, GRU uses only reset gate and update gate. In [11], 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. In [2], the authors 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 [12], the authors employed visualization techniques to study the behavior of LSTM and GRU when performing speech recognition tasks, and concluded that both the LSTM and GRU can accumulate longer memory at higher-level layers but GRU is more robust than LSTM in noisy conditions. In [13], the authors tried LSTM and GRU on language modeling tasks and they confirmed that the LSTM seems to be a better choice than the GRU.

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

- Proposing PGRU for speech recognition, and comparing it with the standard GRU in the bidirectional variant.
- Interleaving PGRU with TDNN [3] into one unidirectional hybrid model, which outperforms the bidirectional PGRU (BPGRU).
- Improving the generalization of PGRU on large-scale data sets by proposing the OPGRU.
- Applying the proposed OPGRU instead of LSTM for both the acoustic model and neural network language model.

The paper is organized as follows: Sect. 2 presents the prior work, Sect. 3 presents the proposed models, Sect. 4 shows the experimental setup, Sect. 5 presents the results for the acoustic models, Sect. 6 presents results for the neural network language models, and the conclusions are provided in Sect. 7.

2. Prior Work

In this paper, we propose the PGRU-based recurrent neural network architectures to build a single ASR system, which contains the one-pass n-gram-based weighted finite-state transducer (WFST) decoder and second-pass neural network language model rescoring; we compare the proposed (O)PGRU with the standard GRU / Projected LSTM (LSTMP) [7]. Before providing the details of the proposed PGRU-based models, we first describe the LSTMP and standard GRU in this section.

2.1 Projected LSTM

LSTM [5] achieves success in sequence modeling tasks, and several modifications to the original LSTM have been made [6], [14], [15]. LSTMP is a popular variant of LSTM and the standard LSTM architecture under Kaldi [16]. 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 the LSTMP unit. The forget gate [6] is also used to allow the LSTMP unit to adaptively forget or reset the memory cell. Unlike the standard LSTM, every LSTMP unit contains one recurrent projection layer and one non-recurrent projection layer. Note that we use one equivalent single projection layer in place of two separate projection layers. The LSTMP is an important component of our baseline system, and its formulations are as follows:

$$
\begin{aligned}
i_t &= \sigma(W_{ix}x_t + W_{is}s_{t-1} + U_{ic}c_{t-1} + b_i) && (1)\\
f_t &= \sigma(W_{fx}x_t + W_{fs}s_{t-1} + U_{fc}c_{t-1} + b_f) && (2)\\
o_t &= \sigma(W_{ox}x_t + W_{os}s_{t-1} + U_{oc}c_t + b_o) && (3)\\
c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_{cx}x_t + W_{cs}s_{t-1} + b_c) && (4)\\
m_t &= o_t \odot \tanh(c_t) && (5)\\
p_t &= W_{pm}m_t && (6)\\
s_t &= W_{sm}m_t && (7)\\
y_t &= (p_t, s_t) && (8)
\end{aligned}
$$

where $\odot$ denotes the element-wise multiplication; σ (the sigmoid function) and tanh are also applied element-wise. U is the learnable vector (diagonal matrix); p_t is the non-recurrent projection part, and s_t is the recurrent projection part. In all reported experiments, p_t and s_t are one quarter of the cell dimension: for example, the cell dimension may be 1024, so p_t and s_t have dimension 256, and the output y_t has dimension 512.

2.2 GRU

GRU was first proposed by Cho et al. [10]. Similar to the LSTM unit, GRU also has a gating mechanism to modulate the information flow through the unit.

In our experiments, we implement the GRU as follows:

$$
\begin{aligned}
r_t &= \sigma(W_{rx}x_t + W_{rh}h_{t-1} + b_r) && (9)\\
z_t &= \sigma(W_{zx}x_t + W_{zh}h_{t-1} + b_z) && (10)\\
\tilde{h}_t &= \tanh(W_{\tilde{h}x}x_t + W_{\tilde{h}h}(r_t \odot h_{t-1}) + b_{\tilde{h}}) && (11)\\
h_t &= (1 - z_t) \odot \tilde{h}_t + z_t \odot h_{t-1} && (12)\\
y_t &= h_t && (13)
\end{aligned}
$$

where the memory cell activation h_t at time t is a linear interpolation of the previous activation h_{t-1}; 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.

For the standard GRU, the candidate activation is computed similarly to the traditional RNN, where z_t decides to what degree the GRU updates its memory cell activation, and r_t is used to forget the previously computed state. Unlike LSTM, GRU does not have a separate memory cell, and the memory cell of GRU is directly exposed to the next-step calculation.

3. Proposed Model

3.1 Projected GRU

Unlike the standard GRU architecture, the PGRU has one projection layer, whose formulation is:

$$
\begin{aligned}
r_t &= \sigma(W_{rx}x_t + W_{rs}s_{t-1} + b_r) && (14)\\
z_t &= \sigma(W_{zx}x_t + W_{zs}s_{t-1} + b_z) && (15)\\
\tilde{h}_t &= \tanh(W_{\tilde{h}x}x_t + W_{\tilde{h}s}(r_t \odot s_{t-1}) + b_{\tilde{h}}) && (16)\\
h_t &= (1 - z_t) \odot \tilde{h}_t + z_t \odot h_{t-1} && (17)\\
y_t &= W_{yh}h_t && (18)\\
s_t &= y_t[0 : s - 1] && (19)
\end{aligned}
$$

where W_{yh} is the projection matrix, which projects h_t onto y_t with a lower dimension; s_t is the recurrent projection; s is the recurrent projection dimension; and y_t is the output of PGRU. The dimension of r_t is identical to the recurrent projection dimension, and the dimension of z_t is identical to the memory cell dimension. PGRU is illustrated in Fig. 1.

In our setup, the projected memory cell y_t of PGRU contains recurrent and non-recurrent projection parts, the recurrent part will be used as the recurrence of PGRU, and all projected memory cells will be fed into the next layer as the output of PGRU. 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 larger dimension while maintaining a small model size. The presence of the non-recurrent part in the projected memory cell can maintain a larger output dimension without obviously increasing the model size.

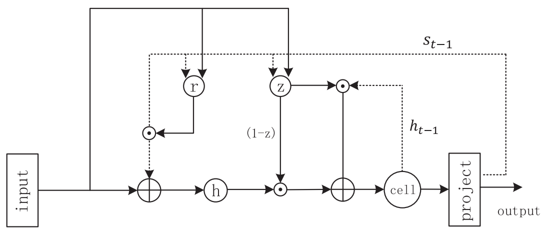

Fig. 1 The structure of PGRU.

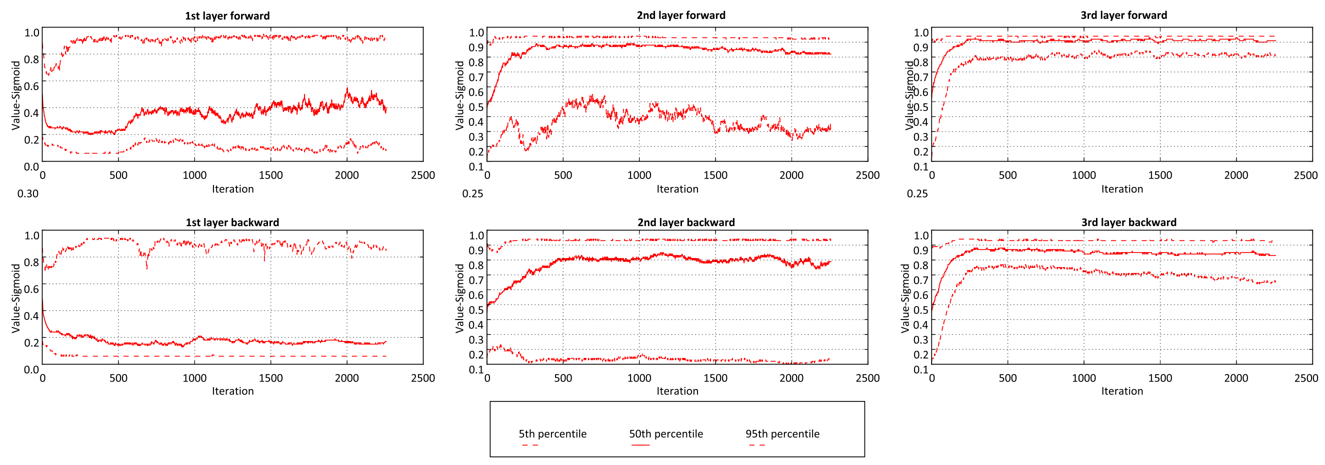

Fig. 2 Sigmoid activation value distribution of different reset gates of 3-layer BPGRU on Fisher + Switchboard ASR task.

3.2 Output-Gate Projected GRU

1. Observation: Fig. 2 shows that the reset gates (sigmoid function) of the 3rd layer of the PGRU tend to become severely saturated, i.e., approach 1. The update gates of PGRU do not saturate, so they are not shown here.
2. Motivation: the reset gate r_t enables the model to delete the past memory by forgetting the previously computed states. The saturated reset gates will tend to remain most of the previously computed states. So the experiments make us believe the reset gate of PGRU is redundant.

We replace the reset gate with an output gate, which helps regulate the projected output of PGRU. And Fig. 4 shows that the output gates of OPGRU are not observed to be saturated. Another modification of PGRU is using h_{t-1} to replace s_{t-1} in Eq. (16). A trainable vector (diagonal matrix), which can be much smaller than matrix, instead of a trainable weight matrix is used to scale h_{t-1}. Our experiments show that OPGRU can achieve better recognition accuracy than PGRU on the large-scale ASR task. Identical to PGRU, the update gates of OPGRU do not exhibit saturation, so they are not shown here.

The equations of the proposed OPGRU are as follows:

$$o_t = \sigma(W_{ox}x_t + W_{os}s_{t-1} + b_o) \tag{20}$$
$$z_t = \sigma(W_{zx}x_t + W_{zs}s_{t-1} + b_z) \tag{21}$$
$$\tilde{h}_t = \tanh(W_{\tilde{h}x}x_t + U_{\tilde{h}h}h_{t-1} + b_{\tilde{h}}) \tag{22}$$
$$h_t = (1 - z_t) \odot \tilde{h}_t + z_t \odot h_{t-1} \tag{23}$$
$$\tilde{y}_t = o_t \odot h_t \tag{24}$$
$$y_t = W_{y\tilde{y}}\tilde{y}_t \tag{25}$$
$$s_t = y_t[0 : s - 1] \tag{26}$$

where $U_{\tilde{h}h}$ is the learnable vector. The dimension of o_t and z_t is identical to the memory cell dimension. The OPGRU is illustrated in Fig. 3.

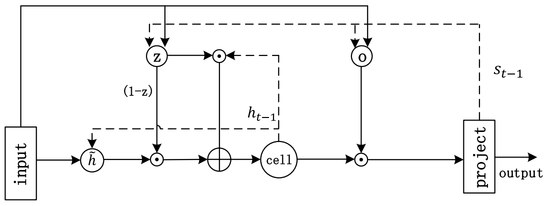

Fig. 3 The structure of OPGRU.

4. Experimental Setup

All of our experiments were conducted using the Kaldi speech recognition toolkit [16]. We focus on the 2000 hr Fisher+Switchboard large vocabulary continuous speech recognition (LVCSR) task, but we also report a part of the results on the 80 hr AMI SDM [17], [18], 200 hr Ted-lium [19], and 300 hr Switchboard (SWBD) LVCSR tasks. The training criterion is phone-level sequence training from scratch using the lattice-free MMI objective [8] on the outputs of frame rate 33 Hz. The 40-dimensional Mel-frequency cepstral coefficients (MFCCs) without cepstral truncation are used as the input into the neural network [20].

The experimental setups for AMI SDM, Ted-lium and Switchboard are the same as those described in [21], the AMI SDM LVCSR system is trained with numerator lattices generated from the parallel AMI individual headset microphone (IHM) data [22]. The Fisher + Switchboard LVCSR system is the same as that in [8]. We use the speed-perturbation technique [23] for the 3-fold data augmentation and iVector to perform the instantaneous adaption of the neural network [24]. The i-Vector is used to provide information about the mean offset of the speaker's data, so the cepstral mean or varaince normalization is not necessary in our setup.

4.1 Neural Network Configuration for the Acoustic Model

The bidirectional LSTMP (BLSTMP) and TDNN-LSTMP

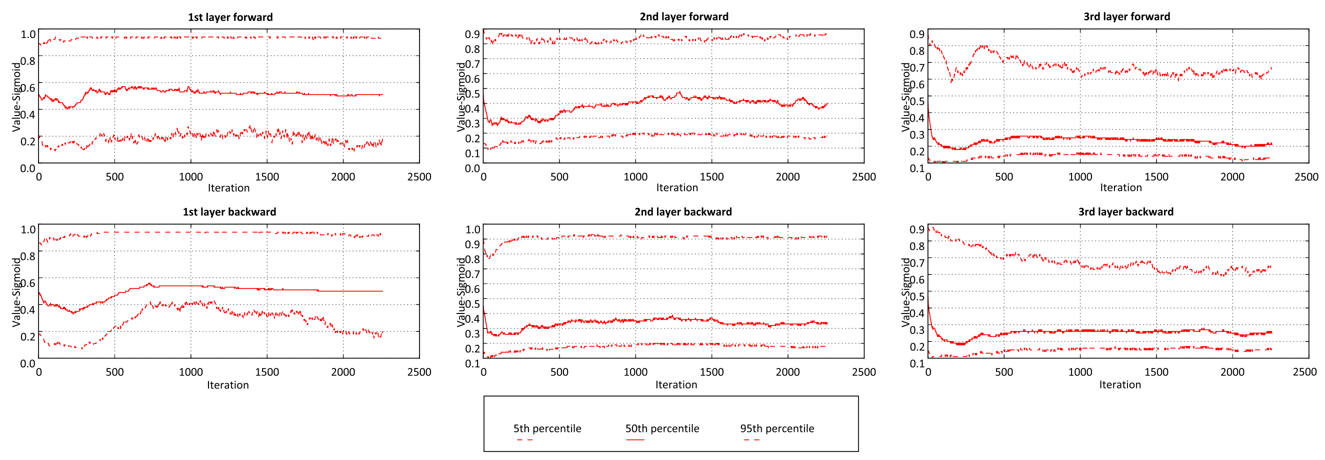

Fig. 4 Sigmoid activation value distribution of different output gates of the 3-layer BOPGRU on Fisher + Switchboard ASR task.

Table 1 Configuration for BLSTMP, BPGRU, TDNN-LSTMP, TDNN-PGRU and TDNN-OPGRU. The unit of latency is millisecond (ms).

Acoustic 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	$T^{100}\ T^{100}\ T^{100}\ L_f\ T\ T\ L_f\ T\ T\ L_f$	210
TDNN-PGRU	$T^{100}\ T^{100}\ T^{100}\ P_f\ T\ T\ P_f\ T\ T\ P_f$	210
TDNN-OPGRU	$T^{100}\ T^{100}\ T^{100}\ O_f\ T\ T\ O_f\ T\ T\ O_f$	210

[1] Forward LSTMP - L_f, backward LSTMP - L_b, forward PGRU - P_f, backward PGRU - P_b, TDNN - T, forward OPGRU - O_f, the default layer frame-rate is 33 Hz, which is the default output frame rate for the 'Chain' model under Kaldi [25]. Other frame rates are specified in the superscript.

[2] The acoustic model decoding latency is affected by input context, chunk-width, chunk contexts and output delay. See [25] for the definition of latency. We use the same training and decoding configuration for BLSTMP as that in [25]. TDNN-PGRU and TDNN-OPGRU have the same configuration as TDNN-LSTM-C in [25] only with the LSTMP layers replaced, so they should have the same latency. We use a broader input context for the unidirectional model, so the latency is 210 ms instead of 200 ms, which is reported in [25].

neural networks are identical to the models described in [21], [25]. Regarding the projection layers in LSTMP, PGRU and OPGRU in this paper, the dimensions of the recurrent and non-recurrent projections are always one-quarter of the cell dimension. The default cell dimension is 1024 unless specified.

All recurrent models are trained and decoded on the fixed length context-sensitive chunks (CSCs) [26]. The acoustic model training cost function is computed using CSCs of width 1500 ms and a left/right context of 400 ms [†]. During decoding, we use the identical chunk-width as training but increase the chunk-context to 500 ms to reduce the WER [25]. The model architectures and latency information are shown in Table 1.

4.2 Language Model

We focused on the large-scale LVCSR task, and the neural network language model (NN-LM) experiments were conducted under the 2000 hr Fisher+Switchboard LVCSR task. Thus, only the language model setup for Fisher+Switchboard LVCSR task is described in this section.

The initial decoding was conducted with a Kaldi WFST decoder using 3-gram LM for the first pass decoding and 4-gram LM for the second pass LM rescoring. The n-gram LMs were trained from the Switchboard and Fisher training audio transcripts with the first 10000 sentences of the training sets as the validation and tuning sets. The perplexity (PPL) of the 3-gram LM and 4-gram LM over the validation set (8.8M words) is 70.4 and 66.9, respectively.

For NN-LM rescoring, we followed the setup described in [27], which directly rescored on the lattice; the word embedding dimension used in our setup is 1024. For the training data sets, in addition to Switchboard and Fisher transcripts, the out-domain Washington conversational Web corpus (191M words) was used. To balance the in-domain and out-domain transcripts, we duplicated the Switchboard / Fisher transcripts 6 / 2 times. The validation set for the NN-LM training was selected every one hundred lines of the Switchboard and Fisher transcripts (24M words). The lattices generated by the 4-gram LM were used for the NN-LM rescoring. The neural network language model and n-gram language model use different validation and training data sets, so the reported PPLs are not comparable across these two systems.

5. Results of the Acoustic Model

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 in the bidirectional variant. For fair comparison, we tuned the parameter size of both BPGRU and BGRU to maintain them with identical model sizes. Extensive experiments from Table 2 show that BPGRU can gain consistent improvement over BGRU. In the AMI SDM LVCSR task, BPGRU achieves a relative WER reduction of approximately 7.5% ~ 8.0% over BGRU,

[†] 1500 ms is equivalent to 150 frames of features.

Table 2 WER for BGRU and BPGRU

Model	AMI SDM					300 Hr Switchboard					
	Parameter Size[1]	Cell Dimension	Layer Number	WER(%)		Parameter Size	Cell Dimension	Layer Number	WER(%) on Eval2000		
				Dev	Eval				SWBD	Callhome	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+Switchboard					
	Parameter Size	Cell Dimension	Layer Number	WER(%)		Parameter Size	Cell Dimension	Layer Number	WER(%) on Eval2000		
				Dev	Eval				SWBD	Callhome	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]. Different LVCSR tasks have different context-dependent state decision trees. This means even with the same cell dimension, the acoustic model of same architecture will vary in model parameter size.

Table 3 WER after interleaving PGRU with TDNN.

Model on AMI SDM	Parameter Size	WER(%)	
		Dev	Eval
BLSTMP	36.4M	39.0	42.3
BPGRU	24.8M	37.8	41.1
TDNN-PGRU	36.5M	35.9	39.0
Model on TED-LIUM	Parameter Size	WER(%)	
		Dev	Eval
BLSTMP	40.6M	8.2	8.6
BPGRU	29.4M	8.1	8.4
TDNN-PGRU	30.2M	8.0	7.7
Model on SWBD	Parameter Size	WER(%) on Eval2000	
		SWBD	Total
BLSTMP	41.2M	9.3	14.3
BPGRU	30.0M	9.4	14.0
TDNN-PGRU	32.7M	9.0	13.3
Model on Fisher+SWBD	Parameter Size	WER(%) on Eval2000	
		SWBD	Total
BLSTMP	41.3M	8.5	12.0
BPGRU	30.1M	9.9	13.9
TDNN-PGRU	32.8M	9.1	12.9

Table 4 WER of various unidirectional acoustic models.

Model on AMI SDM	Parameter Size	WER(%)	
		Dev	Eval
TDNN-LSTMP	43.4M	37.2	40.6
TDNN-PGRU	36.5M	35.9	39.0
TDNN-OPGRU	38.7M	36.3	39.6
Model on Ted-lium	Parameter Size	WER(%)	
		Dev	Test
TDNN-LSTMP	37.1M	8.1	8.4
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	Total
TDNN-LSTMP	39.6M	9.2	14.1
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	Total
TDNN-LSTMP	39.7M	8.2	12.0
TDNN-PGRU	32.8M	9.1	12.9
TDNN-OPGRU	34.9M	8.6	12.0

and the figure is 8.7% ~ 15% on Ted-lium, 6.6% ~ 8.8% on Switchboard, and 2.1% ~ 4.5% 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 and Callhome subset results in the table. We believe that PGRU outperforms standard GRU because the PGRU can preserve a larger memory cell dimension than the standard GRU with the same model size.

5.2 Interleaving TDNN with PGRU

It is challenging to deploy bidirectional recurrent neural networks in a low-latency setting because they must process the entire sample before giving results. We built unidirectional models by combining the forward-only PGRU with TDNN (TDNN-PGRU), which was proven to outperform the BPGRU. Similar to [25], the authors also found that interleaving the unidirectional LSTM with TDNN could perform the same or better than the BLSTM.

Although TDNN is a feed-forward architecture, computing all previous hidden activations at all time steps will be computationally expensive, particularly when one builds a deep and wide-context TDNN. Thus, in our work, we use the sub-sampling TDNN [3] to model the long-term dependency while maintaining a lower computational cost. With the presence of TDNN, at time step t, the input context into one PGRU at layer l will change from the current frame to the time context window.

Table 3 shows that compared with BPGRU, TDNN-PGRU gains a 5.0% relative reduction in WER on the AMI SDM LVCSR task, and the figure is 4.7% on the Switchboard LVCSR task, 5.4% on the Ted-lium LVCSR task, and 7.7% on the Fisher + Switchboard LVCSR task. Overall, the improvement is consistent. By interleaving PGRU and TDNN, we achieved a 5.1% relative reduction in WER over the BPGRU averaged on all LVCSR tasks in Table 3. The improvement indicates that the temporal model is beneficial for the proposed PGRU.

5.3 TDNN-PGRU vs TDNN-OPGRU

Table 3 shows that TDNN-PGRU outperforms BLSTMP on 3 of 4 tested LVCSR tasks but is obviously worse than BLSTMP on the 2000 hr Fisher+Switchboard task. To improve the generalization of the projected-based GRU, particularly on large-scale LVCSR tasks, we proposed OPGRU and tested it in the TDNN hybrid style.

The configuration of TDNN-OPGRU is shown in Table 1. Table 4 shows that for the three small-scale LVCSR tasks (AMI, Ted-lium and Switchboard), TDNN-OPGRU performs almost the same as the TDNN-PGRU, but for the 2000 hr Fisher+Switchboard LVCSR task, TDNN-OPGRU gains an obvious improvement over TDNN-PGRU. From

Table 4, we can conclude that OPGRU generalizes better than PGRU across different datasets.

5.4 Normalization in OPGRU and PGRU

Previous works have applied batch normalization [28] on recurrent neural networks [29]. In general, people avoid directly involving batch normalization in the recurrence. In [29], the author used a rectified linear unit (ReLU) for the GRU and used the batch normalization after the ReLU.

For PGRU and OPGRU, we tested two methods to normalize the (O)PGRU. In the first method, after the projection layer, we applied batch normalization in the feed-forward direction of the projected output, which is denoted as *norm*1. In the second method, we applied batch normalization in the feed-forward direction of the projected output. For the recurrent projection part, we normalized them as Hinton's layer normalization [30], except there is no mean subtraction; there is only variance normalization, which is denoted as *norm*2. Similarly, we also tried the *norm*1 and *norm*2 on the LSTMP.

Table 5 shows that both *norm*1 and *norm*2 can improve the performance of BPGRU/BOPGRU and that *norm*2 is more effective than *norm*1. Thus, *norm*2 is used, and we call PGRU / OPGRU with *norm*2 NormPGRU / NormOPGRU. Table 5 also shows that the TDNN-NormPGRU gains 7.0% and 8.8% relative reduction in WER compared with TDNN-PGRU on the total Eval2000 set and total RT03 set, respectively. Similar to TDNN-PGRU, the figures for TDNN-NormOPGRU over TDNN-OPGRU are 1.7% and 6.0% on total Eval2000 and total RT03, respectively.

In our setup, as Table 5 shows, the proposed normalization methods did not benefit the TDNN-LSTMP or BLSTMP obviously. So the vanilla LSTMP is used as the baseline in the remainder of the paper.

5.5 Comparing NormOPGRU with LSTMP

For comparison, we show the BLSTM results from Xiong et al.[1] in Table 6. For the n-gram LM decoding setup, we only use fisher and switchboard training transcripts to build the 4-gram language model, but Xiong et al. [1] used the extra Washington conversational Web corpus (191M) for the language model. Our BLSTMP baseline is comparable to those of [1].

Table 6 shows that by applying both batch normalization in the TDNN layers and per-frame dropout [21] in the LSTMP / NormOPGRU, we can slightly improve the decoding results. The WERs in bold in Table 6 show that compared with the BLSTMP / TDNN-LSTMP, the proposed TDNN-NormOPGRU can achieve 3.3% / 4.5% relative reduction in WER on the total Eval2000 / RT03 test sets.

5.6 Accelerating Decoding with State Saving

For the standard Kaldi decode setup, we decode the recurrent neural network in the matched way [25], we train the acoustic model on fixed-length CSCs in random order and use the same chunk-width as training for decoding. However, the extra left/right context will introduce a decoding redundancy during the recurrent network decoding. For the unidirectional acoustic model, we decode with the state saving [25] across the chunks to reduce the redundant computation.

As Table 7 shows, in the state-saving scenario, TDNN-

Table 5 Comparison between two normalization methods.

Model on 2000 Hr Fisher+SWBD	Eval2000		RT03	
	SWBD	Total	Fsh	Total
BPGRU	9.9	13.9	10.7	13.4
BPGRU-norm1	9.7	13.6	10.5	12.8
BPGRU-norm2 (BNormPGRU)	8.9	12.6	9.7	11.9
BOPGRU	9.3	13.2	9.9	12.3
BOPGRU-norm1	9.0	13.1	10.0	12.4
BOPGRU-norm2 (BNormOPGRU)	8.7	12.7	9.7	11.8
BLSTMP	8.5	12.0	9.7	11.6
BLSTMP-norm1	8.2	12.1	9.8	11.8
BLSTMP-norm2 (BNormLSTMP)	8.0	12.0	9.4	11.5
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

Table 7 Real time factor (RTF) for various acoustic 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	1.31
+state saving decode [25]	8.1	12.1	9.4	11.4	0.99
TDNN-NormOPGRU+dropout	8.2	11.6	8.5	10.7	0.89
+state saving decode [25]	8.2	11.6	8.6	10.7	0.66

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

Table 6 WER comparison between LSTMP and OPGRU.

Model on 2300 Hr Fisher+SWBD	Eval2000			RT03		
	SWBD	CH	Total	Fsh	SWBD	Total
Xiong et al. [1] BLSTM, spatial smoothing	8.6	15.4	-	-	-	-
Xiong et al. [1] BLSTM, spatial smoothing, 27k senones	8.3	15.3	-	-	-	-
BLSTMP	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	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	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	8.2	14.6	**11.6**	8.5	12.6	**10.7**

NormOPGRU [†] can speed up the decoding by 2.6 times compared with BLSTMP. The state-saving decoding LSTM acoustic models always slightly degrade the performance compared with their non-state-saving counterparts [25]. Thus, in the state-saving decoding setup, the proposed TDNN-NormOPGRU is 4.1% / 6.1% better than TDNN-LSTMP in WER and speeds up the decoding by 1.5 times.

6. Results with Neural Language Model Rescoring

6.1 LSTMP-LM vs OPGRU-LM

Table 8 shows the PPL and WER for various RNN-LM setups. The acoustic model to generate the lattice was BLSTMP. We tested the unidirectional LSTMP-LM / OPGRU-LM of various layers. In Table 8, the multi-layer RNN-LM is always better than the single-layer RNN-LM. In addition, the TDNN-{OPGRU,LSTMP} hybrids, which are similar to the hybrid models used for acoustic models, can further improve the RNN-LM rescoring performance. In terms of PPL, LSTMP-LM outperforms OPGRU-LM on the train set, but OPGRU-LM obtains better generalization on the validation subset. From the WER perspective, the OPGRU-LM always outperforms their LSTMP-LM counterparts in speech recognition. OPGRU-LM gains 0.2% ∼ 0.3% absolute (1.8% ∼ 2.6% relative) WER reduction compared with LSTMP-LM.

The rescoring time is another critical consideration in some scenes, particularly online ASR deployment. Table 9 lists the rescoring time of various RNN-LMs. Compared with LSTMP-LM, OPGRU-LM rescoring is consistently faster: it achieves at most 1.75 times rescoring speed-up.

Table 8 PPL and WER of LSTMP-LM and OPGRU-LM. PPL is computed on the held-out subset of Switchboard and Fisher training transcripts; WERs are shown on total Eval2000 test set and Switchboard subset of total Eval2000. The held-out validation subset of 4-gram baseline LM is different from the one used for NN-LM training, so we do not show the PPL for 4-gram baseline here.

Language Model: Architecture	PPL		Eval2000	
	Train	Dev	SWBD	Total
4-gram baseline	-	-	8.4	12.0
1-layer LSTMP : L_f	75.9	67.4	7.9	11.4
2-layer LSTMP : L_f, L_f	71.6	65.4	7.8	11.2
TDNN-LSTMP : T, L_f, T, L_f	67.3	62.9	7.7	11.1
1-layer OPGRU : O_f	76.9	48.2	7.4	11.1
2-layer OPGRU : O_f, O_f	72.9	49.1	7.4	11.0
TDNN-OPGRU : T, O_f, T, O_f	69.1	47.6	7.3	10.9

Table 9 Neural network language model rescoring speed comparison between LSTMP and OPGRU. The acoustic model (AM) to generate the lattice is 3-layer BLSTMP, the notation $Time$ indicates the amount of time that the NN-LM requires to finish rescoring on the lattice; the unit is minute (min).

Time	1-layer model	2-layer model	TDNN-hybrid model
LSTMP	35 min	63 min	93 min
OPGRU	22 min	41 min	70 min

Table 10 WER (%) for LSTM-based single system and OPGRU-based single system, acoustic models are trained on 2000 hr Switchboard+Fisher data.

Model on Fisher+SWBD	Eval2000		RT03
	SWBD	Total	
BLSTMP	8.4	12.0	11.2
+TDNN-LSTMP-LM	7.7	11.1	10.6
TDNN-LSTMP	8.1	12.1	11.4
+TDNN-LSTMP-LM	7.5	11.0	10.5
TDNN-NormOPGRU	8.2	11.6	10.7
+TDNN-OPGRU-LM	6.7	10.1	9.6

[†]Scripts to reproduce the experiments [31].

6.2 LSTMP Single System vs OPGRU Single System

Unlike [1], our work aims to build a fast and deployable single system with higher accuracy. Hence, we did not try various acoustic model combinations or neural network language model combinations. The final proposed OPGRU-based single system just contains one single TDNN-OPGRU acoustic / language model.

The comparison between the OPGRU-based system and the LSTMP-based system is presented in Table 10. The best LSTMP system is the TDNN-LSTMP hybrid one, which was applied for both acoustic model and language model. It achieves 11.0% / 10.5% on the total Eval2000 / RT03 test sets, which is 1.0% / 0.7% absolute (8.3% / 6.3% relative) better than the 4-gram BLSTMP baseline. The best OPGRU system is the one with the TDNN-NormOPGRU acoustic model and TDNN-OPGRU language model: it reaches 10.1% / 9.6% on total Eval2000 and RT03, which gain 0.9% / 0.9% absolute (8.2% / 8.6% relative) reduction in WER compared with the best LSTMP single system.

Considering the RTF and neural network LM rescoring time, we can conclude that the proposed OPGRU-based single system is faster and more accurate than the LSTMP-based system.

7. Conclusions

In this paper, we described our work on building a more efficient and accurate single ASR system. Instead of attempting various model combination techniques, we focused on a deployable single ASR system. To reduce the complexity of LSTMP, which is the standard recurrent unit under Kaldi, the OPGRU was proposed as an alternative. Compared with LSTMP, the simple OPGRU has only two gates and a smaller model size. On the acoustic model side, to further improve the performance of projected-based GRU, we combined them with TDNN and applied batch normalization and a variant of layer normalization to them. The proposed TDNN-NormOPGRU acoustic model achieves faster decoding speed and higher decoding accuracy than our old LSTMP system. On the neural network language model side, TDNN-OPGRU outperforms the TDNN-LSTMP with higher rescoring efficiency and better recognition accuracy.

Acknowledgments

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).

References

[1] 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.

[2] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, J. Chen, J. Chen, Z. Chen, M. Chrzanowski, A. Coates, G. Diamos, K. Ding, N. Du, E. Elsen, J. Engel, W. Fang, L. Fan, C. Fougner, L. Gao, C. Gong, A. Hannun, T. Han, L.V. Johannes, B. Jiang, C. Ju, B. Jun, P. LeGresley, L. Lin, J. Liu, Y. Liu, W. Li, X. Li, D. Ma, S. Narang, A. Ng, S. Ozair, Y. Peng, R. Prenger, S. Qian, Z. Quan, J. Raiman, V. Rao, S. Satheesh, D. Seetapun, S. Sengupta, K. Srinet, A. Sriram, H. Tang, L. Tang, C. Wang, J. Wang, K. Wang, Y. Wang, Z. Wang, Z. Wang, S. Wu, L. Wei, B. Xiao, W. Xie, Y. Xie, D. Yogatama, B. Yuan, J. Zhan, and Z. Zhu, "Deep speech 2 : End-to-end speech recognition in english and mandarin," vol.48, pp.173–182, June 2016.

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

[4] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," IEEE Trans. Neural Networks, vol.5, no.2, pp.157–166, March 1994.

[5] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Comput., vol.9, no.8, pp.1735–1780, Nov. 1997.

[6] F.A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with lstm," Ninth Int. Conf. Artificial Neural Networks ICANN, pp.850–855 vol.2, 1999.

[7] H. Sak, A.W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," 15th Annual Conference of the International Speech Commun. Association (Interspeech), Singapore, Sept. 14-18, 2014, pp.338–342, 2014.

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

[9] G. Saon, T. Sercu, S. Rennie, and H.J. Kuo, "The IBM 2016 english conversational telephone speech recognition system," CoRR, vol.abs/1604.08242, 2016.

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

[11] 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.

[12] Z. Tang, Y. Shi, D. Wang, Y. Feng, and S. Zhang, "Memory visualization for gated recurrent neural networks in speech recognition," 2017 IEEE Int. Conf. Acoustics, Speech Signal Process. (ICASSP), pp.2736–2740, March 2017.

[13] K. Irie, Z. Tüske, T. Alkhouli, R. Schlüter, and H. Ney, "Lstm, gru, highway and a bit of attention: An empirical overview for language modeling in speech recognition," Interspeech, 2016.

[14] Y. Zhang, G. Chen, D. Yu, K. Yaco, S. Khudanpur, and J. Glass, "Highway long short-term memory rnns for distant speech recognition," 2016 IEEE Int. Conf. Acoustics, Speech Signal Process. (ICASSP), pp.5755–5759, IEEE, 2016.

[15] G. Pundak and T. Sainath, "Highway-lstm and recurrent highway networks for speech recognition," Proc. Interspeech 2017, 2017.

[16] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," IEEE 2011 workshop on automatic speech recognition and understanding, IEEE Signal Processing Society, 2011.

[17] J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal, G. Lathoud, M. Lincoln, A. Lisowska, I. McCowan, W. Post, D. Reidsma, and P. Wellner, "The ami meeting corpus: A pre-announcement," International Workshop on Machine Learning for Multimodal Interaction, pp.28–39, Springer, 2005.

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

[19] A. Rousseau, P. Deleglise, and Y. Esteve, "Ted-lium: an automatic speech recognition dedicated corpus," Proc. Eight Int. Conf. Language Resources and Evaluation (LREC'12), Istanbul, Turkey, May 2012.

[20] 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.

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

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

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

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

[25] V. Peddinti, Y. Wang, D. Povey, and S. Khudanpur, "Low latency acoustic modeling using temporal convolution and lstms," IEEE Signal Process. Lett. , vol.25, no.3, pp.373–377, March 2017.

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

[27] H. Xu, T. Chen, D. Gao, Y. Wang, K. Li, N. Goel, Y. Carmiel, D. Povey, and S. Khudanpur, "A pruned rnnlm lattice-rescoring algorithm for automatic speech recognition," IEEE Int. Conf. Acoustics, Speech Signal Process. (ICASSP), 2018.

[28] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," Proc. 32nd Int. Conf. Machine Learning, vol.37, pp.448–456, July 2015.

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

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

[31] "Code to reproduce the experiment results in this paper can be seen in the official kaldi repository," https://github.com/kaldi-asr/kaldi, accessed April 8, 2018.

Gaofeng Cheng received the B.S. in School of Science from Beijing University of Posts and Telecommunications in 2014. He is currently belonging to the School of Electronic, Electrical and Communication Engineering of University of Chinese Academy of Sciences. He research interests include speech recognition and large-scale deep neural networks.

Pengyuan Zhang received the Ph.D. in the Institute of Acoustic, Chinese Academy of Sciences. He is a now a researcher in the Key Laboratory of Speech Acoustics and Content Understanding, Institute of Acoustics, Chinese Academy of Sciences. His research interests include large vocabulary continuous speech recognition.

Ji Xu received the B.S. and M.S. degrees in Department of Electronic Engineering of Tsinghua University and received Ph.D. in the Institute of Acoustic, Chinese Academy of Sciences. He is a now a researcher in the Key Laboratory of Speech Acoustics and Content Understanding, Institute of Acoustics, Chinese Academy of Sciences. His research interests include signal processing and deep learning.