A PRIORITIZED GRID LONG SHORT-TERM MEMORY RNN FOR SPEECH RECOGNITION

Wei-Ning Hsu, Yu Zhang, and James Glass

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
{wnhsu,yzhang87,glass}@mit.edu

ABSTRACT

Recurrent neural networks (RNNs) are naturally suitable for speech recognition because of their ability of utilizing dynamically changing temporal information. Deep RNNs have been argued to be able to model temporal relationships at different time granularities, but suffer vanishing gradient problems. In this paper, we extend stacked long short-term memory (LSTM) RNNs by using grid LSTM blocks that formulate computation along not only the temporal dimension, but also the depth dimension, in order to alleviate this issue. Moreover, we prioritize the depth dimension over the temporal one to provide the depth dimension more updated information, since the output from it will be used for classification. We call this model the prioritized Grid LSTM (pGLSTM). Extensive experiments on four large datasets (AMI, HKUST, GALE, and MGB) indicate that the pGLSTM outperforms alternative deep LSTM models, beating stacked LSTMs with 4% to 7% relative improvement, and achieve new benchmarks among uni-directional models on all datasets.

Index Terms— speech recognition, recurrent neural network, grid long short-term memory

1. INTRODUCTION

Recently, neural network-based (NN) acoustic models have greatly improved automatic speech recognition (ASR) performance over traditional Gaussian mixture models (GMMs) on a variety of tasks [1, 2, 3, 4]. Further improvement has been observed by applying more advanced neural network architectures, such as convolutional neural networks [5, 6] and time delay neural networks [7]. Among this line of work, recurrent neural networks (RNNs), especially long short-term memory (LSTM) RNNs, that are capable of utilizing dynamically changing contextual windows over the sequence history, have achieved state-of-the-art performance on numerous tasks [8, 9, 10, 11, 12, 13, 14].

As observed in [15], the speech signal encodes acoustic-phonetic information at different time scales. Deep RNNs have been argued to be capable of modeling the temporal relationships at different time granularities [16]. Therefore, we would like to build deeper LSTM RNNs to achieve better performance; however, such models can suffer from the vanishing gradient problem. To alleviate this issue, several architectures have been proposed [11, 17, 18]. One approach named grid LSTM [18] provides a unified framework for both deep and sequential computation by arranging LSTM blocks into multidimensional grids such that each grid contains one LSTM block for each dimension, and a per-dimension gated linear dependence between adjacent cells is introduced.

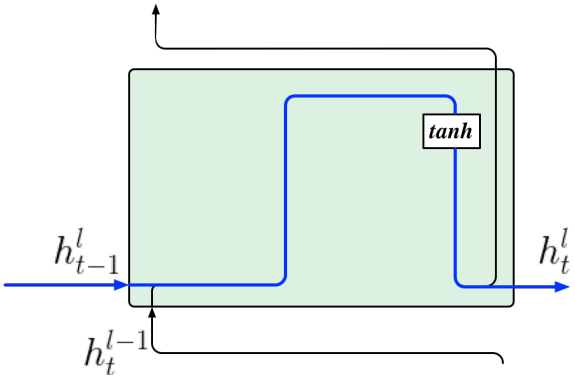
In this paper, we adapt grid LSTM models for acoustic modeling, where each grid contains a time-LSTM and a depth-LSTM for deep computation and temporal computation, respectively. Furthermore, we modify the grid such that the depth dimension is prioritized over the temporal dimension, and call this model a *prioritized* grid LSTM. While the concept of prioritization was described in [18], we believe this paper is the first to apply the concept to speech recognition.

Extensive experiments are conducted on four highly diverse datasets, which range from 100 hours to 1200 hours, include three languages, and cover multiple speech scenarios as well as sampling rates. Initial experiments on two datasets affirm the importance of prioritizing the depth dimension and show that prioritized grid LSTMs outperform LSTMs, highway LSTMs, and residual LSTMs. Further experiments on all datasets indicate that the prioritized grid LSTM consistently outperforms all baseline approaches, and achieves the best performance reported on those datasets among uni-directional models. Lastly, we show that prioritized grid LSTMs can largely benefit from sequence discriminative training.

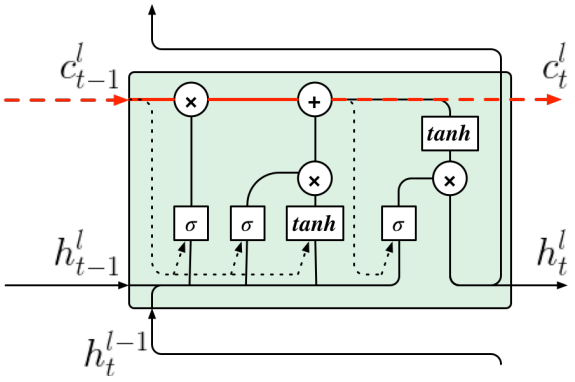
The rest of paper is organized as follows, In Section 2, we describe our models. In Section 3, we briefly discuss related work. The experimental setup is summarized in Section 4, followed the results and discussion in Section 5. Finally, we conclude our work in Section 6.

2. METHOD

In this section, we first explain the working principles of LSTM RNNs, which help us appreciate what problems traditional RNNs have can be alleviated through this design. Next, the vanishing gradient problem encountered in deep neural network structures is illustrated in the context of stacked LSTM models. Subsequently, we introduce an alternative architecture called grid LSTM, which provides a solution to the gradient issue. Finally, we propose two models based on this architecture, which we call prioritized and non-prioritized grid LSTM RNNs, respectively, for acoustic modeling.



(a) RNN block



(b) LSTM block

Fig. 1. Comparison between RNN and LSTM blocks.

2.1. Long Short-Term Memory RNNs

Recurrent neural networks (RNNs) are variants of feed-forward neural networks, which contain feedback loops that feed activations not only to the next layer, but also as the input to the current layer at the next time step. This design enables the network to consider all contexts from the past in order to make a decision about the current frame, which is a desirable property since contextual information plays an important role in acoustic modeling.

However, in practice RNNs cannot preserve information over a long period. This is because commonly-used activation functions, such as the sigmoid function and hyperbolic tangent function, compress the input into a small dynamic range; therefore, the activations of the same layer from n-steps before would have been compressed n times by the time it arrives at the current time step, and thus has a minor influence. While training with back-propagation through time (BPTT), this problem is also known as the vanishing gradient problem in the sense that error signals are not likely to back-propagate along time dimension for many hops.

To address this issue, the LSTM block was proposed in [19] to replace the traditional hidden unit. An LSTM block is composed of an array of memory cells $\mathbf{c}$ as well as three gates: $\mathbf{i}$, $\mathbf{f}$, and $\mathbf{o}$, which are used to control information flow. They are defined as follows:

$$\mathbf{x}_t^l = [\mathbf{h}_t^{l-1}; \mathbf{h}_{t-1}^l] \tag{1}$$

$$\mathbf{i}_t^l = \sigma(\mathbf{W}_i^l \mathbf{x}_t^l + \mathbf{U}_i^l \mathbf{c}_{t-1}^l + \mathbf{b}_i^l) \tag{2}$$

$$\mathbf{f}_t^l = \sigma(\mathbf{W}_f^l \mathbf{x}_t^l + \mathbf{U}_f^l \mathbf{c}_{t-1}^l + \mathbf{b}_f^l) \tag{3}$$

$$\hat{\mathbf{c}}_t^l = \tanh(\mathbf{W}_c^l \mathbf{x}_t^l + \mathbf{U}_c^l \mathbf{c}_{t-1}^l + \mathbf{b}_c^l) \tag{4}$$

$$\mathbf{c}_t^l = \mathbf{f}_t^l \odot \mathbf{c}_{t-1}^l + \mathbf{i}_t^l \odot \hat{\mathbf{c}}_t^l \tag{5}$$

$$\mathbf{o}_t^l = \sigma(\mathbf{W}_o^l \mathbf{x}_t^l + \mathbf{U}_o^l \mathbf{c}_t^l + \mathbf{b}_o^l) \tag{6}$$

$$\mathbf{h}_t^l = \mathbf{W'}_{proj}^l (\mathbf{o}_t^l \odot \tanh(\mathbf{c}_t^l)) \tag{7}$$

where $\mathbf{c}_t^l$, and $\mathbf{h}_t^l$ are the cell state and cell output respectively at time t and layer l; specifically, cell output at 0-th layer h_t^0 refers to the input feature at time t. $\mathbf{W}_*$, $\mathbf{U}_*$ and $\mathbf{b}_*$ are weight matrices and bias vectors connecting different gates. $\odot$ denotes an element-wise product. Note that $\mathbf{U}_*$ are "peephole connections" [20] that only exist within each block, and since the number of cells in each LSTM block is set to one, $\mathbf{U}_*$ are diagonal matrices here. $\mathbf{W'}_{proj}$ is a projection matrix as proposed in [9].

A graphical comparison between RNN blocks and LSTM blocks is shown in Figure 1. Solid lines are used to denote inputs that are shared among all LSTM blocks, while dashed lines are used to denote inputs that are only passed to the corresponding LSTM blocks at the next time step. In addition, we use dotted lines to denote peephole connections. The key difference is the approach of storing and propagating memories. LSTM blocks store memories in the form of memory cell states, as opposed to RNN memories which are in the form of hidden state activations; more importantly, a gated linear dependence is introduced between memory cell states across two consecutive time steps, as indicated by the bold red line in Figure 1(b), which allows memories to be preserved.

2.2. Vanishing Gradient Along Depth Dimension

As mentioned earlier, we would like the LSTM RNNs to model temporal relationships in the speech signal at different time granularities, since deep RNNs have been argued to be able to learn them [16]. The most naïve way to build a deep LSTM RNN is by stacking multiple LSTM layers, as illustrated in Figure 2.

We denote the input signal flow along the depth dimension with the bold blue line in Figure 2; the error signals on the other hand flow in the opposite direction along depth dimension. As we increase the depth of the model, we can see how deep LSTM RNN models can suffer from the vanishing gradient problem, not along the time dimension, but along the depth dimension. Empirically, it is also shown in [11] that deep LSTM models deteriorate significantly when increasing the depth from 3 layers to 8 layers.

2.3. Grid Long Short-Term Memory RNNs

The grid LSTM RNN was first introduced in [18]. Unlike traditional LSTM RNN models, which organize LSTM blocks as a temporal chain, grid LSTM RNN models arrange LSTM

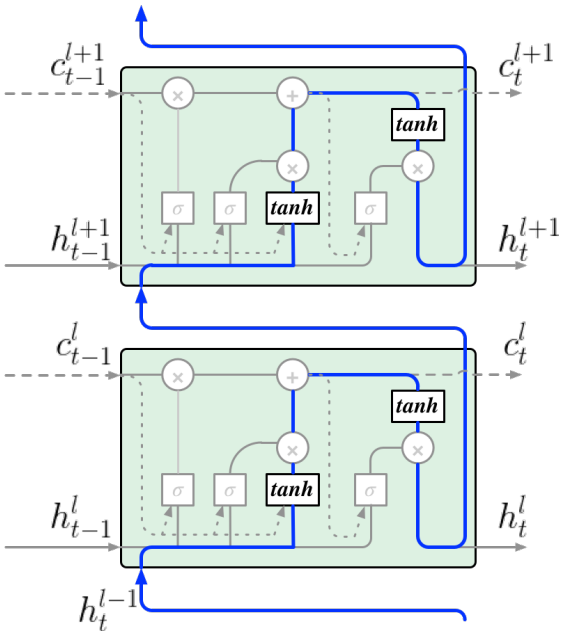


Fig. 2. Illustration of a simple stacked LSTM RNN model.

blocks into multidimensional grids such that each grid contains one set of LSTM blocks for each dimension, including the depth dimension. This architecture introduces per-dimension gated linear dependencies between adjacent cell states, which mitigates the vanishing gradient problem along all dimensions.

Here we consider a two-dimensional grid LSTM model for acoustic modeling, which are time and depth dimensions respectively, as illustrated in Figure 3. The computations in each grid are defined as follows:

$$\mathbf{x}_{t,l} = [\mathbf{h}_{t,l-1}^D; \mathbf{h}_{t-1,l}^T] \tag{8}$$

$$(\mathbf{h}_{t,l}^T, \mathbf{c}_{t,l}^T) = \text{TIME-LSTM}(\mathbf{x}_{t,l}, \mathbf{c}_{t-1,l}^T, \Theta^T) \tag{9}$$

$$(\mathbf{h}_{t,l}^D, \mathbf{c}_{t,l}^D) = \text{DEPTH-LSTM}(\mathbf{x}_{t,l}, \mathbf{c}_{t,l-1}^D, \Theta^D) \tag{10}$$

Note that we slightly change the notation here by using subscripts to denote both time and depth, and using superscripts to indicate a specific set of LSTM blocks. $c_{t,l}^i$ and $h_{t,l}^i$ are cell state and cell output respectively at time t and layer l of i-LSTM, while Θ^i denotes all the parameters of i-LSTM. The cell output of DEPTH-LSTM at the last layer, $h_{t,L}^D$, is passed to the softmax layer for classification.

One last thing that needs to be tackled is $c_{t,0}^D$, of which the value is undetermined. The easiest solution would be setting the value to zero, which gives a flat initialization of cell states regardless of input value. Instead, we apply a linear transform such that

$$\mathbf{c}_{t,0}^D = \mathbf{V}\mathbf{h}_{t,0}^D \tag{11}$$

which achieves better performance empirically.

2.4. Prioritized Grid Long Short-Term Memory RNNs

In Equation 10, we notice that the cell output from TIME-LSTM is not being utilized for classification of the current

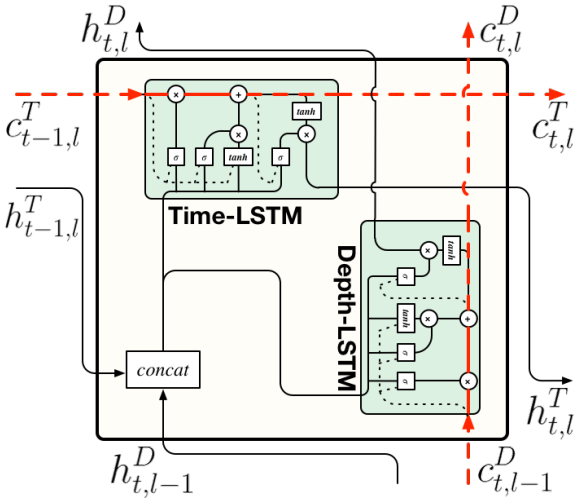


Fig. 3. GLSTM

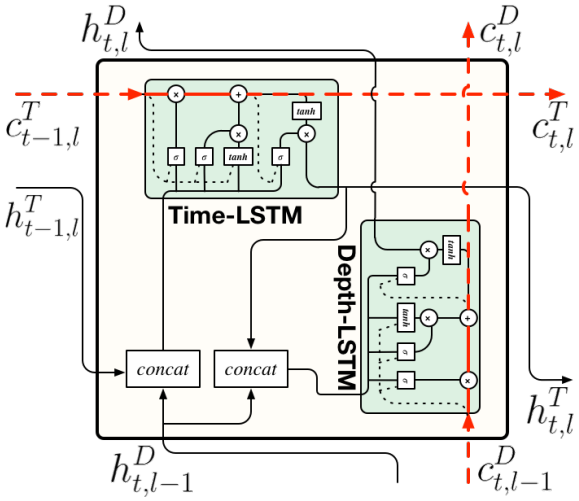


Fig. 4. Prioritized GLSTM

time step. In other words, we would like the depth dimension to know the output from other dimensions at the current grid such that it is implicitly deeper in terms of the number of transformations before being used for classification. We slightly modify the formulation from Section 2.3 as follows:

$$\mathbf{x}_{t,l}^T = [\mathbf{h}_{t,l-1}^D; \mathbf{h}_{t-1,l}^T] \tag{12}$$

$$(\mathbf{h}_{t,l}^T, \mathbf{c}_{t,l}^T) = \text{TIME-LSTM}(\mathbf{x}_{t,l}^T, \mathbf{c}_{t-1,l}^T, \Theta^T) \tag{13}$$

$$\mathbf{x}_{t,l}^D = [\mathbf{h}_{t,l-1}^D; \mathbf{h}_{t,l}^T] \tag{14}$$

$$(\mathbf{h}_{t,l}^D, \mathbf{c}_{t,l}^D) = \text{DEPTH-LSTM}(\mathbf{x}_{t,l}^D, \mathbf{c}_{t,l-1}^D, \Theta^D) \tag{15}$$

where the input to DEPTH-LSTM is updated after TIME-LSTM of the same grid is processed. We call this model prioritized grid LSTM (pGLSTM), and illustrate it in Figure 4, as opposed to non-prioritized grid LSTM (npGLSTM) in Figure 3.

3. RELATED WORK

Several alternative approaches have been proposed to address the vanishing gradient problem, and enable training of deeper neural networks. One model, named highway long short-term memory (HLSTM), was proposed in [11], introduced direct gated connections between cell states in adjacent layers, and is controlled by a "depth-gate". The depth gate function is determined by the cell outputs from the previous layer, as well as the cell states from the previous layer and previous time step.

Another model, called a residual network, that was proposed in [17], introduces "shortcut connections" that add up the input to one layer and an output from some upper layer as the new output of that upper layer, in the sense that the upper layers are obliged to learn the residual function of the ideal mapping. We therefore name the LSTM model with shortcut connections the residual LSTM (RLSTM) model.

The most significant difference between GLSTM models and these two alternative models is that a GLSTM model not only goes deeper, but also elegantly utilizes vertical information with an LSTM, providing all the functionality that an LSTM possesses along the depth dimension.

4. EXPERIMENT SETUP

4.1. Dataset

Our experiments to study the behavior of grid LSTMs were based on four different speech corpora. These corpora span a wide variety of configurations, ranging from 100 hours to 1200 hours, include three languages, multiple accents, and recording under different scenarios and sampling rates. We describe these datasets in the following sections.

4.1.1. AMI

The AMI corpus comprises approximately 100 hours of meeting recordings in instrumented meeting rooms [21]. Multiple microphones were used, including individual headset microphones, lapel microphones, and one or more microphone arrays. In this work, we use the single distant microphone (SDM) condition for our experiments. Our systems follow the split recommended in the corpus release: 80/9/9 hours for train/dev/test respectively. For our training, we use all segments provided by the corpus, including those with overlapping speech. Our models are evaluated on the test set only. NIST's asclite tool [22] is used for scoring.

4.1.2. HKUST

The HKUST Mandarin Telephone Speech (LDC2005S15) dataset is a medium-sized corpus containing 150 hours of conversational telephone speech from Mandarin speakers, recorded at an 8k sampling rate. The two callers do not know each other in advance, and similar topics to those in Fisher English (LDC2004S13) were used to initiate a conversation. The release is split into training and development sets with 873 calls and 24 calls respectively, of which we use the development for evaluation.

4.1.3. GALE Mandarin

We merged GALE Phase 2 Chinese Broadcast Conversation Speech (LDC2013S04), GALE Phase 3 Chinese Broadcast Conversation Speech Part 1 (LDC2014S09) and Part 2 (LDC2015S06), and GALE Phase 2 Chinese Broadcast News Speech (LDC2013S08), to create a large 500-hour Mandarin corpus. All four corpora are recorded at a 16k sampling rate from Chinese broadcast programs. We use the same 3-hour evaluation set as in [12].

4.1.4. Arabic MGB

This dataset is provided by the 2016 Arabic MGB Challenge[1], containing about 1200 hours of Arabic broadcast programs taken from Aljazeera TV over 10 years, recorded at a 16k sampling rate. Ten hours of data are partitioned as the official development set for the challenge, and will be used for evaluation in this work.

4.2. Model Setup

Here we consider three baseline recurrent neural network models: (1) LSTM (2) HLSTM, and (3) RLSTM. For all models, we carefully follow the configurations reported in [11]. Each layer contains 1024 memory cells, and a 512-node linear projection layer is added on top of each layer's output. Specifically, for RLSTM, we add shortcut connections from each layer's input to its output. The first two baseline models will be used for all experiments, while the last one will only be used when comparing deep LSTM variants.

For our non-prioritized and prioritized grid LSTM models (npGLSTM/pGLSTM), we chose the same configuration for both time-LSTM and depth-LSTM as for the aforementioned baseline models.

4.3. Training

We use Kaldi [23] for feature extraction, decoding, and training of initial HMM-GMM models. Maximum likelihood-criterion context-dependent speaker adapted acoustic models with Mel-Frequency Cepstral Coefficient (MFCC) features are trained with standard Kaldi recipes. Forced alignment is performed to generate labels for neural network acoustic model training. A feed-forward neural network model for each dataset is then trained and used to re-generate forced alignment, which is fixed for the rest of the experiments.

The Computational Network Toolkit (CNTK) [24] is used for the rest of neural network training. As [10] suggests, all weights are randomly initialized from the uniform distribution with range $[-0.05, 0.05]$, and all biases are initialized to 0 without generative or discriminative pretraining [1]. All neural network models, unless explicitly stated otherwise, are trained with a cross-entropy (CE) criterion, using truncated back-propagation-through-time (BPTT) [25] for optimization, which unrolls 20 frames and parallelizes 40 utterances in each mini-batch. No momentum is used for the first epoch, and a momentum of 0.9 is used for subsequent epochs

[1] http://www.mgb-challenge.org/arabic.html

[26]. $L2$ constraint regularization [27] with weight 10^{-5} is applied.

For *AMI*, *HKUST* and *GALE Mandarin*, ten percent of the training data is held out as a validation set, which is used to control the learning rate. When no gain is observed after an epoch, the learning rate is halved, and the model with the lowest validation loss is reloaded. For *Arabic MGB*, all data is used for training, and the learning rate is halved after each epoch; in addition, we use 4-GPU parallel training with model-averaging stochastic gradient descent (SGD) method [28]. Specifically, as [29] suggests, we start a seed model trained with standard SGD method for one epoch to achieve better performance.

Sequence discriminative training of Grid LSTM models using state-level minimum Bayes risk (sMBR) [30] criterion is conducted on the largest *Arabic MGB* dataset in order to examine the benefit of sequence discriminative training on our proposed model.

The input features for all models are 80 dimensional log Mel filterbank features computed every 10 ms, with an additional 3 dimensional pitch features. The output targets are context-dependent triphone states, of which the numbers are determined by the last HMM-GMM training stage. Table 1 shows the number of output targets in each dataset.

	AMI	HKUST	GALE	MGB
#states	3943	2825	4198	3711

Table 1. Number of output targets in each dataset.

5. RESULTS

The performance of various models are reported in character error rate (CER) for Chinese corpora and word error rate (WER) for the rest.

5.1. Prioritized/Non-Prioritized Grid LSTM

We first compare the two grid LSTM models along with baseline models on two medium-sized datasets: *HKUST* and *GALE Mandarin*. The CER of the different models are shown in Table 2. Both grid LSTM architectures outperform the vanilla LSTM model, as well as the highway LSTM model. Specifically, a 3% to 5% relative gain is achieved for the non-prioritized grid LSTM compared to the vanilla LSTM. The result suggests that grid LSTM models are empirically better solutions for introducing gated linear dependencies across the depth dimension.

Between the two grid architectures, the prioritized grid LSTM model consistently shows better ASR performance, providing an additional 1% relative gain compared to the non-prioritized model. This result supports our hypothesis that the LSTM whose output is fed into the final softmax layer should be prioritized in order to obtain more recent information.

5.2. Comparisons with Alternative Deep LSTMs

We compared alternative deep LSTM architectures with the prioritized Grid LSTM on the AMI corpus, when increasing model depth. Table 3 shows the detailed results. When increasing the number of layers from 3 to 8, the vanilla LSTM deteriorates significantly; on the other hand, the performance of the HLSTM only degrades slightly and levels off after 8 layers.

Model	#layers	HKUST	GALE
LSTM	3	33.29	23.96
HLSTM	3	32.86	23.33
npGLSTM	3	32.32	22.80
pGLSTM	3	**32.06**	**22.54**

Table 2. Performance of Grid LSTMs and baseline models.

In contrast, both the RLSTM and pGLSTM benefit from increasing the depth. The pGLSTM consistently performs better than all the other models, and it is worth noting that the 3-layer pGLSTM model achieves roughly the same accuracy as the 16-layer RLSTM model, while the latter takes much longer to train and much more space. We argue that utilizing the vertical (depth) information with an LSTM is essential for achieving good performance. We were not able to train a pGLSTM model with 16 layers. Careful parameter initialization may be required.

Model	#layers	#params	with overlap	no overlap
LSTM	3	12M	50.7	41.7
LSTM	8	36M	52.6	43.8
HLSTM	3	14M	50.4	41.2
HLSTM	8	40M	50.7	41.3
HLSTM	16	82M	50.7	41.2
RLSTM	3	12M	51.3	42.0
RLSTM	8	36M	50.5	40.8
RLSTM	16	74M	49.9	40.4
pGLSTM	3	25M	49.8	40.5
pGLSTM	8	72M	**49.0**	**39.6**

Table 3. Performance of different deep LSTM models.

5.3. Deeper Prioritized Grid LSTM

We conducted extensive experimentation to verify the effectiveness of the prioritized Grid LSTM on all four datasets. Table 4 summarizes the results of the baseline models as well as the proposed models, and includes references to other models tested on the same dataset that are reported in the literature.

The performance of the pGLSTM models are consistently superior to the baseline models, with gains being observed when increasing the number of layers from 3 to 5 on all datasets, even on the smallest AMI corpus. In addition, the 5 layer pGLSTM models set new benchmark results on the HKUST and GALE datasets. As for AMI, the best result is the state-of-the-art uni-directional recurrent model. As [11] shows that the bi-directional version gives about 2% absolute error reduction, we would hope that modifying the proposed pGLSTM model to be a bi-directional one would lead to better performance than BHLSTMP. This remains to be demonstrated in future work.

Model	#layers	#params	AMI	HKUST	GALE	MGB
BHLSTMP [11]	3	11M	**48.3**	-	-	-
Stacked maxout LSTMPs [31]	3	-	-	33.89	-	-
Highway CLDNN [12]	11	44M	-	-	22.41	-
LSTM	3	12M	50.7	33.29	23.96	23.56
HLSTM	3	14M	50.4	32.86	23.33	23.32
HLSTM	5	24M	50.7	32.40	22.63	23.12
pGLSTM	3	25M	49.8	32.06	22.54	22.36
pGLSTM	5	44M	48.6	**31.36**	**22.33**	**22.18**

Table 4. Model comparison on all four datasets.

5.4. Prioritized Grid LSTM with Sequence Training

Finally, we perform sequence training for the prioritized Grid LSTM on the 1200 hour Arabic MGB dataset. Detailed results are shown in Table 5. The results suggest that the prioritized Grid LSTM model can largely benefit from sequence training. Here an 8.8% relative improvement in WER is observed for sequence training of the 3-layer model, while a slightly larger 9.3% relative improvement is observed on a 5-layer model.

Model	#layers	MGB
pGLSTM	3	22.36
pGLSTM	5	22.18
pGLSTM (sMBR)	3	20.40
pGLSTM (sMBR)	5	**20.11**

Table 5. Performance of sequence training on pGLSTM

6. CONCLUSION

In this paper, we present two grid LSTM models for speech recognition, which utilize information along the depth dimension as an LSTM and simultaneously alleviate the vanishing gradient problem. Extensive experiments are conducted on four highly diverse datasets. Results suggest that (1) prioritizing the depth dimension is essential for achieving better performance, (2) our prioritized grid LSTM model outperforms two alternative designs for deep LSTM models, (3) sets new benchmarks for uni-directional models on all four datasets, and (4) greatly benefits from sequence discriminative training.

For future work, we plan to extend the prioritized grid LSTM model by combining it with complimentary neural network layers, such as convolutional layers and feed-forward layers. We also would like to investigate three dimensional grid LSTM models, which are time, frequency, and depth dimension respectively.

Acknowledgements

This research was supported in part by Ping-An and by the Qatar Computing Research Institute (QCRI).

7. REFERENCES

[1] Frank Seide, Gang Li, Xie Chen, and Dong Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*. IEEE, 2011, pp. 24–29.

[2] Frank Seide, Gang Li, and Dong Yu, "Conversational speech transcription using context-dependent deep neural networks.," in *Interspeech*, 2011, pp. 437–440.

[3] Michael L Seltzer, Dong Yu, and Yongqiang Wang, "An investigation of deep neural networks for noise robust speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 7398–7402.

[4] Bo Li and Khe Chai Sim, "Modeling long temporal contexts for robust DNN-based speech recognition," in *Interspeech*, 2014, pp. 353–357.

[5] Tara N Sainath, Abdel-rahman Mohamed, Brian Kingsbury, and Bhuvana Ramabhadran, "Deep convolutional neural networks for LVCSR," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 8614–8618.

[6] Ossama Abdel-Hamid, Li Deng, and Dong Yu, "Exploring convolutional neural network structures and optimization techniques for speech recognition.," in *Interspeech*, 2013, pp. 3366–3370.

[7] Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Interspeech*, 2015, pp. 2440–2444.

[8] Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 273–278.

[9] Hasim Sak, Andrew W Senior, and Françoise Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling.," in *Interspeech*, 2014, pp. 338–342.

[10] Tara N Sainath, Oriol Vinyals, Andrew Senior, and Hasim Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4580–4584.

[11] Yu Zhang, Guoguo Chen, Dong Yu, Kaisheng Yaco, Sanjeev Khudanpur, and James Glass, "Highway long short-term memory RNNs for distant speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5755–5759.

[12] Wei-Ning Hsu, Yu Zhang, Ann Lee, and James Glass, "Exploiting depth and highway connections inconvolutional recurrent deep neural networks for speech recognition.," in *Interspeech*, 2016, pp. 395–399.

[13] Jinyu Li, Abdelrahman Mohamed, Geoffrey Zweig, and Yifan Gong, "LSTM time and frequency recurrence for automatic speech recognition," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 187–191.

[14] Jinyu Li, Abdelrahman Mohamed, Geoffrey Zweig, and Yifan Gong, "Exploring multidimensional LSTMs for large vocabulary ASR," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4940–4944.

[15] Su-Lin Wu, Brian ED Kingsbury, Nelson Morgan, and Steven Greenberg, "Incorporating information from syllable-length time scales into automatic speech recognition," in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*. IEEE, 1998, vol. 2, pp. 721–724.

[16] Michiel Hermans and Benjamin Schrauwen, "Training and analysing deep recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2013, pp. 190–198.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[18] Nal Kalchbrenner, Ivo Danihelka, and Alex Graves, "Grid long short-term memory," *arXiv preprint arXiv:1507.01526*, 2015.

[19] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[20] Felix A Gers, Nicol N Schraudolph, and Jürgen Schmidhuber, "Learning precise timing with LSTM recurrent networks," *Journal of machine learning research*, vol. 3, no. Aug, pp. 115–143, 2002.

[21] Jean Carletta, "Unleashing the killer corpus: experiences in creating the multi-everything ami meeting corpus," *Language Resources and Evaluation*, vol. 41, no. 2, pp. 181–190, 2007.

[22] Jonathan G Fiscus, Jerome Ajot, Nicolas Radde, and Christophe Laprun, "Multiple dimension levenshtein edit distance calculations for evaluating asr systems during simultaneous speech," in *in Proc LREC*. Citeseer, 2006.

[23] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. Dec. 2011, IEEE Signal Processing Society, IEEE Catalog No.: CFP11SRW-USB.

[24] Dong Yu, Adam Eversole, Mike Seltzer, Kaisheng Yao, Zhiheng Huang, Brian Guenter, Oleksii Kuchaiev, Yu Zhang, Frank Seide, Huaming Wang, et al., "An introduction to computational networks and the computational network toolkit," Tech. Rep., Tech. Rep. MSR, Microsoft Research, 2014, http://codebox/cntk, 2014.

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

[26] Yu Zhang, Dong Yu, Michael L Seltzer, and Jasha Droppo, "Speech recognition with prediction-adaptation-correction recurrent neural networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5004–5008.

[27] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[28] Daniel Povey, Xiaohui Zhang, and Sanjeev Khudanpur, "Parallel training of DNNs with natural gradient and parameter averaging," *arXiv preprint arXiv:1410.7455*, 2014.

[29] Kai Chen and Qiang Huo, "Scalable training of deep learning machines by incremental block training with intra-block parallel optimization and blockwise model-update filtering," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5880–5884.

[30] Karel Veselỳ, Arnab Ghoshal, Lukás Burget, and Daniel Povey, "Sequence-discriminative training of deep neural networks.," in *Interspeech*, 2013, pp. 2345–2349.

[31] Xiangang Li and Xihong Wu, "Improving long short-term memory networks using maxout units for large vocabulary speech recognition," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4600–4604.