IMPROVING LAYER TRAJECTORY LSTM WITH FUTURE CONTEXT FRAMES

Jinyu Li, Liang Lu, Changliang Liu, Yifan Gong

Microsoft Speech and Language Group

ABSTRACT

In our recent work, we proposed a layer trajectory long short-term memory (ltLSTM) model which decouples the tasks of temporal modeling and senone classification with time-LSTMs and depth-LSTMs. The ltLSTM model achieved significant accuracy improvement over the traditional multi-layer LSTM models from our previous study. Considering the future context frames carrying valuable information for predicting the target label evidenced by the success of bi-directional LSTMs, in this work we investigate how to incorporate this kind of information with hidden vectors from either time-LSTM or depth-LSTM. Trained with 30 thousand hours of EN-US Microsoft internal data, the best ltLSTM model with future context frames can improve the baseline ltLSTM with up to 11.5% relative word error rate (WER) reduction and improve the baseline LSTM with up to 24.6% relative WER reduction across different tasks.

Index Terms— LSTM, layer trajectory, future context frames, temporal modeling, senone classification

1. INTRODUCTION

There has been a significant progress in automatic speech recognition (ASR) since the transition from the deep feedforward neural networks (DNNs) [1, 2, 3, 4, 5] to recurrent neural networks (RNNs) with long short-term memory (LSTM) units [6]. LSTMs alleviate the gradient vanishing or exploding issues in standard RNNs by using input, output and forget gates, thus improving the capacity of the network to capture long temporal context information in audio sequences. LSTM-RNNs [7, 8, 9, 10, 11, 12] have been shown to outperform DNNs on a variety of ASR tasks [13], and since then, considerable amount of efforts have been devoted to improving the structure of LSTM for ASR, such as convolutional LSTM DNN (CLDNN) [14], time-frequency LSTM-RNNs [15, 16, 17, 18], grid LSTMs [19, 20], residual LSTMs [21, 22], and highway LSTMs [23, 24].

Recently, we proposed a layer trajectory LSTM (ltLSTM) model [25] which is equipped with a depth-LSTM that scans the hidden states of time-LSTMs for senone (tied triphone states) classification. This architecture decouples the tasks of time recurrence modeling and senone classification for standard LSTMs. Furthermore, the depth-LSTM creates auxiliary connections for gradient flow, thereby making it easier to train deeper LSTMs. This model achieved significant accuracy improvement from traditional LSTMs or residual LSTMs in our previous experiments. In [26], we extended our previous work by proposing the generalized ltLSTM architecture with the concept of depth processing block. We integrated gated feedforward units as well as max-pooling [27, 28, 29] feedforward units designed for the depth processing block with lower computational cost.

In this study, we aim at further improving the accuracy of ltLSTM by using future context frames, inspired by the success of recent works of incorporating future context frames, such as latency-control bi-directional LSTM (LC-BLSTM) [23], time-delay neural network (TDNN) [30], and feedforward sequential memory network (FSMN) [31]. LC-BLSTM reduces the latency of BLSTM by chunk-wise forward and backward LSTM computation. TDNN and FSMN are similar in terms of model structure, which can be viewed as 1-D convolutional neural networks taking a window of acoustic frames as input. These models can benefit from future acoustic frames to improve recognition accuracy, though at the cost of higher latency compared with standard uni-directional recurrent models. In this work, we focus on improving the uni-directional recurrent models by taking advantage of future context frames, which is only partially addressed in [32].

The unique structure of ltLSTM provides more flexibility to incorporate the future context information compared to other model structures as mentioned above. In particular, the future information can be utilized by either time-LSTM or depth-LSTM, and both options will be investigated in this work. Our experiments were performed using around 30 thousand hours of anonymized EN-US data, which is a collection of Microsoft Cortana and Conversation data with mixed close-talk and far-field audios. Our results show that the ltLSTM with future context frames can significantly outperform the baseline ltLSTM without future information with up to 11.5% relative word error rate (WER) reduction and improve the baseline LSTM with up to 24.6% relative WER reduction.

The rest of the paper is organized as follows. In Section 2, we briefly overview the standard multi-layer LSTM and our previously proposed ltLSTM. In Section 3, we describe how we incorporate context future frames into ltLSTMs with different options. We evaluate the proposed models in Section 4, and conclude our study in Section 5.

2. LAYER TRAJECTORY LSTM

2.1. LSTM

We refer to the standard LSTM as time-LSTM since it does temporal modeling via time recurrence by taking the output of previous time step as the input to the current time step. At the time step t, the computation of the l-th layer LSTM units can be described as:

$$\mathbf{i}_t^l = \sigma(\mathbf{W}_{ix}^l \mathbf{x}_t^l + \mathbf{W}_{ih}^l \mathbf{h}_{t-1}^l + \mathbf{p}_i^l \odot \mathbf{c}_{t-1}^l + \mathbf{b}_i^l) \tag{1}$$

$$\mathbf{f}_t^l = \sigma(\mathbf{W}_{fx}^l \mathbf{x}_t^l + \mathbf{W}_{fh}^l \mathbf{h}_{t-1}^l + \mathbf{p}_f^l \odot \mathbf{c}_{t-1}^l + \mathbf{b}_f^l) \tag{2}$$

$$\mathbf{c}_t^l = \mathbf{f}_t^l \odot \mathbf{c}_{t-1}^l + \mathbf{i}_t^l \odot \phi(\mathbf{W}_{cx}^l \mathbf{x}_t^l + \mathbf{W}_{ch}^l \mathbf{h}_{t-1}^l + \mathbf{b}_c^l) \tag{3}$$

$$\mathbf{o}_t^l = \sigma(\mathbf{W}_{ox}^l \mathbf{x}_t^l + \mathbf{W}_{oh}^l \mathbf{h}_{t-1}^l + \mathbf{p}_o^l \odot \mathbf{c}_t^l + \mathbf{b}_o^l) \tag{4}$$

$$\mathbf{h}_t^l = \mathbf{o}_t^l \odot \phi(\mathbf{c}_t^l) \tag{5}$$

where $\mathbf{x}_t^l$ is the input vector for the l-th layer with

$$\mathbf{x}_t^l = \begin{cases} \mathbf{h}_t^{l-1}, & \text{if } l > 1 \\ \mathbf{s}_t, & \text{if } l = 1 \end{cases} \tag{6}$$

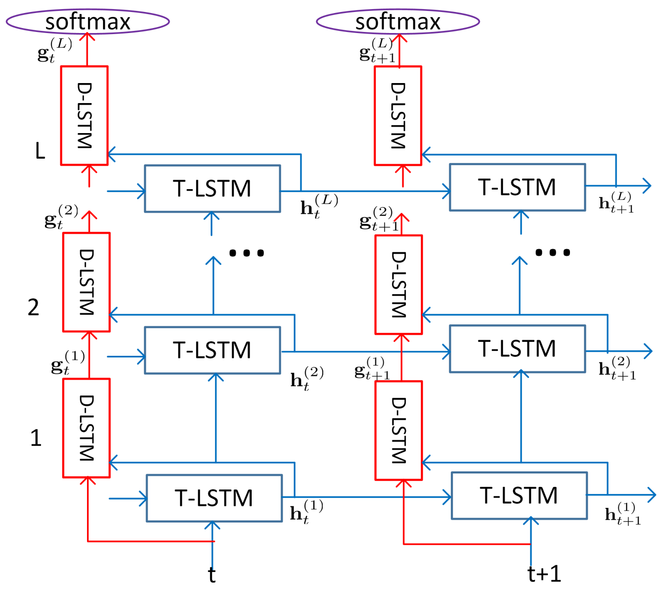

Fig. 1. Diagram of layer trajectory LSTM (ltLSTM). Depth-LSTM (D-LSTM) is used to scan the outputs of time-LSTM (T-LSTM) across all layers at the current time step to get summarized layer trajectory information for senone classification. Note that There is no time recurrence in D-LSTM, which only occurs in T-LSTMs.

$\mathbf{s}_t$ is the speech spectrum input at the time step t; $l = 1...L$, where L is the total number of hidden layers. The vectors $\mathbf{i}_t^l$, $\mathbf{o}_t^l$, $\mathbf{f}_t^l$, $\mathbf{c}_t^l$ are the activations of the input, output, forget gates, and memory cells, respectively. $\mathbf{h}_t^l$ is the output of the time-LSTM. $\mathbf{W}_{\cdot x}^l$ and $\mathbf{W}_{\cdot h}^l$ are the weight matrices for the inputs $\mathbf{x}_t^l$ and the recurrent inputs $\mathbf{h}_{t-1}^l$, respectively. $\mathbf{b}_\cdot^l$ are bias vectors. $\mathbf{p}_i^l$, $\mathbf{p}_o^l$, $\mathbf{p}_f^l$ are parameter vectors associated with peephole connections. The functions σ and ϕ are the logistic sigmoid and hyperbolic tangent nonlinearity, respectively. The operation $\odot$ represents element-wise multiplication of vectors.

2.2. Layer trajectory LSTM

The layer trajectory LSTM (ltLSTM) model [25] performs temporal modeling using time-LSTM and senone classification using depth-LSTM separately, which was proven to perform better for acoustic modeling than using the time-LSTM alone. The model structure is illustrated by Figure 1. The formulation of time-LSTM is the same as in Section 2.1. In this model, the l-th layer of the depth-LSTM can be expressed as:

$$\mathbf{j}_t^l = \sigma(\mathbf{U}_{jh}^l \mathbf{h}_t^l + \mathbf{U}_{jg}^l \mathbf{g}_t^{l-1} + \mathbf{q}_j^l \odot \mathbf{m}_t^{l-1} + \mathbf{d}_j^l) \quad (7)$$

$$\mathbf{e}_t^l = \sigma(\mathbf{U}_{eh}^l \mathbf{h}_t^l + \mathbf{U}_{eg}^l \mathbf{g}_t^{l-1} + \mathbf{q}_e^l \odot \mathbf{m}_t^{l-1} + \mathbf{d}_e^l) \quad (8)$$

$$\mathbf{m}_t^l = \mathbf{e}_t^l \odot \mathbf{m}_t^{l-1} + \mathbf{j}_t^l \odot \phi(\mathbf{U}_{sh}^l \mathbf{h}_t^l + \mathbf{U}_{sg}^l \mathbf{g}_t^{l-1} + \mathbf{d}_s^l) \quad (9)$$

$$\mathbf{v}_t^l = \sigma(\mathbf{U}_{vh}^l \mathbf{h}_t^l + \mathbf{U}_{vg}^l \mathbf{g}_t^{l-1} + \mathbf{q}_v^l \odot \mathbf{m}_t^l + \mathbf{d}_v^l) \quad (10)$$

$$\mathbf{g}_t^l = \mathbf{v}_t^l \odot \phi(\mathbf{m}_t^l) \quad (11)$$

$\mathbf{g}_t^l$ is the output of the depth-LSTM at time t and layer l. Comparing Eqs. (1) – (5) with Eqs. (7) – (11), we can see the biggest difference is that the depth-LSTM takes its previous layer output $\mathbf{g}_t^{l-1}$ and the time-LSTM's current layer output $\mathbf{h}_t^l$ as inputs, while the time-LSTM takes its previous layer output $\mathbf{h}_t^{l-1}$ and previous time step output $\mathbf{h}_{t-1}^l$ as inputs.

Note that there is no recurrence in depth-LSTM at different time steps. The time recurrence only occurs in time-LSTM across time. The structural difference between time-LSTM and depth-LSTM helps them to deal with different aspects of the learning problem, in particular, sequential modeling versus senone classification. Although the total computational cost of the ltLSTM is almost doubled compared to that using the time-LSTM alone, it can be significantly reduced by using the gated deep neural network units instead of LSTM units for the depth processing, which is shown to be effective in [26].

3. LAYER TRAJECTORY LSTM WITH FUTURE CONTEXT FRAMES

In this section, we study two approaches to incorporating the future context frames into ltLSTMs for higher recognition accuracy. The key idea is to get a fixed size vector representation of variable future frames, referred to as *lookahead embedding*, as an additional feature to the network. In this paper, we study the use a linear transform and attention weights for this purpose.

3.1. Lookahead embedding through linear transform

Recall that $\mathbf{h}_t^l$ and $\mathbf{g}_t^{l-1}$ are the inputs for computing $\mathbf{g}_t^l$ in Eqs. (7) – (11), here we seek to replace either $\mathbf{h}_t^l$ or $\mathbf{g}_t^{l-1}$ by the lookahead embedding vector in order to incorporate the future context information when computing $\mathbf{g}_t^l$.

A simple approach to obtaining the embedding vector, denoted as $\boldsymbol{\eta}_t^l$, is to use linear transforms. Suppose we have τ hidden vectors $\mathbf{h}_{t+1:t+\tau}^l$ from the time-LSTM corresponding to the future frames, we can compute the embedding vector as:

$$\boldsymbol{\eta}_t^l = \sum_{\delta=0}^{\tau} \mathbf{H}_\delta^l \mathbf{h}_{t+\delta}^l, \quad (12)$$

where $\mathbf{H}_\delta^l$ is a weight matrix corresponding to the time step $t + \delta$. If we share the weight across all time steps, it is equivalent to a 1-D convolutional operation. As it does not save computation, nor improves accuracy, we did not focus on the sharing approach in this study. Given the embedding vector, we then replace $\mathbf{h}_t^l$ by $\boldsymbol{\eta}_t^l$ in Eqs. (7) – (11) to calculate $\mathbf{g}_t^l$. In this study, we apply Eq. (12) to all the hidden layers, and refer to this model as ltLSTM-Tτ, where τ indicates how many future frames we look ahead in time-LSTM.

Similarly, we can compute the embedding vector from the depth-LSTM, such as

$$\boldsymbol{\zeta}_t^{l-1} = \sum_{\delta=0}^{\tau} \mathbf{G}_\delta^{l-1} \mathbf{g}_{t+\delta}^{l-1}, \quad (13)$$

where $\mathbf{G}_\delta^{l-1}$ denotes the weight matrix, and for clarity, we denote this embedding vector as $\boldsymbol{\zeta}_t^{l-1}$. In this case, we replace $\mathbf{g}_t^{l-1}$ by $\boldsymbol{\zeta}_t^{l-1}$ in Eqs. (7) – (11) to compute $\mathbf{g}_t^l$. Again, we apply Eq. (13) to all hidden layers in our experiments, and refer to this model as ltLSTM-Dτ. When we apply both Eq. (12) and Eq. (13) at the same time, we will refer to the model as ltLSTM-TτDτ.

In Figure 2 and Figure 3, we show the computational steps to update the $l + 1$th layer depth-LSTM output $\mathbf{g}_t^{l+1}$ when incorporating τ future frames from time-LSTM and depth-LSTM, respectively. When we incorporate the future frames from time-LSTM only as Figure 2 shows, the evaluation of $\mathbf{g}_t^{l+1}$ depends on $\mathbf{g}_t^l$ and $\boldsymbol{\eta}_t^{l+1}$ which is generated from $[\mathbf{h}_t^{l+1}......\mathbf{h}_{t+\tau}^{l+1}]$ as in Eq. (12). When

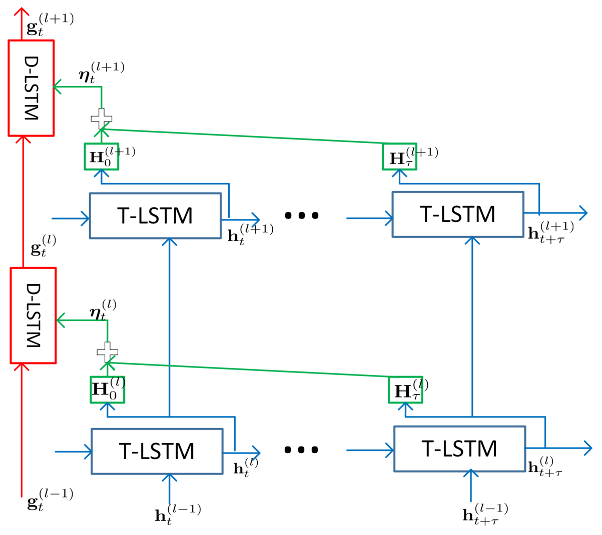

Fig. 2. The process used to evaluate the $l+1$th layer depth-LSTM output when incorporating τ future frames of hidden vectors from time-LSTM at every layer.

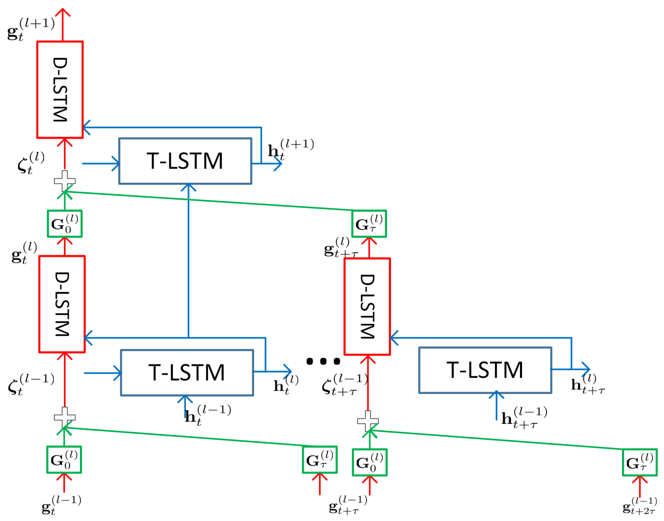

Fig. 3. The process used to evaluate the $l+1$th layer depth-LSTM output when incorporating τ future frames of hidden vectors from depth-LSTM at every layer.

we stack multiple layers, there is no latency accumulation, so the total number of lookahead frames in this case is still τ. However, when incorporating the future frames from the depth-LSTM as Figure 3, there is latency accumulation when we stack multiple layers. For a L layer ltLSTM with τ future context frames, the total number of additional lookahead frames will be $L\tau$.

3.2. Lookahead embedding through attention weights

Another way to incorporate future frames is to use the attention mechanism to generate an embedding vector of a context window with input-dependent weights. This method was used successfully to improve the accuracy of connectionist temporal classification (CTC) modeling in [33]. The only difference from [33] is that we apply the attention modeling to all the hidden layers instead of the top layer only.

In the following, we briefly summarize the process of using attention to calculate $\boldsymbol{\eta}_t^l$ which is then used to replace $\mathbf{h}_t^l$ in Eqs. (7) – (11) to calculate $\mathbf{g}_t^l$. We can use a similar process to calculate $\boldsymbol{\zeta}_t^{l-1}$ which is later used to replace $\mathbf{g}_t^{l-1}$.

We first transform $\mathbf{h}_\delta^l$ in a context window for each $\delta \in [t-\tau,\ t+\tau]$ as

$$\mathbf{r}_\delta = \mathbf{W}'_{t-\delta}\mathbf{h}_\delta^l \tag{14}$$

Then define the energy signal for location-based attention as

$$\mathbf{e}_{t,\delta} = \tanh(\mathbf{W}\mathbf{r}_\delta + \mathbf{V}\boldsymbol{\beta}_{t-1} + \mathbf{b}) \tag{15}$$

where $\boldsymbol{\beta}_{t-1} = \mathbf{F} * \boldsymbol{\alpha}_{t-1}$, with the elements of $\boldsymbol{\alpha}$ defined as

$$\alpha_{t,\delta,j} = \frac{\exp(e_{t,\delta,j})}{\sum_{\delta'=t-\tau}^{t+\tau}\exp(e_{t,\delta',j})}, \quad j = 1, \cdots, n. \tag{16}$$

Here, $\alpha_{t,\delta,j}$ can be interpreted as the amount of contribution from $r_\delta(j)$ in computing $\eta_t^l(j)$. Now, the context vector $\boldsymbol{\eta}_t^l$ can be computed using

$$\boldsymbol{\eta}_t^l = \gamma \sum_{\delta=t-\tau}^{t+\tau} \boldsymbol{\alpha}_{t,\delta} \odot \mathbf{r}_\delta. \tag{17}$$

Hence, our proposed method is a dimension-wise location-based attention.

4. EXPERIMENTS

In this section, we evaluate the effectiveness of ltLSTM modeling using future context frames. Our baseline models are the vanilla LSTM, residual LSTM (ResLSTM), and ltLSTM without future context frames. The ResLSTM model uses a direct shortcut path across layers to alleviate the gradient vanishing issue in the multiple layer LSTM. In our experiments, all models are uni-directional, and were trained with 30 thousand hours of anonymized and transcribed Microsoft production data, including Cortana and Conversation data, recorded in both close-talk and far-field conditions. All LSTM models use 1024 hidden units and the output of each LSTM layer is reduced to 512 using a linear projection layer. The softmax layer has 9404 nodes to model the senone labels. The target senone label is delayed by 5 frames as in [8]. The backpropagation through time (BPTT) [34] with truncation size 16 is used to train all models. The input feature is 80-dimension log Mel filter bank for every 10 milliseconds (ms) speech. We applied frame skipping by a factor of 2 [12] to reduce the runtime cost, which corresponds to 20ms per frame. The language model is a 5-gram with around 100 million (M) ngrams.

We evaluate all cross entropy (CE) trained models with Microsoft Cortana and Conversation test sets. Both sets contain mixed close-talk and far-field recordings, with 439k and 111k words, respectively. The Cortana test set has shorter utterances related to voice search and commands, while the Conversation test set has longer utterances from conversations. We also evaluate the models on the third test set named as DMA with 29k words, which is not in Cortana or Conversation domain. The DMA domain was unseen during the model training, and is served to evaluate the generalization capacity of the model.

Table 1 shows WERs of all the models. Among LSTMs with different number of layers, the 6-layer model performs the best. When increasing the number of layers to 10, we observed considerable ac-

Table 1. WERs of all models on Cortana, Conversation, and DMA test sets. All test sets are mixed with close-talk and far-field recordings.

	Cortana	Conversation	DMA
4-layer LSTM	10.37	19.41	20.66
6-layer LSTM	9.85	19.20	20.19
10-layer LSTM	10.58	19.92	21.62
6-layer ResLSTM	9.99	18.85	19.49
10-layer ResLSTM	9.68	18.15	18.62
12-layer ResLSTM	9.59	18.19	18.78
6-layer ltLSTM	9.28	17.47	17.61
6-layer ltLSTM-T4	9.15	17.17	16.68
6-layer ltLSTM-D4	8.78	16.51	15.58
6-layer ltLSTM-T4D4	8.75	16.31	15.53
6-layer ltLSTM-T4 with attention	8.99	17.14	17.02

curacy degradation, which is consistent with the observations the in literature [23, 20]. The 6-layer ResLSTM is close to the 6-layer LSTM in terms of WERs, there are consistent improvements by increasing to 10 layers for ResLSTM model, but no further improvement when increasing to 12 layers.

The 6-layer ltLSTM clearly outperforms all the LSTM and ResLSTM models, with 9.28%, 17.47% ad 17.61% WERs on Cortana, Conversation, and DMA test sets, respectively. We use this model as the baseline model to evaluate the effectiveness of ltLSTM modeling with future context frames.

First, the 6-layer ltLSTM-T4 model which incorporates future 4 frames of time-LSTM output at each layer achieves 9.15%, 17.17%, and 16.68% WERs on those 3 test sets. However, adding the attention module does not bring as much improvements as we observed in the end-to-end CTC modeling [33, 35]. The reason maybe that the attention scheme can relax the hard alignment modeling to alleviate the frame independence assumption of CTC, which may not be a key problem for the hybrid models in this study as the decoding space is constrained with a lexicon and language model.

When incorporating 4 future frames at each layer in depth-LSTM, the 6-layer ltLSTM-D4 model can achieve larger improvements, which are 8.78%, 16.51%, 15.58% WERs on Cortana, Conversation, and DMA test sets, respectively. This standards for relative 5.4%, 5.5%, and 11.5% WER reductions from the baseline 6-layer ltLSTM and relative 10.9%, 14.9%, and 24.6% WER reductions from the baseline 6-layer LSTM on those three test sets, respectively. It is interesting to see that all these models with future context frames achieved the largest improvement on the unseen DMA test set. This may be because the future context frames provide much more information for unseen scenario to predict senones labels as the baseline models have already been trained very well to handle the matched test data. Adding future frames to both time-LSTM and depth-LSTM, however, we only observe slight improvement over the 6-layer ltLSTM-D4.

In Table 2, we compare the runtime costs of the baseline 6-layer ltLSTM model and all 6-layer ltLSTM models with future context frames as well as latency requirements in terms of the number of lookahead frames. The ltLSTM baseline model does not use any future context frame, while the ltLSTM-T4 model only requires overall 4 lookahead frames because it does not have latency accumulation across the depth. Given its accuracy improvement with small increase in latency, this model is appealing for scenarios with strict latency requirement. In contrast, ltLSTM-D4 has totally 24 lookahead frames due to the context expansion in depth-LSTM. The additional future context information helps to push down the WER further, and it is therefore desirable for applications with less strict latency requirement. Both ltLSTM-T4 and ltLSTM-D4 have 6M more parameters than the baseline ltLSTM model. Note that we can improve the recognition accuracy of the ltLSTM-T4 model by having an ltLSTM-T24 model with 24 frames lookahead in each layer, however this will significantly increase the model size to around 100 M parameters. Therefore, we didn't pursue such a model. The ltLSTM-T4 attention model also only requires 4 lookahead frames, but it is computationally more expensive than the ltLSTM-T4 model.

Table 2. Runtime cost of ltLSTM and its variants in terms of additional lookahead frames relevant to the standard LSTM and the total number of parameters. All models have 6 hidden layers.

	lookahead frames	parameters (M)
ltLSTM	0	57
ltLSTM-T4	4	63
ltLSTM-D4	24	63
ltLSTM-T4 Attention	4	69
ltLSTM-T4D4	24	71

5. CONCLUSIONS

In this paper, we investigated incorporating future context frames into ltLSTMs for higher accuracy acoustic models. We have shown that the lookahead embeddings from either time-LSTM or depth-LSTM improves recognition accuracy. We presented two approaches to obtaining this embedding vector, i.e., by linear transforms and attention mechanism. From our experiments with around 30k hours of Microsoft internal speech training data, the 6-layer ltLSTM with 4 future frames from the time-LSTM at each layer improved the baseline by relative 1.4%, 1.7%, and 5.3% WER reductions on Cortana, Conversation, and DMA sets, respectively. The overall additional latency is only 4 frames, which is desirable for applications with strict latency requirement. On the other hand, the 6-layer ltLSTM with 4 future frames from the depth-LSTM at each layer improved the baseline 6-layer ltLSTM by relative 5.4%, 5.5%, and 11.5% WER reductions compared to the baseline, respectively. However, this model requires 24 lookahead frames in total, and is suitable for application without strict latency requirement. We also evaluate the effectiveness of attention modeling, and found its gain for ltLSTM is mostly due to the introduction of future context frames.

Note the extreme case of using lookahead frames is the bi-directional modeling [36] which has an additional backward LSTM running from the end of utterances to the beginning. In the future study, we will compare the proposed contextual ltLSTM with bi-directional LSTM, and investigate whether we can further improve bi-directional LSTM with the layer trajectory modeling.

6. REFERENCES

[1] Dong Yu, Li Deng, and George E. Dahl, "Roles of pre-training and fine-tuning in context-dependent DBN-HMMs for real-world speech recognition," in *NIPS 2010 Workshop on Deep Learning and Unsupervised Feature Learning*, 2010.

[2] T. Sainath, B. Kingsbury, B. Ramabhadran, et al., "Making deep belief networks effective for large vocabulary continuous speech recognition," in *Proc. ASRU*, 2011, pp. 30–35.

[3] N. Jaitly, P. Nguyen, A. Senior, and V. Vanhoucke, "Application of pretrained deep neural networks to large vocabulary speech recognition," in *Proc. Interspeech*, 2012.

[4] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdelrahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[5] Li Deng, Jinyu Li, Jui-Ting Huang, Kaisheng Yao, Dong Yu, Frank Seide, Michael Seltzer, Geoff Zweig, Xiaodong He, Jason Williams, et al., "Recent advances in deep learning for speech research at microsoft," in *Proc. ICASSP*, 2013, pp. 8604–8608.

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

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

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

[9] Haşim Sak, Oriol Vinyals, Georg Heigold, Andrew Senior, Erik McDermott, Rajat Monga, and Mark Mao, "Sequence discriminative distributed training of long short-term memory recurrent neural networks," in *Proc. Interspeech*, 2014.

[10] Xiangang Li and Xihong Wu, "Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition," in *Proc. ICASSP*, 2015, pp. 4520–4524.

[11] Y. Miao and F. Metze, "On speaker adaptation of long short-term memory recurrent neural networks.," in *Proc. Interspeech*, 2015, pp. 1101–1105.

[12] Y. Miao, J. Li, Y. Wang, S. Zhang, and Y. Gong, "Simplifying long short-term memory acoustic models for fast training and decoding," in *Proc. ICASSP*, 2016.

[13] D. Yu and J. Li, "Recent progresses in deep learning based acoustic models," *IEEE/CAA J. of Autom. Sinica.*, vol. 4, no. 3, pp. 399–412, July 2017.

[14] Tara N Sainath, Oriol Vinyals, Andrew Senior, and Haşim Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Proc. ICASSP*, 2015, pp. 4580–4584.

[15] J. Li, A. Mohamed, G. Zweig, and Y. Gong, "LSTM time and frequency recurrence for automatic speech recognition," in *ASRU*, 2015.

[16] J. Li, A. Mohamed, G. Zweig, and Y. Gong, "Exploring multidimensional LSTMs for large vocabulary ASR," in *Proc. ICASSP*, 2016.

[17] Tara N Sainath and Bo Li, "Modeling time-frequency patterns with LSTM vs. convolutional architectures for LVCSR tasks," in *Proc. Interspeech*, 2016.

[18] Bo Li and Tara N Sainath, "Reducing the computational complexity of two-dimensional LSTMs," in *Proc. Interspeech*, 2017.

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

[20] Wei-Ning Hsu, Yu Zhang, and James Glass, "A prioritized grid long short-term memory RNN for speech recognition," in *Spoken Language Technology Workshop (SLT), 2016 IEEE*. IEEE, 2016, pp. 467–473.

[21] Yuanyuan Zhao, Shuang Xu, and Bo Xu, "Multidimensional residual learning based on recurrent neural networks for acoustic modeling," in *Proc. Interspeech*, 2016, pp. 3419–3423.

[22] Jaeyoung Kim, Mostafa El-Khamy, and Jungwon Lee, "Residual LSTM: Design of a deep recurrent architecture for distant speech recognition," *arXiv preprint arXiv:1701.03360*, 2017.

[23] Yu Zhang, Guoguo Chen, Dong Yu, Kaisheng Yao, Sanjeev Khudanpur, and James Glass, "Highway long short-term memory rnns for distant speech recognition," *ICASSP*, 2016.

[24] Golan Pundak and Tara N Sainath, "Highway-LSTM and recurrent highway networks for speech recognition," in *Proc. of Interspeech*, 2017.

[25] Jinyu Li, Changliang Liu, and Yifan Gong, "Layer trajectory LSTM," in *Proc. Interspeech*, 2018.

[26] Jinyu Li, Liang Lu, Changliang Liu, and Yifan Gong, "Exploring layer trajectory LSTM with depth processing units and attention," in *Proc. IEEE SLT*, 2018.

[27] Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio, "Maxout networks," *arXiv preprint arXiv:1302.4389*, 2013.

[28] Yajie Miao, Florian Metze, and Shourabh Rawat, "Deep maxout networks for low-resource speech recognition," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 398–403.

[29] Pawel Swietojanski, Jinyu Li, and Jui-Ting Huang, "Investigation of maxout networks for speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 7649–7653.

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

[31] Shiliang Zhang, Cong Liu, Hui Jiang, Si Wei, Lirong Dai, and Yu Hu, "Nonrecurrent neural structure for long-term dependence," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 25, no. 4, pp. 871–884, 2017.

[32] Jie Li, Xiaorui Wang, Yuanyuan Zhao, and Yan Li, "Gated recurrent unit based acoustic modeling with future context," in *Proc. Interspeech*, 2018.

[33] A. Das, J. Li, R. Zhao, and Y. Gong, "Advancing connectionist temporal classification with attention modeling," in *Proc. ICASSP*, 2018.

[34] Herbert Jaeger, *Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the" echo state network" approach*, vol. 5, GMD-Forschungszentrum Informationstechnik Bonn, 2002.

[35] J. Li, G. Ye, A. Das, R. Zhao, and Y. Gong, "Advancing acoustic-to-word CTC model," in *Proc. ICASSP*, 2018.

[36] Mike Schuster and Kuldip K Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.