Supplementary material: Temporal Pyramid Recurrent Neural Network

Qianli Ma,[1] Zhenxi Lin,[1] Enhuan Chen,[1] Garrison W. Cottrell[2]
[1]School of Computer Science and Engineering, South China University of Technology, Guangzhou, China
[2]Department of Computer Science and Engineering, University of California, San Diego, CA, USA
qianlima@scut.edu.cn, zhenxi_lin@foxmail.com, ceh930603@gmail.com, gary@ucsd.edu

Experiment setup

In this section, we supplement detailed experiment setups for the tasks.

Hyper-parameters setting

The length of input subsequences L and the aggregation granularity g are two specific and important hyper-parameters of TP-RNN, which can control the length of the longest gradient feedback path from inputs to the output. Specifically, it equals to the number of levels of the sub-pyramids ($J = \log_g L$) plus the length of the shortcut path ($N = T/L$), which is always much shorter than the input sequence length as in conventional RNNs (i.e., $J + N << T$). As a result, the larger the L and g, the shorter the length of the longest gradient feedback path, and the faster the convergence rate and the higher the accuracy. More details about the effect of these two hyper-parameters can be found in later Section.

Generally, we set L according to the length of input sequence T, so that the number of input subsequences N is controlled to no more than 20. g controls the number of levels of the sub-pyramids (J) based on L. For simplicity, we control J to 2, so g is set to the square root of L. L and g of the higher layer are generally smaller due to the shorter input sequence (the aggregated state sequence) produced by the sub-pyramids in the previous layer. L and g of TP-RNN-M for each task are shown in Table 1. And L and g of TP-RNN-S are the same as those of the first layer of TP-RNN-M.

Table 1: The subsequence lengths and aggregation granularities for the 3 layers of TP-RNN-M.

Task		Subsequence length L	Aggregation granularity g
Masked addition problem		100, 25, 4	10, 5, 2
Pixel-by-pixel image classification		49, 16, 4	7, 4, 2
Signal recognition		100, 25, 4	10, 5, 2
Speaker identification	MFCC	25, 4, 2	5, 2, 2
	Raw	100, 25, 9	10, 5, 3

The setup of HM-LSTM

Chung *et al.* (2017) proposed a hierarchical multi-scale RNN (HM-LSTM) specially designed for language modeling, which can model latent multi-scale structures in language sequences by binary boundary detector. The boundary detector can implicitly learn word boundary information. In our experiments, we use 3 layers HM-LSTM as in (Chung, Ahn, and Bengio 2017) and set the hidden size to 100 as the same as TP-RNN-M for a comparable number of parameters. We equipped an output module to aggregate the output of each layer at each time step through the attention mechanism as in (Chung, Ahn, and Bengio 2017), where the hidden size of output embedding is 100. The binary boundary detector is learned by straight-through estimator with slope annealing trick. The slope is set to 1 and slowly increase the slope when training until it reaches a threshold with an appropriate scheduling. We speculated that only limited information is retained in the last aggregate state due to boundary gate flushing, so we average pooling all aggregation states to classify for fair comparison. Our experimental results show that this hierarchical structure specially designed for a particular task lacks generalization for other tasks. For examples, HM-LSTM failed to converge when learning directly from raw audio waves in speaker identification.

Masked addition problem

We follow the same setup as (Le, Jaitly, and Hinton 2015; Arjovsky, Shah, and Bengio 2016; Xia et al. 2018) to randomly generate 100,000 training samples and 10,000 test samples and use three different lengths, T = 200, 500, 1000. Particularly, when T = 200, Dilated LSTM uses 62 hidden state units and 8 layers with the lengths of skip connections from 1 to 128, due to the longest skip length must be less than the sequence length 200. For other situations, it still uses 59 hidden state units and 9 layers.

Speaker identification

On this task, Dilated LSTM also uses 9 layers with the lengths of skip connections from 1 to 256. In addition, we also try to expand the skip lengths of Dilated LSTM in order to further shorten the lengths of gradient feedback paths. We use two additional configurations when using raw audio waves with

1000Hz sampling rate. First, we use 8 layers with the skip lengths from 8 to 1024, which is consistent with (Chang et al. 2017). But the result shown in Table 2 indicates that, when the skip length does not start from 1, the performance will drop significantly due to missing dependencies. So we try another way and use 11 layers with the skip lengths from 1 to 1024. Unfortunately, the result shows that it failed to converge. This indicates that Dilated LSTM may suffer from the vanishing gradient problem when expanding the skip lengths by stacking more layers.

Table 2: The accuracies (%) of Dilated LSTM with different configurations on the VCTK data set. "/" represents the network failed to converge.

Model	Layers	Min skip length	Max skip length	Accuracy
Dilated LSTM	9	1	256	**90.7**
	8	8	1024	76.5
	11	1	1024	/

Effect of hyper-parameters

In this section, we study the effect of some specific hyper-parameters such as L, J, K. Once we set the length of the subsequence L and the number of level J of sub-pyramid, we can get the number of sub-pyramid N from $N = T/L$ and the aggregation granularity g from $J = \log_g L$. K is the number of layers of TP-RNNs. According to the previous description, the longest gradient feedback path from inputs to output is $J + N$. Theoretically, the smaller J or N is (the larger L is), the shorter the longest feedback path is, and the model has a faster convergence rate. Our experimental results confirm this conclusion.

Impact of the length of subsequence L

We fix the number of level J of sub-pyramid to 2, so the aggregation granularity g is set to be the square root of L. According to $L = T/N$, the larger N is, the smaller L is. We increase the size of L of TP-RNN-S successively, and the results tested on pMNIST are shown in Figure 1. As shown in Figure 1, the larger the L and g are (the smaller N is), the faster the convergence speed and the higher the accuracy, due to the shorter gradient feedback paths. This is consistent with our conclusion above.

Impact of the number of level J

We fix L to 64 and test J from 1 to 6. We use TP-RNN-S to test on pMNIST. Figure 2 (a) shows that the smaller J is, the higher the accuracy, due to the shorter feedback path.

Imapct the number of layers K

We set the maximum to 6 layers and test on pMNIST. Figure 2 (b) shows that the accuracy increases with the number of layers until saturation. The best result is achieved when K is equal to 3, so the number of layers of TP-RNN-M is set to 3 in our experiment.

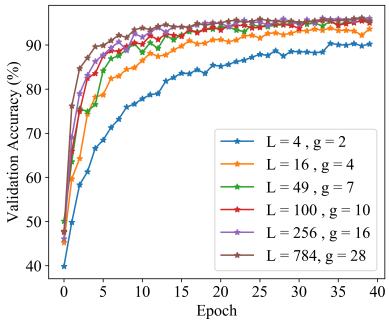

Figure 1: The validation accuracy curves of TP-RNN-S with different L and g on pMNIST data set.

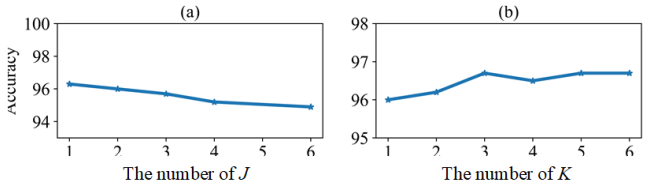

Figure 2: The result of pMNIST with different J and K

Convergence analysis

In this section, we analyze the convergence of TP-RNN compared with the baselines on three pixel-by-pixel image classification tasks.

MNIST and pMNIST

Figure 3 shows the validation accuracy curves of TP-RNNs with the baselines during training on MNIST and pMNIST data sets. We can clearly observe that the convergence speed of TP-RNNs is faster than LSTMs on both data sets. This verifies that the proposed pyramid-like structure can greatly alleviate the gradient vanishing problem and enable RNNs to be trained faster for learning long-term dependencies. Compared with Dilated LSTM and HM-LSTM, TP-RNN-M consistently converges faster with a comparable number of parameters. The convergence speed of TP-RNN-S is close to that of Dilated LSTM and faster than HM-LSTM on pMNIST. It is worth noting that the number of parameters of TP-RNN-S is only one-fifth of that of Dilated LSTM and one-eighth of that of HM-LSTM, so it is very lightweight. In addition, multi-layer TP-RNN converging faster than single-layer one further shows that learning multi-scale dependencies is helpful for training sequence modeling.

Noisy MNIST

The results with three setups, sequence length T = 1000, 2000 and 3000, on noisy MNIST are shown in Figure 4. We can observe that LSTMs and HM-LSTM fail to converge at any length. When T is below 2000, Dilated LSTM and TP-RNNs both can converge to high accuracies quickly. When T increases to 3000, TP-RNNs can still converge quickly and the accuracy remains around 98%, while Dilated LSTM fails to converge. We believe the consistently better performance

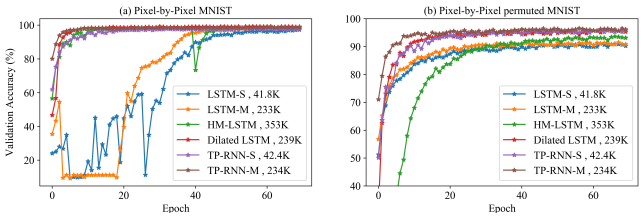

Figure 3: The validation accuracy curves on MNIST and pMNSIT data sets.

of TP-RNN across different sequence length T is benefit from the gradient feedback short-paths provided by the pyramid-like structure.

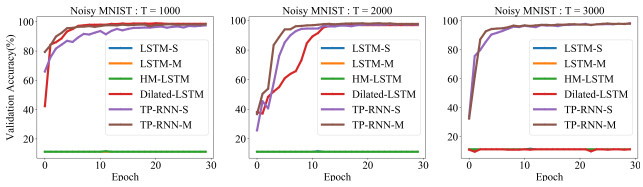

Figure 4: The validation accuracy curves on noisy MNIST data set for three setups, T = 1000 (left), 2000 (center) and 3000 (right).

Advantages discussion

In this section, we discuss the advantages of TP-RNN over existing hierarchical RNNs. Most of existing hierarchical RNNs were proposed to learn multi-scale dependencies with a multi-layer structure by gating mechanism (Chung et al. 2015; Kim, Singh, and Lee 2016) or updating mechanism (Chung, Ahn, and Bengio 2017). However, these existing models didnt consider the vanishing gradient problem in the deep RNN structure. The longest gradient feedback path of each layer is still equal to the input sequence length T, while TP-RNN shortens it to $J + N$. As for Dilated RNN (Chang et al. 2017), it employed skip connections with different skip lengths in different layers to learn multi-scale dependencies and shortened gradient feedback paths of each layer. However, the input sequence length T of each layer of Dilated RNN was still not shortened, while TP-RNN shortens it to T/g^j. A shorter input sequence can also shorten the gradient feedback path.

References

Arjovsky, M.; Shah, A.; and Bengio, Y. 2016. Unitary evolution recurrent neural networks. In *International Conference on Machine Learning*, 1120–1128.

Chang, S.; Zhang, Y.; Han, W.; Yu, M.; Guo, X.; Tan, W.; Cui, X.; Witbrock, M.; Hasegawa-Johnson, M.; and Huang, T. S. 2017. Dilated recurrent neural networks. In *Advances in Neural Information Processing Systems*, 77–87.

Chung, J.; Ahn, S.; and Bengio, Y. 2017. Hierarchical multi-scale recurrent neural networks. In *International Conference on Learning Representations*.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2015. Gated feedback recurrent neural networks. In *International Conference on Machine Learning*, 2067–2075.

Kim, M.; Singh, D. M.; and Lee, M. 2016. Towards abstraction from extraction: Multiple timescale gated recurrent unit for summarization. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, 70–77.

Le, Q. V.; Jaitly, N.; and Hinton, G. E. 2015. A simple way to initialize recurrent networks of rectified linear units. *arXiv preprint arXiv:1504.00941*.

Xia, W.; Zhu, W.; Liao, B.; Chen, M.; Cai, L.; and Huang, L. 2018. Novel architecture for long short-term memory used in question classification. *Neurocomputing* 299:20–31.