Time Delay Recurrent Neural Network for Speech Recognition

Boji Liu\textsuperscript{1,a}, Weibin Zhang\textsuperscript{1,b}, Xiangming Xu\textsuperscript{1} and Dongpeng Chen\textsuperscript{2}

\textsuperscript{1}School of Electronic and Information Engineering South China University of Technology, GuangZhou, China
\textsuperscript{2}VoiceAI Technologies, Shenzhen, China

Email: \textsuperscript{a}msboji@mail.scut.edu.cn, \textsuperscript{b}eweibin@scut.edu.cn

Abstract. In Automatic Speech Recognition (ASR), Time Delay Neural Network (TDNN) has been proven to be an efficient network structure for its strong ability in context modeling. In addition, as a feed-forward neural architecture, it is faster to train TDNN, compared with recurrent neural networks such as Long Short-Term Memory (LSTM). However, different from recurrent neural networks, the context in TDNN is carefully designed and is limited. Although stacking Long Short-Term Memory (LSTM) together with TDNN in order to extend the context information have been proven to be useful, it is too complex and is hard to train. In this paper, we focus on directly extending the context modeling capability of TDNNs by adding recurrent connections. Several new network architectures were investigated. The results on the Switchboard show that the best model significantly outperforms the base line TDNN system and is comparable with TDNN-LSTM architecture. In addition, the training process is much simpler than that of TDNN-LSTM.

1. Introduction
Intelligent virtual assistants such as Siri, Alexa and Cortana are becoming smarter and more capable. Many people become more reliant on them since they make our lives easier. One of the key components in these intelligent products is speech recognition, converting speech into text automatically by computers. Nowadays, neural networks have been applied in almost all commercial speech recognition systems to achieve state-of-art recognition accuracy. Speech is a signal with long temporal contexts. Thus it is very important for the acoustic model to capture the long-term temporal dependencies of speech. Many efforts have been spent in improving the temporal modeling capability of acoustic models.

At the feature level, feature representations such as TRAPs [1], wavelet based multi-scale spectro-temporal representations [2] and deep scattering spectra [3] have been proposed to improve the context modeling capability of the system. These features can be spliced and fed into a feed-forward neural network in order to model wider temporal contexts.

Model-based approach, which is the focus of this paper, can also be utilized to address this problem. Recurrent neural networks (RNNs) have cycle connections in the hidden layers [4]. Ideally, history information will be kept in the recurrent hidden nodes and theoretically unlimited contexts information can be utilized. Unfortunately, the gradient vanishing problem substantially deteriorate the performance of RNNs. This is because the gradient vanishing or exploding problem limit the capability of RNNs to model the long range context dependencies to 5-10 discrete time steps between
relevant input signals and output [5]. Variants of RNNs such as long short-term memory (LSTM, [4], [5], [6], [7], [8]) have been successfully applied to speech recognition to achieve state-of-the-art recognition accuracy. But LSTM needs much more time to train, compared with other feed-forward networks.

Time Delay Neural Network (TDNN) uses a feed-forward architecture, and has been proven to be powerful in handling the context information of speech signal [9]. The long range context information of speech signal is utilized through a carefully designed hierarchical structure. In a TDNN architecture, the first layer process input from narrow contexts of the speech signal. The deeper layers will process input by slicing the output of the hidden activations from the previous layer in order to learn wider temporal relationships. However, splicing continuous windows of frames in traditional TDNN structure leads to overlap and redundancy. To improve efficiency, sub-sampling is proposed [9]. Sub-sampling is a method allowing gaps between feature frames at each layer [9]. It can help decrease the number of parameters and increase the computation efficiency.

The success of TDNN shows that the most important information related to the recognition of current frame lies in a relatively narrow context. On the other hand, efforts to combine LSTM and TDNN have been done and improvement is observed [6], [8]. Especially in [8], the authors conducted a lot of experiments to evaluate different stacking structures of TDNN-LSTM network. The improvement by using TDNN-LSTM indicates the necessity of utilizing longer context information.

In this paper, we focus on other ways to extend the context modeling capability of TDNN. Because of the complexity of LSTM, we prefer not to use it in our structures. Mainly two methods are investigated: 1) instead of using LSTM, a RNN layer is used in a TDNN-RNN network. 2) direct recurrent connections are added and the new network is called Time-Delay Recurrent Network (TDRNN). Besides, the following issues are investigated in this paper:

- How to combine TDNN and RNN?
- The number of layers of RNN in TDNN model.
- Locations to add the recurrent connections in a TDNN model.
- Exploration of more complicate recurrent structure.

The following of this paper is organized as follows. In section 2 we describe the proposed structure in details, followed by the experimental setup in Section 3. In section 4 we present the experimental results and then the conclusions are given in section 5. Finally we will have some discussion about the redundancy.

2. Proposed method

2.1. Time Delay Neural Network

In a time delay neural network, the temporal context is modeled by using a hierarchical architecture. Each layer in a TDNN operates at a different temporal resolution. The outputs of the activation from previous hidden layer are spliced as the input of the current layer. Therefore, the current layer operates at a much wider context, compared with the previous layer. As we go to higher layers of the network, increasingly wide context is seen by the network.

Similar to Convolutional Neural Networks (CNNs [9]), the transforms in the same layer of a TDNN are tied across time in order to reduce the number of parameters and make the transformation invariant to time shift of the input [9]. TDNNs are seen as a precursor to the CNNs. [9] proposed a method to subsample the TDNN network. The splicing configuration {-1,1} means that we splice the input at current time step minus 1 and the current time step plus 1 (i.e. the current frame is dropped). Sub-sampling reduces the dimension of the input and thus the model size. Figure 1 shows a TDNN with sub-sampling.

The overall input contexts of TDNNs are limited, for example, asymmetric context windows of up to 16 frames in past and 9 frames in the future are investigated in [9]. The success of TDNNs indicates that the most valuable information for the recognition of the current frame lies in a relatively narrow context. This is true even when recurrent models are used. Truncated Back Propagation Through Time
is widely used in LSTM training to limit the context [8], [10], [11]. In an unidirectional LSTM [5], [8], [11], the left context length is usually set to 20.

![Figure 1](image1.png)  
**Figure 1.** A sub-sampling TDNN structure with config string "{-1,1 -1,1 -3,3 -3,3 -3,3 -3,3}".

![Figure 2](image2.png)  
**Figure 2.** Adding RNN layer in a TDNN structure then we get a TDNN-RNN structure. It is similar to TDNN-LSTM structure but is more efficient.

2.2. **TDNN-RNN**

The improvement by using TDNN-LSTM [6] indicates the necessity of utilizing longer context information. Due to the complexity of LSTM, it takes much more time to train the TDNN-LSTM model. We believe that the TDNN architecture has captured the most valuable context information. There is no need to add another very complicate component. Instead of TDNN-LSTM, we will explore the effectiveness of the TDNN-RNN structure as show in Figure 2. In this architecture, we add another RNN layer in the middle of a TDNN. The added RNN component might be able to utilize additional context to further improve the recognition accuracy.

2.3. **TDRNN**

Another architecture that we explored in this paper is the one with direct recurrent connection in the TDNN layer. Figure 3 shows an example of this new type of architecture which we call Time Delay Recurrent Neural Network (TDRNN). Same as the TDNN, the recurrent layer is tied across different time steps. As the most important contexts have been modeled in TDNN, adding limited additional context may be enough for the TDRNN to achieve the best performance.

Finally, we empirically found that it is better to add another transform to the recurrent connection as shown in Figure 4. The output of the TDRNN layer in figure 4 is further handled by one full-connect neural network, and then serves as the input of next time step. We call this structure as deep recurrent edge, and the optimal number of full-connect neural network should be investigated.
3. Experimental setup

All of the models are evaluated on the 300 hours Switchboard conversational telephone speech task [12] and the Nnet3 recipe in Kaldi toolkit [13] is used to build our experimental systems. The feature frame alignment is performed using GMM-HMM baseline recognition system as described in [9] 40-dimension Mel-frequency cepstral coefficients (MFCCs) without cepstral truncation are used as input. The input features are spliced by concatenating the "{-2,1,0,1,2}" frames. Feature adaptation is utilized by appending 100-dimension iVector with the MFCC input. Finally, the resulting 300-dimension feature is transformed by a 300-dimension linear discriminant analysis (LDA) and is used as the model input. Data augmentation technique is adopted to generate three copies of the training data with speed perturbation rates of 0.9, 1.0 and 1.1.

We uses a symmetric context configuration as shown in Table 1 for our baseline TDNN model, and the complete baseline TDNN structure is shown is figure 1. Using this configuration, the left and right contexts of the input signal are all 16. Cross entropy training criteria is used to trained all the models reported in this paper. The number of hidden nodes for the baseline TDNN is set to 1024.

**Table 1.** Layer-wise context config for sub-sampled TDNN baseline.

| layer | 1     | 2     | 3     | 4     | 5     | 6     |
|-------|-------|-------|-------|-------|-------|-------|
| context | {-1,1} | {-1,1} | {-3,3} | {-3,3} | {-3,3} | {-3,3} |

We firstly compare the performance of TDNN-RNN and TDNN-LSTM. To simplify the following description, we use the notations such as TDNN-RNN(3) to indicate inserting a RNN layer on top of the 3rd TDNN layer. More notations can be found in Table 2. The cell dimension of a LSTM layer is set to 1024, and both the recurrent projection unit and non-recurrent projection unit is set to 256. Truncated Back Propagation Through Time (BPTT) with 40 time steps is used in both TDNN-RNN and TDNN-LSTM models.
Table 2. Example config for TDRNN and TDNN-RNN models, where "T" means a TDNN layer, "TR" means a TDRNN layer, "L" means a LSTM layer and "R" means a RNN layer. TDRNN(2,4) means replace the TDNN on layer 2 and layer 4 by TDRNN while TDNN-RNN(3) means inserting a RNN on layer 3.

| model           | Layer config |
|-----------------|--------------|
| TDNN-LSTM(3)    | T T L T T T T |
| TDNN-RNN(3)     | T T TR T T T |
| TDRNN(2,4)      | T TR T TR T T |

We then go on to compare TDRNN and the baseline TDNN. The context length of recurrent layer at this step is limited to the context length of TDNN baseline (i.e. 16). The context length at this step is much shorter than the one in TDNN-LSTM model mentioned above, so is more efficient. To improve the training speed, the Classical Block Momentum(CBM) Blockwise Model-Update Filtering (BMUF) algorithm [14] is applied with 16 parallel jobs and the block momentum factor of 0.9.

4. Experimental Results

We present results on the Switchboard subset (labeled as swbd) and the complete Hub5'00 evaluation set (labeled as hub5). A develop set (labeled as dev) with 4000 sentences is selected randomly from the full training data set. Language model is built from Fisher transcripts as described in [9]. Experimental results are reported in word error rate (WER).

Results of TDNN-LSTM and TDNN-RNN models are presented in table 3. We also investigated RNN layers with different depth. From the table we can see that deep1-TDNN-RNN model performs comparable to TDNN-LSTM models. The training of the TDNN-RNN model is much faster than the TDNN-LSTM model and TDNN-RNN has fewer parameters.

Table 3. Comparison of TDNN-LSTM and TDNN-RNN. Deep1-TDNN-RNN means adding one full-connect layer on the recurrent edge.

| model            | WER    |     |
|------------------|--------|-----|
| TDNN-LSTM(3)     | 14.7   | 11.6| 17.0|
| TDNN-RNN(3)      | 14.5   | 11.7| 17.3|
| Deep1-TDNN-RNN(3)| 14.5   | 11.4| 17.1|

Results of step two is shown in table 4 and table 5. In table 4 we present the performance of TDRNN and RNN with different configurations. The table consist of six TDRNN structure with one TDRNN layer, seven TDNN-RNN structure with one RNN layer and four TDRNN structure with multiple TDRNN layer. Comparing one-layer TDRNN and one-layer TDNN-RNN, we found that the TDRNN performs a little better than the TDNN-RNN. By replacing RNN with TDRNN, we decrease the number of neural network layer in the model and improvement is obtained. On the other hand, we discover that experiment with recurrent layers near input and output performs bad, like TDRNN(6),TDNN-RNN(1) and TDNN-RNN(7). This phenomenon may be due to the mismatch of dimensions between recurrent layers and input/output.

As for the results of multiple TDRNN layer, we find that adding more TDRNN layers show little impact to WER, or even a little worse, and one recurrent layer should be suitable. The result is different from the TDNN-LSTM experiment presented in [8], and the latter one have three LSTM layer.
Table 4. Results with respect to different positions of recurrent edges.

| model          | WER   |
|----------------|-------|
|                | dev   | swbd | hub5 |
| TDNN           | 15.1  | 12.1 | 17.9 |
| TDRNN(1)       | 15.2  | 11.8 | 17.2 |
| TDRNN(2)       | 14.3  | 11.3 | **17.1** |
| TDRNN(3)       | 14.5  | 11.6 | 17.2 |
| TDRNN(4)       | 14.1  | 11.6 | 17.2 |
| TDRNN(5)       | 14.9  | 11.9 | 17.9 |
| TDRNN(6)       | 15.4  | 11.9 | 18.3 |

| model          | WER   |
|----------------|-------|
| TDNN-RNN(1)    | 16.4  | 12.6 | 18.2 |
| TDNN-RNN(2)    | 14.3  | 11.8 | 17.4 |
| TDNN-RNN(3)    | 14.2  | 11.6 | 17.1 |
| TDNN-RNN(4)    | 14.6  | 11.5 | 17.4 |
| TDNN-RNN(5)    | 14.3  | 11.6 | 17.4 |
| TDNN-RNN(6)    | 15.0  | 11.9 | 17.9 |
| TDNN-RNN(7)    | 15.4  | 12.3 | 18.7 |

| model          | WER   |
|----------------|-------|
| TDRNN(2)       | 14.1  | 11.6 | 17.2 |
| TDRNN(4,6)     | 14.3  | 11.6 | 17.5 |
| TDRNN(2,4,6)   | 14.3  | 11.4 | 17.1 |
| TDRNN(2,3,5,6) | 14.3  | 11.3 | **17.1** |

Table 5 investigates about the information processing power of recurrent structure by adding full-connected neural network (NN) layer on the recurrent edge. We define the number of NN layer on the recurrent edge as depth so deep1-TDRNN(2) is a structure with TDRNN on layer 2 and depth 1. The results show that TDRNN with depth 1 perform well, and adding depth couldn't decrease the WER. Besides, TDRNN still outperform TDNN-RNN.

Table 5. Depth of deep recurrent edge, deep2 means there are two full-connected neural network layer on the recurrent edge.

| model          | WER   |
|----------------|-------|
|                | dev   | swbd | hub5 |
| TDRNN(2)       | 14.3  | 11.3 | 17.1 |
| deep1-TDRNN(2) | 14.2  | 11.2 | **16.9** |
| deep2-TDRNN(2) | 14.3  | 11.6 | 17.1 |
| deep3-TDRNN(2) | 14.1  | 11.5 | 16.9 |
| TDNN-RNN(3)    | 14.2  | 11.6 | 17.1 |
| deep1-TDNN-RNN(3) | 14.2  | 11.2 | 17.1 |
| deep2-TDNN-RNN(3) | 14.4  | 11.6 | 17.2 |
| deep3-TDNN-RNN(3) | 14.4  | 11.4 | 17.1 |

In order to compare the training efficiency, we also draw the likelihood-training time curve, as shown in figure 5. From the figure we can see the training time of TDRNN models with different layers. The training time statistics is done roughly in same computer cluster, but the relative efficient is comparable. Experiment in step one is included in the figure with dotted line. Comparison of likelihood between experiment in step one and step two is meaningless, because the training procedure are different. But we can see a significant improvement on training convergence time in step two. The result shows that TDRNN can be trained mush faster than TDNN-LSTM.
5. Conclusion & Discussion

In this paper, we investigate adding recurrent connections directly on the TDNN structure. The new model is called Time Delay Recurrent Neural Network. TDRNN performs a little better than TDNN-LSTM and also has lesser layer. We try to improve the modeling ability for TDRNN layer and we find that TDRNN with one neural network layer on the recurrent edge (i.e. deep1-TDRNN) perform best. It obtains about 6% relative improvement, compared with the TDNN baseline. We didn't apply the complex training process for LSTM in our experiment, instead we use the TDNN training process to train the TDRNN model, so the model is very efficient. On the other hand, we reach this improvement without considering the extra context for RNN, instead we keep the context length same as TDNN baseline so the context length is reduced. The TDRNN model has the potential to outperform TDNN-LSTM model.

At this part we discuss about the redundancy of TDNN-LSTM model. Many experiment have shown that LSTM is good at long term information modeling. However, the successful application of context truncation when computing gradient BackPropagation Through Time (BPTT) also indicates that the context that are far from current frame show little impact to WER. TDNN is also good at long-term modeling, but the context is finite. The best combination of TDNN and recurrent models should not only success the efficient context modeling ability of TDNN, but also develop the infinite context modeling ability of recurrent model. However, combination of TDNN and LSTM may have the redundancy because the complexity of LSTM, and that's why we investigate the deep TDRNN. In the future, we would like to make a global evaluation on these combination.

Acknowledgments

This work is supported in part by the National Natural Science Founding of China (61601187, U1636218).
References

[1] H. Hermansky and S. Sharma 1999 Temporal patterns (traps) in asr of noisy speech IEEE International Conference on Acoustics, Speech, and Signal Processing, 1999. Proceedings pp 289–292 vol 1
[2] N. Mesgarani, M. Slaney, and S. A. Shamma 2006 Discrimination of speech from nonspeech based on multiscale spectro-temporal modulations IEEE Transactions on Audio, Speech, and Language Processing vol 14 no 3 pp 920–930
[3] J. Andn and S. Mallat 2014 Deep scattering spectrum IEEE Transactions on Signal Processing vol 62 no 16 pp 4114–4128
[4] A. Graves, A. Mohamed, and G. E. Hinton 2013 Speech recognition with deep recurrent neural networks IEEE International Conference on Acoustics, Speech, and Signal Processing pp 6645–6649
[5] H. Sak, A. W. Senior, and F. Beaufays 2014 Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition Computer Science pp 338–342 02
[6] G. Cheng, V. Peddinti, D. Povey, V. Manohar, S. Khudanpur, and Y. Yan 2017 An exploration of dropout with lstms Interspeech pp 1586–1590
[7] W. Xiong, J. Droppo, X. Huang, F. Seide, M. L. Seltzer, A. Stolcke, D. Yu, and G. Zweig 2017 The microsoft 2016 conversational speech recognition system IEEE International Conference on Acoustics, Speech, and Signal Processing pp 5255–5259
[8] V. Peddinti, Y. Wang, D. Povey, and S. Khudanpur 2018 Low latency acoustic modeling using temporal convolution and lstms IEEE Signal Processing Letters vol 25 no 3 pp 373–377
[9] V. Peddinti, D. Povey, and S. Khudanpur 2015 A time delay neural network architecture for efficient modeling of long temporal contexts Interspeech pp 3214–3218
[10] K. Chen and Q. Huo 2016 Training deep bidirectional lstm acoustic model for lvcsr by a context-sensitive-chunk bptt approach IEEE Transactions on Audio, Speech, and Language Processing vol 24 no 7 pp 1185–1193
[11] Y. Zhang, G. Chen, D. Yu, K. Yaco, S. Khudanpur, and J. R. Glass 2016 Highway long short-term memory rnns for distant speech recognition International Conference on Acoustics, Speech, and Signal Processing pp 5755–5759
[12] J. Godfrey, E. Holliman, and J. McDaniel 1992 Switchboard: telephone speech corpus for research and development IEEE International Conference on Acoustics, Speech, and Signal Processing vol 1 04 pp 517–520
[13] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz et al 2011 The kaldi speech recognition toolkit
[14] K. Chen and Q. Huo 2016 Scalable training of deep learning machines by incremental block training with intra-block parallel optimization and blockwise model-update filtering pp 5880–5884