# Crossed-Time Delay Neural Network for Speaker Recognition

## Abstract

Time Delay Neural Network is an well-performing structure for DNN-based speaker recognition methods. In this paper we introduce a novel structure Cross-Time Delay Neural Network (CTDNN) updating the performance of TDNN. Inspired by the multi-filter setting of convolution layer from computer vision, we set different time delay units at the bottom layer each with different context size and construct a multilayer parallel network which has been proved global optimality. It gives significant improvements over TDNN as we tested in speaker identification tasks especially in few shots condition where CTDNN doubles the accuracy of original TDNN. We also compare the proposed CTDNN with TDNN-F, which shows that our model has a 22% absolute accuracy improvement under few shots condition and can also better utilize the calculation resources with a faster training speed.

Index terms: time delay neural network, speaker recognition, feature extraction, acoustic modeling

## Introduction

Speaker recognition system verifies or identifies a speaker’s identity based on speaker’s voice. It can be divided into speaker verification and speaker identification, where speaker verification aims to verify whether an utterance corresponds to a given identity and speaker identification aims to identify a speech from all enrolled speakers. According to the different testing scenario, speaker recognition can also be categorized into closed-set or open-set settings. For closed-set scenario, all testing identities are enrolled in the training set, therefore it can be regarded as a classification problem. For open-set scenario, the testing identities are not previously seen in the training set, which is closer to real world application since new identities will be added to the system continually. To address that problem, each utterance must be mapped into an embedding space where cosine similarity is used to evaluate whether two utterances correspond to one same identity. This paper and most of others mainly focus on the open-set speaker recognition problem.

Recently, deep neural network has been widely applied to learning speakers’ embedding through the learning process of classification such as X-Vectors[[]] and have shown great priority in performance [] than traditional statistical models. Time delay layer is an important component among most of the DNN-based models.

Time Delay Neural Network was regarded as the ancestor of convolution neural network [1]. It is effective in capturing features from long range temporal contexts and is widely used in speech related field such as speaker recognition system [11] automatic speech recognition [12] and voice conversion []. The TDNN architecture, shown in Figure 1, uses a modular and incremental method to create larger networks from sub-components [10]. The time delay architecture can be regarded as a convolution on sequence data where a 1-dimension filter scans through the input sequence and generate an output at each step with the strategy of weight-sharing. Many related works have focused on TDNN such as TDNN-LSTM [13], TDNN-BLSTM [14], CNN-LSTM-TDNN [15] and TDNN-F [2]. [13,14,15] focus on combining TDNN with different components to construct better model and [2] proposed a variant of TDNN through low-rank matrix factorization to overcome gradient explosion problem for TDNN-based network structures.

We propose the cross-time delay neural network as a variant of TDNN, named CTDNN. The multiple-filters mechanism of a convolution layer in CNN inspires us to set different time delay units in the bottom layer of the network. In CNN, each filter with different parameters in the same convolution layer captures different characteristics of the input, which ultimately helps to classify the input image. In the original TDNN, there is only one filter in one layer, which restricts the models feature extraction and generalization ability according to our analysis and experiments. Our CTDNN structure has three main advantages:

* The time delay units with different context size in the same layer help to extract more heterogeneous features.
* The structure is wider and shallower, but not deeper, which avoids gradient explosion and vanishing problem arising occasionally in the training process and guarantees the generalization ability.
* Our model works well with large batch, compared to TDNN-F [2], which enables it to utilize calculation resources in a more efficient way without alternating the batches frequently.

这里对Speaker Identification研究现状的阐述不足。

应该明确Speaker Identification的主要方法类型，神经网络的优势，用于Speaker Identification的典型神经网络的现状，是否有不足，本文的出发点，目标是解决Speaker Identification领域的什么难点，后文所提方法的合理性。

## Baseline Models

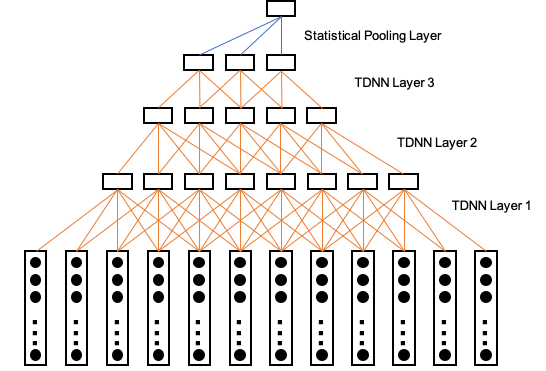


Figure 1 \*\*\*\*（图中文字太小）

The network architecture of our speaker identification baseline systems are the same as the original TDNN architecture in [1] and the improved architecture TDNN-F in [2]. In a TDNN architecture shown in Figure 1, the shallow layers are learned on relatively narrow contexts and each deeper layer can process the input from a wider temporal context (shallow layers, deeper layer没有交代清楚). The time delay architecture can be regarded as a one-dimension convolution on sequence data where a 1-d filter in different layers scans through the input sequence by the strategy of weight-sharing. After the time delay layers (前面需要有一个对TDNN整体的清晰描述，不然后面类似“After the time delay layers”就不明确) is the statistical pooling layer which computes the statistical feature, followed by fully connected layers and softmax to project the sequence into speaker’s identity. During back-propagation, the lower time delay layers are updated by a gradient accumulated over all the time steps of the input sequence. Thus the lower layers of the network are forced to learn translation invariant feature transforms [3].

The TDNN-F is a factored form of TDNN which is structurally the same as a TDNN whose layers have been compressed via SVD (第一次出现全拼) to reduce the number of parameters, and uses shortcut connection [4] and highway connections in order to avoid gradient diffusion problems in deeper network.

## Crossed-Time Delay Neural Network

The proposed CTDNN is shown in Figure 2, which is more of a wide and shallow structure rather than a narrow and deep structure. It combines the Crossed-Time delay layers and the statistical layers.

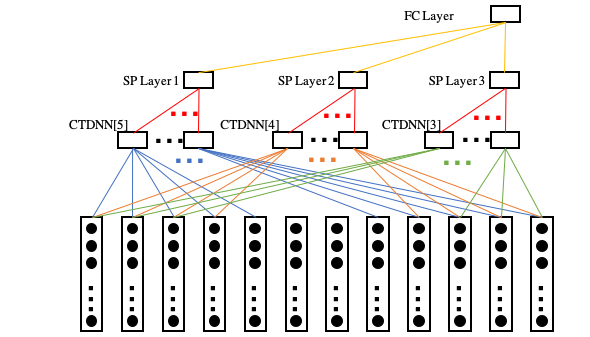


Figure 2 \*\*

**3.1 Crossed-Time Delay Layer**

We set different time delay units in the bottom layer to directly extract features from the input sequence. Each unit has a different context frame size from wide to narrow(前面缺少铺垫，不是很清晰). In Figure 2, the context frame sizes are 5, 4 and 3 marked as CTDNN[5], CTDNN[4] and CTDNN[3]（编号容易与文献编号混淆）. Each unit scans the input sequence separately and output a fixed size vector at each step till the end of the sequence. In other words, we can regard the different time delay units as different filters and each of them take the sequence as input and outputs different feature maps in computer vision terms. The CTDNN layers can also be stacked vertically to form a deeper hierarchy structure, in this case, each feature map should be allocated a new time delay unit.

It seems against the consensus that the deep and narrow network is better than the wide and shallow one as discussed in [5]. However, the extension of the layer width is not to simply add more neurons and connections, but to extract features at different frequencies or paces. We exploit the strength of the structure from two perspectives including the heterogeneous feature extraction and ……..

3.1.1 The heterogeneous feature extraction

Using crossed-time delay units can extract more heterogeneous feature than that of a TDNN. Since the raw audio is viewed as short-time stationary signal, it has to be framed to short-time pieces at a fixed frequency to further analyze the audio and extract other features like MFCC. In original TDNN models, the time delay units are stacked vertically, and each unit has fixed reception field1 and parameters within connections. This single-line structure has the bottom layer to domain the feature extraction capacity, which limits the generalization ability of the model.

Take the model shown in Figure 1 as an example. The bottom layer has a context size of 5, so it take in 5 frames of feature at a time. The second layer has a input size of 4, and it takes in four features from the bottom layer as input, which enlarges its context size to 8 due to the tree-like vertical structure. However the second layer does not actually takes input from a context size of 8 but the linear combination of 4 short sequences at the size of 5. So does the deeper layer. So the key of the model is up to the bottom layer. With a fixed set of parameters and context size, the feature it get is homogeneous since there are features that range more or less than 5 frames because of the short-time stationary property of audio signal and those features cannot be captured by one fixed-context-size time delay unit.

As shown in Figure 2, we set 3 time delay units (图2没有看出来呢) each with a different context size at the bottom layer. During back-propagation, due to the different context size, the lower layers of the network are updated by a gradient accumulated over different time steps of the input temporal context. Thus the lower layers of the network are forced to learn different feature transforms, which enlarges the feature extraction capacity of the model to have better generalization performance.

Second, shallow networks are more feasible to train and converge, specially on small datasets. Training might suffer from gradients vanishing or exploding problems during the process of back-propagation in deep neural network [6]. The literature [7] found that relatively small network sizes have obvious computational advantages when training on small dataset. We leverage the depth and width of CTDNN in our experiments and find that building two CTDNN layers can outperform 5 normal TDNN layers in both common and few-shots learning tasks.

**3.2 Statistical Concatenation**

Since the context size of time delay unit differs in the bottom layer, the output of the units will have different length. Instead of doing statistical pooling on all the output in one time, we compute the mean and standard deviation for each time delay unit’s output and concatenate the results parallel before the fully connected layer.

## 4. Experiments

We conduct our experiments on the open VCC2016 dataset. It includes 10 speakers and each speaker has 162 different utterances lasting from 1s to 6s. In the first experiment the ratio of training data and test data is 7:3 and in the second experiment the ratio is 1:9 to fully test different models’ generalization ability under few samples condition.

4.1 Preprocessing

The acoustic features were 13-dimensional MFCC features extracted every 10 ms and the frame size for short-time Fourier transform (STFT) was 25 ms. In order to obtain the same length inputs, we duplicate the short-length input and cut off the extra length to make all the input at the same length. Our model was implemented with PyTorch.

4.2 Model Configurations

Table 1. \*\*\*\*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer | TDNN | CTDNN | Layer | TDNN-F |
| 1 | TDNN[-2,2] | CTDNN[-4,4];CTDNN[-2,2];CTDNN[-1,1] | 1 | TDNN[-2,2] |
| 2 | TDNN[-1,2] | CTDNN[-1,1];CTDNN[-1,1];CTDNN[-1,1] | 2 | FTDNN |
| 3 | TDNN[-3,+3] | SP | ... | (6 FTDNN) |
| 4 | TDNN[-7,2] | FC | 9 | FTDNN |
| 5 | SP | FC | 10 | FC |
| 7 | FC | Softmax | 11 | SP |
| 8 | FC |  | 12 | FC |
| 9 | Softmax |  | 13 | FC |
|  |  |  | 14 | Softmax |

In Table 1, FC stands for the Fully Connected layer and SP for Statistical Pooling Layer. We construct the TDNN and TDNN-F structure the same as [1] and [2]. TDNN structure combines of 4 TDNN layers. TDNN-F has up to 14 layers, i.e., the deepest structure in our experiments. All the TDNN, CTDNN, and FTDNN layers in the three models have batch normalized input and are activated by ReLU function. Dropout and skip-connection policies are only involved in FTDNN (FTDNN和TDNN-F含义不同？) not in TDNN and CTDNN. To be recognized, the proposed CTDNN has a wider and shallower structure. We set 3 time-delay units in the first and second layer. And the units in first layer have different context size to directly extract heterogeneous features from the input.

4.3 Training Parameters Setting

We used cross entropy as the loss function. Adam optimizer [8] was used and the training batch size was 128. The learning rate was fixed to 0.0001 for CTDNN and TDNN, 0.0001 for FTDNN and Pre-training models were not involved in the experiments.

## 5. Results

Table 2. The Best Top1 Test Accuracy

|  |  |  |
| --- | --- | --- |
| Structure | Exp. 1 (Sample Ratio 7:3) | Exp. 2 (Sample Ratio 1:9) |
| TDNN | 0.778 | 0.448 |
| CTDNN | **0.992** | **0.904** |
| FTDNN 32 | 0.965 | 0.662 |
| FTDNN 128 | 0.608 | 0.681 |

Table 2 shows the results on two experiments. For the experiment 1 (Exp. 1) and the experiment 2 (Exp. 2), the ratios of training to testing samples are 7:3 and 1:9, respectively. In both experiments, our CTDNN outperforms the other structures, especially in few shots learning in which the accuracy is more than 2 times of the original TDNN. Moreover, the experiments shows that the performance of FTDNN gets worse when batch size grows and can’t converge with the batch size of 128. We then tune the batch size and find 32 is the most proper setting for the first experiment. However, FTDNN can’t work well with any batch size compared to CTDNN in few shots condition.

The reason for FTDNN can’t do well with large batch size is actually a general problem as discussed in [9]. There is still no consensus on how to tune batch size for different models. Different models might have different best batch size on different tasks. A large batch can significantly speed up the training while might suffer from loss in accuracy compared with small batch. From that perspective, the fact that CTDNN can achieve higher accuracy with large batch size also suggests that it can take full advantage of the GPU resources and speed up the training process.

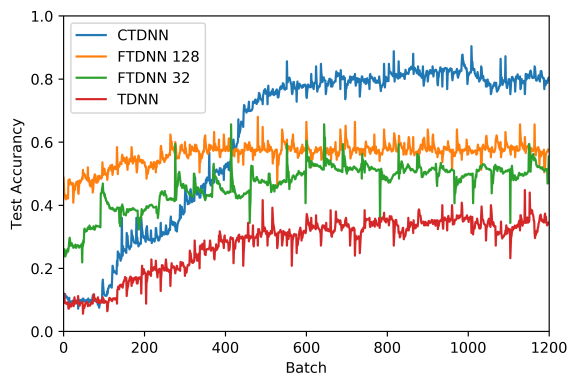
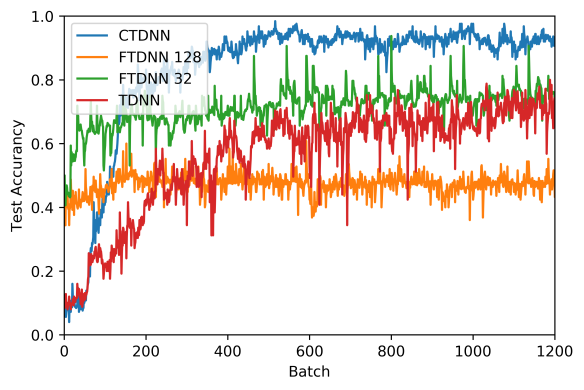


Figure 3. Training Curve of Exp. 1 with Sample Ratio 7:3 (left) and Exp. 2 with Sample Ratio 1:9 (right)

Figure 3 shows the curves of test accuracy during training. It can be seen that CTDNN comes to convergence with higher accuracy than other models in both experiments. And the curve is much more smooth for CTDNN, which suggests the structure is more stable (这个特点不太容易看出来，是否暂且不提).

## 6.Conclusion

In this work, we analyzed and examined the performance the new structure CTDNN. Our analysis suggests that the cross-time delay units can extract heterogeneous features therefore achieve better generalization ability. And our experiments proved our analysis and showed the of large batch capacity of CTDNN .

TDNN was once the precursor of convolution neural network, now we apply the characteristics from CNN to improve TDNN and gain improvements. In the future we will explore more application of CTDNN such as using it to improve different TDNN based model and combine it with embedding extraction system like x-vector to find out its effect on embedding.

Reference:

1. V. Peddinti, D. Povey, and S. Khudanpur, “A time delay neural network architecture for efﬁcient modeling of long temporal contexts,” in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2015, vol. 2015-Janua, pp. 2–6.
2. D. Povey et al., “Semi-orthogonal low-rank matrix factorization for deep neural networks,” Proc. Annu. Conf. Int. Speech Commun. Assoc. INTERSPEECH, vol. 2018-September, no. 2, pp. 3743–3747, 2018.
3. A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, “Phoneme recognition using time-delay neural networks,” IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 37, no. 3, pp. 328–339, Mar. 1989.
4. K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings ofthe IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
5. H. Mhaskar, Q. Liao, and T. Poggio, “When and why are deep networks better than shallow ones?,” 31st AAAI Conf. Artif. Intell. AAAI 2017, pp. 2343–2349, 2017.
6. R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks,” 30th Int. Conf. Mach. Learn. ICML 2013, no. PART 3, pp. 2347–2355, 2013.
7. M. Olson, A. J. Wyner, and R. Berk, “Modern neural networks generalize on small data sets,” Adv. Neural Inf. Process. Syst., vol. 2018-December, no. NeurIPS, pp. 3619–3628, 2018.
8. D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” Computer Science, 2014.
9. F. He, T. Liu, and D. Tao, “Control Batch Size and Learning Rate to Generalize Well : Theoretical and Empirical Evidence,” no. NeurIPS, 2019.
10. A. Waibel, “Modular construction of time-delay neural networks for speech recognition,” Neural computation, vol. 1, no. 1, pp. 39–46, 1989.
11. “D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudan- pur, “X-vectors: Robust dnn embeddings for speaker recognition,” in 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2018, pp. 5329–5333.
12. D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, “Speaker recognition for multi-speaker conversations using x-vectors,” in Proceedings of ICASSP, Brighton, UK, May 2019.
13. Chen, C.-P., Zhang, S.-Y., Yeh, C.-T., Wang, J.-C., Wang, T., & Huang, C.-L. (2019). Speaker Characterization Using TDNN-LSTM Based Speaker Embedding. ICASSP 2019
14. Y. Qin, T. Lee, S. Feng, and A. P. Hin Kong, “Automatic speech assessment for people with aphasia using TDNN-BLSTM with multi-task learning,” Proc. Annu. Conf. Int. Speech Commun. Assoc. INTERSPEECH, vol. 2018-September, no. September, pp. 3418–3422, 2018.
15. X. Miao, I. McLoughlin, and Y. Yan, “A new time-frequency attention mechanism for TDNN and CNN-LSTM-TDNN, with application to language identification,” Proc. Annu. Conf. Int. Speech Commun. Assoc. INTERSPEECH, vol. 2019-September, pp. 4080–4084, 2019.