Dynamic temporal residual network for sequence modeling

Ruijie Yan, Liangrui Peng, Shanyu Xiao, Michael T. Johnson & Shengjin Wang

International Journal on Document Analysis and Recognition (IJDAR)

ISSN 1433-2833
Volume 22
Number 3

IJDAR (2019) 22:235-246
DOI 10.1007/s10032-019-00328-x

Springer

International Journal on Document Analysis and Recognition (IJDAR) (2019) 22:235–246
https://doi.org/10.1007/s10032-019-00328-x

SPECIAL ISSUE PAPER

Dynamic temporal residual network for sequence modeling

Ruijie Yan[1] · Liangrui Peng[1] · Shanyu Xiao[1] · Michael T. Johnson[2] · Shengjin Wang[1]

Received: 16 November 2018 / Revised: 15 April 2019 / Accepted: 11 June 2019 / Published online: 2 July 2019

Abstract
The long short-term memory (LSTM) network with gating mechanism has been widely used in sequence modeling tasks including handwriting and speech recognition. As an LSTM network can be unfolded along the temporal dimension and its temporal depth is equal to the length of the input feature sequence, the introduction of gating might not be sufficient to completely model the dynamic temporal dependencies in sequential data. Inspired by the residual learning in ResNet, this paper proposes a dynamic temporal residual network (DTRN) by incorporating residual learning into an LSTM network along the temporal dimension. DTRN involves two networks: Its primary network consists of modified LSTM units with weighted shortcut connections for adjacent temporal outputs, while its secondary network generates dynamic weights for the shortcut connections. To validate the performance of DTRN, we conduct experiments on three commonly used public handwriting recognition datasets (IFN/ENIT, IAM and Rimes) and one speech recognition dataset (TIMIT). The experimental results show that the proposed DTRN has outperformed previously reported methods.

Keywords Long short-term memory · Residual learning · Off-line handwriting recognition · Speech recognition

1 Introduction

Because temporal sequences are neither always uniformly structured, nor uniformly predictable [9], it is challenging to capture dynamic dependencies in sequence modeling tasks such as handwriting and speech recognition. The long short-term memory (LSTM) [18] network has made considerable progress in sequence modeling by introducing a gating mechanism for the vanilla recurrent neural network (RNN), which alleviates the gradient vanishing or exploding problem [17] in training through the back-propagation through time (BPTT) method. In real applications, an LSTM network not only has multiple stacked layers in a spatial structure, but also has the same depth as the length of the feature sequence when unfolded in time [24], which indicates that there is a possibility for further improvements in the temporal structure to better capture dynamic temporal dependencies in sequential data.

Residual learning has been shown to be effective for deep feed-forward neural networks by models such as ResNet [16]. Efforts are already underway to simplify RNN optimization by constructing shortcut connections between stacked RNN layers [35,39], which is similar to ResNet. However, because the temporal depth of an unfolded LSTM network is usually deeper than the spatial depth of its stacked layers, it makes sense to explore the temporal residual learning mechanism for LSTM. Although other methods with skip connections exist that can also reuse prior LSTM outputs [4,15], these methods do not follow the residual learning formula.

Following the residual learning concept, we construct a temporal residual learning architecture for LSTM that is the equivalent of adding shortcut connections between the adjacent temporal outputs of LSTM units. Different from the existing temporal residual learning mechanism for classical RNNs [38], which uses gates to control the residual shortcuts,

✉ Liangrui Peng
penglr@tsinghua.edu.cn

Ruijie Yan
yrj17@mails.tsinghua.edu.cn

Shanyu Xiao
xiaosy15@mails.tsinghua.edu.cn

Michael T. Johnson
mike.johnson@uky.edu

Shengjin Wang
wgsgj@tsinghua.edu.cn

1 Beijing National Research Center for Information Science and Technology, Department of Electronic Engineering, Tsinghua University, Beijing 100084, China

2 Department of Electrical and Computer Engineering, University of Kentucky, Lexington, KY 40506, USA

we investigate dynamic weights generated by a separate secondary network. We name this proposed method a dynamic temporal residual network (DTRN).

We explore two structures for the secondary network, i.e., an LSTM network and a self-attention network [34]. The secondary network is designed to capture the dynamic dependency by exploring the contextual information in the input sequence.

The main contributions of this paper are summarized as follows:

1. We propose a temporal residual learning mechanism for the LSTM network by adding weighted shortcut connections between the temporally adjacent outputs, which introduces additional nonlinear modeling ability.
2. A secondary network is proposed to generate the dynamic coefficients for the weighted shortcut connections. Two different structures including an LSTM network and a self-attention network are explored for the secondary network to model the dynamic temporal dependencies.
3. The experimental results on both off-line handwriting recognition and speech recognition show that the proposed method has outperformed previously reported methods and has better generalization ability than does the classical LSTM network.

The remainder of this paper is organized as follows. Section 2 provides a brief review of related work. Section 3 presents our methods. Section 4 provides the experimental results and analyses, and Sect. 5 concludes the paper.

2 Related work

Off-line handwriting recognition and speech recognition are classical sequence modeling problems. With the emergence of deep learning, gated RNNs such as the LSTM networks have provided promising solutions.

For off-line handwriting recognition, multidimensional LSTM (MDLSTM) with connectionist temporal classification (CTC) has been shown to be effective for a variety of languages, including Arabic [12], French [23] and Chinese [36]. Recently, CNN-LSTM architectures [8,19,29,31] have achieved higher recognition accuracy than MDLSTM.

LSTM-CTC architectures have also been adopted for deep learning-based automatic speech recognition (ASR) systems on both small [13] and large [2,5] datasets.

Recently, deep neural networks with attention [3,6] or self-attention [34] mechanism have shown improved performance in sequence modeling tasks. Attention in deep learning can be broadly interpreted as a vector of importance of the contextual information. The self-attention network is adopted in our method as an alternative way to generate dynamic weights.

The ResNet [16] and the highway network [32] with shortcut connections have successfully eased the training of deep forward neural networks. The ResNet with identity shortcut connections can be regarded as a special case of the highway network with shortcut connections controlled by gates. To improve the performance of RNN models, some methods also consider shortcut connections for RNNs. For the spatial structure, attempts have been made to add shortcut connections between adjacent stacked LSTM layers in machine translation [35] and speech recognition [39] systems.

For the temporal structure, skip connections inside classical RNN or LSTM units [4,15] have been explored to overcome the gradient vanishing and explosion problems and to capture the long-term dependencies. The skip RNNs use skip shortcut connections for previous hidden states across multiple time steps with preset fixed weights, and they do not follow the residual learning formula.

Some existing methods also explore temporal residual learning for classical RNNs. Yue et al. [38] propose temporal residual connections that are either identical (similar to the ResNet) or controlled by gates (similar to the highway network). For dynamic weights generated by a separate network, Pei et al. [27] propose recurrent attention-gated units for sequence classification. Attention scores are calculated by using an attention module to localize the salient observations in the input sequence. The recurrent attention-gated units use the attention scores as the weights for the convex summation of the previous hidden states and the candidate hidden states. For the primary network, these two methods focus on connecting the hidden states of a classical RNN, while the proposed method explores shortcut connections between temporal LSTM outputs. For the secondary network, the main difference is that our secondary network is an LSTM network or a self-attention network, while the attention module in [27] is a classical RNN.

3 Methodology

3.1 Dynamic temporal residual learning

We can consider $\mathcal{H}(x_t) = LSTM(x_t)$ as an underlying mapping to be fit by an LSTM unit, where x_t is the current input at time step t. We explicitly let the unit approximate a residual function: $\mathcal{F}(x_t) = \mathcal{H}(x_t) - \mathcal{H}(x_{t-1})$, where $\mathcal{H}(x_{t-1})$ is the previous output of LSTM at time step $t-1$. For brevity, we abbreviate $\mathcal{F}(x_t)$ as $\mathcal{F}_t$ and $\mathcal{H}(x_t)$ as $\mathcal{H}_t$. The original function thus becomes $\mathcal{H}_t = \mathcal{F}_t + \mathcal{H}_{t-1}$. The temporal residual learning form is designed to simplify the learning process by letting $\mathcal{F}_t$ focus on the difference between $\mathcal{H}_t$ and $\mathcal{H}_{t-1}$.

Temporal residual learning can be implemented by adding a shortcut connection between the previous output and the

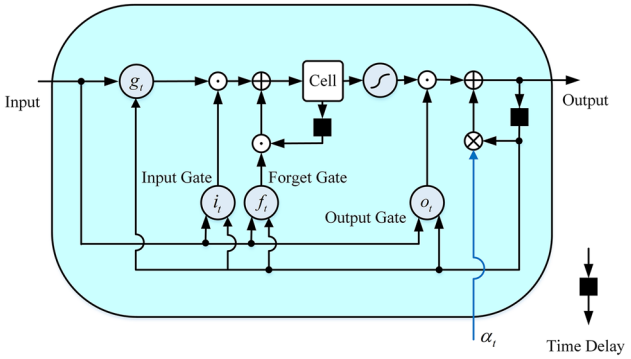


Fig. 1 Computational graph of a modified LSTM unit with the temporal residual learning mechanism

current output of an LSTM unit. To better model the dynamic dependency, we further introduce a dynamic weight α_t for the shortcut connection, i.e., $\mathcal{H}_t = \mathcal{F}_t + \alpha_t \mathcal{H}_{t-1}$. With the initial hidden state $\mathcal{H}_0 = \mathbf{0}$, we have:

$$\begin{aligned} \mathcal{H}_t &= \mathcal{F}_t + \alpha_t \mathcal{H}_{t-1} \\ &= \mathcal{F}_t + \sum_{k=1}^{t-1} \left(\prod_{i=k+1}^{t} \alpha_i \right) \mathcal{F}_k, \ 1 \le t \le T \end{aligned} \tag{1}$$

Equation (1) indicates that the desired underlying mapping of the current time step t can be decomposed into two additive parts. The first part, $\mathcal{F}_t$, is the nonlinear representation newly learned from the current input, while the second part contains the additional contextual information of the previous time steps.

In the second part of Eq. (1), $\mathcal{F}_k$ is weighted by $\prod_{i=k+1}^{t} \alpha_i$. If $\alpha_t \equiv 0$, the result is the same as the classical LSTM. When $\alpha_t \equiv 1$, identity mapping is constructed. When $0 < \alpha_t < 1$, the closer time step contributes more to $\mathcal{H}_t$, while the more distant time steps rarely affect the current time step due to the continuous multiplication. This property complies with the intuitive observation for most sequential data in handwriting and speech recognition tasks. Thus, by adding the dynamic weight α_t, the model can better capture complex dependencies by exploring the additional contextual information in the sequential data.

3.2 Dynamic temporal residual network

The proposed DTRN consists of a primary network and a secondary network; the primary network is a modified LSTM network, and the secondary network is a dynamic coefficient generator.

The core of a modified LSTM unit in the primary network is the shortcut connection between the temporally adjacent outputs. A detailed computational graph of the modified LSTM units is shown in Fig. 1.

The following recurrent transition functions describe the formula for a classical LSTM unit:

$$g_t = \phi(W_{xg} x_t + W_{hg} h_{t-1} + b_g) \tag{2}$$
$$i_t = \sigma(W_{xi} x_t + W_{hi} h_{t-1} + b_i) \tag{3}$$
$$f_t = \sigma(W_{xf} x_t + W_{hf} h_{t-1} + b_f) \tag{4}$$
$$o_t = \sigma(W_{xo} x_t + W_{ho} h_{t-1} + b_o) \tag{5}$$
$$c_t = i_t \odot g_t + f_t \odot c_{t-1} \tag{6}$$
$$h_t = o_t \odot \phi(c_t) \tag{7}$$

where $\phi(\cdot)$ and $\sigma(\cdot)$ refer to the hyperbolic tangent (tanh) function and the sigmoid function, respectively. The symbol $\odot$ represents the elementwise product.

With the temporal residual connections, the current output h_t is computed by adding the weighted delayed output h_{t-1} to the output of a classical LSTM unit; thus, Eq. (7) is modified into the following equation:

$$h_t = o_t \odot \phi(c_t) + \alpha_t h_{t-1} \tag{8}$$

where $o_t \odot \phi(c_t)$ is the nonlinear representation that the network should learn at time step t. The weight of the shortcut connection at time step t is denoted by α_t.

3.3 Dynamic weight generation

There are two ways to obtain α_t, i.e., statically or dynamically. For the static method, we manually set a constant value for all α_t (i.e., $\alpha_t \equiv \alpha$ $(0 < \alpha \le 1)$). In this situation, we call the network a static temporal residual network (STRN).

For dynamic weights, we propose adopting a secondary network in DTRN to generate α_t dynamically from the input sequences. At each time step, an internal vector representation z_t is first learned by a network from the input feature x_t and then converted to a scalar α_t by a fully connected layer with one neuron. The scalar is used as the weight of the shortcut connection for the corresponding time step in the primary network. In a general form, the dynamic weights are generated as the following equations show:

$$z_t = g(x_t) \tag{9}$$
$$\alpha_t = \sigma(w^T z_t + b) \tag{10}$$

where $g(\cdot)$ is the composite mapping function to learn the internal vector representation z_t and w and b are the trainable parameters of the fully connected layer. We use the sigmoid activation function $\sigma(\cdot)$ to obtain α_t $(0 < \alpha_t < 1)$.

We investigate two structures for the secondary network, i.e., an LSTM network and a self-attention network. For

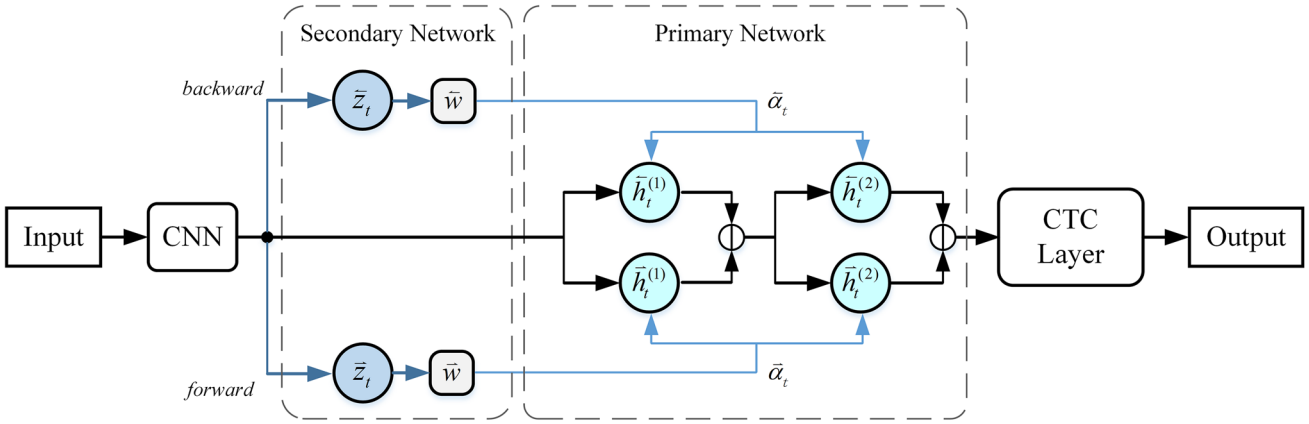


Fig. 2 System framework of a bidirectional DTRN with a two-layer primary network. The primary network is a modified LSTM network; the symbol ⓘ refers to feature concatenation. The secondary network can be an LSTM network or a self-attention network along with a fully connected layer. The outputs of the secondary network are used as weights for the shortcut connections in the primary network. A feature extractor such as a CNN can be added into DTRN

LSTM, we have $z_t = LSTM(x_t)$; for self-attention, the computational flow of z_t is as follows:

$$q_t = \phi(W^Q x_t + b^Q) \tag{11}$$

$$k_i = \phi(W^K x_i + b^K),\ 1 \le i \le T \tag{12}$$

$$v_i = \phi(W^V x_i + b^V),\ 1 \le i \le T \tag{13}$$

$$a_i = softmax(\frac{1}{\sqrt{d_k}} q_t^T k_i),\ 1 \le i \le T \tag{14}$$

$$z_t = \sum_{i=1}^{T} a_i v_i \tag{15}$$

where $W^Q, W^K, W^V \in \mathbb{R}^{d_k \times n}$ are three learnable matrices. Here, we denote the dimension of the input as n and the dimension of the queries, keys and values as d_k. At each time step, the values are weighted and summed, where the weights are computed by the keys and the query at the corresponding time step.

We use a discount factor to constrain the maximum weights of the shortcut connections for DTRN. As a result, Eq. (8) actually becomes:

$$h_t = o_t \odot \phi(c_t) + \gamma \cdot \alpha_t h_{t-1} \tag{16}$$

where γ $(0 < \gamma < 1)$ in Eq. (16) is the preset discount factor.

3.4 System implementation

Figure 2 shows a schematic diagram of DTRN. The primary and secondary networks in DTRN are both bidirectional; thus, for each direction in the secondary network, there is a corresponding LSTM or self-attention network. The obtained $\overrightarrow{\alpha_t}$ and $\overleftarrow{\alpha_t}$ are used for the corresponding directions of the primary network.

A feature extractor can be added into DTRN. We use a CNN as the feature extractor for off-line handwriting recognition and a Mel frequency cepstral coefficients (MFCC) [7] feature extractor for speech recognition.

The two networks in DTRN are trained jointly by setting a common loss function. We use the CTC loss [11] in our model. Let the input sequence be x and the target sequence be y; S represents the training set, and we get the CTC loss as follows:

$$\mathcal{L} = -\ln \prod_{(x,y)\in S} p(y|x) = -\sum_{(x,y)\in S} \ln p(y|x) \tag{17}$$

where $p(y|x)$ is the probability that the network will generate sequence y given sequence x through all possible paths.

4 Experiments

We apply DTRN to the off-line handwriting recognition task using three popular public datasets: the IFN/ENIT Arabic handwriting dataset [26], the IAM English handwriting dataset [22] and the Rimes French handwriting dataset [14]. To verify the effectiveness of DTRN on other sequence modeling tasks, we also conduct speech recognition experiments on the TIMIT dataset [10].

Table 1 lists the information about the datasets used in the experiments. Note that the character/phoneme set of each dataset includes a blank symbol for the CTC decoding.

We use edit distance as the evaluation index. Character error rate (CER) is used to measure the performance of the model for the offline handwriting experiments, while

Table 1 Four datasets used in the experiments

Task	Dataset	Level	#Alphabet	#Train	#Val	#Test
Arabic handwriting recognition	IFN/ENIT	Word	121	26,459	–	6033
English handwriting recognition	IAM	Line	80	6161	966	2915
French handwriting recognition	Rimes	Line	99	10,171	1162	778
Speech recognition	TIMIT	Sentence	62	3696	–	192

phoneme error rate (PER) is used for the speech recognition experiments. To focus solely on the comparison of recognition ability, all the results are reported without language models or data augmentation.

We use the RmsProp algorithm [33] to optimize the models for all the experiments. As there are random factors in the training process, all the experiments in this section are executed three times. The major results are reported as the average, maximum and minimum error rates for each experiment.

We implement our method using the PyTorch deep learning framework [25] and use Tesla P100 GPUs for parallel computation. The source code will be available at https://github.com/RuijieJ/DTRN-pytorch.

4.1 Arabic handwriting recognition

We compare the performance of various model structures and configurations on the IFN/ENIT dataset.

4.1.1 Dataset

The original publicly available IFN/ENIT dataset is divided into five sets (*a*, *b*, *c*, *d* and *e*). One commonly used test scenario is *abcd-e* (i.e., the training set consists of set *a* through set *d*, and the test set is set *e*). Detailed information of the IFN/ENIT dataset is listed in Table 1.

In our experiments, all the input images are preprocessed using the center-normalizer method provided by the OCRopus system[1] [37] and resized to a unified height of 48 pixels while preserving the original aspect ratio.

4.1.2 Experimental settings

The baseline LSTM model for Arabic handwriting recognition has a CNN for feature extraction. The CNN contains five layers, and its configuration is listed in Table 2. Batch normalization (BN) [20] is used for each convolution layer. The activation function after BN is leaky ReLU.

For the baseline LSTM, STRN and the primary network in DTRN, the first and second RNN layers have 64 units, and other RNN layers have 128 units. For example, the numbers of units in a five-layer LSTM are 64, 64, 128, 128 and 128, respectively. Dropout is used for each layer with a fixed dropout rate of 0.5.

For each experiment, the model is trained for 300 epochs with the learning rate of 1×10^{-4}.

[1] https://github.com/tmbdev/ocropy.

Table 2 Configuration of the CNN feature extractor for Arabic handwriting recognition

Configuration	Values
No. of conv. filters	16–32–48–64–80
Max pooling	Y–Y–N–N–N
Dropout	0–0–0.2–0.2–0.2
Conv. kernel/stride	3×3 / 1×1
Max pooling kernel/stride	2×2 / 2×2

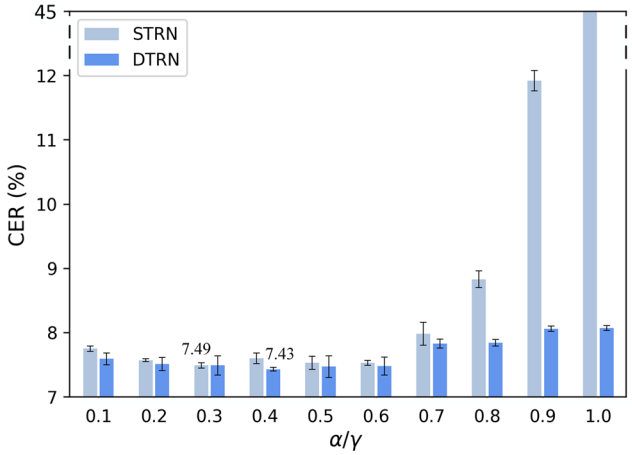


Fig. 3 Comparison of recognition performance for the STRN and DTRN models on the test set of the IFN/ENIT dataset

4.1.3 Comparisons of STRN and DTRN

For STRN, we set $\alpha_t = \alpha$ for all the time steps $1 \le t \le T$. For DTRN, the discount factor γ controls the maximum weights of the shortcut connections. In this experiment, we compare different values of α for STRN and different discount factors for DTRN. We fix the number of RNN layers in STRN and the primary network of DTRN to 5. For DTRN, we use the LSTM as the secondary network, which has one layer with 32 units.

Figure 3 shows the comparison results on the test set. The vertical line on the top of each bar indicates the standard

Table 3 Comparison of different configurations for the secondary LSTM network on the IFN/ENIT dataset

#Layers	#Units	#Params	Training CER (%)	Test CER (%)
1	8	1.87M	0.82	7.52
			[0.81–0.84]	[7.46–7.58]
	16	1.93M	0.81	**7.41**
			[0.80–0.83]	[7.36–7.45]
	32	2.06M	0.79	7.44
			[0.76–0.81]	[7.39–7.50]
	64	2.33M	0.81	**7.41**
			[0.79–0.82]	[7.34–7.47]
2	32–32	2.08M	0.80	7.49
			[0.78–0.84]	[7.43–7.53]
	32–64	2.12M	0.77	7.42
			[0.77–0.78]	[7.41–7.43]
	64–32	2.37M	0.80	7.57
			[0.79–0.81]	[7.47–7.64]

Bold values indicate the best results for the corresponding experimental configurations

deviation from three independent experiments. For STRN, large α values degrade the model's performance. DTRN is more stable to variations of the discount factor, and it achieves slightly higher accuracy than the best result of STRN. Based on the results in Fig. 3, we set $\alpha = 0.3$ for STRN and $\gamma = 0.4$ for DTRN for the subsequent experiments.

4.1.4 Different configurations of the secondary network

For a DTRN, the secondary network performs a simpler task than does the primary network; thus, we set both its numbers of layers and units to smaller values than those in the primary network.

We compare different configurations for the two types of secondary network. The number of RNN layers in the primary network is fixed at 5. For the secondary LSTM network, different numbers of RNN layers and units are compared, and the results are listed in Table 3. For the secondary self-attention network, we set the dimension d_k of the three learnable matrices in Eq. (15) to different values, and the results are shown in Table 4.

The comparison results indicate that an appropriate size for the secondary network may achieve a slightly better performance on the test set. Using an LSTM as the secondary network yields a slightly lower CER on the test set for Arabic handwriting recognition.

4.1.5 Comparison of different numbers of layers in the primary network

We also compare the performance of baseline LSTM and DTRN with different numbers of layers in the primary network. To explore the performance of models with a controlled number of parameters, we adopt a one-layer LSTM with 32 units as the secondary network of DTRN. In this way, the DTRN has slightly fewer parameters than does the baseline LSTM network with one more layer.

Table 4 Comparison of different configurations for the secondary self-attention network on the IFN/ENIT dataset

d_k	#Params	Training CER (%)	Test CER (%)
64	1.99M	0.83	7.50
		[0.82–0.86]	[7.44–7.62]
128	2.17M	0.83	7.47
		[0.82–0.85]	[7.38–7.57]
256	2.54M	0.82	7.44
		[0.78–0.86]	[7.38–7.49]

The comparison results are listed in Table 5. Compared with the baseline LSTM networks, DTRNs with similar (or even smaller) numbers of network parameters achieve better results, which indicates that the introduction of the temporal residual learning mechanism can bring improved performance.

The seven-layer DTRN with a training CER of 0.72% and a test CER of 6.91% achieves the best result, showing that DTRN has better nonlinear sequence modeling and generalization ability.

4.1.6 Comparisons with other work

The comparison of the seven-layer DTRN with some previously published works is shown in Table 6. We also add experiments using a seven-layer STRN and a seven-layer DTRN with a secondary self-attention network. DTRN with a secondary self-attention network is denoted by DTRN-ATTN in Table 6. The dimension d_k of the three learnable matrices in the self-attention network is 128. The DTRN method with

Table 5 Comparison of baseline LSTM networks and DTRNs with different numbers of layers on the IFN/ENIT Arabic handwriting dataset

RNN	#Layers	#Params	Training CER (%)	Test CER (%)
Baseline LSTM	4	1.41M	1.35	8.29
			[1.31–1.44]	[8.09–8.52]
	5	1.80M	1.04	7.84
			[1.03–1.05]	[7.79–7.89]
	6	2.20M	0.99	7.71
			[0.93–1.03]	[7.51–7.83]
	7	2.59M	0.92	7.52
			[0.88–0.98]	[7.40–7.59]
	8	2.99M	0.97	7.74
			[0.95–0.99]	[7.57–7.83]
DTRN	4	1.66M	1.05	7.74
			[1.01–1.09]	[7.64–7.87]
	5	2.06M	0.79	7.44
			[0.76–0.81]	[7.39–7.50]
	6	2.45M	0.75	7.39
			[0.74–0.76]	[7.35–7.45]
	7	2.85M	**0.72**	**6.91**
			[0.69–0.75]	[6.83–7.01]
	8	3.24M	0.72	7.53
			[0.68–0.76]	[7.36–7.73]

Bold values indicate the best results for the corresponding experimental configurations

Table 6 Comparisons of different methods on the IFN/ENIT Arabic handwriting dataset

Method	#Params	Test CER (%)
1DLSTM [37]	0.86M	14.32
MDLSTM [1]	–	15.05
JU-OCR2 [1]	–	13.42
Baseline LSTM	2.59M	7.52
STRN	2.59M	7.37
DTRN-ATTN	2.96M	7.38
DTRN	2.85M	**6.91**

Bold values indicate the best results for the corresponding experimental configurations

an LSTM as the secondary network achieves state-of-the-art CER on the test scenario "abcd-e" for the IFN/ENIT dataset.

4.2 English and French handwriting recognition

We further test the effectiveness of DTRN on both the IAM English handwriting dataset and the Rimes French handwriting dataset.

4.2.1 Dataset

The details of the IAM and Rimes datasets are listed in Table 1. We use the tools provided in [30] for sample preprocessing, including denoising and deskewing. All the images are resized to a unified height of 128 pixels while preserving the original aspect ratio.

4.2.2 Experimental settings

Following the work in [29] which is implemented in Torch [30], we reimplement the CNN-LSTM architecture in PyTorch as our baseline LSTM model. The feature extractor is a five-layer CNN. Compared with the CNN used in Arabic handwriting experiments, the only difference is that max pooling operation is adopted for the first three layers instead of the first two.

For each LSTM layer, the number of units is 256 and the dropout rate is set to 0.5. STRN and the primary network in DTRN have the same network configuration as the corresponding part of the baseline LSTM. We adopt a one-layer LSTM with 128 units as the secondary network of DTRN. In this way, the number of parameters in a DTRN is slightly less than that in a baseline LSTM with one more layer.

To obtain these results, the batch size is 16 and the learning rate is 3×10^{-4}. We train each model until the validation CER stops decreasing, which is usually about 300 epochs.

4.2.3 Experimental results

A comparison of the baseline LSTM and DTRN models on the IAM and Rimes datasets is shown in Table 7. The results show that DTRN with a similar (or even smaller) number of

Table 7 Comparison of baseline LSTM networks and DTRNs on the IAM English handwriting dataset and Rimes French handwriting dataset

Configuration			IAM		Rimes	
RNN	#Layers	#Params	Validation CER (%)	Test CER (%)	Validation CER (%)	Test CER (%)
Baseline LSTM	3	6.44M	5.31	8.78	3.44	3.36
			[5.28–5.33]	[8.69–8.85]	[3.37–3.51]	[3.33–3.38]
	4	8.01M	4.95	8.07	3.05	2.99
			[4.87–5.04]	[8.02–8.12]	[2.98–3.09]	[2.95–3.06]
	5	9.59M	4.58	7.62	2.75	2.63
			[4.46–4.65]	[7.54–7.69]	[2.67–2.79]	[2.49–2.71]
	6	11.17M	4.50	7.35	2.65	2.55
			[4.45–4.55]	[7.22–7.52]	[2.49–2.79]	[2.53–2.57]
DTRN	3	7.88M	4.73	7.75	2.79	2.77
			[4.66–4.84]	[7.68–7.86]	[2.72–2.84]	[2.74–2.85]
	4	9.46M	4.42	7.29	2.50	2.37
			[4.38–4.49]	[7.24–7.36]	[2.36–2.65]	[2.35–2.41]
	5	11.04M	**4.16**	6.95	**2.36**	**2.28**
			[4.09–4.25]	[6.87–7.10]	[2.29–2.47]	[2.20–2.35]
	6	12.61M	4.24	**6.91**	2.43	2.30
			[4.22–4.25]	[6.85–6.98]	[2.39–2.48]	[2.26–2.37]

Bold values indicate the best results for the corresponding experimental configurations

Table 8 English handwriting recognition results on the IAM dataset

Method	#Params	Validation CER (%)	Test CER (%)
MDLSTM [28]	–	7.40	10.80
LSTM [29]	9.59M	4.70	7.90
Baseline LSTM	11.17M	4.50	7.35
STRN	9.59M	4.31	7.12
DTRN-ATTN	10.08M	4.37	7.00
DTRN	12.61M	**4.24**	**6.91**

Bold values indicate the best results for the corresponding experimental configurations

Table 9 French handwriting recognition results on the Rimes dataset

Method	#Params	Validation CER (%)	Test CER (%)
MDLSTM [28]	–	5.90	6.80
LSTM [29]	9.59M	2.60	2.50
Baseline LSTM	11.17M	2.65	2.55
STRN	9.59M	2.48	2.39
DTRN-ATTN	10.08M	2.43	2.31
DTRN	11.04M	**2.36**	**2.28**

Bold values indicate the best results for the corresponding experimental configurations

network parameters has outperformed the classical LSTM. On the IAM dataset, the best test CER of 6.91% is achieved by the six-layer DTRN. On the Rimes dataset, the best test CER of 2.28% is achieved by the five-layer DTRN.

The results of STRN, DTRN and some previously published results are listed in Tables 8 and 9. All the methods are compared without language models or data augmentation. We also add experiments using a five-layer STRN and a five-layer DTRN with self-attention-based dynamic weights. The dimension d_k of the three learnable matrices in the self-attention network is set to 128. DTRN with a secondary LSTM network achieves state-of-the-art test CERs.

4.3 Speech recognition experiments

Our proposed method is flexible and applicable to other sequence modeling tasks. To verify the effectiveness of the proposed methods, we also conduct speech recognition experiments on the TIMIT dataset.

Table 10 Speech recognition results on the TIMIT dataset

Method	#Params	Training PER (%)	Test PER (%)
LSTM [13]	4.3M	–	17.70
Baseline LSTM	3.62M	28.05	15.22
		[27.68–28.41]	[14.70–15.71]
STRN	3.62M	24.42	14.85
		[23.98–24.79]	[14.31–15.25]
DTRN–ATTN	3.80M	23.97	**14.59**
		[23.78–24.09]	[14.50–14.67]
DTRN	3.67M	**23.72**	14.74
		[23.61–23.88]	[14.52–15.16]

Bold values indicate the best results for the corresponding experimental configurations

4.3.1 Dataset

The details of the TIMIT dataset are listed in Table 1. The input feature is a conventional MFCC with a dimension of 39. All 61 phoneme labels are used during training and decoding and mapped to 39 classes following the method in [21] to obtain test PERs.

4.3.2 Experimental settings

The baseline model is a three-layer LSTM network slightly modified from the model used in [13]. Each LSTM layer has 256 units (while each LSTM layer has 250 units in [13]). STRN and the primary network in DTRN have the same configuration as the baseline LSTM network.

When using an LSTM as the secondary network of DTRN, a one-layer LSTM with 64 units is adopted. When generating dynamic weights using a self-attention network, the dimension d_k of the three learnable matrices in the self-attention network is set to 64.

We set the dropout rates for all the layers to 0.5. For each experiment, the model is trained for 150 epochs with the learning rate of 1×10^{-4}.

4.3.3 Experimental results

As shown in Table 10, STRN and DTRN have also outperformed the classical LSTM network. DTRN with a self-attention-based dynamic weights generator has achieved the best performance on the test set with the CER of 14.59%, which is different from the results in handwriting recognition experiments. It seems that the self-attention method has better sequence modeling ability for the one-dimensional speech signals. This result indicates that designing appropriate secondary network structures for different tasks can be beneficial to the performance of DTRN.

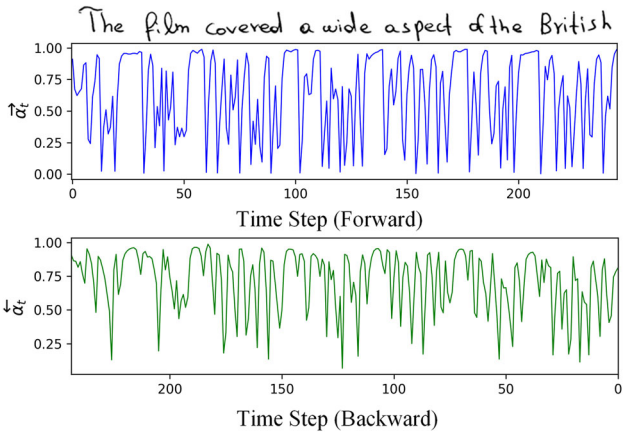


Fig. 4 Examples of α_t in DTRN for both forward and backward directions

4.4 Visualization

4.4.1 The dynamic weights for temporal shortcuts in the modified LSTM units

Figure 4 shows the values of α_t in a bidirectional DTRN with respect to a sample from the IAM dataset. Larger values of the dynamic coefficient α_t indicate that more prior outputs are passed directly to the outputs of the current time step. α_t usually obtains its maximum value in uniformly structured areas (e.g., empty space) and obtains its minimum value while encountering a new character, which complies with the intuitive observation for most sequential data.

4.4.2 The statistics of the modified LSTM unit outputs

In ResNet, the learned residual functions generally have small responses [16], and we find a similar phenomenon in DTRN. Figure 5 shows the standard deviation of the learned nonlinear responses [$\mathcal{F}_t$ in Eq. (1)] at each time step of both a five-layer baseline LSTM and five-layer DTRN with respect to the same input image. At most time steps, the nonlin-

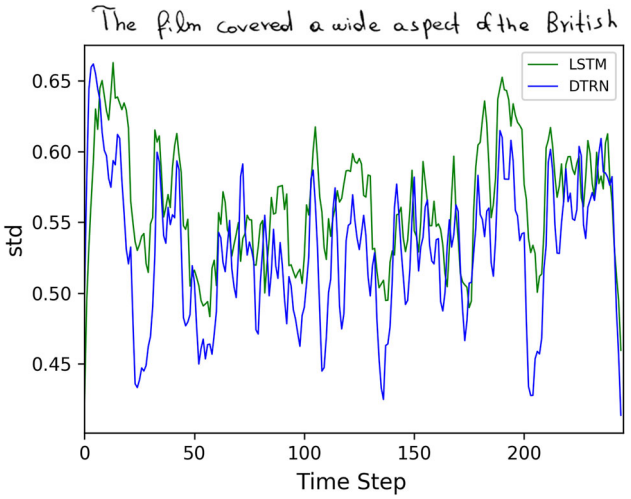


Fig. 5 Standard deviations (std) of the nonlinear responses of the baseline LSTM and DTRN for the same input image

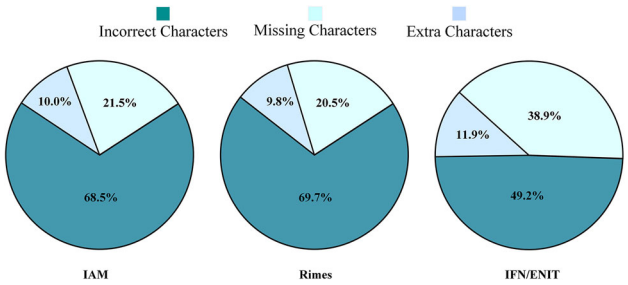


Fig. 6 Proportions of the three error types on the test set of three handwriting datasets using DTRNs

ear responses that DTRN outputs are smaller than those of the baseline LSTM network, which indicates that DTRN can learn more concise representations. This phenomenon might explain why DTRNs with similar or fewer parameters achieve a better performance than do baseline LSTM networks.

4.4.3 Error analysis

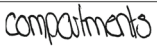
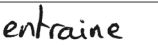
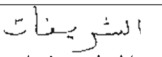
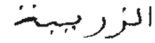
We analyze the test recognition errors of five-layer DTRNs on the three handwriting datasets. When using the edit distance to compute the CER for off-line handwriting recognition, three types of errors may occur: incorrect characters, missing characters and extra characters. Table 11 shows some examples for each case.

The proportion of each error type is shown in Fig. 6. Incorrect characters are the dominant factor, and most of them are caused by similar characters, such as (a, o), (i, Î) and (ت, ث). The missing character errors generally occur when the neighboring characters are touching or cursive, both of which are common in Arabic handwriting samples. The extra character errors occur less frequently and usually appear along with incorrect character errors.

5 Conclusion

In this paper, we propose the DTRN architecture, which extends the LSTM network with a residual learning mechanism along the temporal dimension. The temporal shortcut connections allow the model to explore additional contextual information that is useful for the subsequent predictions. The experimental results on off-line handwriting and speech recognition tasks show that the proposed method has outperformed previously reported methods. In the future, we will explore additional possible structures for both the primary network and the secondary network. We will also investigate attention-based methods as an alternative to CTC decoding method. DTRN can be further applied to other sequence modeling tasks including large character set Chinese handwriting recognition.

Table 11 Examples of the three error types on three handwriting datasets

Database	Data	Incorrect Characters	Missing Characters	Extra Characters
IAM	Input Image			
	Ground Truth	*compartments*	*understand*	*careful*
	Prediction	*compostments*	*undertand*	*careferl*
Rimes	Input Image			
	Ground Truth	*entraine*	*rembourser*	*ouverture*
	Prediction	*entraîne*	*rembourer*	*ouvertiure*
IFN/ENIT	Input Image			
	Ground Truth	تالة	الشريفات	الزريبة
	Prediction	شالة	الشريات	الزريبة

Acknowledgements The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper. This research is supported by National Natural Science Foundation of China under Grant U1636124, 61471214 and 61573028. This research is also supported by China Scholarship Council.

References

1. Abandah, G.A., Jamour, F.T., Qaralleh, E.A.: Recognizing handwritten Arabic words using grapheme segmentation and recurrent neural networks. Int. J. Doc. Anal. Recognit. **17**(3), 275–291 (2014)
2. Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., et al.: Deep speech 2: end-to-end speech recognition in English and Mandarin. In: Proceedings of the International Conference on Machine Learning, pp. 173–182 (2016)
3. Bluche, T., Louradour, J., Messina, R.: Scan, attend and read: end-to-end handwritten paragraph recognition with MDLSTM attention. In: Proceedings of the International Conference on Document Analysis and Recognition, vol. 1, pp. 1050–1055 (2017)
4. Chang, S., Zhang, Y., Han, W., Yu, M., Guo, X., Tan, W., Cui, X., Witbrock, M., Hasegawa-Johnson, M.A., Huang, T.S.: Dilated recurrent neural networks. In: Advances in Neural Information Processing Systems, pp. 77–87 (2017)
5. Chen, K., Huo, Q.: Training deep bidirectional LSTM acoustic model for LVCSR by a context-sensitive-chunk BPTT approach. IEEE Trans. Audio Speech Lang. Process. **24**(7), 1185–1193 (2016)
6. Chorowski, J., Jaitly, N.: Towards better decoding and language model integration in sequence to sequence models (2016). arXiv preprint arXiv:1612.02695
7. Davis, S., Mermelstein, P.: Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE Trans. Acoust. Speech Signal Process. **28**(4), 357–366 (1980)
8. Ding, H., Chen, K., Yuan, Y., Cai, M., Sun, L., Liang, S., Huo, Q.: A compact CNN-DBLSTM based character model for offline handwriting recognition with Tucker decomposition. In: Proceedings of the International Conference on Document Analysis and Recognition, pp. 507–512 (2017)
9. Elman, J.L.: Finding structure in time. Cognit. Sci. **14**(2), 179–211 (1990)
10. Garofolo, J.S., Lamel, L.F., Fisher, W.M., Fiscus, J.G., Pallett, D.S., Dahlgren, N.L.: DARPA TIMIT acoustic-phonetic continuous speech corpus CD-ROM. NIST speech disc 1-1.1. NASA STI/Recon technical report 93 (1993)
11. Graves, A.: Connectionist temporal classification. In: Supervised Sequence Labelling with Recurrent Neural Networks, pp. 5–13. Springer (2012)
12. Graves, A.: Offline Arabic handwriting recognition with multidimensional recurrent neural networks. In: Guide to OCR for Arabic Scripts, pp. 297–313. Springer (2012)
13. Graves, A., Mohamed, A., Hinton, G.: Speech recognition with deep recurrent neural networks. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing, pp. 6645–6649 (2013)
14. Grosicki, E., Carre, M., Brodin, J.M., Geoffrois, E.: RIMES evaluation campaign for handwritten mail processing. In: Proceedings of the International Conference on Frontiers in Handwriting Recognition, pp. 1–6 (2008)
15. Gui, T., Zhang, Q., Zhao, L., Lin, Y., Peng, M., Gong, J., Huang, X.: Long short-term memory with dynamic skip connections (2018). arXiv preprint arXiv:1811.03873
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
17. Hochreiter, S.: The vanishing gradient problem during learning recurrent neural nets and problem solutions. Int. J. Uncertain. Fuzziness Knowl. Based Syst. **6**(2), 107–116 (1998)
18. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
19. Hu, W., Cai, M., Chen, K., Ding, H., Sun, L., Liang, S., Mo, X., Huo, Q.: Sequence discriminative training for offline handwriting recognition by an interpolated CTC and lattice-free MMI objective function. In: Proceedings of the International Conference on Document Analysis and Recognition, pp. 61–66 (2017)
20. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift (2015). arXiv preprint arXiv:1502.03167
21. Lee, K.F., Hon, H.W.: Speaker-independent phone recognition using hidden Markov models. IEEE Trans. Acoust. Speech Signal Process. **37**(11), 1641–1648 (1989)
22. Marti, U., Bunke, H.: The IAM-database: an English sentence database for offline handwriting recognition. Int. J. Doc. Anal. Recognit. **5**(1), 39–46 (2002)
23. Menasri, F., Louradour, J., Bianne-Bernard, A.L., Kermorvant, C.: The A2iA French handwriting recognition system at the Rimes-ICDAR2011 competition. In: Proceedings of Document Recognition and Retrieval, p. 8297Y (2012)
24. Pascanu, R., Gulcehre, C., Cho, K., Bengio, Y.: How to construct deep recurrent neural networks (2013). arXiv preprint arXiv:1312.6026
25. Paszke, A., Gross, S., Soumith, C., et. al.: Automatic differentiation in PyTorch. In: NIPS 2017 Autodiff Workshop (2017)
26. Pechwitz, M., Maddouri, S.S., Märgner, V., Ellouze, N., Amiri, H., et al.: IFN/ENIT-database of handwritten Arabic words. In: Proceedings of the Colloque International Francophone sur l'Ecrit et le Document, vol. 2, pp. 127–136 (2002)
27. Pei, W., Baltrusaitis, T., Tax, D.M., Morency, L.P.: Temporal attention-gated model for robust sequence classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 820–829 (2017)
28. Pham, V., Bluche, T., Kermorvant, C., Louradour, J.: Dropout improves recurrent neural networks for handwriting recognition. In: Proceedings of the International Conference on Frontiers in Handwriting Recognition, pp. 285–290 (2014)
29. Puigcerver, J.: Are multidimensional recurrent layers really necessary for handwritten text recognition? In: Proceedings of the International Conference on Document Analysis and Recognition, pp. 67–72 (2017)
30. Puigcerver, J., Martin-Albo, D., Villegas, M.: Laia: a deep learning toolkit for HTR. https://github.com/jpuigcerver/Laia (2016). GitHub repository
31. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Trans. Pattern Anal. Mach. Intell. **39**(11), 2298–2304 (2017)
32. Srivastava, R.K., Greff, K., Schmidhuber, J.: Highway networks (2015). arXiv preprint arXiv:1505.00387
33. Tieleman, T., Hinton, G.: Lecture 6.5-RmsProp: divide the gradient by a running average of its recent magnitude. In: COURSERA: Neural Networks for Machine Learning (2012)
34. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
35. Wu, Y., Schuster, M., Chen, Z., et al.: Google's neural machine translation system: bridging the gap between human and machine translation (2016). arXiv preprint arXiv:1609.08144

36. Wu, Y.C., Yin, F., Chen, Z., Liu, C.L.: Handwritten Chinese text recognition using separable multi-dimensional recurrent neural network. In: Proceedings of the International Conference on Document Analysis and Recognition, pp. 79–84 (2017)
37. Yousefi, M.R., Soheili, M.R., Breuel, T.M., Stricker, D.: A comparison of 1D and 2D LSTM architectures for the recognition of handwritten Arabic. In: Proceedings of Document Recognition and Retrieval, p. 94020H (2015)
38. Yue, B., Fu, J., Liang, J.: Residual recurrent neural networks for learning sequential representations. Information **9**(3), 56 (2018)
39. Zhang, Y., Chen, G., Yu, D., Yaco, K., Khudanpur, S., Glass, J.: Highway long short-term memory RNNs for distant speech recognition. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing, pp. 5755–5759 (2016)