J Grid Computing (2020) 18:169–176
https://doi.org/10.1007/s10723-018-9444-4

Residual Recurrent Highway Networks for Learning Deep Sequence Prediction Models

Tehseen Zia · **Saad Razzaq**

Received: 20 February 2018 / Accepted: 28 May 2018 / Published online: 6 June 2018

Abstract A contemporary approach for acquiring the computational gains of depth in recurrent neural networks (RNNs) is to hierarchically stack multiple recurrent layers. However, such performance gains come with the cost of challenging optimization of hierarchal RNNs (HRNNs) which are deep both hierarchically and temporally. The researchers have exclusively highlighted the significance of using shortcuts for learning deep hierarchical representations and deep temporal dependencies. However, no significant efforts are made to unify these finding into a single framework for learning deep HRNNs. We propose residual recurrent highway network (R2HN) that contains highways within temporal structure of the network for unimpeded information propagation, thus alleviating gradient vanishing problem. The hierarchical structure learning is posed as residual learning framework to prevent performance degradation problem. The proposed R2HN contain significantly reduced data-dependent parameters as compared to related methods. The experiments on language modeling (LM) tasks have demonstrated that the proposed architecture leads to design effective models. On LM experiments with Penn TreeBank, the model achieved 60.3 perplexity and outperformed baseline and related models that we tested.

Keywords Deep learning · Recurrent neural networks · Sequence modeling · Highway networks · Residual learning

T. Zia (✉)
Comsats Institute of Information Technology, Islamabad, Pakistan
e-mail: Tehseen.zia@comsats.edu.pk

S. Razzaq
University of Sargodha, Sargodha, Pakistan
e-mail: saadrazzaq@uos.edu.pk

1 Introduction

Recurrent neural networks (RNNs) are specifically designed for processing data in sequential steps. This allows the network to seize sequential associations between data instances. The RNNs also possess the flexibility to process variable size inputs and produce outputs of varying lengths. These abilities have enabled RNNs to perform impressively in several machine learning problems such as language modeling [1–3], online handwritten recognition [4], speech recognition [5, 6] and learning word embedding [7, 8].

An established incapability of RNNs is its restricted tendency to capture longer dependencies between instances due to vanishing gradient problem [9]. A contemporary approach to deal with this inadequacy is to use long short-term memory (LSTM) [10] – a specifically designed architecture, or its variants such as gated recurrent unit (GRU) [11]. An important essence of these architectures is to deploy shortcut (linear) channel between layers for unhindered gradient transmission. Though the architectures

are quite effective, they employ an indirect mechanism for deploying shortcut channels and involve relatively large data-dependent parameters. The contemporary approaches in shortcut based architectures emphasize on direct shortcuts channels [12, 13]. Most promising examples of these architectures are residual network and highway network (HN) [12]. These deep feed-forward networks have shown significant improvement in visual recognition tasks [13], speech recognition [14, 15] and language modeling [16, 17].

However, the concept of depth is not obvious for RNNs as for feed-forward networks [18]. Though, RNNs can be regarded as deep when unfolded in time because of multiple non-linear layers between input and output at time. Still, RNNs can be viewed as swallow regarding hierarchal structure because of the computation at a timestamp which is simply a linear projection tailed by non-linearity. Nowadays, to achieve computational benefits of depth, recurrent layers are often stacked hierarchically [19–21]. However, such performance advances come with the cost of challenging optimization of deep hierarchal RNNs (HRNNs) [18, 22].

Because HRNNs are deep spatially (i.e. hierarchically) and temporally, the appropriate architecture must allow unimpeded information transient across both spatial and temporal depth. Recently, the researchers have extended LSTMs with shortcut channels along hierarchical structure to ease optimization of deeper HRNNs [14, 15]. However, the resultant architectures are computationally intensive due to abundance of data-dependent parameters. Other recently proposed architectures are restricted to using shortcut channels to optimize temporal structure or hierarchical structure, but not both [22, 23].

Stimulating from the excellence of residual learning in hierarchal feed-forward networks [13] and highways networks as simpler alternative of LSTMs [22], we propose residual recurrent highway networks (R2HN). The R2HN employ structure of highway networks to enable optimization of temporal (or transition) function. This highway based temporal function contains one gating function as compared with three gating function of LSTMs, hence saving one-third of parameters. To facilitate optimization of hierarchical structure, the framework of residual learning is employed. Both highway and residual frameworks are unified into a single R2HN framework. Key contributions of this work are: 1) the development of a novel lightweight deep RNN architecture for learning sequence models 2) empirical and complexity analysis of various RNNs architectures on a benchmark language modeling dataset. Organization of the paper is as follows: related work and background is given in Sections 2 and 3 respectively. The proposed HRHN architecture is presented in Section 4. Experimental setup is described in Section 5 and results and analysis is presented in Section 6. Finally the conclusion of the paper is given in Section 7.

2 Related Work

This section justifies building deep RNN architectures and highlights shortcomings of existing deep RNN architectures.

2.1 Advantages of Depth for Recurrent Neural Networks

Deep learning field is developed on a hypothesis that hierarchically deep structures bring forth computational advantages [24]. While the hypothesis is well-justified for feed-forward neural networks [13, 24]; it may not be properly vindicated for its efficacy in RNNs [18]. This was perhaps due to the reason that, because RNNs are already deep networks (i.e. they can be stated as composition of several non-linear layers once unfolded in time), it has been argued that RNN are already enjoying the computational gains of depth [25]. However, recently several studies have been conducted to investigate the legitimacy of the hypothesis for hierarchical RNN. A number of theoretical and empirical evidences are reported in the favor of hierarchically deep RNN structures. For example, using RNNs' closely related recurrent arithmetic circuits (RACs) [26], it is demonstrated that depth yield significant advantage in the capacity of recurrent networks to capture long-term dependencies. Additionally, a number of empirical evidences are recently reported to support this hypothesis for HRNNs [6, 14, 15, 18, 27–29]. These results are establishing the computational advantages of deep RNN architectures.

2.2 Deep Recurrent Neural Network Architectures

The hierarchical RNNs (HRNNs) are extensively used now in range of applications containing image and

video interpretation [21, 27], language modeling [25, 28–30], speech recognition [31, 32], machine translation [33] and text classification [34, 35], etc. The idea of stacking multiple RNNs was formerly proposed by Schmidhuber et al. [19]. It was shown that HRNNs lead to improve computational and learning efficacy of the network as it decrease updating frequency of layers in order of depth. The rewards of organizing temporal dependencies hierarchically are further established in [20]. In a recent study HRNN is explicitly modeled with less frequent updating of higher layers than lower layers [32] in order to explicitly analyze the outcomes of [35]. Chung et al. has recently enabled HRNNs to learn temporal connectivity structure [28]. This development permits the network to learn hierarchy structure dynamically. HRNNs are further enabled to discover latent hierarchical structure in temporal data without explicit boundary information [25]. In [18] it is observed that HRNNs are still shallow networks with respect to input-to-hidden transition, hidden-to-hidden transition and hidden-to-output transitions. Deep variants of the functions are also proposed in the study.

Although the rewards of depth is generally recognized for HRNNs, however it is shown that naive stacking often led to performance degradation in HRNNs [14, 15, 22, 36], mainly due to vanishing gradient problem. As HRNNs are deep hierarchically and temporally, the optimization of their architectures remains a challenging task. While the idea of using shortcut connection between layers to ease the optimization of deep architectures is known for quite a time, the approach is well-adopted in contemporary architectures. One of the most successful architecture in this category is long short-term memory (LSTM) [10]. In LSTM, the shortcut connection is provided through an indirect mechanism (i.e. using memory cells). Though, LSTM has shown significant improvements over RNNs, e.g. [33, 37, 38]. However, recently direct shortcut connection based approaches such as recurrent highway networks [22] and recurrent residual networks [23] have either outperformed LSTMs or shown comparative performance with significantly reduced parameters. The essence of the architectures is to reduce data-dependent parameters and computations while retaining core component of LSTM (i.e. shortcut connection between layers for unhampered gradient propagation). Zilly et al. [22] proposed recurrent highway network (RHN) where a gated highway is provided across layers in the spatial domain. Similar architectures are proposed in [23, 39, 40] where the ungated highway is deployed. Though the methods enabled learning of deep hierarchical structure, nonlinear output-input interface between hierarchical layers remains a bottleneck. Because, the nonlinear function is prune to vanishing gradient problem [9], it restricts the ability of network to learn temporal abstraction and longer dependencies. Recently, the problem is rectified by using LSTM with an additional shortcut connection between hierarchical layers [14, 15]. However, as mentioned, direct shortcut connection based approaches have outdone LSTMs in several recent studies. Further, LSTM involves relatively large data dependent parameters and have complex structure which leads to poor understanding of its sources of success and failures [41]. Therefore, by replacing LSTM with direct shortcut connections into recurrent structure of RNNs, computational benefits can be achieved.

3 Background

In this section, we give a brief review of residual and highway networks and three existing highway architectures.

3.1 Recurrent Neural Networks

In conventional RNNs, the activation function is usually composed of element-wise nonlinearity tailed by affine transformation as:

$$h_t = \sigma(W_x \chi_t + W_h h_{t-1}) \tag{1}$$

Where $W_*, * = \{x, h\}$ denotes weight matrices and σ symbolizes sigmoid function. The output of RNN is defined as an activation of hidden state:

$$o_t = \sigma(W_y h_t)$$

The parameters of RNNs can be optimized by minimizing a cost function $J(W)$ with respect to a dataset of N training sequences $D = \{(x_1^{(n)}, y_1^{(n)}), \ldots, (x_{T_n}^{(n)}, y_{T_n}^{(n)})\}_{n=1}^{N}$ as:

$$J(W) = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T} d(y_t, o_t)$$

Where $d(y_t, o_t)$ is pre-specified measure of divergence between y_t and o_t such as entropy or Euclidean

distance. The computation is performed by using stochastic gradient descent where partial derivate of loss function is calculated using backpropgation through time [42].

3.2 Deep Recurrent Neural Networks

Deep RNNs are formed by stacking multiple layers of RNNs. Precisely; the lower RNN layer l activations h_t^l are fed to the higher layer $l+1$ as input x_t^{l+1}. Although each RNN layer is deep when unfolded in time (i.e. a feedforward network with shared weights between layers), deep RNNs still achieve significant performance gains over single-layer RNNs. It is recently reported that deep RNNs better utilize parameters through multiple layers by distributing them over the space [8]. It is important to note that in the conventional deep RNNs the interface between different RNN layers is output-input connection which vanish the gradients and therefore not effective.

3.3 Residual Network (RN)

RN resolves the notorious gradient vanishing problem associated with conventional output-input interface between layers by providing shortcut paths as identity mapping. The shortcut paths assist gradients to back-propagate through many layers without vanishing. Instead of direct mapping from input x to output y by using a function $F(x; W)$ with W parameters, the RN learns residual mapping from x to $y - x$ with the same function. The original mapping recasts as:

$$y = F(x;\ W) + x$$

The RN eases optimization of deep neural networks with hundreds of layers and shown record-setting performances on variety of computer vision tasks [13].

3.4 Highway Network (HN)

A HN is another way for incorporating shortcut channels into deep neural networks [12]. Unlike to RN, the information propagation over the channels is regulated with an adaptive gating function known as transfer gate. The formulation of HN is defined as:

$$y = H(x;\ W_h) \cdot T(x;\ W_T) + x \cdot (1 - T(x;\ W_T))$$

Where $H(x; W_h)$ is output of layer(s) and $T(x; W_T)$ is transfer gate which is defined as:

$$T(x;\ W_T) = \sigma(W_T x + b_T) \tag{2}$$

3.5 Recurrent Highway Network (RHN)

The RHN is an adaptation of highway networks for RNNs [22]. The RHN is designed as a simpler variant of LSTMs while retaining its vital mechanism: to use multiplicative gating function for regulating the information flow over self-connected additive cells. RHN outperformed LSTM in experimentations on language modeling and machine translation with significantly reduced parameters. The computation of the RHN at any timestamp t on a layer l is defined as:

$$S_t^l = H_t^l \cdot T_t^l + S_{t-1}^l \cdot (1 - T_t^l) \tag{3}$$

Where

$$H_t^l(x_t;\ W_h) = tanh(W_h x_t + R_h S_{t-1}^l)$$
$$T_t^l = \sigma(W_T x_t + R_T S_{t-1}^l)$$

4 Residual Recurrent Highway Networks (R2HN)

The architecture is designed with an objective to integrate recent developments in learning deep hierarchical and temporal representations for learning deep recurrent neural networks. The proposed residual recurrent highway network (R2HN) is schematically illustrated in Fig. 1. There are three key components of the architecture: recurrent neural network block, highway block and residual block. The RNN and HN blocks are jointly referred as RHN and used as an alternative of LSTM networks. The RHN is preferred as transition function because it outperformed LSTM in typical discrete sequence modeling tasks while using significantly lower number of parameters. The residual block is elected to ease the learning of deep hierarchical representations because of its state-of art performances in learning deeper hierarchical representations [12]. The formulation of R2HN can be defined as:

$$S_t^l = x_t^l \cdot T_t^l + S_{t-1}^l \cdot (1 - T_t^l) + S_t^{l-1} \tag{4}$$

where $x_t^l = S_t^{l-1}$ and S_t^l is input and output of the network at timestamp t, S_{t-1}^l is previous hidden state of the network and T_t^l is transfer gate defined in (2).

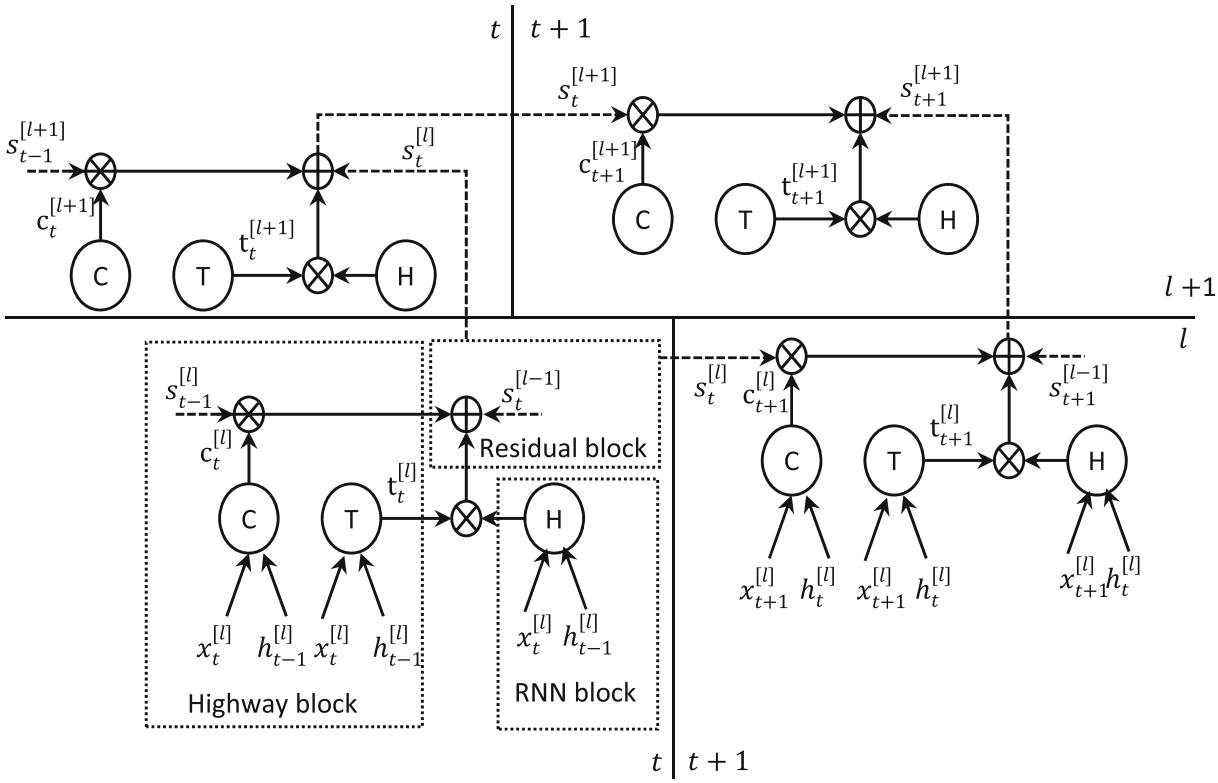

Fig. 1 A schematic illustration of R2HN architecture

It can be shown through a simple complexity analysis that R2HN use significantly less number of parameters as compared with recent proposed deep RNNs architectures: highway LSTM [14] and residual LSTM [15]. Consider m and n as input dimension and number of hidden states respectively. The weight matrices $W_{m\times n}$ and $R_{n\times n}$ are used for transforming input and hidden states (i.e. at time $t-1$). As LSTM uses three gating functions (i.e. input, forget and output gate) and a state updating function [10], the number of parameters involve in the computation of a single layer at a time step is $4(mn + n^2)$. As residual connection carries no additional parameter, the number of parameters for residual LSTM remains the same. In, highway LSTM, an additional gate (i.e. carry gate) is employed to regulate the information flow from lower hierarchical layer. Consequently, the parameters involve in the computation are $5(mn + n^2)$. Comparatively in RHN [22], one gating functions is involved (i.e. transfer gate) along with a state computing function. Hence, the number of parameters required for the computation is $2(mn + n^2)$. As no additional parameters are used in R2HN, the number of parameters remains $2(mn + n^2)$. Consider input dimensionality $m = 100$, number of hidden states $n = 10$ and number of stacked layer $= 5$, R2HN offers 25% parameters reduction over residual LSTM and 40% over highway LSTM at each time step.

5 Experimental Setup

We considered discrete sequence modeling task for evaluating worthiness of R2HN model. The task is equivalent to modeling a multinomial distribution. In RNNs, the task is commonly achieved by enabling the network to predict the probability of next symbol x_{t+1}, given the hidden state h_t where h_t is a function of all the former symbols $x_1, \ldots, x_{t-1}$ and present symbol x_t as:

$$p(x_{t+1}|x_1, x_2, \ldots, x_t) = g(h_t) \tag{5}$$

By following the common practice, we parameterized the distribution with softmax function at output layer. The

parameters for the softmax function are optimized in order to maximize log likelihood of training dataset as:

$$\max_{\theta} \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \log p(x_t^n | x_1^n, \ \ldots \ , x_{t-1}^n) \tag{6}$$

Where θ indicates parameters. Performance of the model is gauged on the basis of perplexity which measures capability of a probability distribution function to predict a sample. Mathematically, the perplexity can be defined as:

$$2^{H(p)} = 2^{-\sum p(x_{t+1}|x_t) \log_2 p(x_{t+1}|x_t)} \tag{7}$$

H(p) symbolizes entropy of the distribution. Experiments are conducted on word-level language modeling (LM) tasks on Penn TreeBank [43] dataset. The Penn TreeBank dataset consists of news stories from Wall Street Journal. We used a publically available version of this dataset as used in [7] and available on web.[1] The dataset consists of 500K training words and 50K test words.

The models are trained on dual-socket 8 cores Intel(R) Xeon(R) machines. Tensorflow CPU toolkit is employed to implement these models [44]. The standard stochastic gradient descent algorithm is used for the optimization of models where the gradients are computed using back-propagation through time (BPTT) [42]. For batch processing, the dataset is distributed into sequences of length 1K bytes through splitting the single one-hot matrix of whole dataset row-wise. The internal state of network (i.e. h_t) is reset after 10 sequences, thus allowing the models to hold information of past 10K characters. The objective of this setting is to analyze the ability of the model to capture and use long term dependencies. However, for computational elegance, the gradient signal is restricted to back-propagated only to the start of each 1K byte sequence (i.e. the network is unrolled 1K times). Therefore, the computation of gradient is approximate. This approach is a commonly used in sequence modeling tasks and referred as truncated BPTT. The weight initialization is performed by sampling uniformly from the range (-0.02, 0.02). The learning rates are decayed exponentially during the training process. Each network is trained for 500 epochs; however, we set early stopping criteria on step tolerance where the tolerance value is kept 1e-9. In order to perform a fair and unbiased comparison with baselines and related networks; all the networks are trained and evaluated on same dataset. Lastly, in order to reduce computations, a single transfer gate is used T_t^l while the other gate is defined in terms of the transfer gate as: $(1 - T_t^l)$. This approach is used in RHN to decrease the number of parameters [22].

[1] http://www.fit.vutbr.cz/~imikolov/rnnlm/simple-examples.tgz

6 Results and Analysis

The R2HN is mainly designed to ease the optimization of deeper RNNs. Therefore, the main investigation that we want is to test whether the proposed network can learn superior models than related networks with gradient based algorithm. It is observed that published results in other studies use models with widely diverse sizes, regularization methods and data sizes. Consequently, an unbiased comparison of these models cannot be performed on the basis of these results [22]. Relying on the recommendations of [17, 18, 25] for a fair comparison, the baseline and related works are implemented with similar settings.

The R2HN is compared with RHN, LSTM and two recently proposed deep variants of LSTM: highway LSTM (HighwayLSTM) and residual LSTM (ResLSTM). Two types of models are learned with each network: 3 layer model (M3) and 5 layer model (M5). The input and output layers in each network is respectively embedding and softmax layer. The size of hidden units in 3 and 5 layers models are 1000 and 500 respectively. Table 1 shows results on test dataset. In both experimentations, R2HN achieved best perplexity

Table 1 Test set perplexity of word-level language models on the Penn Treebank dataset

Network	Depth	Perplexity	Elapse time
RHN	3	168.0	22:11:30
	5	181.2	14:42:37
LSTM	3	156.7	–
	5	172.9	–
HighwayLSTM	3	90.0	22:26:52
	5	73.6	18:28:10
ResLSTM	3	148.4	20:39:04
	5	61.5	09:54:09
R2HN	3	109.9	15:07:43
	5	**60.3**	**05:29:20**

The bold emphasis shows results of proposed method

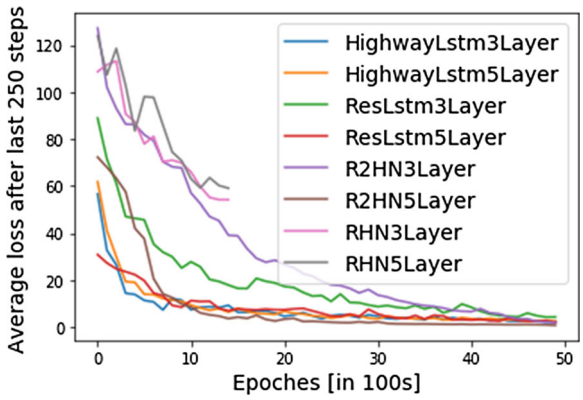

Fig. 2 Learning curves of the models. The final values of RHN models are shown for better illustration

than other networks. Although, ResLSTM (M5) revealed a comparative performance, R2HN has an advantage of fast convergence (i.e. deep R2HN is converged in approximately half of the time than deep ResLSTM). Consequently, these results validate that even for small datasets exploiting parameters for increasing network depth can gain computational benefits even with reduced size of the RNN "state". HighwayLSTM has shown an advantage over other models in relatively swallow architecture, however, its takes maximum elapse time to converge as compared to other models. Finally, by comparing the results of RHN and R2HN, and LSTM and ResLSTM, we can endorse that depth of hierarchical structure of RNNs play a vital part in the generalization of sequence models beside recurrent depth. The learning curves of the networks are shown in Fig. 2. The results illustrate that deep R2HN optimize models faster than other networks. Despite performance evaluation of these models, we also monitored the training elapse time of the models. The results are shown in Table 1. As can be seen, the R2HN is optimizing the LM much faster than other compared networks, particularly when the model is deep (i.e, M5), R2HN is approximately 50 times faster than most nearest ResLSTM (M5) network. Moreover, deep R2HN is 300 times faster than it's comparatively swallow version (i.e. M3) whereas the other deep networks are 50 times at maximum than their swallow models. However, a key assumption underlying such hierarchical/deep RNNs is that sequences contain hierarchical structure which may not be always the case.

7 Conclusion

A novel deep RNN architecture residual recurrent highway network (R2HN) is presented. Highways connections are used to prevent vanishing gradient problem and residual connections are employed to avoid degradation problem while learning deep hierarchical structure. It is shown through a complexity and empirical analysis and that R2HN is computationally efficient and effective as compared with contemporary RNN models. These results also demonstrated that exploiting parameters for increasing network depth can gain computational benefits even with reduced size of the RNN "state". However, to further establish the efficacy of proposed R2HN, additional experimentations on other datasets are required. We are interested to analyze the behavior of individual residual, highway and RNN blocks in different settings to gain insights of model behaviors and improve performance limiting factors.

References

1. Graves, A.: Generating sequences with recurrent neural networks. arXiv:1308.0850 (2013)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv:1409.0473 (2014)
3. Mikolov, T.: Statistical language models based on neural networks. Presentation at Google, Mountain View (2012)
4. Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A novel connectionist system for unconstrained handwriting recognition. IEEE Trans. Pattern Anal. Mach. Intell. **31**(5), 855–868 (2009)
5. Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A.R., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., Kingsbury, B.: Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal Process. Mag. **29**(6), 82–97 (2012)
6. Graves, A., Mohamed, A.R., Hinton, G.: Speech recognition with deep recurrent neural networks. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 6645–6649 (2013)
7. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
8. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv:1301.3781 (2013)
9. Pascanu, R., Mikolov, T., Bengio, Y.: On the difficulty of training recurrent neural networks. In: International Conference on Machine Learning, pp. 1310–1318 (2013)

10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
11. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555 (2014)
12. Srivastava, R.K., Greff, K., Schmidhuber, J.: Highway networks. arXiv:1505.00387 (2015)
13. 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)
14. Zhang, Y., Chen, G., Yu, D., Yaco, K., Khudanpur, S., Glass, J.: Highway long short-term memory rnns for distant speech recognition. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 5755–5759 (2016)
15. Kim, J., El-Khamy, M., Lee, J.: Residual LSTM: Design of a deep recurrent architecture for distant speech recognition. arXiv:1701.03360 (2017)
16. Kim, Y., Jernite, Y., Sontag, D., Rush, A.M.: Character-aware neural language models. In: AAAI, pp. 2741–2749 (2016)
17. Jozefowicz, R., Zaremba, W., Sutskever, I.: An empirical exploration of recurrent network architectures. In: Proceedings of the 32nd International Conference on Machine Learning (ICML-15), pp. 2342–2350 (2015)
18. Pascanu, R., Gulcehre, C., Cho, K., Bengio, Y.: How to construct deep recurrent neural networks. arXiv:1312.6026 (2013)
19. Schmidhuber, J.: Learning complex, extended sequences using the principle of history compression. Learning, **4**(1) (2008)
20. El Hihi, S., Bengio, Y.: Hierarchical recurrent neural networks for long-term dependencies. In: Advances in Neural Information Processing Systems, pp. 493–499 (1996)
21. Du, Y., Wang, W., Wang, L.: Hierarchical recurrent neural network for skeleton based action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1110–1118 (2015)
22. Zilly, J.G., Srivastava, R.K., Koutník, J., Schmidhuber, J.: Recurrent highway networks. arXiv:1607.03474 (2016)
23. Wang, Y., Tian, F.: Recurrent residual learning for sequence classification. In: EMNLP, pp. 938–943 (2016)
24. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436 (2015)
25. Chung, J., Ahn, S., Bengio, Y.: Hierarchical multiscale recurrent neural networks. arXiv:1609.01704 (2016)
26. Levine, Y., Sharir, O., Shashua, A.: Benefits of depth for long-term memory of recurrent networks. arXiv:1710.09431 (2017)
27. Pan, P., Xu, Z., Yang, Y., Wu, F., Zhuang, Y.: Hierarchical recurrent neural encoder for video representation with application to captioning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1029–1038 (2016)
28. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Gated feedback recurrent neural networks. In: International Conference on Machine Learning, pp. 2067–2075 (2015)
29. Aharoni, Z., Rattner, G., Permuter, H.: Gradual learning of deep recurrent neural networks. arXiv:1708.08863 (2017)
30. Serban, I.V., Sordoni, A., Bengio, Y., Courville, A.C., Pineau, J.: Building end-to-end dialogue systems using generative hierarchical neural network models. In: AAAI, pp. 3776–3784 (2016)
31. Sainath, T.N., Vinyals, O., Senior, A., Sak, H.: Convolutional, long short-term memory, fully connected deep neural networks. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4580–4584 (2015)
32. Chan, W., Jaitly, N., Le, Q., Vinyals, O.: Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4960–4964 (2016)
33. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, pp. 3104–3112 (2014)
34. Zhang, X.Y., Yin, F., Zhang, Y.M., Liu, C.L., Bengio, Y.: Drawing and recognizing chinese characters with recurrent neural network. IEEE Trans. Pattern Anal. Mach. Intell. **40**(3), 849–862 (2018)
35. Yogatama, D., Dyer, C., Ling, W., Blunsom, P.: Generative and discriminative text classification with recurrent neural networks. arXiv:1703.01898 (2017)
36. Koutnik, J., Greff, K., Gomez, F., Schmidhuber, J.: A clockwork rnn. In: International Conference on Machine Learning, pp. 1863–1871 (2014)
37. Graves, A., Schmidhuber, J.: Offline handwriting recognition with multidimensional recurrent neural networks. In: Advances in Neural Information Processing Systems, pp. 545–552 (2009)
38. Sak, H., Senior, A., Beaufays, F.: Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. arXiv:1402.1128 (2014)
39. Goel, H., Melnyk, I., Banerjee, A.: R2N2: Residual recurrent neural networks for multivariate time series forecasting. arXiv:1709.03159 (2017)
40. Baskar, M.K., Karafiát, M., Burget, L., Veselý, K., Grézl, F., Černocký, J.: Residual memory networks: Feed-forward approach to learn long-term temporal dependencies. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4810–4814 (2017)
41. Karpathy, A., Johnson, J., Fei-Fei, L.: Visualizing and understanding recurrent networks. arXiv:1506.02078 (2015)
42. Werbos, P.J.: Backpropagation through time: What it does and how to do it. Proc. IEEE **78**(8), 1550–1560 (1990)
43. Marcus, M.P., Marcinkiewicz, M.A., Santorini, B.: Building a large annotated corpus of english: The penn treebank. Comput. Linguist. **19**(1), 313–330 (1993)
44. Goldsborough, P.: A tour of tensorflow. arXiv:1610.01178 (2016)