Recurrent Residual Learning for Sequence Classification

Yiren Wang*
University of Illinois at Urbana-Champaign
yiren@illinois.edu

Fei Tian
Microsoft Research
fetia@microsoft.com

Abstract

In this paper, we explore the possibility of leveraging Residual Networks (ResNet), a powerful structure in constructing extremely deep neural network for image understanding, to improve recurrent neural networks (RNN) for modeling sequential data. We show that for sequence classification tasks, incorporating residual connections into recurrent structures yields similar accuracy to Long Short Term Memory (LSTM) RNN with much fewer model parameters. In addition, we propose two novel models which combine the best of both residual learning and LSTM. Experiments show that the new models significantly outperform LSTM.

1 Introduction

Recurrent Neural Networks (RNNs) are powerful tools to model sequential data. Among various RNN models, Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) is one of the most effective structures. In LSTM, gating mechanism is used to control the information flow such that gradient vanishing problem in vanilla RNN is better handled, and long range dependency is better captured. However, as empirically verified by previous works and our own experiments, to obtain fairly good results, training LSTM RNN needs carefully designed optimization procedure (Hochreiter et al., 2001; Pascanu et al., 2013; Dai and Le, 2015; Laurent et al., 2015; He et al., 2016; Arjovsky et al., 2015), especially when faced with unfolded very deep architectures for fairly long sequences (Dai and Le, 2015).

From another perspective, for constructing very deep neural networks, recently Residual Networks (ResNet) (He et al., 2015) have shown their effectiveness in quite a few computer vision tasks. By learning a residual mapping between layers with identity skip connections (Jaeger et al., 2007), ResNet ensures a fluent information flow, leading to efficient optimization for very deep structures (e.g., with hundreds of layers). In this paper, we explore the possibilities of leveraging residual learning to improve the performances of recurrent structures, in particular, LSTM RNN, in modeling fairly long sequences (i.e., whose lengths exceed 100). To summarize, our main contributions include:

1. We introduce residual connecting mechanism into the recurrent structure and propose recurrent residual networks for sequence learning. Our model achieves similar performances to LSTM in text classification tasks, whereas the number of model parameters is greatly reduced.

2. We present in-depth analysis of the strengths and limitations of LSTM and ResNet in respect of sequence learning.

3. Based on such analysis, we further propose two novel models that incorporate the strengths of the mechanisms behind LSTM and ResNet. We demonstrate that our models outperform LSTM in many sequence classification tasks.

*This work was done when the author was visiting Microsoft Research Asia.

Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 938–943,
Austin, Texas, November 1-5, 2016.

2 Background

RNN models sequences by taking sequential input $x = \{x_1, \cdots, x_T\}$ and generating T step hidden states $h = \{h_1, \cdots, h_T\}$. At each time step t, RNN takes the input vector $x_t \in \mathcal{R}^n$ and the previous hidden state vector $h_{t-1} \in \mathcal{R}^m$ to produce the next hidden state h_t.

Based on this basic structure, LSTM avoids gradient vanishing in RNN training and thus copes better with long range dependencies, by further augmenting vanilla RNN with a memory cell vector $c_t \in \mathcal{R}^m$ and multiplicative gate units that regulate the information flow. To be more specific, at each time step t, an LSTM unit takes x_t, c_{t-1}, h_{t-1} as input, generates the input, output and forget gate signals (denoted as i_t, o_t and f_t respectively), and produces the next cell state c_t and hidden state h_t:

$$\begin{aligned} \tilde{c}_t &= tanh(W_c[h_{t-1}, x_t] + b_c) \\ i_t &= \sigma(W_i[h_{t-1}, x_t] + b_i) \\ f_t &= \sigma(W_f[h_{t-1}, x_t] + b_f) \\ o_t &= \sigma(W_o[h_{t-1}, x_t] + b_o) \\ c_t &= f_t \otimes C_{t-1} + i_t \otimes \tilde{c}_t \\ h_t &= o_t \otimes tanh(c_t) \end{aligned} \tag{1}$$

where $\otimes$ refers to element-wise product. $\sigma(x)$ is the sigmoid function $\sigma(x) = 1/(1+\exp(-x))$. $W_j (j \in \{i, o, f, c\})$ are LSTM parameters. In the following part, such functions generating h_t and c_t are denoted as $h_t, c_t = LSTM(x_t, h_{t-1}, c_{t-1})$.

Residual Networks (ResNet) are among the pioneering works (Szegedy et al., 2015; Srivastava et al., 2015) that utilize extra identity connections to enhance information flow such that very deep neural networks can be effectively optimized. ResNet (He et al., 2015) is composed of several stacked residual units, in which the l^{th} unit takes the following transformation:

$$h_{l+1} = f(g(h_l) + \mathcal{F}(h_l; W_l)) \tag{2}$$

where h_l and h_{l+1} are the input and output for the l^{th} unit respectively. $\mathcal{F}$ is the residual function with weight parameters W_l. f is typically the ReLU function (Nair and Hinton, 2010). g is set as identity function, i.e., $g(h_l) = h_l$. Such an identity connection guarantees the direct propagation of signals among different layers, thereby avoids gradient vanishing. The recent paper (Liao and Poggio, 2016) talks about the possibility of using shared weights in ResNet, similar to what RNN does.

3 Recurrent Residual Learning

The basic idea of recurrent residual learning is to force a direct information flow in different time steps of RNNs by identity (skip) connections. In this section, we introduce how to leverage residual learning to 1) directly construct recurrent neural network in subsection 3.1; 2) improve LSTM in subsection 3.2.

3.1 Recurrent Residual Networks (RRN)

The basic architecture of Recurrent Residual Network (**RRN** for short) is illustrated in Figure 1, in which orange arrows indicate the identity connections from each h_{t-1} to h_t, and blue arrows represent the recurrent transformations taking both h_t and x_t as input. Similar to equation (2), the recurrent transformation in RRN takes the following form (denoted as $h_t = RRN(x_t, h_{t-1})$ in the following sections):

$$h_t = f(g(h_{t-1}) + \mathcal{F}(h_{t-1}, x_t; W)), \tag{3}$$

where g is still the identity function s.t. $g(h_{t-1}) = h_{t-1}$, corresponding to the orange arrows in Figure 1. f is typically set as *tanh*. For function $\mathcal{F}$ with weight parameters W (corresponding to the blue arrows in Figure 1), inspired by the observation that higher recurrent depth tends to lead to better performances (Zhang et al., 2016), we impose K deep transformations in $\mathcal{F}$:

$$\begin{aligned} y_1^t &= \sigma(x_t W_1 + h_{t-1} U_1 + b_1) \\ y_2^t &= \sigma(x_t W_2 + y_1^t U_2 + b_2) \\ &\cdots \\ y_K^t &= \sigma(x_t W_K + y_{K-1}^t U_K + b_K) \\ \mathcal{F}(h_{t-1}, x_t) &= y_K^t \end{aligned} \tag{4}$$

where x_t is taken at every layer such that the input information is better captured, which works similarly to the mechanism of highway network (Srivastava et al., 2015). K is the recurrent depth defined in (Zhang et al., 2016). The weights W_m ($m \in \{1, \cdots, K\}$) are shared across different time steps t.

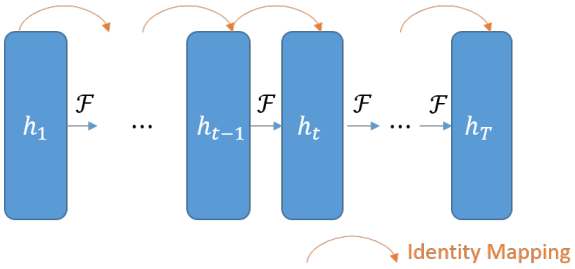

Figure 1: The basic structure of Recurrent Residual Networks.

RRN forces the direct propagation of hidden state signals between every two consecutive time steps with identity connections g. In addition, the multiple non-linear transformations in $\mathcal{F}$ guarantees its capability in modelling complicated recurrent relationship. In practice, we found that $K = 2$ yields fairly good performances, meanwhile leads to half of LSTM parameter size when model dimensions are the same.

3.2 Gated Residual RNN

Identity connections in ResNet are important for propagating the single input image information to higher layers of CNN. However, when it comes to sequence classification, the scenario is quite different in that there is a new input at every time step. Therefore, a forgetting mechanism to "forget" less critical historical information, as is employed in LSTM (controlled by the forget gate f_t), becomes necessary. On the other hand, while LSTM benefits from the flexible gating mechanism, its parametric nature brings optimization difficulties to cope with fairly long sequences, whose long range information dependencies could be better captured by identity connections.

Inspired by the success of the gating mechanism of LSTM and the residual connecting mechanism with enhanced information flow of ResNet, we further propose two Gated Residual Recurrent models leveraging the strengths of the two mechanisms.

3.2.1 Model 1: Skip-Connected LSTM (SC-LSTM)

Skip-Connected LSTM (**SC-LSTM** for short) introduces identity connections into standard LSTM to enhance the information flow. Note that in Figure 1, a straightforward approach is to replace $\mathcal{F}$ with an LSTM unit. However, our preliminary experiments do not achieve satisfactory results. Our conjecture is that identity connections between consecutive memory cells, which are already sufficient to maintain short-range memory, make the unregulated information flow overly strong, and thus compromise the merit of gating mechanism.

To reasonably enhance the information flow for LSTM network while keeping the advantage of gating mechanism, starting from equation (1), we propose to add skip connections between two LSTM hidden states with a wide range of distance L (e.g., $L = 20$), such that $\forall t = \{1, 1+L, 1+2L, \cdots, 1+\lfloor\frac{T-L-1}{L}\rfloor L\}$:

$$h_{t+L} = tanh(c_{t+L}) \otimes o_{t+L} + \alpha h_t \tag{5}$$

Here α is a scalar that can either be fixed as 1 (i.e., identity mapping) or be optimized during training process as a model parameter (i.e., parametric skip connection). We refer to these two variants as **SC-LSTM-I** and **SC-LSTM-P** respectively. Note that in SC-LSTM, the skip connections only exist in time steps $1, 1+L, 1+2L, \cdots, 1+\lfloor\frac{T-L-1}{L}\rfloor L$. The basic structure is shown in Figure 2.

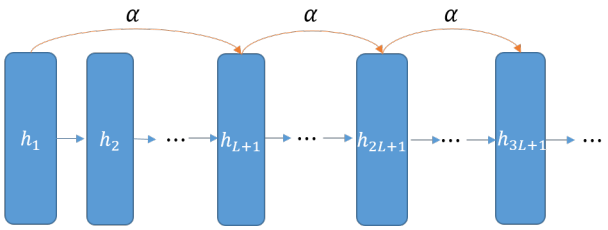

Figure 2: The basic structure of Skip-Connected LSTM.

3.2.2 Model 2: Hybrid Residual LSTM (HRL)

Since LSTM generates sequence representations out of flexible gating mechanism, and RRN generates representations with enhanced residual historical information, it is a natural extension to combine the two representations to form a signal that benefits from both mechanisms. We denote this model as Hybrid Residual LSTM (**HRL** for short).

In HRL, two independent signals, h_t^{LSTM} generated by LSTM (equation (1)) and h_t^{RRN} generated by RRN (equation (3)), are propagated through LSTM and RRN respectively:

$$\begin{aligned} h_t^{LSTM}, c_t &= LSTM(x_t, h_{t-1}^{LSTM}, c_{t-1}) \\ h_t^{RRN} &= RRN(x_t, h_{t-1}^{RRN}) \end{aligned} \tag{6}$$

The final representation h_T^{HRL} is obtained by the mean pooling of the two "sub" hidden states:

$$h_T^{HRL} = \frac{1}{2}(h_T^{LSTM} + h_T^{RRN}) \quad (7)$$

h_T^{HRL} is then used for higher level tasks such as predicting the sequence label. Acting in this way, h_T^{HRL} contains both the *statically* forced and *dynamically* adjusted historical signals, which are respectively conveyed by h_t^{RRN} and h_t^{LSTM}.

4 Experiments

We conduct comprehensive empirical analysis on sequence classification tasks. Listed in the ascending order of average sequence lengths, several public datasets we use include:

1. **AG's news corpus**[1],a news article corpus with categorized articles from more than $2,000$ news sources. We use the dataset with 4 largest classes constructed in (Zhang et al., 2015).

2. **IMDB movie review dataset**[2], a binary sentiment classification dataset consisting of movie review comments with positive/negative sentiment labels (Maas et al., 2011).

3. **20 Newsgroups** (**20NG** for short), an email collection dataset categorized into 20 news groups. Simiar to (Dai and Le, 2015), we use the post-processed version[3], in which attachments, PGP keys and some duplicates are removed.

4. **Permuted-MNIST** (**P-MNIST** for short). Following (Le et al., 2015; Arjovsky et al., 2015), we shuffle pixels of each MNIST image (LeCun et al., 1998) with a fixed random permutation, and feed all pixels sequentially into recurrent network to predict the image label. Permuted-MNIST is assumed to be a good testbed for measuring the ability of modeling very long range dependencies (Arjovsky et al., 2015).

[1] http://www.di.unipi.it/~gulli/AG_corpus_of_news_articles.html

[2] http://ai.stanford.edu/~amaas/data/sentiment/

[3] http://ana.cachopo.org/datasets-for-single-label-text-categorization

Detailed statistics of each dataset are listed in Table 1. For all the text datasets, we take every word as input and feed word embedding vectors pre-trained by Word2Vec (Mikolov et al., 2013) on Wikipedia into the recurrent neural network. The top most frequent words with 95% total frequency coverage are kept, while others are replaced by the token "UNK". We use the standard training/test split along with all these datasets and randomly pick 15% of training set as dev set, based on which we perform early stopping and for all models tune hyper-parameters such as dropout ratio (on non-recurrent layers) (Zaremba et al., 2014), gradient clipping value (Pascanu et al., 2013) and the skip connection length L for SC-LSTM (cf. equation (5)). The last hidden states of recurrent networks are put into logistic regression classifiers for label predictions. We use Adadelta (Zeiler, 2012) to perform parameter optimization. All our implementations are based on Theano (Theano Development Team, 2016) and run on one K40 GPU. All the source codes and datasets can be downloaded at `https://publish.illinois.edu/yirenwang/emnlp16source/`.

We compare our proposed models mainly with the state-of-art standard LSTM RNN. In addition, to fully demonstrate the effects of residual learning in our HRL model, we employ another hybrid model as baseline, which combines LSTM and GRU (Cho et al., 2014), another state-of-art RNN variant, in a similar way as HRL. We use **LSTM+GRU** to denote such a baseline. The model sizes (word embedding size $\times$ hidden state size) configurations used for each dataset are listed in Table 2. In Table 2, "Non-Hybrid" refers to LSTM, RRN and SC-LSTM models, while "Hybrid" refers to two methods that combines two basic models: HRL and LSTM+GRU. The model sizes of all hybrid models are smaller than the standard LSTM. All models have only one recurrent layer.

4.1 Experimental Results

All the classification accuracy numbers are listed in Table 3. From this table, we have the following observations and analysis:

1. RRN achieves similar performances to standard LSTM in all classification tasks with only

Dataset	Ave. Len	Max Len	#Classes	#Train : #Test
AG's News	34	211	4	$120,000 : 7,600$
IMDB	281	$2,956$	2	$25,000 : 25,000$
20NG	267	$11,924$	20	$11,293 : 7,528$
P-MNIST	784	784	10	$60,000 : 10,000$

Table 1: Classification Datasets.

	AG's News	*IMDB*	*20NG*	*P-MNIST*
Non-Hybird	256×512	256×512	500×768	1×100
Hybrid	256×384	256×384	256×512	1×80

Table 2: Model Sizes on Different Dataset.

Model/Task	*AG's News*	*IMDB*	*20NG*	*P-MNIST*
LSTM	91.76%	88.88%	79.21%	90.64%
RRN	91.19%	89.13%	79.76%	88.63%
SC-LSTM-P	92.01%	90.74%	**82.98%**	94.46%
SC-LSTM-I	**92.05%**	90.67%	81.85%	**94.80%**
LSTM+GRU	91.05%	89.23%	80.12%	90.28%
HRL	91.90%	**90.92%**	81.73%	90.33%

Table 3: Classification Results (Test Accuracy).

half of the model parameters, indicating that residual network structure, with connecting mechanism to enhance the information flow, is also an effective approach for sequence learning. However, the fact that it fails to significantly outperform other models (as it does in image classification) implies that forgetting mechanism is desired in recurrent structures to handle multiple inputs.

2. Skip-Connected LSTM performs much better than standard LSTM. For tasks with shorter sequences such as AG's News, the improvement is limited. However, the improvements get more significant with the growth of sequence lengths among different datasets[4], and the performance is particularly good in P-MNIST with very long sequences. This reveals the importance of skip connections in carrying on historical information through a long range of time steps, and demonstrates the effectiveness of our approach that adopts the residual connecting mechanism to improve LSTM's capability of handling long-term dependency. Furthermore, SC-LSTM is robust with different hyperparameter values: we test $L = 10, 20, 50, 75$ in P-MNIST and find the performance is not sensitive w.r.t. these L values.

3. HRL also outperforms standard LSTM with fewer model parameters[5]. In comparison, the hybrid model of LSTM+GRU cannot achieve such accuracy as HRL. As we expected, the additional long range historical information propagated by RRN is proved to be good assistance to standard LSTM.

5 Conclusion

In this paper, we explore the possibility of leveraging residual network to improve the performance of LSTM RNN. We show that direct adaptation of ResNet performs well in sequence classification. In addition, when combined with the gating mechanism in LSTM, residual learning significantly improve LSTM's performance. As to future work, we plan to apply residual learning to other sequence tasks such as language modeling, and RNN based neural machine translation (Sutskever et al., 2014) (Cho et al., 2014).

[4]t-test on SC-LSTM-P and SC-LSTM-I with $p\_value < 0.001$.

[5]t-test on HRL with $p\_value < 0.001$.

References

Martin Arjovsky, Amar Shah, and Yoshua Bengio. 2015. Unitary evolution recurrent neural networks. *arXiv preprint arXiv:1511.06464*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems*, pages 3061–3069.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Identity mappings in deep residual networks. *arXiv preprint arXiv:1603.05027*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jrgen Schmidhuber. 2001. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies.

Herbert Jaeger, Mantas Lukoševičius, Dan Popovici, and Udo Siewert. 2007. Optimization and applications of echo state networks with leaky-integrator neurons. *Neural Networks*, 20(3):335–352.

César Laurent, Gabriel Pereyra, Philémon Brakel, Ying Zhang, and Yoshua Bengio. 2015. Batch normalized recurrent neural networks. *arXiv preprint arXiv:1510.01378*.

Quoc V Le, Navdeep Jaitly, and Geoffrey E Hinton. 2015. A simple way to initialize recurrent networks of rectified linear units. *arXiv preprint arXiv:1504.00941*.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Qianli Liao and Tomaso Poggio. 2016. Bridging the gaps between residual learning, recurrent neural networks and visual cortex. *arXiv preprint arXiv:1604.03640*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of The 30th International Conference on Machine Learning*, pages 1310–1318.

Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *Advances in Neural Information Processing Systems*, pages 2368–2376.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657.

Saizheng Zhang, Yuhuai Wu, Tong Che, Zhouhan Lin, Roland Memisevic, Ruslan Salakhutdinov, and Yoshua Bengio. 2016. Architectural complexity measures of recurrent neural networks. *arXiv preprint arXiv:1602.08210*.