

---

# Recurrent neural network regularization

---

**Wojciech Zaremba**

Google & New York University

**Ilya Sutskever**

Google

**Oriol Vinyals**

Google

WOJ.ZAREMBA@GMAIL.COM

ILYASU@GOOGLE.COM

VINYALS@GOOGLE.COM

## Abstract

We present a simple regularization technique for Recurrent Neural Networks (RNNs) with Long Short-Term Memory (LSTM) units. The technique is based on dropout and gives a tremendous reduction in overfitting. We show that it is useful in a variety of sequence modelling problems that include language modeling, speech recognition, and machine translation.

## 1. Introduction

RNNs yield the state of the art performance on many sequence modelling tasks, including language modelling (?), speech recognition (Graves et al., 2013), and machine translation (?). However, there have been no good ways of regularizing them. As a result, practical applications tend to use RNNs that are too small, as large RNNs would overfit. To date, existing regularization methods give relatively small improvements on RNNs (Graves, 2013). Dropout is a highly effective way of regularizing feedforward neural networks (Srivastava, 2013) that had enjoyed considerable success. However, it is not clear how to use dropout on RNNs, because a naive application of dropout does not yield good results. In this work, we show how to correctly apply dropout to LSTMs, and demonstrate that this results in great reduction in overfitting.

## 2. Related work

Dropout (Srivastava, 2013) is a recently regularization method that has enjoyed a lot of success in applications of feedforward neural networks. While there has been a lot of work extending Dropout (Wang & Manning, 2013; Wan et al., 2013), there has been relatively little research

so far in applying dropout to RNNs. The only paper on the topic is (Bayer et al., 2013) which focuses on “marginalized dropout” (from (Wang & Manning, 2013)) which is a noiseless deterministic approximation to conventional dropout. (Bayer et al., 2013) claims that conventional dropout cannot be successfully used in RNNs due to its large variance which hurts learning and causes poor convergence. In this work, we show that a specific application of dropout greatly reduces the overfitting of RNNs.

There has been extensive work on using RNNs for language modelling (Mikolov, 2012; Sutskever, 2013). Moreover, there have been a number of architectural variations on the RNN that are better suited for learning on data with long term dependencies (Hochreiter & Schmidhuber, 1997; Graves et al., 2009; Cho et al., 2014; ?; ?). This work focuses on the LSTM which is the most widespread variant of the RNN, although it is likely that our findings are valid for other models.

In this paper, we focus on the following RNN tasks: language modelling, speech recognition, and machine translation. Language modelling is the first task where RNNs achieved substantial success (Mikolov et al., 2010; 2011; Pascanu et al., 2013). RNNs have also been successfully used for speech recognition (Robinson et al., 1996; Graves et al., 2013) and have been applied to machine translation, where they are typically used for language modelling, re-ranking, or phrase modelling (?Cho et al., 2014; ?; ?).

## 3. Regularizing RNN with LSTM cells

In this section we describe the deep LSTM 3.1. Next, we show how to regularize them 3.2, and we provide an intuition for why our regularization scheme works.

We use upperscript to denote a layer number and lowerscript to denote a timestep. All our states are  $n$ -dimensional. Let  $h_k^l \in \mathbb{R}^n$  be a hidden state in layer  $l$  in step  $k$ . Moreover, let  $T_{n,m} : \mathbb{R}^n \rightarrow \mathbb{R}^m$  be a linear transform and with a bias ( $Wx + b$  for some  $W$  and  $b$ ). Let  $\odot$  be

a element-wise multiplication and let  $h_k^0$  be an input word vector. We use the activations  $h_k^L$  to predict  $y_k$ , since  $L$  is the number of layers in our deep LSTM.

### 3.1. Long-short term memory units

The RNNs dynamics can be described in terms of deterministic transitions from previous hidden states to the hidden states in the next step. The transition is a function

$$\text{RNN} : h_k^{l-1}, h_{k-1}^l \rightarrow h_k^l$$

For classical RNNs, this function is given by

$$h_k^l = f(T_{n,n}h_k^{l-1} + T_{n,n}h_{k-1}^l), \text{ where } f \in \{\text{sigm}, \text{tanh}\}$$

The LSTM has relatively complicated dynamics that make it easy to “memorize” information for extended number of time steps. The “long term” memory is stored in a vector of *memory cells*  $c_k^l \in \mathbb{R}^n$ . Although there are many LSTM architectures that differ in their connectivity structure and activation functions, all LSTM architectures have explicit memory cells that make it easy to store information for extended periods of time. The LSTM can decide to overwrite this information, retrieve it, or keep it for the next time step. The LSTM architecture used in our experiments is given by the following equations (Graves et al., 2013):

$$\begin{aligned} \text{LSTM} : h_k^{l-1}, h_{k-1}^l, c_{k-1}^l &\rightarrow h_k^l, c_k^l \\ \begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} &= \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \text{tanh} \end{pmatrix} T_{2n,4n} \begin{pmatrix} h_k^{l-1} \\ h_{k-1}^l \end{pmatrix} \\ c_k^l &= f \odot c_{k-1}^l + i \odot g \\ h_k^l &= o \odot \tanh(c_k^l) \end{aligned}$$

In these equations,  $\text{sigm}$  and  $\text{tanh}$  are applied element-wise. Diagram 1 illustrates the LSTM equations.

### 3.2. Regularization with Dropout

The main contribution of this paper is the discovery that carefully placed dropout tremendously improves the generalization ability of LSTMs. To be effective, dropout must be placed on the non-recurrent connections 2. The following equation describes it more precisely, where  $\mathbf{D}$  is the dropout operator that sets a random subset of its argument to zero:

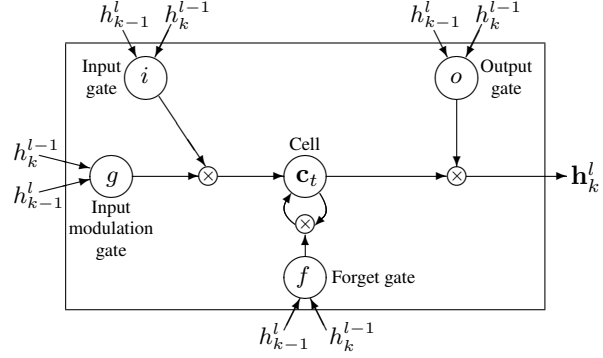


Figure 1. Graphical representation of LSTM memory cells used in this paper (there are minor differences in compare to (Graves, 2013)).

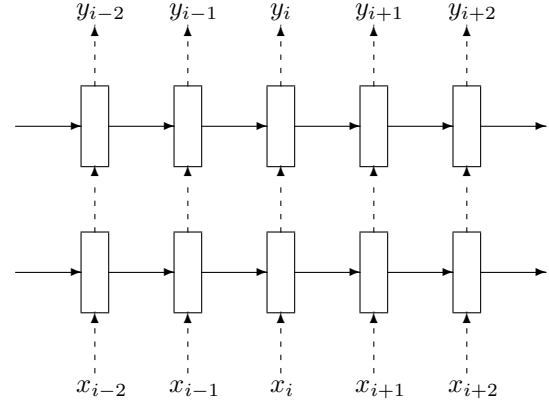


Figure 2. Regularized multilayer RNN. Dashed arrows indicate connections with applied dropout, while solid lines indicate connections where dropout is not applied.

$$\begin{aligned} \begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} &= \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \text{tanh} \end{pmatrix} T_{2n,4n} \begin{pmatrix} \mathbf{D}(h_k^{l-1}) \\ h_{k-1}^l \end{pmatrix} \\ c_k^l &= f \odot c_{k-1}^l + i \odot g \\ h_k^l &= o \odot \tanh(c_k^l) \end{aligned}$$

The dropout operator corrupts the information carried by the units, which forces them to perform their intermediate computations in a more robust manner. At the same time, we do not want to erase all of the information. We would especially like to remember events that occurred

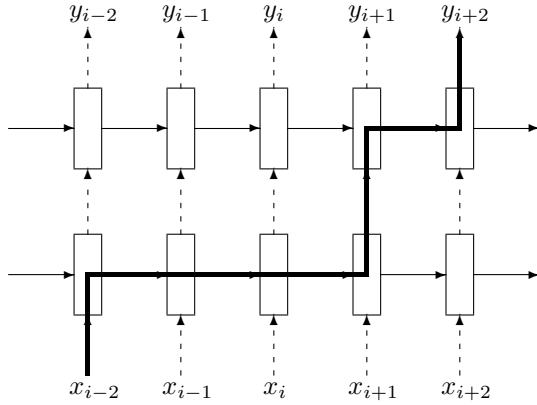


Figure 3. Thick line indicates an exemplary information flow in RNN. Information flow line is crossed  $L + 1$  times, where  $L$  is depth of network.

many timesteps in the past. Figure 3 shows a possible flow of information from an event that occurred at  $x_{i-2}$  to the prediction in the step  $i + 2$ . We can see that the information is corrupted by dropout only  $L + 1$  times, and it is independent of how far in past event occurred. Previous regularization techniques would perturb the recurrent connections or the recurrent hidden state, which would greatly reduce the LSTM’s memory capacity. By not using dropout on the recurrent connections, the LSTM is able to get most of the benefit of dropout without sacrificing its valuable ability to learn and store information for long periods of time.

## 4. Experiments

We present here results in three domains: language modeling 4.1, speech recognition 4.2, and machine translation 4.3.

### 4.1. Language modeling

We have conducted word-level prediction experiments on Penn tree bank (PTB) dataset (?). This dataset consists of 929k training words, 73k validation words, and 82k test words. It has 10k words in vocabulary.

Our model is a two-layer LSTM with 650 units [so 325 units per layer?] whose parameters are initialized uniformly in  $[-0.05, 0.05]$ . We apply 50% dropout on the non-recurrent connections. We train for 39 epochs, starting with learning rate of 1, and after 6 epochs we decrease it by a factor of 1.2 in every epoch. We unroll the RNN for 35 steps, and clip the norm of the gradients (normalized

Model	Validation set	Test set
A single model		
(Pascanu et al., 2013)		107.5
(Cheng et al.)		100.0
Non-regularized LSTM	120.7	114.5
Regularized LSTM	86.2	<b>82.7</b>
Model averaging		
(Mikolov, 2012)	83.5	89.4
(Cheng et al.)		80.6
2 non-regularized LSTMs	100.4	96.1
5 non-regularized LSTMs	87.9	84.1
10 non-regularized LSTMs	83.5	80.0
2 regularized LSTMs	80.6	77.0
5 regularized LSTMs	76.7	73.3
10 regularized LSTMs	75.2	<b>72.0</b>

Table 1. Word-level perplexity on Penn-tree-bank dataset.

by minibatch size) at 5. We set mini-batch to 20. Table 1 compares various models with our models.

### 4.2. Speech recognition

### 4.3. Machine translation

## 5. Discussion

We present amazing performance boosts with methods, which we only intuitively understand. We think, that it crucial to derive our models analytically, rather than based only on intuition. However, so far this problems seem to be very difficult to tackle.

## References

- Bayer, Justin, Osendorfer, Christian, Chen, Nutan, Urban, Sebastian, and van der Smagt, Patrick. On fast dropout and its applicability to recurrent networks. *arXiv preprint arXiv:1311.0701*, 2013.
- Cheng, Wei-Chen, Kok, Stanley, Pham, Hoai Vu, Chieu, Hai Leong, and Chai, Kian Ming A. Language modeling with sum-product networks.
- Cho, Kyunghyun, van Merriënboer, Bart, Gulcehre, Caglar, Bougares, Fethi, Schwenk, Holger, and Bengio, Yoshua. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.
- Graves, Alex. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.
- Graves, Alex, Liwicki, Marcus, Fernández, Santiago, Bertolami, Roman, Bunke, Horst, and Schmidhuber, Jürgen. A novel connectionist system for unconstrained

- handwriting recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(5):855–868, 2009.
- Graves, Alex, Mohamed, Abdel-rahman, and Hinton, Geoffrey. Speech recognition with deep recurrent neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 6645–6649. IEEE, 2013.
- Hochreiter, Sepp and Schmidhuber, Jürgen. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- Mikolov, Tomáš. *Statistical language models based on neural networks*. PhD thesis, Ph. D. thesis, Brno University of Technology, 2012.
- Mikolov, Tomas, Karafiát, Martin, Burget, Lukas, Cernocký, Jan, and Khudanpur, Sanjeev. Recurrent neural network based language model. In *INTERSPEECH*, pp. 1045–1048, 2010.
- Mikolov, Tomas, Deoras, Anoop, Povey, Daniel, Burget, Lukas, and Cernocký, Jan. Strategies for training large scale neural network language models. In *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*, pp. 196–201. IEEE, 2011.
- Pascanu, Razvan, Gulcehre, Caglar, Cho, Kyunghyun, and Bengio, Yoshua. How to construct deep recurrent neural networks. *arXiv preprint arXiv:1312.6026*, 2013.
- Robinson, Tony, Hochberg, Mike, and Renals, Steve. The use of recurrent neural networks in continuous speech recognition. In *Automatic speech and speaker recognition*, pp. 233–258. Springer, 1996.
- Srivastava, Nitish. *Improving neural networks with dropout*. PhD thesis, University of Toronto, 2013.
- Sutskever, Ilya. *Training Recurrent Neural Networks*. PhD thesis, University of Toronto, 2013.
- Wan, Li, Zeiler, Matthew, Zhang, Sixin, Cun, Yann L, and Fergus, Rob. Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pp. 1058–1066, 2013.
- Wang, Sida and Manning, Christopher. Fast dropout training. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pp. 118–126, 2013.