Notes on LSTMs

Chris Dyer
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA, 15213
cdyer@cs.cmu.edu

Abstract

Long Short Term Memory (LSTM) Recurrent Neural Nets (RNNs) are a variant of neural networks designed to avoid problems recurrent neural networks have with learn long-range temporal dependencies (Hochreiter and Schmidhuber, 1997).

1 Introduction

This document briefly sketches classic recurrent neural networks and long short-term memory (LSTM) RNNs, which address some of the issues. Both of these computational devices are map a series of inputs $(\mathbf{x}_1, \mathbf{x}_2, \ldots)$ to a series of outputs $(\mathbf{y}_1, \mathbf{y}_2, \ldots)$. A classic example is language modeling where the input at time t, $\mathbf{x}_t$, is a representation of the previous word and $\mathbf{y}_t$ is some representation of the prediction of the next word in the vocabulary. LSTMs address some difficulties associated with capturing long-range dependencies with classic RNNs due to the so-called "vanishing gradients".

1.1 Notion

Matrices and higher-order tensors are designated with boldface capital Latin or Greek letters, e.g. $\mathbf{X}$, $\boldsymbol{\Sigma}$, a column vector is a boldface lowercase letter, e.g. $\mathbf{x}$, $\boldsymbol{\mu}$, a scalar is a lowercase letter, e.g. x, μ. A structured object, such as a sequence or tree, is bold script $\boldsymbol{w} = \langle w_1, w_2, \ldots, w_\ell \rangle$. The element at i, j in a matrix $\mathbf{A}$ is written a_{ij} or $(\mathbf{A})_{ij}$. Let $\odot$ designate the element-wise (Hadamard) product.

2 RNNs

A **recursive neural network** is defined as follows. At time t it receives an input vector $\mathbf{x}_t \in \mathbb{R}^k$ of observations, an input vector $\mathbf{h}_{t-1} \in \mathbb{R}^k$ representing the previous hidden state, it produces a new hidden representation

$$\mathbf{h}_t = \sigma(\mathbf{C}_x \mathbf{x}_t + \mathbf{C}_h \mathbf{h}_{t-1}).$$

The RNN then produces an output for time t:

$$\mathbf{y}_t = f(\mathbf{W}\mathbf{h}_t + \mathbf{b})$$

where f is some element-wise nonlinearity.

3 LSTMs

A **long short-term memory** is defined as follows. At time t it receives an input vector $\mathbf{x}_t \in \mathbb{R}^k$ of observations, an input vector $\mathbf{h}_{t-1} \in \mathbb{R}^k$ representing the previous hidden state, and a memory state $\mathbf{c}_{t-1} \in \mathbb{R}^k$ from the previous time step. The computes three gates $\mathbf{i}_t$, $\mathbf{f}_t$, and $\mathbf{o}_t$ controlling, respectively, how strongly to consider the observed input, how much of the memory to retrain, and how much information to pass to the output. It additionally computes a new value for the memory $\mathbf{c}_t$ and a new hidden representation as follows:

$$
\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{I}_x\mathbf{x}_t + \mathbf{I}_h\mathbf{h}_{t-1} + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{F}_x\mathbf{x}_t + \mathbf{F}_h\mathbf{h}_{t-1} + \mathbf{b}_f) \\
\mathbf{o}_t &= \sigma(\mathbf{O}_x\mathbf{x}_t + \mathbf{O}_h\mathbf{h}_{t-1} + \mathbf{b}_o) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot g(\mathbf{C}_x\mathbf{x}_t + \mathbf{C}_h\mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{h}_t &= \mathbf{o}_t \odot g(\mathbf{c}_t)
\end{aligned}
$$

where σ is the element-wise logistic sigmoid function and g is an element-wise nonlinearity (usually $\tanh$). The behavior of the network is controlled by the parameters $\mathbf{I}_x, \mathbf{I}_h, \mathbf{F}_x, \mathbf{F}_h, \mathbf{O}_x, \mathbf{O}_h, \mathbf{C}_x$, and $\mathbf{C}_h$ which are all in $\mathbb{R}^{k\times k}$. The base values $\mathbf{h}_0 = \mathbf{c}_0 = \mathbf{0}$. Finally, a new output is computed:

$$\mathbf{y}_t = f(\mathbf{W}\mathbf{h}_t + \mathbf{b})$$

References

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, pages 1735–1780.

Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. In *Proc. ICML*.