Review on RNN

LSTM/BRNN and NTM

Lei Yuntong 1700012893

December 12, 2018

EECS, PKU

RNN overview

RNN overview

RNN, namely Recurrent Neural Network (in some place also means Recursive NN, but they have little difference), mainly deals with sequential inputs and outputs.

Recurrent neural networks are feedforward neural networks augmented by the inclusion of edges that span adjacent time step.

Basic RNN is like below:

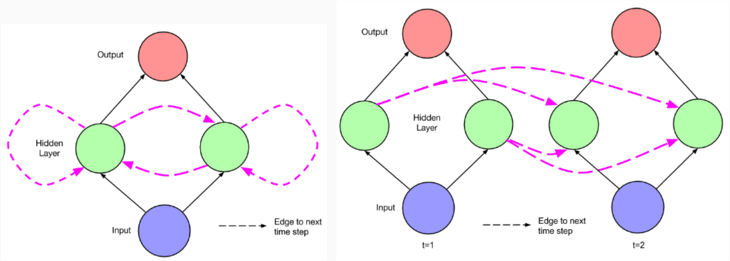

Figure 1: folded and unfolded layer of a RNN example[1]

RNN overview

Input of RNN is usually a sequence

$$\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, ..., \boldsymbol{x}^{(T)}, ...\}$$

and output of RNN can also be a sequence

$$\{\hat{\boldsymbol{y}}^{(1)}, \hat{\boldsymbol{y}}^{(2)}, ..., \hat{\boldsymbol{y}}^{(T)}, ...\}$$

RNN overview

Typically we have such equations in a RNN.

$$\boldsymbol{s}_t = W^{hx}\boldsymbol{x}^{(t)} + W^{hh}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_h$$

$$\boldsymbol{h}^{(t)} = \sigma(\boldsymbol{s}_t)$$

$$\hat{\boldsymbol{y}}^{(t)} = \mathrm{softmax}(W^{yh}\boldsymbol{h}^{(t)} + \boldsymbol{b}_y)$$

$$E_t = -\boldsymbol{y}_t^T log(\hat{\boldsymbol{y}}_t)$$

$$E = \sum_t^T E_t$$

Where $\mathrm{softmax}(\boldsymbol{x}) = \dfrac{e^{x_i}}{\sum_{j=0}^{n} e^{x_j}}$

RNN overview

Back propagation of RNN is called BPTT(Back Propagation Through Time)

When we back propagate, we want to optimize parameters in matrices W^{hx}, W^{yh} and W^{hh}.

Thus we need to compute $\frac{\partial E}{\partial W^{hx}}$, $\frac{\partial E}{\partial W^{yh}}$ and $\frac{\partial E}{\partial W^{hh}}$.

Note that $E = \sum_{t=0}^{T} E_t$, so we just need to compute $\frac{\partial E_t}{\partial W^{**}}$ and add them up together.

RNN overview

The results are

$$\frac{\partial E_t}{\partial W^{yh}} = (\hat{\boldsymbol{y}} - \boldsymbol{y}_t) \otimes \boldsymbol{h}_t, \quad \frac{\partial E}{\partial W^{yh}} = \sum_{t=0}^{T} (\hat{\boldsymbol{y}} - \boldsymbol{y}_t) \otimes \boldsymbol{h}_t$$

Similarly

$$\frac{\partial E}{\partial W^{hh}} = \sum_{t=0}^{T} \sum_{k=0}^{t} \delta_k \otimes \boldsymbol{h}_{k-1}$$

$$\frac{\partial E}{\partial W^{hx}} = \sum_{t=0}^{T} \delta_k \otimes \boldsymbol{x}_k$$

Where $\delta_k = (W^{yhT}(\hat{\boldsymbol{y}} - \boldsymbol{y}_t) \odot (1 - \boldsymbol{h}_t \odot \boldsymbol{h}_t)$.

And we can use these results to update each parameter matrix.

RNN overview

However, people face severe Gradient Vanishment.
To solve this, we have several optimizations to naive RNN, such as Long Short-Term Memory, Bidirectonal RNN and so on.

LSTM: Long Short-Term Memory

LSTM

A direct comparison between nodes of normal RNN and LSTM.

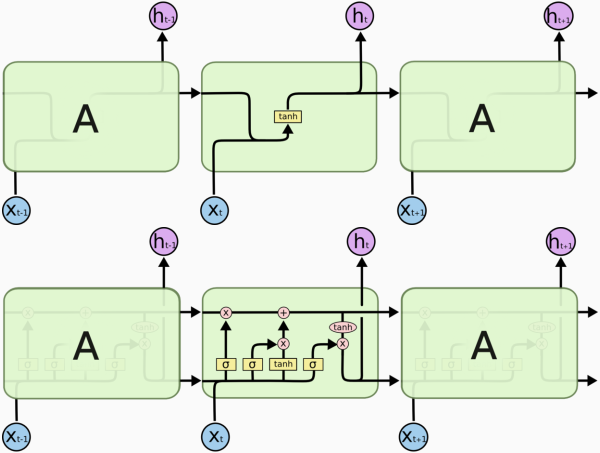

Figure 2: normal RNN vs LSTM

LSTM

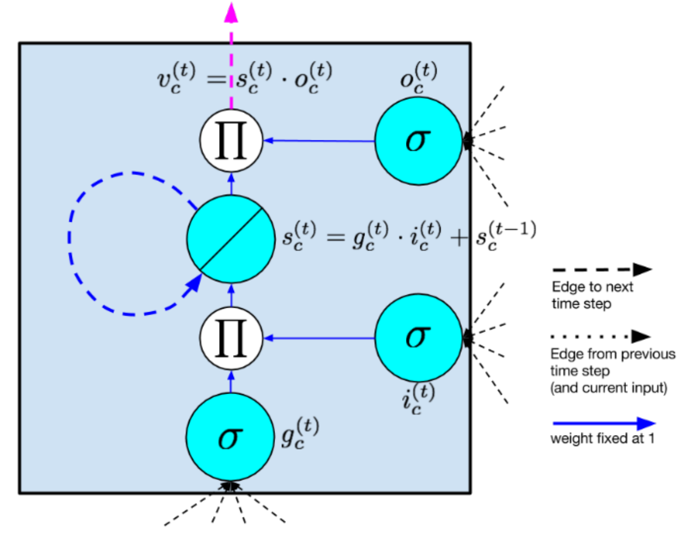

Figure 3: folded memory cell of LSTM[1]

LSTM

The equations to describe the memory cell is like below.

Input node

$$\boldsymbol{g}^{(t)} = \phi(W^{gx}\boldsymbol{x}^{(t)} + W^{gh}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_g)$$

Input gate

$$\boldsymbol{i}^{(t)} = \sigma(W^{ix}\boldsymbol{x}^{(t)} + W^{ih}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_i)$$

Internal state

$$\boldsymbol{s}^{(t)} = \boldsymbol{g}^{(t)} \odot \boldsymbol{i}^{(t)} + \boldsymbol{s}^{(t-1)}$$

Output gate

$$\boldsymbol{o}^{(t)} = \sigma(W^{ox}\boldsymbol{x}^{(t)} + W^{oh}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_o)$$

And hidden layer

$$\boldsymbol{h}^{(t)} = \phi(\boldsymbol{s}^{(t)}) \odot \boldsymbol{o}^{(t)}$$

LSTM

One more variant of LSTM is that added a forget gate.

$$\boldsymbol{f}^{(t)} = \sigma(W^{fx}\boldsymbol{x}^{(t)} + W^{fh}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_f)$$

and then internal state changes accordingly

$$\boldsymbol{s}^{(t)} = \boldsymbol{g}^{(t)} \odot \boldsymbol{i}^{(t)} + \boldsymbol{s}^{(t-1)} \odot \boldsymbol{f}^{(t)}$$

The memory cell is like below.

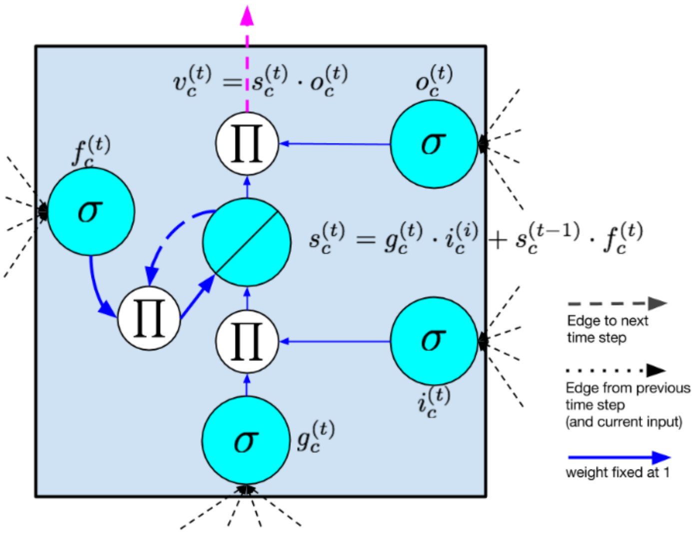

Figure 4: folded memory cell of LSTM with a forget gate[1]

LSTM

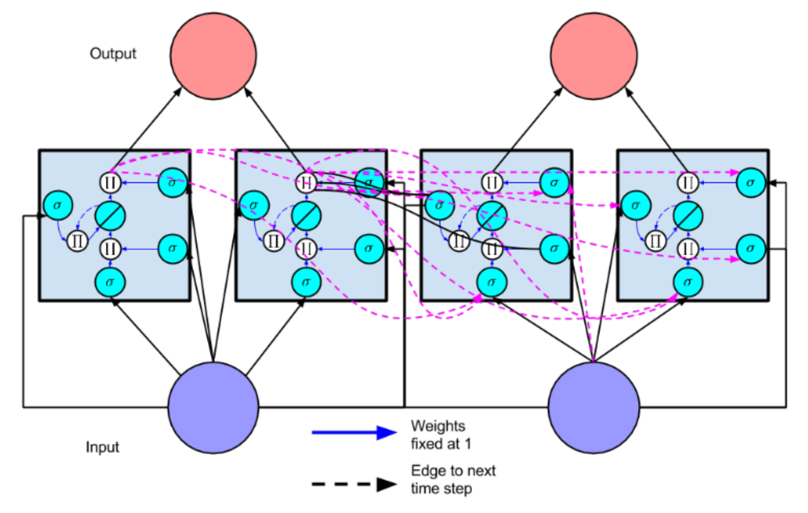

Figure 5: unfolded memory cell[1]

BRNN: Bidirectonal RNN

BRNN

BRNN stands for Bidirectonal Recurrent Neural Network. A simple BRNN is like this.

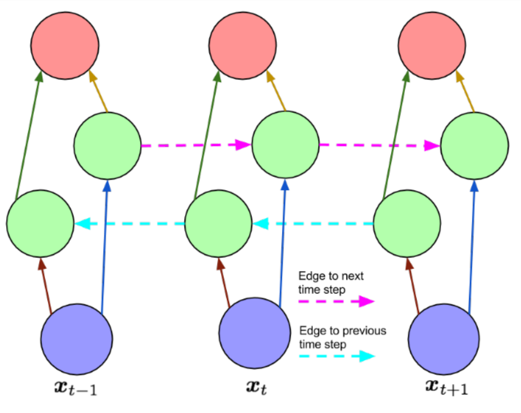

Figure 6: BRNN with 1 input node, 1 hidden node and 1 output node[1]

BRNN

The equations that describe this network is like Value of hidden layer

$$\boldsymbol{h}^{(t)} = \sigma(W^{hx}\boldsymbol{x}^{(t)} + W^{hh}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_h)$$

Value of hidden layer from another direction

$$\boldsymbol{z}^{(t)} = \sigma(W^{zx}\boldsymbol{x}^{(t)} + W^{zh}\boldsymbol{z}^{(t+1)} + \boldsymbol{b}_z)$$

Output

$$\hat{\boldsymbol{y}}^{(t)} = \sigma(W^{yh}\boldsymbol{h}^{(t)} + W^{yz}\boldsymbol{z}^{(t)} + \boldsymbol{b}_y)$$

References i

e. a. Zachary C. Lipton.
A critical review of recurrent neural networks for sequence learning.
arXiv: 1506.00019, 2015.