RNN and LSTM equations

Kushagra Bansal *

June 25, 2021

0.1 Introduction

In this peresentation, we will look at the training equations of a Recurrent Neural Network(RNN) derived using the Backpropagation through time (BPTT) algorithm. Then we will explore the numerical difficulty known as the Vanishing Gradients problem that arise while training an RNN.

This will help us to arrive at the Vanilla Long Short Term Memory model by reasoning. Then we will look at the forward propagation equations of an LSTM cell and finally look at all the training equation using the BPTT algorithm.

0.2 Training the RNN via BPTT

From section 2, we know that the RNN system as given by eq.(32) and eq. (33) as

$$s[n] = W_r * r[n-1] + W_x * x[n] + b_s$$

where

$$r[n] = tanh(s[n])(33)$$

From the first proposition we saw that the truncated RNN system unrolled for $[K_m$ steps is also enough to minimize the loss function L. Here for simplicity we will drop the subscript m and assume that the system is trained for K_m steps only, unless stated otherwise. The essence of Backprop algorithm is to calculate the gradients of the objective function E with respect to all the parameters of the model defined by $\theta = [W_r, W_x, b_s]$ and use these gradient

*Funded by SRF

values in gradient descent learning algorithm to update the parameters and minimize the loss function over several iterations.

Firstly, since the loss function E depends upon the readout signal $r[n]$ we will measure its gradient wrt $r[n]$.

$$\chi[n] = \frac{\partial E}{\partial r[n]}$$

and since $r[n]$ explicitly depends upon the state signal $s[n]$ we shall also measure the gradient of E wrt $s[n]$.

$$\psi[n] = \frac{\partial E}{\partial s[n]}$$

From here we might say that $\psi[n] = \chi[n] * \frac{dr[n]}{ds[n]}$, but we should also incorporate the contribution coming from the next layer that is from, $s[n+1]$ as well. This means our expression for $\psi[n]$ will be

$$\psi[n] = \frac{\partial E[r[n]]}{\partial r[n]} + W_r \psi[n+1]$$

and

$$\chi[n] = \psi[n] * \frac{ds[n]}{dr[n]}$$

which implies

$$\chi[n] = \left(\frac{\partial E[r[n]]}{\partial r[n]} + W_r \psi[n+1] \right) * \frac{ds[n]}{dr[n]}$$

Just like in the forward propagating sequence, we will initialize this sequence as well with some initial condition as $\psi[n = K_m] = 0$.

Now expressing the gradient of E in terms of $\psi[n]$ will greatly simplify our task.

$$\frac{\partial E}{\partial W_r}[n] = \frac{\partial E}{\partial s[n]} * \frac{\partial s[n]}{\partial W_r}$$

$$\frac{\partial E}{\partial W_r}[n] = \psi[n] * r^T[n-1]$$

similarly,

$$\frac{\partial E}{\partial W_x}[n] = \frac{\partial E}{\partial s[n]} * \frac{\partial s[n]}{\partial W_x}$$

$$\frac{\partial E}{\partial W_x}[n] = \psi[n] * x^T[n]$$

and

$$\frac{\partial E}{\partial b_s}[n] = \frac{\partial E}{\partial s[n]} * \frac{\partial s[n]}{\partial b_s}$$

$$\frac{\partial E}{\partial b_s}[n] = \psi[n]$$

These are the gradients at a single step with index n. The expression for the total gradients across all training steps would be:

$$\frac{\partial E}{\partial \theta} = \sum_{n=0}^{K_m - 1} \frac{\partial E}{\partial \theta}[n]$$

0.2.1 Vanishing and exploding gradients

From the expression of $\psi[n]$, we can see the $< \psi[k] >$ is a backward moving sequence. This means that $\psi[n]$ for arbitrarily large value of index n can influence the entire sub sequence $< \psi[k] >$ for $0 \leq k < n$.
So we are interested in the proportion of $\psi[n]$ retained in back propagating to $\psi[k]$ where $n >> k$. So we calculate the derivative

$$\frac{\partial \psi[k]}{\partial \psi[n]}$$

as

$$\frac{\partial \psi[k]}{\partial \psi[n]} = \prod_{i=k+1}^{n} W_r * \frac{dr[i]}{ds[i]}$$

(*Refer to handwritten notes for derivation using an example*)
Now in section II, while analysing the stability of the RNN network, we came to the conlusion that W_r is a diagonal matrix with small positive entries so $|W_r| \to 0$. Also the derivative of the hyperbolic activation fucntion is pretty close to zero for most values. Thus when $n - k$ is very large the expression $\frac{\partial \psi[k]}{\partial \psi[n]} \to 0$ This is known as the vanishing gradients problem.
Conversely, if W_r violates the stability considerations then for large $n-k$ the expression tends to ∞ , resulting in exploding gradients problem.

1 Vanilla LSTM

The most effective solution to these training difficulties is the **Long Short Term Memory network** which we shall explore next. The essence of the **LSTM** network is to have more control over the flow of information via a gating mechanism. That is to have some non linear functions control what amount of information is passed through and what is "forgotten".
To arrive at the vanilla LSTM network, let's look at the canonical RNN cell equations.

$$s[n] = W_s s[n-1] + W_r r[n-1] + W_x x[n] + b_s$$

If we examine the equation (30), we can see that both terms, $W_s s[n-1]$ which represents the contribution from the previous cell state and the term $W_r r[n-1] + W_x x[n] + b_s$ which represents the historical information as well as the current step's input signal information, contribute equally to the state signal. We can have a gate vector ranging between 0 and 1 to selectively let the information pass. so let

$$F_s[s[n-1]] = W_s s[n-1]$$

and

$$F_u[r[n-1], x[n]] = W_r r[n-1] + W_x x[n] + b_s$$

and now

$$s[n] = g_{cs} * F_s[s[n-1]] + g_{cu} * F_u[r[n-1], x[n]]$$

where both g_{cs} and g_{cr} are vectors between 0 and 1. This helps us have more control over the flow of information.
Also we may let $W_s = I$, as long as g_{cs} is parametrizable. (Since it is also a diagonal matrix)
Now let's look at the F_u term. Let the cell's observable output value to controlled by another similar gate g_{cr}, so

$$v[n] = g_{cr} * r[n]$$

$$0 \leq g_{cr} \leq 1$$

and the input signal be controlled by g_{cx} so the term $F_u[r[n-1], x[n]$ changes to

$$F_u[v[n-1], x[n]] = W_r * v[n-1] + g_{cx} * x[n] + b_s$$

And applying the activation function tanh gives us

$$u[n] = tanh(W_r * v[n-1] + g_{cx} * x[n] + b_s)$$

which is the update signal comprising of information from the input at current step and a mix of historical signals, both of which are controlled by gates. For simplicity let $g_{cx} = 1$
The expression for $s[n]$ becomes

$$s[n] = g_{cs} * s[n-1] + g_{cu} * u[n]$$

We will refer to g_{cu} as the **update** gate, g_{cs} as the **forget** gate and g_{cr} as the **output** gate.

Now we shall derive the expressions for the gates. For this we will utilize all the signal either at the current step, or in the previous step. Then we would wrap these expression via an activation function. A good choice is the sigmoid logistic function defined as

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

Now let z_{cs}, z_{cu} and z_{cr} be the accumulation values for the gates g_{cs}, g_{cu} and g_{cr} respectively. So

$$z_{cs}[n] = W_{x_{cs}} x[n] + W_{s_{cs}} s[n-1] + W_{v_{cs}} v[n-1] + b_{cs}$$

$$z_{cu}[n] = W_{x_{cu}} x[n] + W_{s_{cu}} s[n-1] + W_{v_{cu}} v[n-1] + b_{cu}$$

$$z_{cr}[n] = W_{x_{cr}} x[n] + W_{s_{cr}} s[n] + W_{v_{cr}} v[n-1] + b_{cr}$$

Note that we have used the freshly calcuated $s[n]$ value in the z_{cr} node, becuase we have it available at the current step.
Now applying the sigmoid function will yield the expression for the gates.

$$g_{cs}[n] = \sigma(z_{cs}[n])$$

$$g_{cu}[n] = \sigma(z_{cu}[n])$$

$$g_{cr}[n] = \sigma(z_{cr}[n])$$

Next we shall look into the Vanilla LSTM forward and backward equations.

2 Long short term memory network

2.1 Overview

The key principle of the LSTM cell is centered around two main objectives: data and control of data. The data components prepare the candidate data signals, while the control components prepare the throttle signals. Hence if control signal is 0, no amount of candidate data will pass through and if it is 1, all the data shall pass, and for intermediate values, corresponding percentage of data will pass through the gate.

2.2 Control/Throttling "Gates"

The vanilla LSTM cell uses three types of gates:

1. control the amount of update candidate signal used to comprise the state signal of the cell at the present step,we shall refer it as *update gate*.

2. control the amount of state signal at the adjacent lower indexed step (n-1) used to comprise the state signal of the cell at the present step,we shall refer it as *forget gate*

3. control the amount of readout signal to release as externally observable signal at the present step,we shall refer it as *output gate*

2.3 Activation functions

For the data candidate signals we will be using the hyperbolic tangent activation function defined as

$$tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

and for the control gates we will use the sigmoid function defined as

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

2.4 Parameters of the LSTM cell

Note: We shall use the terms a_{xx} to denote the accumulation value for the respective gate g_{xx} or the respective candidate signal. This will help us to easily write out the equations.

1. for $a_{cu}[n]$, accumulation value for the update gate $g_{cu}[n]$
    - $W_{x_{cu}}$: connecting with $x[n]$
    - $W_{s_{cu}}$: connecting with $s[n-1]$
    - $W_{v_{cu}}$: connecting with $v[n-1]$
    - b_{cu}: bias term
2. for $a_{cs}[n]$, accumulation value for the forget gate $g_{cs}[n]$
    - $W_{x_{cs}}$: connecting with $x[n]$
    - $W_{s_{cs}}$: connecting with $s[n-1]$
    - $W_{v_{cs}}$: connecting with $v[n-1]$
    - b_{cs}: bias term
3. for $a_{cr}[n]$, accumulation value for the output gate $g_{cr}[n]$
    - $W_{x_{cr}}$: connecting with $x[n]$
    - $W_{s_{cr}}$: connecting with $s[n]$
    - $W_{v_{cr}}$: connecting with $v[n-1]$
    - b_{cr}: bias term
4. for $a_{du}[n]$, accumulation value for the update candidate signal $u[n]$
    - $W_{x_{du}}$: connecting with $x[n]$
    - $W_{v_{du}}$: connecting with $v[n-1]$
    - b_{du}: bias term

So, the parameter matrix Θ becomes:

$$\Theta = \begin{pmatrix} W_{x_{cu}} & W_{s_{cu}} & W_{v_{cu}} & b_{cu} \\ W_{x_{cs}} & W_{s_{cs}} & W_{v_{cs}} & b_{cs} \\ W_{x_{cr}} & W_{s_{cr}} & W_{v_{cr}} & b_{cr} \\ W_{x_{du}} & 0 & W_{v_{du}} & b_{du} \end{pmatrix}$$

2.5 Forward pass

The forward pass equations governing the LSTM cell.

$$a_{cu} = W_{x_{cu}} * x[n] + W_{s_{cu}} * s[n-1] + W_{v_{cu}} * v[n-1] + b_{cu} \tag{1}$$

$$a_{cs} = W_{x_{cs}} * x[n] + W_{s_{cs}} * s[n-1] + W_{v_{cs}} * v[n-1] + b_{cs} \tag{2}$$

$$a_{cr} = W_{x_{cr}} * x[n] + W_{s_{cr}} * s[n] + W_{v_{cr}} * v[n-1] + b_{cr} \tag{3}$$

$$a_{du} = W_{x_{du}} * x[n] + W_{v_{du}} * v[n-1] + b_{du} \tag{4}$$

$$u[n] = tanh(a_{du}) \tag{5}$$

$$g_{cu} = \sigma(a_{cu}) \tag{6}$$

$$g_{cs} = \sigma(a_{cs}) \tag{7}$$

$$g_{cr} = \sigma(a_{cr}) \tag{8}$$

$$s[n] = g_{cs} * s[n-1] + g_{cu} * u[n] \tag{9}$$

$$r[n] = tanh(s[n]) \tag{10}$$

$$v[n] = g_{cr}[n] \odot r[n] \tag{11}$$

The equations (1),(2) and (3) represent the accumulation nodes of update,forget and output gate respectively, which after applying the sigmoid function given the update, forget and output gates in (6),(7) and (8) respectively.
Equation (4) gives the accumulation value for the update candidate signal and after applying the *tanh* activation function, gives the update candidate signal in (5)
And finally, (9) gives the expression for the state signal at the current step with index n.

2.6 Backward pass

Much like the pass in the RNN network, here also we will make uses of intermediate gradients to calculate the derivatives of E with respect to Θ. This will help us in easily writing out the expressions for the gradients.
Also, as in the RNN model, the gradients as we shall see will be directly proportional to the quantity $\psi[n]$.
First let's start by calculating the derivatives of the two activation functions *tanh* and *sigmoid*.

$$tanh'(z) = 1 - tanh(z)^2$$

and

$$\sigma'(z) = \sigma(z)(1 - \sigma(z))$$

Now let's define the intermediate gradient variables as below:

$$\psi[n] = \frac{\partial E}{s[n]} \tag{1}$$

$$\rho[n] = \frac{\partial E}{r[n]} \tag{2}$$

$$\alpha_{xx}[n] = \frac{\partial E}{a_{xx}[n]} \tag{3}$$

$$\gamma_{xx}[n] = \frac{\partial E}{g_{xx}[n]} \tag{4}$$

As in the RNN system, we shall see that the gradient $\psi[n]$ holds special significance as all the other gradints will dervie thiru values using it.

$$\chi[n] = (\frac{\partial y}{\partial v[n]})^T(\frac{\partial E}{\partial y[n]}) + f_\chi[n+1] \tag{5}$$

$$\rho[n] = (\frac{\partial E}{\partial v[n]})(\frac{\partial v}{\partial r[n]}) = \chi[n]g_{cr}[n] \tag{6}$$

$$\gamma_{cr}[n] = (\frac{\partial E}{\partial v[n]})(\frac{\partial v}{\partial g_{cr}[n]}) = \chi[n]r[n] \tag{6}$$

$$\alpha_{cr}[n] = (\frac{\partial E}{\partial v[n]})(\frac{\partial v[n]}{\partial g_{cr}[n]})(\frac{\partial g_{cr}[n]}{\partial a_{cr}[n]}) = \gamma_{cr}[n]\sigma'[a_{cr}[n]] \tag{7}$$

$$\psi[n] = \rho[n](\frac{\partial r[n]}{\partial s[n]}) + (\frac{\partial a_{cr}[n]}{\partial s[n]})\alpha_{cr}[n] + f_\psi[n+1] \tag{8}$$

$$\psi[n] = \chi[n] \odot g_{cr}[n] \odot tanh'(s[n]) + W_{s_{cr}}[n] \odot \alpha_{cr}[n] + f_\psi[n+1] \tag{9}$$

$$\alpha_{cu}[n] = \frac{\partial E}{\partial s[n]} \odot \frac{\partial s[n]}{\partial g_{cu}[n]} \odot \frac{\partial g_{cu}[n]}{\partial a_{cu}[n]} = \psi[n] \odot u[n] \odot \sigma'[a_{cu}[n]] \tag{10}$$

$$\alpha_{cs}[n] = \frac{\partial E}{\partial s[n]} \odot \frac{\partial s[n]}{\partial g_{cs}[n]} \odot \frac{\partial g_{cs}[n]}{\partial a_{cs}[n]} = \psi[n] \odot s[n-1] \odot \sigma'[a_{cs}[n]] \tag{11}$$

$$\alpha_{du}[n] = \frac{\partial E}{\partial s[n]} \odot \frac{\partial s[n]}{\partial u[n]} \odot \frac{\partial u[n]}{\partial a_{du}[n]} = \psi[n] \odot g_{cu}[n] \odot tanh'[a_{du}[n]] \tag{12}$$

where $f_\psi[n+1]$ and $f_\chi[n+1]$ are the contribution from the right terms in the backward moving gradient sequences. (Contribution from further terms) Now, we shall calculate the gradients of E with respect to Θ. Calculating these intermediate gradient sequences has greatly simplified our task.

$$\frac{\partial E[n]}{\partial W_{x_{cu}}} = \frac{\partial E}{\partial a_{cu}[n]}\frac{\partial a_{cu}[n]}{\partial W_{x_{cu}}} = \alpha_{cu}[n]x^T[n] \tag{13}$$

$$\frac{\partial E[n]}{\partial W_{s_{cu}}} = \frac{\partial E}{\partial a_{cu}[n]}\frac{\partial a_{cu}[n]}{\partial W_{s_{cu}}} = \alpha_{cu}[n]s^T[n-1] \tag{14}$$

$$\frac{\partial E[n]}{\partial W_{v_{cu}}} = \frac{\partial E}{\partial a_{cu}[n]}\frac{\partial a_{cu}[n]}{\partial W_{v_{cu}}} = \alpha_{cu}[n]v^T[n-1] \tag{15}$$

$$\frac{\partial E[n]}{\partial b_{cu}} = \frac{\partial E}{\partial a_{cu}[n]}\frac{\partial a_{cu}[n]}{\partial b_{cu}} = \alpha_{cu}[n] \tag{16}$$

$$\frac{\partial E[n]}{\partial W_{x_{cs}}} = \frac{\partial E}{\partial a_{cs}[n]}\frac{\partial a_{cs}[n]}{\partial W_{x_{cs}}} = \alpha_{cs}[n]x^T[n] \tag{17}$$

$$\frac{\partial E[n]}{\partial W_{s_{cs}}} = \frac{\partial E}{\partial a_{cs}[n]}\frac{\partial a_{cs}[n]}{\partial W_{s_{cs}}} = \alpha_{cs}[n]s^T[n-1] \tag{18}$$

$$\frac{\partial E[n]}{\partial W_{v_{cs}}} = \frac{\partial E}{\partial a_{cs}[n]}\frac{\partial a_{cs}[n]}{\partial W_{v_{cs}}} = \alpha_{cs}[n]v^T[n-1] \tag{19}$$

$$\frac{\partial E[n]}{\partial b_{cs}} = \frac{\partial E}{\partial a_{cs}[n]}\frac{\partial a_{cs}[n]}{\partial b_{cs}} = \alpha_{cs}[n] \tag{20}$$

$$\frac{\partial E[n]}{\partial W_{x_{cr}}} = \frac{\partial E}{\partial a_{cr}[n]}\frac{\partial a_{cr}[n]}{\partial W_{x_{cr}}} = \alpha_{cr}[n]x^T[n] \tag{21}$$

$$\frac{\partial E[n]}{\partial W_{s_{cr}}} = \frac{\partial E}{\partial a_{cr}[n]}\frac{\partial a_{cr}[n]}{\partial W_{s_{cr}}} = \alpha_{cr}[n]s^T[n] \tag{22}$$

(*Note the change is index of* $s[n-1]$)

$$\frac{\partial E[n]}{\partial W_{v_{cr}}} = \frac{\partial E}{\partial a_{cr}[n]}\frac{\partial a_{cr}[n]}{\partial W_{v_{cr}}} = \alpha_{cr}[n]v^T[n-1] \tag{23}$$

$$\frac{\partial E[n]}{\partial b_{cr}} = \frac{\partial E}{\partial a_{cr}[n]}\frac{\partial a_{cr}[n]}{\partial b_{cr}} = \alpha_{cr}[n] \tag{24}$$

$$\frac{\partial E[n]}{\partial W_{x_{du}}} = \frac{\partial E}{\partial a_{du}[n]}\frac{\partial a_{du}[n]}{\partial W_{x_{du}}} = \alpha_{du}[n]x^T[n] \tag{25}$$

$$\frac{\partial E[n]}{\partial W_{v_{du}}} = \frac{\partial E}{\partial a_{du}[n]} \frac{\partial a_{du}[n]}{\partial W_{v_{du}}} = \alpha_{du}[n] v^T[n-1] \tag{26}$$

$$\frac{\partial E[n]}{\partial b_d u} = \frac{\partial E}{\partial a_{du}[n]} \frac{\partial a_{du}[n]}{\partial b_{du}} = \alpha_{du}[n] \tag{27}$$

We can see how expressing the gradients in terms of intermediate gradient values, greatly simplified our task and made the equations much much easier to understand and implement.
These are the gradient values for a single step with index n. Combining these values for all unrolled steps will give us the expression for the gradient of E with respect to Θ as:

$$\frac{\partial E}{\partial \Theta} = \sum_{n=0}^{K-1} \frac{\partial E}{\partial \Theta}[n]$$

These values can now be used by the gradient descent or any other optimization algorithm to obtain the parameters that minimize the cost function.

3 Conclusion

The feedback neural networks like Recurrent neural networks allow us to deal with a large variety of sequence data. Their recursive architecture makes them very robust for different types of applications like speech recognition, image/video captioning, music generation, machine translation and many many more. However, the numerical difficulties that arrive while training these networks don't allow us to make models that can capture long dependencies in the data (which is the general case in almost all of real world data)
The Long Short Term Memory network proposed by Sepp Hochreiter and Jurgen Schimdhuber in 1997, solves this issue by giving us more control over the flow of signals through gating mechanisms.
Many researchers have tried experimenting with these Vanilla LSTM models and have often produced astonishing results. For example: The Bidirectional Recurrent Neural network(BRNN), which essentially takes into account information from later values in the sequence as well, is shown to have perform very well in speech and audio recognition tasks. The newly proposed Gated Recurrent Unit or GRU's which is essentially a simplified version of the LSTM cell is also known to have produce equivalent results but with much less computational complexity. These days attention models are also being studied

for a variety of different tasksand remain an exciting research prospect.

These document was written as part of the author's summer research internship in correspondance with Professor Shayan Srinivasa Garani of Indian Institute of Science Bengaluru. For complete document: refer Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network by Alex Sherstinsky. *https://arxiv.org/pdf/1808.03314.pdf*
Thank You.