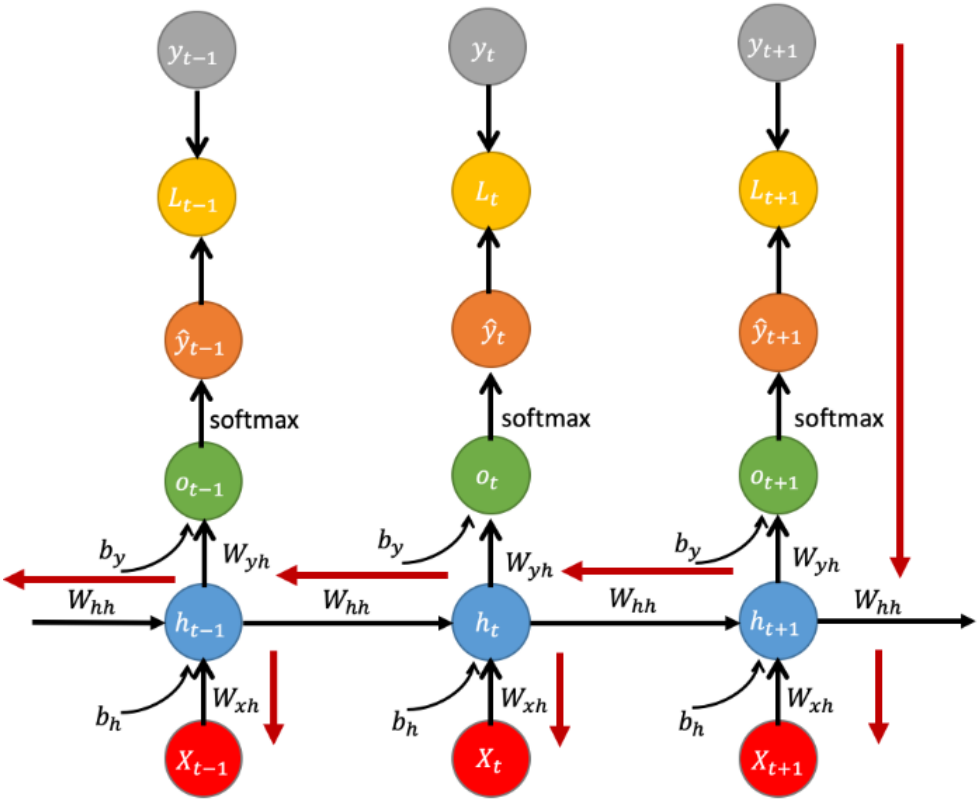
Backpropagation through time for RNN

Qazi Zarif Ul Islam

April 2, 2024



Unfolded Recurrent Neural Net
(Credit: [2])

This document is meant to be treated as supplementary to [1] and assumes that the reader understands the working principle of an RNN. If the reader requires such understanding, we recommend reading the cited work first and only then reading this document. The aim of this document is thus not to provide explanation on the RNN but rather to focus on one mathematical aspect the cited work has failed to explain with sufficient readability and that is- The gradient of the loss of RNN w.r.t. the *hidden weights* W_{hh}. In truth, we shall

derive the gradient aka differentiate the loss at a particular time instant wrt *the concatenated weight matrix* $[W_{xh}\ W_{hh}]^T$ and ultimately obtain an equation for a particularly elusive-to-understand term in the gradient (there are 3 terms in the gradient, this is one of them), the $\frac{\partial h_t}{\partial w_h}$ The concatenation of W_{xh} and W_{hh} does not make any difference from a computational perspective. [1, 2] and so, we shall denote it as w_h. Lastly, no distinction is made in terms of notation for vectors, scalars or matrices. This simplifies the derivation process however, we shall produce a work in the future that incorporates vector-matrix notation.

Let's start

The emprical loss of the neural network is,

$$\hat{L} = \frac{1}{T}\sum_{t=1}^{T} \ell(y_t, d_t) = \frac{1}{T}(l_1 + l_2 + ... + l_t ... + l_T) \tag{1}$$

For an RNN, the system is defined by,

$$h_t = f(X_t, h_{t-1}) = \phi_h(W_{xh} \cdot X_t + W_{hh} \cdot h_{t-1} + b_h) \tag{2}$$

$$\hat{o}_t = f_o(h_t) = \phi_o(W_{hy} \cdot h_t + b_y) \tag{3}$$

Let $w_h = [W_{xh}\ W_{hh}]^T$.

Thus,

$$\frac{\partial \hat{L}}{\partial w_h} = \frac{1}{T}(\frac{\partial l_1}{\partial w_h} + \frac{\partial l_2}{\partial w_h} + ... + \frac{\partial l_t}{\partial w_h} ... + \frac{\partial l_T}{\partial w_h}) \tag{4}$$

Now the loss at a particular time instant t follows the chain rule of derivatives.

$$\frac{\partial l_t}{\partial w_h} = \frac{\partial l_t}{\partial o_t}\frac{\partial o_t}{\partial h_t}\frac{\partial h_t}{\partial w_h} \tag{5}$$

But h_t is a function of h_{t-1} and w_h as well (besides X_t). Furthermore, h_{t-1} is again a function of w_h. Thus, by the multivariable chain rule,

If $h_t = f_1(h_{t-1}, w_h), h_{t-1} = g_1(h_{t-2}, w_h), w_h = h_1(w_h)$ [1] (Read as "The function f_1 takes h_{t-1} and w_h as input, g_1 takes h_{t-2} and w_h as input and h_1 takes w_h as input.)

$$\frac{\partial h_t}{\partial w_h} = \frac{\partial f_1}{\partial w_h} + \frac{\partial f_1}{\partial h_{t-1}}\frac{\partial h_{t-1}}{\partial w_h} \tag{6}$$

Similarly, if $h_{t-1} = f_2(h_{t-2}, w_h), h_{t-2} = g_2(h_{t-3}, w_h), w_h = h_2(w_h)$

[1] (Note: h_1 (the function, *not* the hidden state) is merely the identity function. We've formulated it so just to be consistent with the formulation of "case 1" in [3]. Eqn 6 would still be valid even if we hadn't defined it as a separate function.

$$\frac{\partial h_{t-1}}{\partial w_h} = \frac{\partial f_2}{\partial w_h} + \frac{\partial f_2}{\partial h_{t-2}} \frac{\partial h_{t-2}}{\partial w_h} \tag{7}$$

Similarly, if $h_{t-2} = f_3(h_{t-3}, w_h), h_{t-3} = g_3(h_{t-4}, w_h), w_h = h_3(w_h)$

$$\frac{\partial h_{t-2}}{\partial w_h} = \frac{\partial f_3}{\partial w_h} + \frac{\partial f_3}{\partial h_{t-3}} \frac{\partial h_{t-3}}{\partial w_h} \tag{8}$$

We can find similar expressions for time steps further back h_{t-3}, h_{t-4} all the way to h_1. Now, from equation 6 and equation 7,

$$\frac{\partial h_t}{\partial w_h} = \frac{\partial f_1}{\partial w_h} + \frac{\partial f_1}{\partial h_{t-1}} (\frac{\partial f_2}{\partial w_h} + \frac{\partial f_2}{\partial h_{t-2}} \frac{\partial h_{t-2}}{\partial w_h}) \qquad \text{(we won't expand } \frac{\partial h_{t-2}}{\partial w_h}) \tag{9}$$

$$= \frac{\partial f_1}{\partial w_h} + \frac{\partial f_1}{\partial h_{t-1}} \frac{\partial f_2}{\partial w_h} + \frac{\partial f_1}{\partial h_{t-1}} \frac{\partial f_2}{\partial h_{t-2}} \frac{\partial h_{t-2}}{\partial w_h} \tag{10}$$

$$= \frac{\partial h_t}{\partial w_h} + \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial w_h} + \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial h_{t-2}} \frac{\partial h_{t-2}}{\partial w_h} \qquad (f_1 = h_t,\ f_2 = h_{t-1}) \tag{11}$$

which is equivalent to,

$$\boxed{\frac{\partial h_t}{\partial w_h} = \frac{\partial h_t}{\partial w_h} + \sum_{i=1}^{t-1} \prod_{j-1=i}^{t} (\frac{\partial h_j}{\partial h_{j-1}}) \frac{\partial h_i}{\partial w_h}}$$

Which is the same as equation 9.7.7 in [1].

How do we arrive at the final form?: In equation 11, if we expand $\frac{\partial h_{t-2}}{\partial w_h}$ (and subsequently expand more terms that recursively emerge), observe that every term's last (rightmost) term is partial of h_i as i **progresses** from the beginning and ends at t as we move from right to left in the summation. Hence, i is our index of summation (The term that "progresses" should be what we sum over). Now, every term is again a cascade of products, decreasing in number of multipliers as we move from right to left in the summation. Every multiplier term is a partial of h_j wrt h_{j-1}. (Note that it is tempting to set our index of product as i, as it was for the summation, but doing this would not achieve the final form)

References

[1] *Backpropagation Through Time.* `https://d2l.ai/chapter_recurrent-neural-networks/bptt.html`. Accessed: 2024-03-24.

[2] Murat Karakaya. *Backpropagation Through Time of RNN.* `https://mmuratarat.github.io/2019-02-07/bptt-of-rnn`. Accessed: 2024-03-24. 2019.

[3] Paul's online notes. *Chain rule*. https://tutorial.math.lamar.edu/classes/calciii/chainrule.aspx. Accessed: 2024-03-24. 2018.