

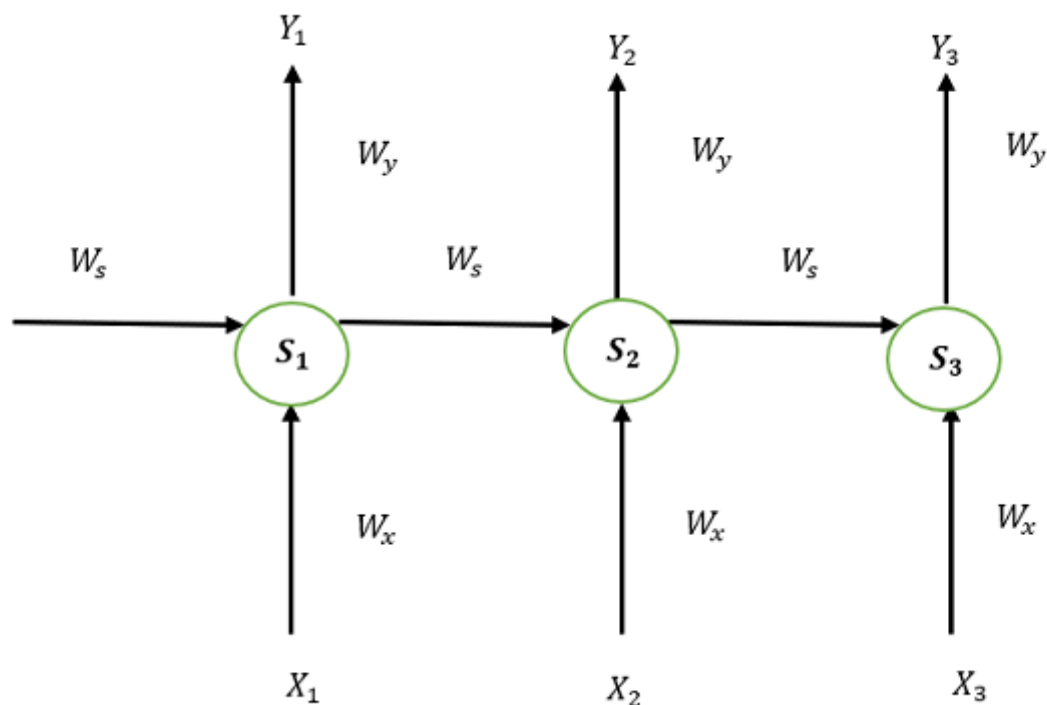
## Back Propagation through time – RNN

### Introduction:

Recurrent Neural Networks are those networks that deal with sequential data. They can predict outputs based on not only current inputs but also considering the inputs that were generated prior to it. The output of the present depends on the output of the present and the memory element (which includes the previous inputs).

To train these networks, we make use of traditional backpropagation with an added twist. We don't train the system on the exact time "t". We train it according to a particular time "t" as well as everything that has occurred prior to time "t" like the following: t-1, t-2, t-3.

Take a look at the following illustration of the RNN:



$S_1$ ,  $S_2$ , and  $S_3$  are the states that are hidden or memory units at the time of  $t_1$ ,  $t_2$ , and  $t_3$ , respectively, while  $W_s$  represents the matrix of weight that goes with it.

$X_1$ ,  $X_2$ , and  $X_3$  are the inputs for the time that is  $t_1$ ,  $t_2$ , and  $t_3$ , respectively, while  $W_x$  represents the weighted matrix that goes with it.

The numbers  $Y_1$ ,  $Y_2$ , and  $Y_3$  are the outputs of  $t_1$ ,  $t_2$ , and  $t_3$ , respectively as well as  $W_y$ , the weighted matrix that goes with it.

For any time,  $t$ , we have the following two equations:

$$S_t = g_1(W_x x_t + W_s S_{t-1})$$

$$Y_t = g_2(W_y S_t)$$

where  $g_1$  and  $g_2$  are activation functions.

We will now perform the back propagation at time  $t = 3$ .

Let the error function be:

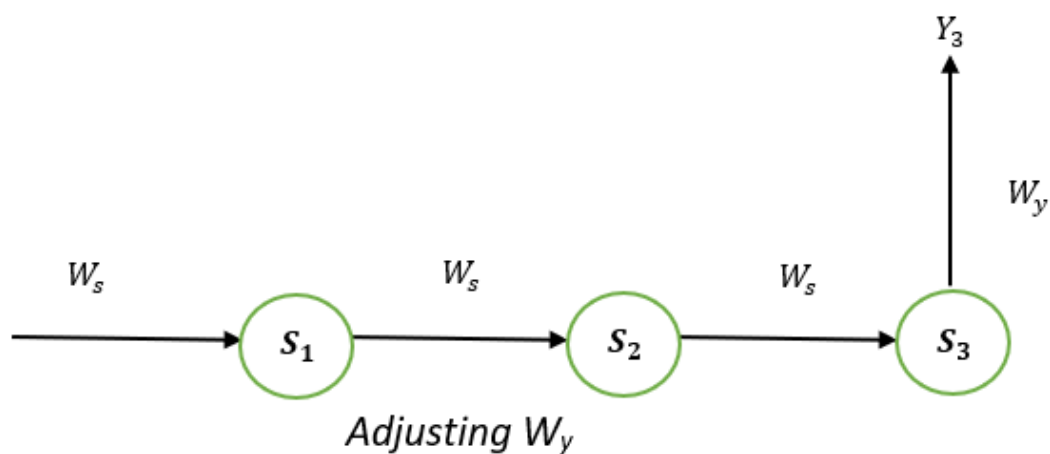
$$E_t = (d_t - Y_t)^2$$

Here, we employ the squared error, in which  $D_3$  is the desired output at a time  $t = 3$ .

In order to do backpropagation, it is necessary to change the weights that are associated with inputs, memory units, and outputs.

## Adjusting $W_y$

To better understand, we can look at the following image:



## Formula:

$$\frac{\partial E_3}{\partial W_y} = \frac{\partial E_3}{\partial Y_3} \cdot \frac{\partial Y_3}{\partial W_y}$$

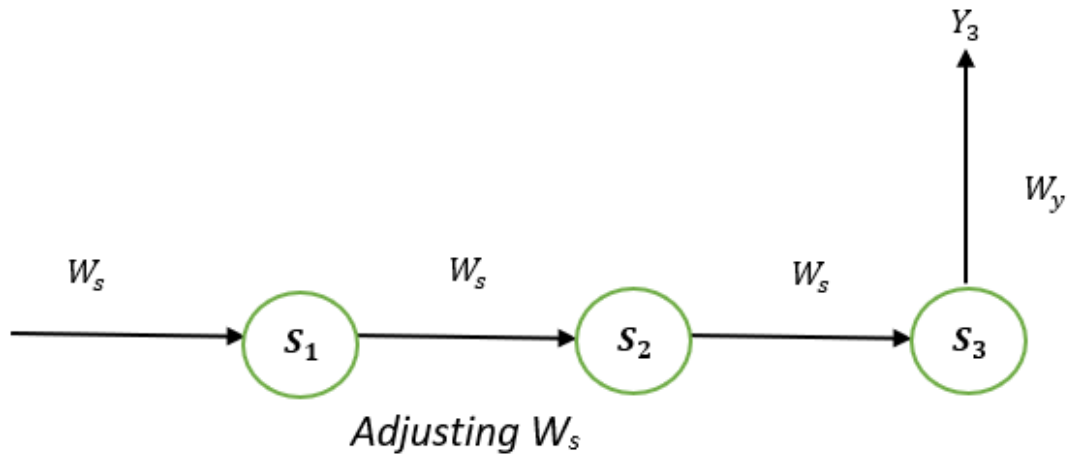
Explanation:

E3 is a function of Y3. Hence, we differentiate E3 with respect to Y3.

Y3 is a function of W3. Hence, we differentiate Y3 with respect to W3.

## Adjusting Ws

To better understand, we can look at the following image:



## Formula:

$$\frac{\partial E_3}{\partial W_y} = \left( \frac{\partial E_3}{\partial Y_3} \cdot \frac{\partial Y_3}{\partial S_3} \cdot \frac{\partial S_3}{\partial W_y} \right) + \left( \frac{\partial E_3}{\partial Y_3} \cdot \frac{\partial Y_3}{\partial S_3} \cdot \frac{\partial S_3}{\partial S_2} \cdot \frac{\partial S_2}{\partial W_s} \right) + \left( \frac{\partial E_3}{\partial Y_3} \cdot \frac{\partial Y_3}{\partial S_3} \cdot \frac{\partial S_3}{\partial S_2} \cdot \frac{\partial S_2}{\partial S_1} \cdot \frac{\partial S_1}{\partial W_s} \right)$$

Explanation:

E3 is a function of the Y3. Therefore, we distinguish the E3 with respect to Y3. Y3 is a function of the S3. Therefore, we differentiate between Y3 with respect to S3.

S3 is an element in the Ws. Therefore, we distinguish between S3 with respect to Ws.

But it's not enough to stop at this, therefore we have to think about the previous steps in time. We must also differentiate (partially) the error function in relation to the memory units S2 and S1, considering the weight matrix Ws.

It is essential to be aware that a memory unit, such as St, is the result of its predecessor memory unit, St-1.

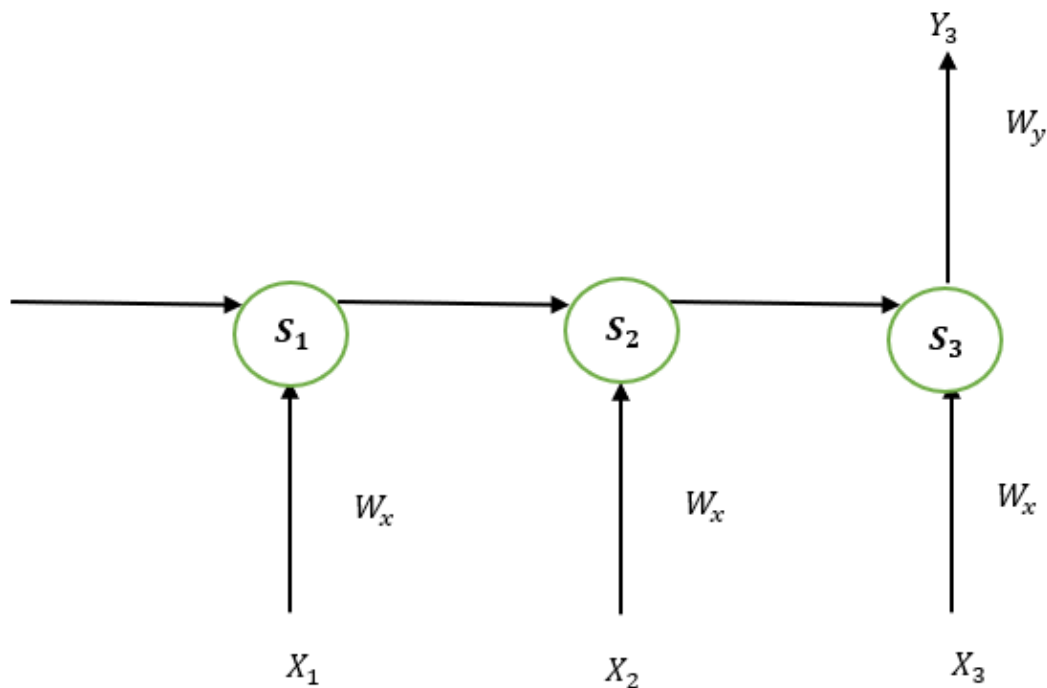
Therefore, we distinguish S3 from S2 and S2 from S1.

In general, we can describe this formula in terms of:

$$\frac{\partial E_N}{\partial W_s} = \sum_{i=1}^N \frac{\partial E_N}{\partial Y_N} \cdot \frac{\partial Y_N}{\partial S_i} \cdot \frac{\partial S_i}{\partial W_s}$$

### Adjusting WX:

To better understand, we can look at the following image:



*Adjusting  $W_x$*

### Formula:

$$\frac{\partial E_3}{\partial W_x} = \left( \frac{\partial E_3}{\partial Y_3} \cdot \frac{\partial Y_3}{\partial S_3} \cdot \frac{\partial S_3}{\partial W_x} \right) + \left( \frac{\partial E_3}{\partial Y_3} \cdot \frac{\partial Y_3}{\partial S_3} \cdot \frac{\partial S_3}{\partial S_2} \cdot \frac{\partial S_2}{\partial W_x} \right) + \left( \frac{\partial E_3}{\partial Y_3} \cdot \frac{\partial Y_3}{\partial S_3} \cdot \frac{\partial S_3}{\partial S_2} \cdot \frac{\partial S_2}{\partial S_1} \cdot \frac{\partial S_1}{\partial W_x} \right)$$

Explanation:

E3 is an effect in the Y3. Therefore, we distinguish the E3 with respect to Y3. Y3 is an outcome that is a function of the S3. Therefore, we distinguish the Y3 with respect to S3.

S3 is an element in the WX. Thus, we can distinguish the S3 with respect to WX.

We can't just stop at this, and therefore we also need to think about the preceding time steps. Therefore, we separate (partially) the error function in relation to the memory unit S2 and S1, considering the WX weighting matrix.

In general, we can define this formula in terms of:

$$\frac{\partial E_N}{\partial W_S} = \sum_{i=1}^N \frac{\partial E_N}{\partial Y_N} \cdot \frac{\partial Y_N}{\partial S_i} \cdot \frac{\partial S_i}{\partial W_X}$$

## **Limitations:**

This technique that uses the back Propagation over time (BPTT) is a method that can be employed for a limited amount of time intervals, like 8 or 10. If we continue to backpropagate and the gradient gets too small. This is known as the "Vanishing gradient" problem. This is because the value of information diminishes geometrically with time. Therefore, if the number of time steps is greater than 10 (Let's say), the data is effectively discarded.