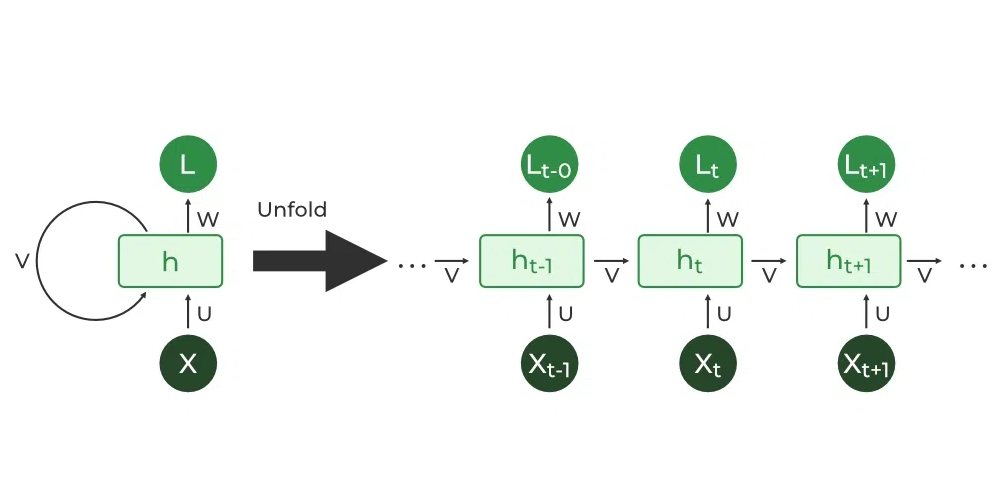
**DAY 11**

**30.06.2023**

**Recurrent Neural Networks**

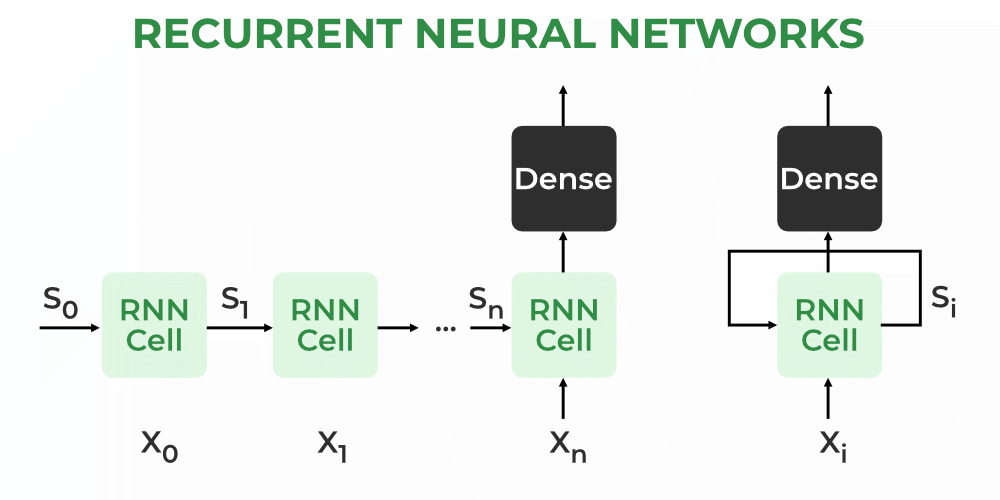
Recurrent Neural Network (RNN) is a type of [**Neural Network**](https://www.geeksforgeeks.org/tag/neural-network/) where the output from the previous step is fed as input to the current step. In traditional neural networks, all the inputs and outputs are independent of each other, but in cases when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words.



Thus, RNN came into existence, which solved this issue with the help of a Hidden Layer. The main and most important feature of RNN is its **Hidden state**, which remembers some information about a sequence. The state is also referred to as *Memory State*since it remembers the previous input to the network. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output. This reduces the complexity of parameters, unlike other neural networks.

***Architecture***

RNNs have the same input and output architecture as any other deep neural architecture. However, differences arise in the way information flows from input to output. Unlike Deep neural networks where we have different weight matrices for each Dense network in RNN, the weight across the network remains the same.



It calculates state hidden state Hifor every input Xi.By using the following formulas:

***h = σ (UX + Wh-1 + B)***

***Y = O(Vh +C) Hence***

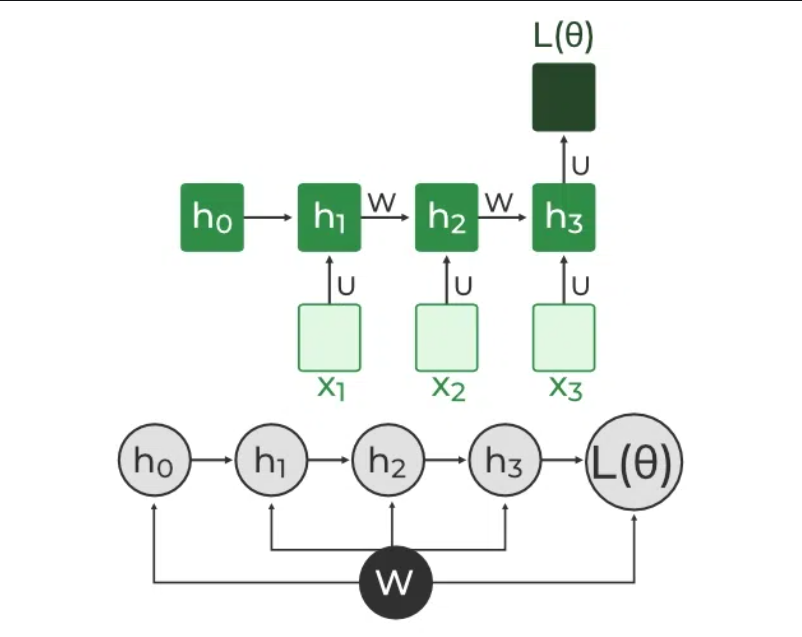
***Y = f(X, h, W, U, V, B, C)***

Here S is the state matrix which has element si as the state of the network at timestep i. The parameters in the network are W, U, V, c, b which are shared across timestep.

***Training***

* A single-time step of the input is provided to the network.
* Then calculate its current state using a set of current input and the previous state.
* The current ht becomes ht-1 for the next time step.
* One can go as many time steps according to the problem and join the information from all the previous states.
* Once all the time steps are completed the final current state is used to calculate the output.
* The output is then compared to the actual output i.e the target output and the error is generated.
* The error is then back-propagated to the network to update the weights and hence the network (RNN) is trained using [Backpropagation](https://www.geeksforgeeks.org/ml-back-propagation-through-time/) through time.







***Backpropagation Through Time (BPTT) In RNN***

***Advantages***

* An RNN remembers each and every piece of information through time. It is useful in time series prediction only because of the feature to remember previous inputs as well. This is called Long Short-Term Memory.
* Recurrent neural networks are even used with convolutional layers to extend the effective pixel neighbourhood.

***Disadvantages***

* Gradient vanishing and exploding problems.
* Training an RNN is a very difficult task.
* It cannot process very long sequences if using tanh or ReLU as an activation function.

***Applications***

* Language Modelling and Generating Text
* Speech Recognition
* Machine Translation
* Image Recognition, Face detection
* Time series Forecasting

**References**

[https://www.ibm.com/topics/recurrent-neural-networks](https://www.ibm.com/topics/recurrent-neural-networks#:~:text=A%20recurrent%20neural%20network%20(RNN,data%20or%20time%20series%20data)

<https://www.analyticsvidhya.com/blog/2022/03/a-brief-overview-of-recurrent-neural-networks-rnn/>

[https://towardsdatascience.com/recurrent-neural-networks](https://towardsdatascience.com/recurrent-neural-networks-rnns-3f06d7653a85)

<https://builtin.com/data-science/recurrent-neural-networks-and-lstm>