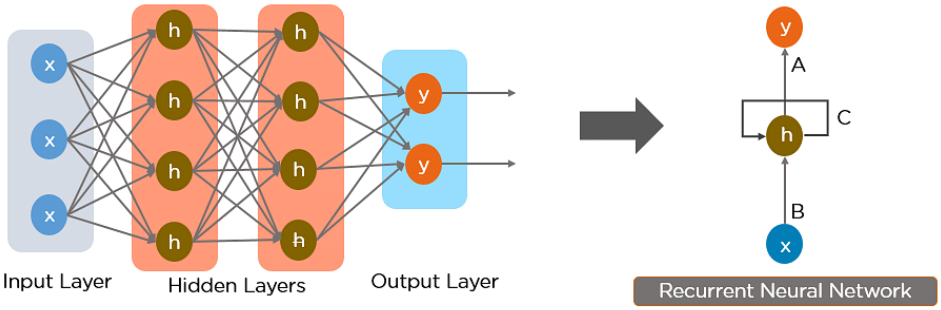
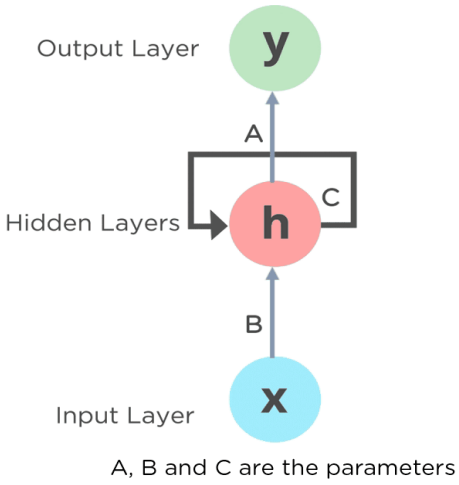
What Is a Recurrent Neural Network (RNN)?

RNN works on the principle of saving the output of a particular layer and feeding this back to the input in order to predict the output of the layer.



The nodes in different layers of the neural network are compressed to form a single layer of recurrent neural networks. A, B, and C are the parameters of the network.



Why Recurrent Neural Networks?

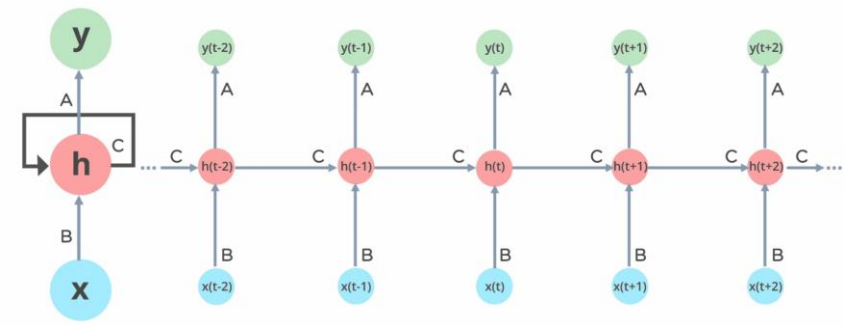
RNN were created because there were a few issues in the feed-forward neural network:

- Cannot handle sequential data
- Considers only the current input
- Cannot memorize previous inputs

The solution to these issues is the RNN. An RNN can handle sequential data, accepting the current input data, and previously received inputs. RNNs can memorize previous inputs due to their internal memory.

How Does Recurrent Neural Networks Work?

In Recurrent Neural networks, the information cycles through a loop to the middle hidden layer.



The input layer ‘x’ takes in the input to the neural network and processes it and passes it onto the middle layer.

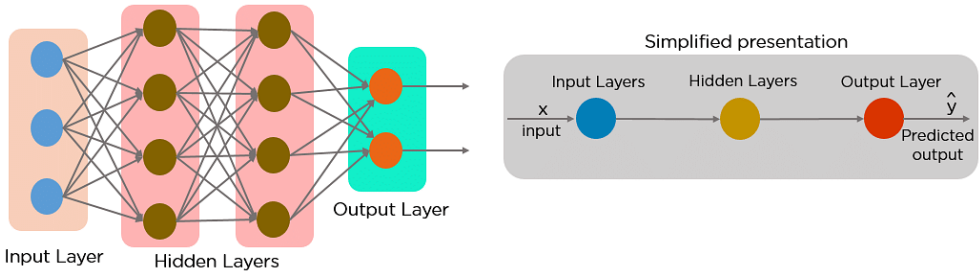
The middle layer ‘h’ can consist of multiple hidden layers, each with its own activation functions and weights and biases. If you have a neural network where the various parameters of different hidden layers are not affected by the previous layer,

ie: the neural network does not have memory, then you can use a recurrent neural network.

The Recurrent Neural Network will standardize the different activation functions and weights and biases so that each hidden layer has the same parameters. Then, instead of creating multiple hidden layers, it will create one and loop over it as many times as required.

Feed-Forward Neural Networks vs Recurrent Neural Networks

A feed-forward neural network allows information to flow only in the forward direction, from the input nodes, through the hidden layers, and to the output nodes. There are no cycles or loops in the network.



In a feed-forward neural network, the decisions are based on the current input. It doesn't memorize the past data, and there's no future scope. Feed-forward neural networks are used in general regression and classification problems.

Applications of Recurrent Neural Networks

Image Captioning

RNNs are used to caption an image by analyzing the activities present.

Time Series Prediction

Any time series problem, like predicting the prices of stocks in a particular month, can be solved using an RNN.

Natural Language Processing

Text mining and Sentiment analysis can be carried out using an RNN for Natural Language Processing (NLP).

Machine Translation

Given an input in one language, RNNs can be used to translate the input into different languages as output.

Machine Translation

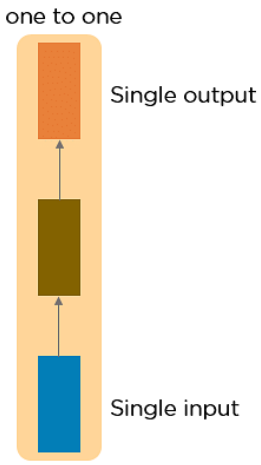
Types of Recurrent Neural Networks

There are four types of Recurrent Neural Networks:

1. One to One
2. One to Many
3. Many to One
4. Many to Many

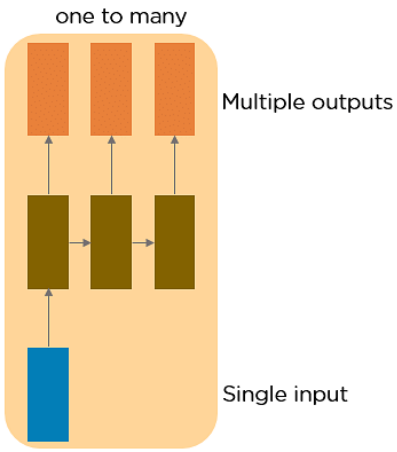
One to One RNN

This type of neural network is known as the Vanilla Neural Network. It's used for general machine learning problems, which has a single input and a single output.



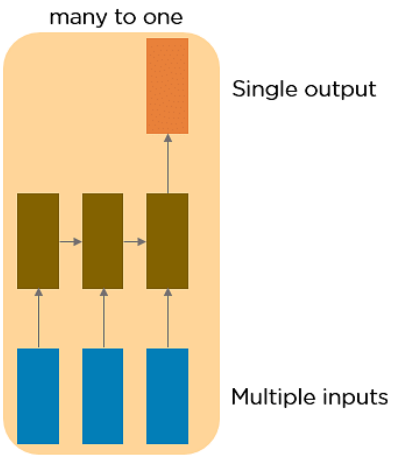
One to Many RNN

This type of neural network has a single input and multiple outputs. An example of this is the image caption.



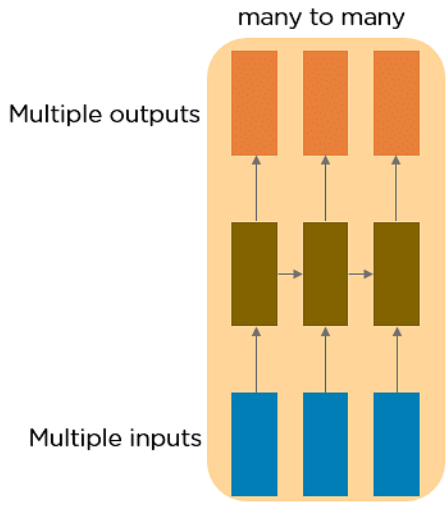
Many to One RNN

This RNN takes a sequence of inputs and generates a single output. Sentiment analysis is a good example of this kind of network where a given sentence can be classified as expressing positive or negative sentiments.



Many to Many RNN

This RNN takes a sequence of inputs and generates a sequence of outputs. Machine translation is one of the examples.

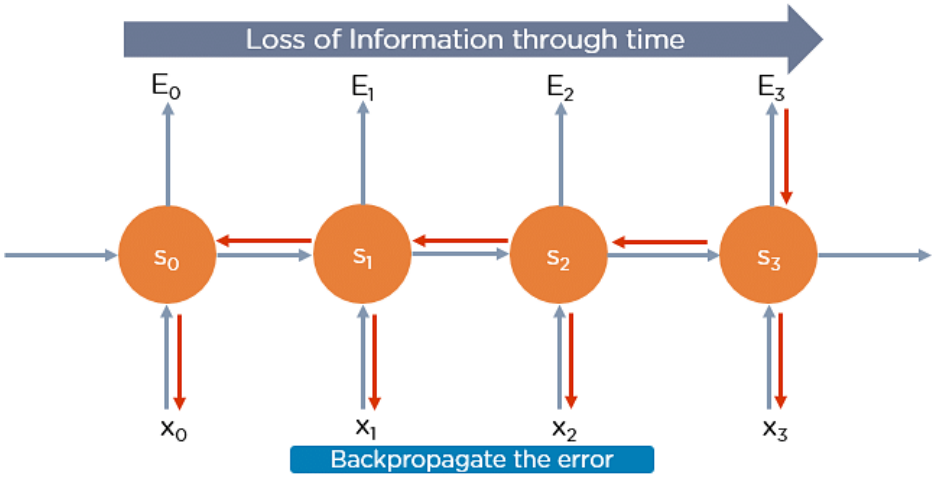


Two Issues of Standard RNNs

1. Vanishing Gradient Problem

Recurrent Neural Networks enable you to model time-dependent and sequential data problems, such as stock market prediction, machine translation, and text generation. You will find, however, RNN is hard to train because of the gradient problem.

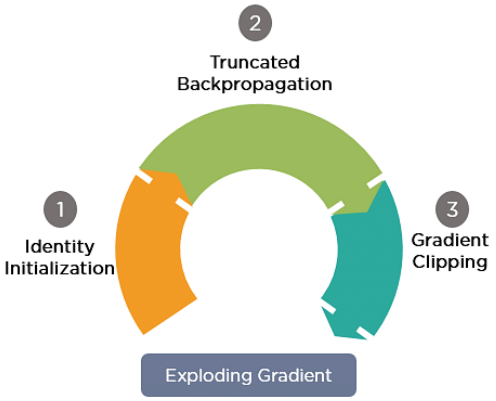
RNNs suffer from the problem of vanishing gradients. The gradients carry information used in the RNN, and when the gradient becomes too small, the parameter updates become insignificant. This makes the learning of long data sequences difficult.

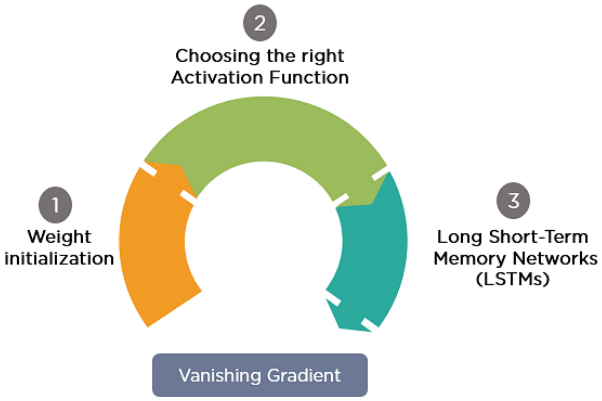


2. Exploding Gradient Problem

While training a neural network, if the slope tends to grow exponentially instead of decaying, this is called an Exploding Gradient. This problem arises when large error gradients accumulate, resulting in very large updates to the neural network model weights during the training process.

Long training time, poor performance, and bad accuracy are the major issues in gradient problems.

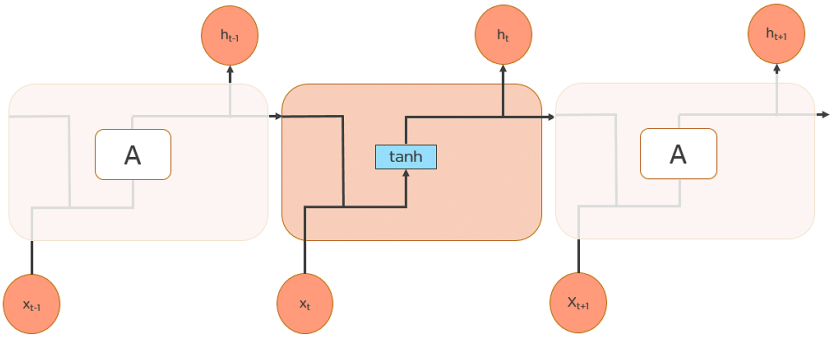




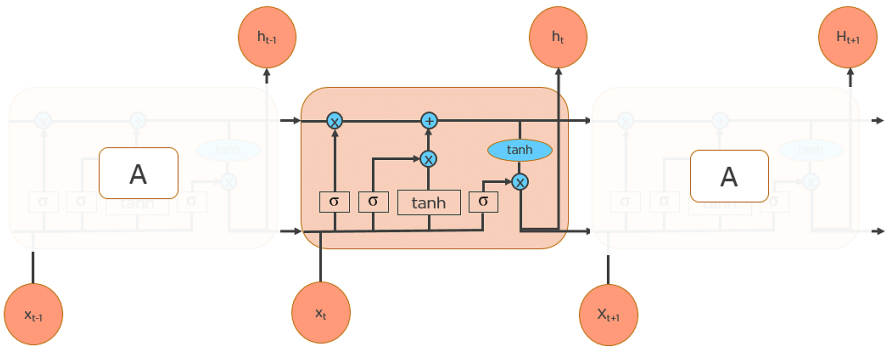
Long Short-Term Memory Networks

LSTMs are a special kind of RNN — capable of learning long-term dependencies by remembering information for long periods is the default behavior.

All RNN are in the form of a chain of repeating modules of a neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.



LSTMs also have a chain-like structure, but the repeating module is a bit different structure. Instead of having a single neural network layer, four interacting layers are communicating extraordinarily.


h_{t-1}
h_t
H_{t+1}
A
A
σ
σ
tanh
σ
tanh
x_{t-1}
x_t
X_{t+1}