Artificial Intelligence Masterclass

Recurrent Neural Network (RNN)

H.M. Samadhi Chathuranga Rathnayake

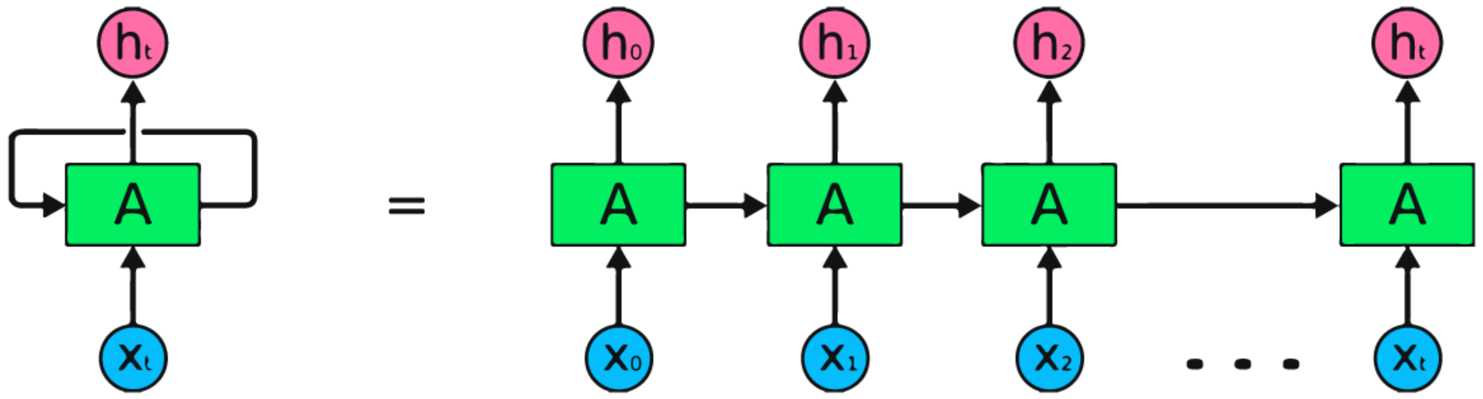
M.Sc in CS (SU), PG.Dip in SML (Othm), PG.Dip in HRM (LRN), B.Sc (Hons) in IS (UOC), B.Eng (Hons) in SE (LMU), P. Dip EP & SBO (ABE), Dip SE, Dip IT, Dip IT & E-Com, Dip B.Mgt, Dip HRM, Dip Eng

Introduction to RNN

- RNN remembers past inputs due to an internal memory which is useful for predicting stock prices, generating text, transcriptions, and machine translation.
- The inputs and outputs of a typical neural network are independent of one another, however in an RNN, the outcome is reliant on earlier elements in the sequence.
- Recurrent networks also share parameters across each layer of the network.
- In Feedforward Multi Layer Perceptron networks, there are different weights across each node.
- Whereas RNN shares the same weights within each layer of the network and during gradient descent, the weights and basis are adjusted individually to reduce the loss.

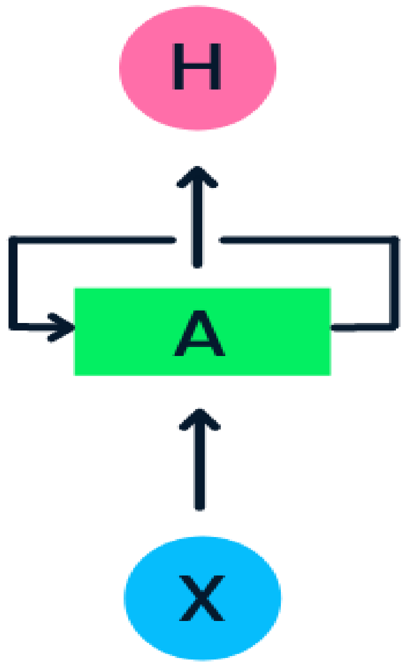
Introduction to RNN

- If we are forecasting stock prices using simple data [45,56,45,49,50,...], each input from X0 to Xt will contain a past value.
- For example, X0 will have 45, X1 will have 56, and these values are used to predict the next number in a sequence.



Process of RNN

- In RNN, the information cycles through the loop, so the output is determined by the current input and previously received inputs.



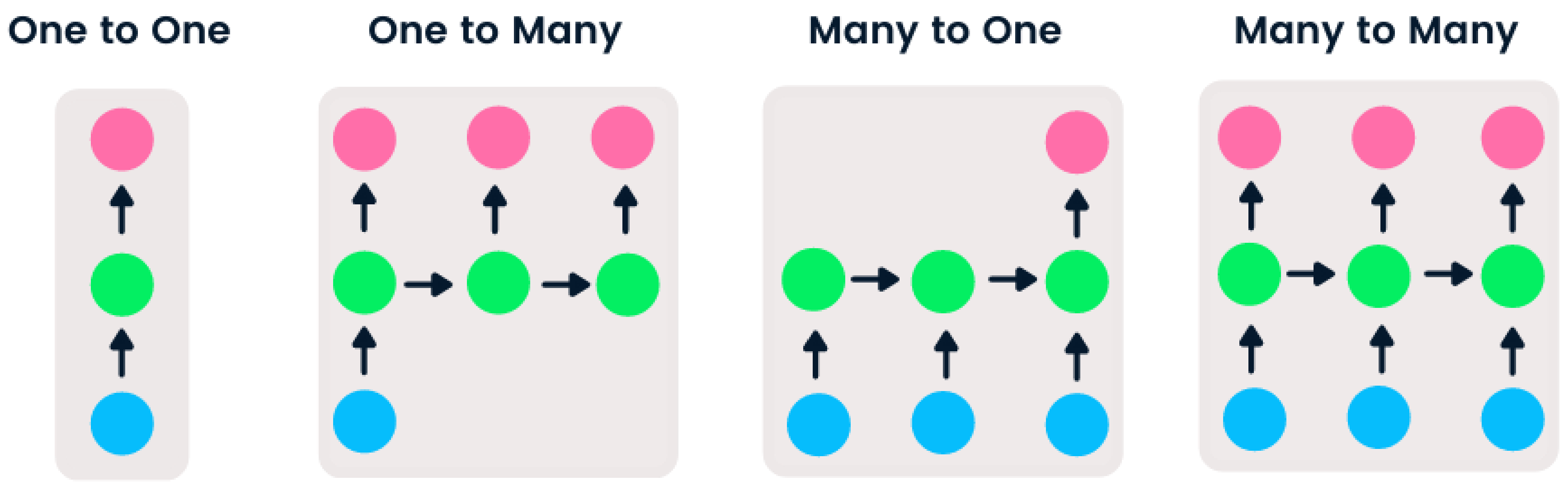
Process of RNN

- The input layer X processes the initial input and passes it to the middle layer A.
- The middle layer consists of multiple hidden layers, each with its activation functions, weights, and biases.
- These parameters are standardized across the hidden layer so that instead of creating multiple hidden layers, it will create one and loop it over.
- Instead of using traditional backpropagation, recurrent neural networks use backpropagation through time (BPTT) algorithms to determine the gradient.
- In backpropagation, the model adjusts the parameter by calculating errors from the output to the input layer.
- BPTT sums the error at each time step as RNN shares parameters across each layer.

Types of RNN

- There are four types of RNN based on different lengths of inputs and outputs.
  - **One-to-one** is a simple neural network. It is commonly used for machine learning problems that have a single input and output.
  - **One-to-many** has a single input and multiple outputs. This is used for generating image captions.
  - **Many-to-one** takes a sequence of multiple inputs and predicts a single output. It is popular in sentiment classification, where the input is text and the output is a category.
  - **Many-to-many** takes multiple inputs and outputs. The most common application is machine translation.

Types of RNN



Differences Between CNN and RNN

- CNN is applicable for sparse data like images. RNN is applicable for time series and sequential data.
- While training the model, CNN uses a simple backpropagation and RNN uses backpropagation through time to calculate the loss.
- RNN can have no restriction in length of inputs and outputs, but CNN has finite inputs and finite outputs.
- CNN has a feedforward network and RNN works on loops to handle sequential data.
- CNN can also be used for video and image processing. RNN is primarily used for speech and text analysis.

Limitations of RNN

- Simple RNN models usually run into two major issues. These issues are related to gradient, which is the slope of the loss function along with the error function.

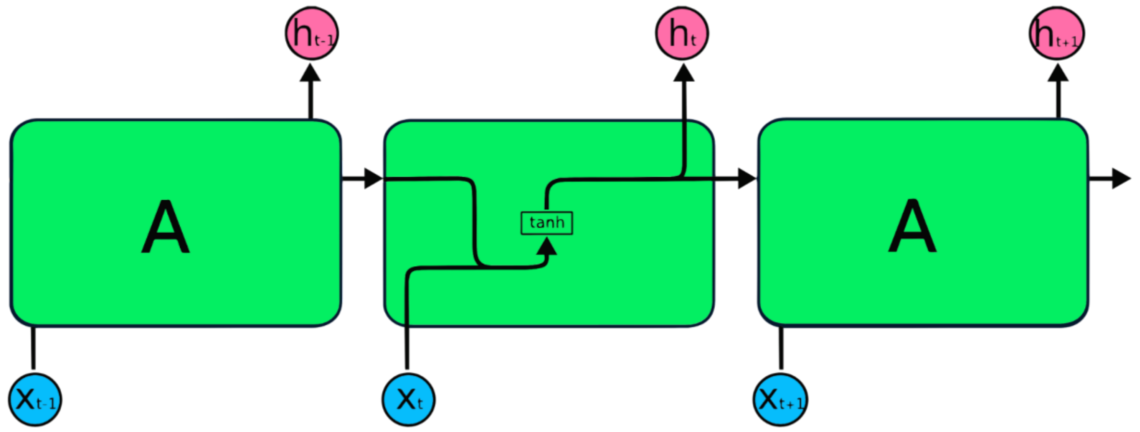
  - **Vanishing Gradient problem** occurs when the gradient becomes so small that updating parameters becomes insignificant; eventually the algorithm stops learning.
  - **Exploding Gradient problem** occurs when the gradient becomes too large, which makes the model unstable. In this case, larger error gradients accumulate, and the model weights become too large. This issue can cause longer training times and poor model performance.

Limitations of RNN

- The simple solution to these issues is to reduce the number of hidden layers within the neural network, which will reduce some complexity in RNNs.
- These issues can also be solved by using advanced RNN architectures such as **LSTM**

Long Short Term Memory (LSTM)

- The simple RNN repeating modules have a basic structure with a single tanh layer.
- RNN simple structure suffers from short memory, where it struggles to retain previous time step information in larger sequential data.

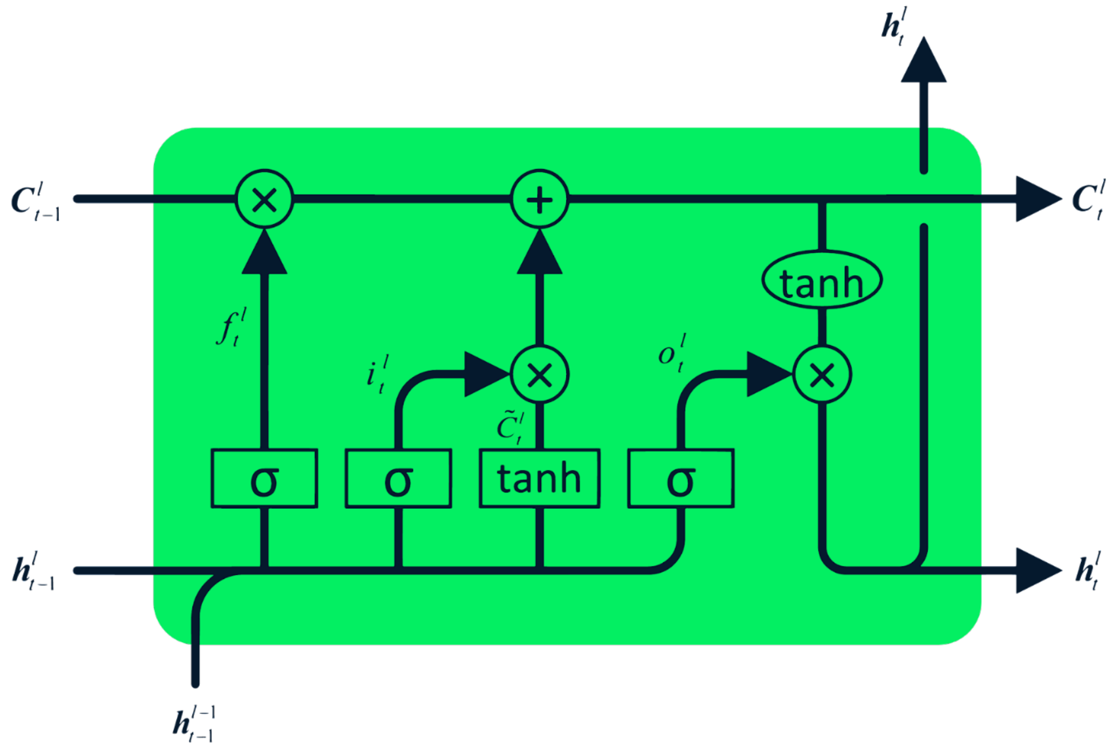


- These problems can easily be solved by long short term memory (LSTM) as it is capable of remembering long periods of information.

Long Short Term Memory (LSTM)

- The Long Short Term Memory (LSTM) is the advanced type of RNN, which was designed to prevent both decaying and exploding gradient problems.
- Just like RNN, LSTM has repeating modules, but the structure is different.
- Instead of having a single layer of tanh, LSTM has four interacting layers that communicate with each other.
- This four-layered structure helps LSTM retain long-term memory and can be used in several sequential problems including machine translation, speech synthesis, speech recognition, and handwriting recognition.

Long Short Term Memory (LSTM)



h_t^l
C_{t-1}^l
C_t^l
tanh
f_t^l
i_t^l
o_t^l
$\tilde{C}_t^l$
σ
σ
tanh
σ
h_{t-1}^l
h_t^l
h_{t-1}^{l-1}