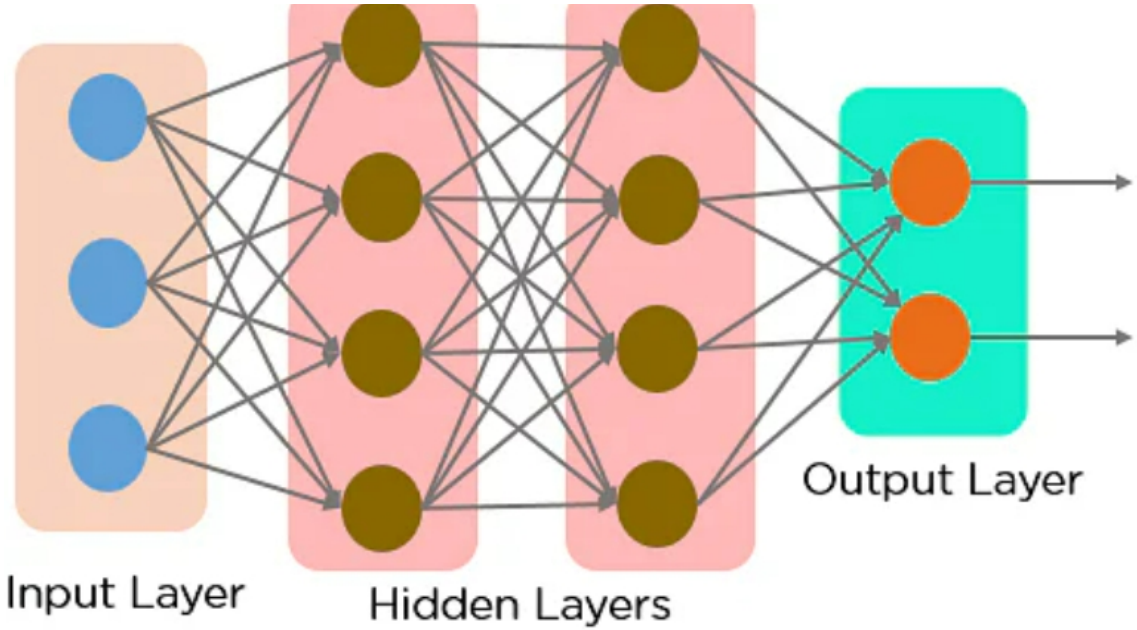
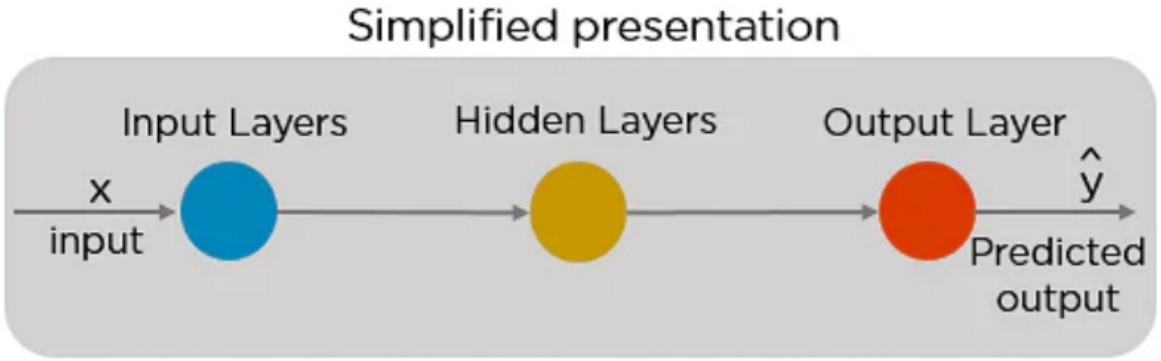
Recurrent Neural Networks

Feed-Forward Neural Networks





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

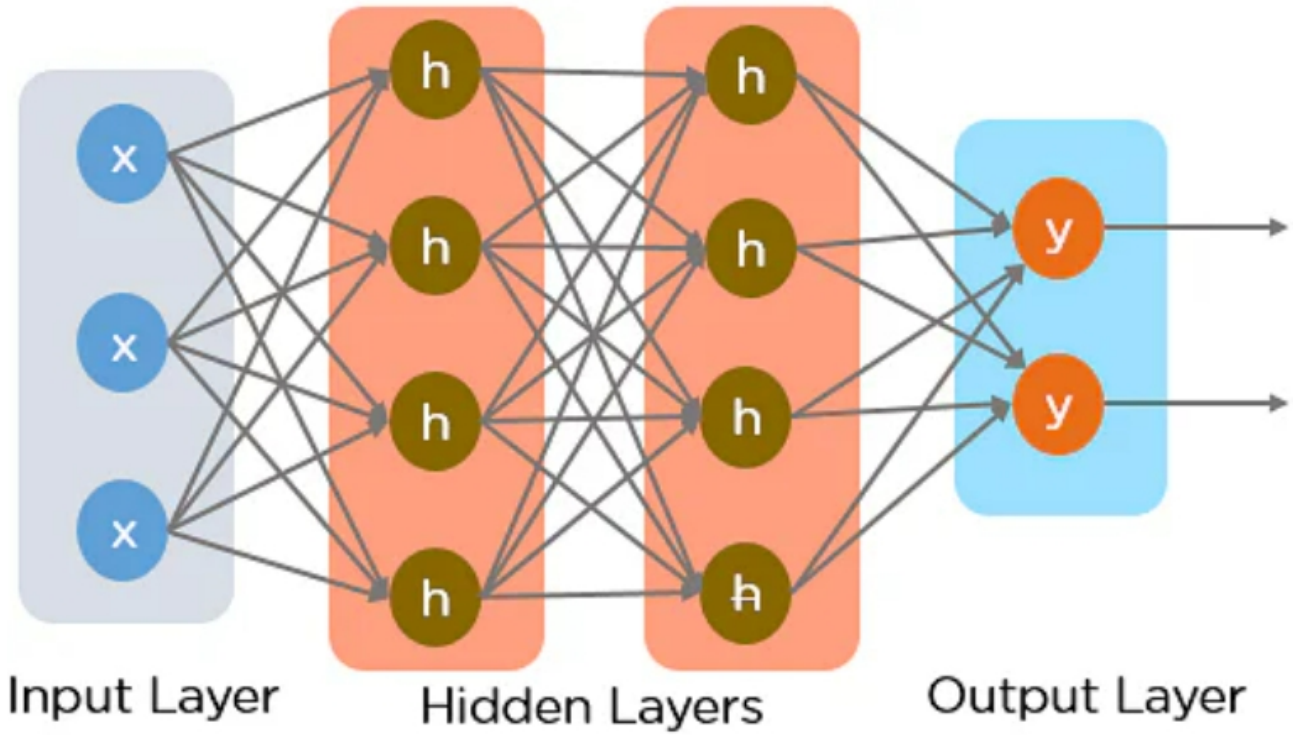
issues in the feed-forward neural network:

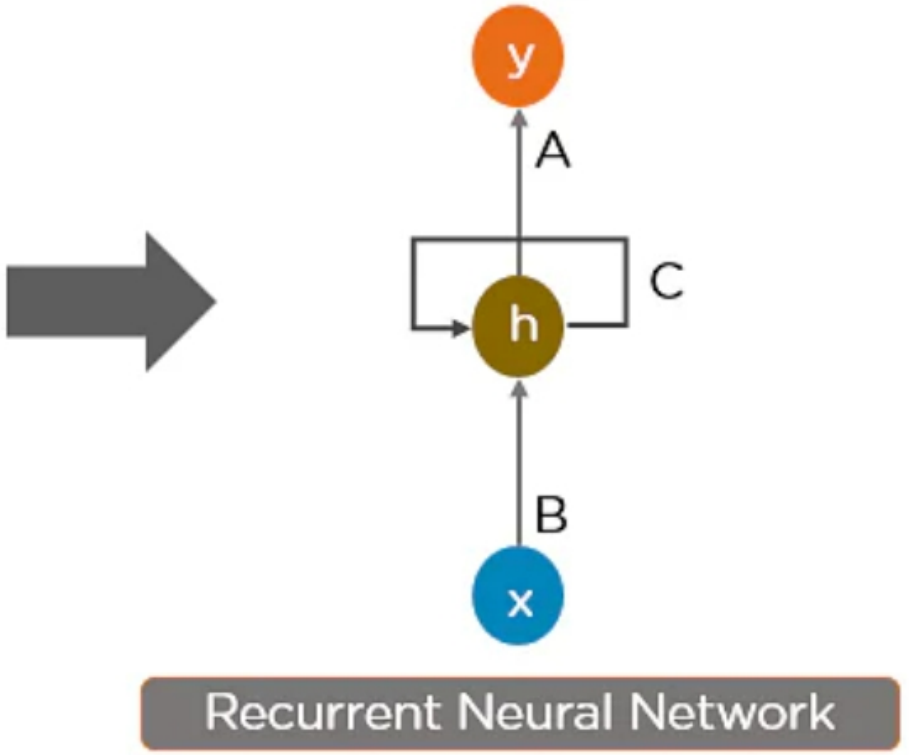
- Cannot handle sequential data
- Considers only the current input
- Cannot memorize previous inputs

- 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.

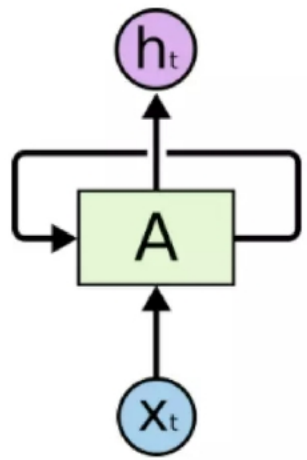
- RNN works on the principle of saving the output of a particular layer and feeding this back to the input in order to predict the output of the layer.

RNN ARCHITECTURE

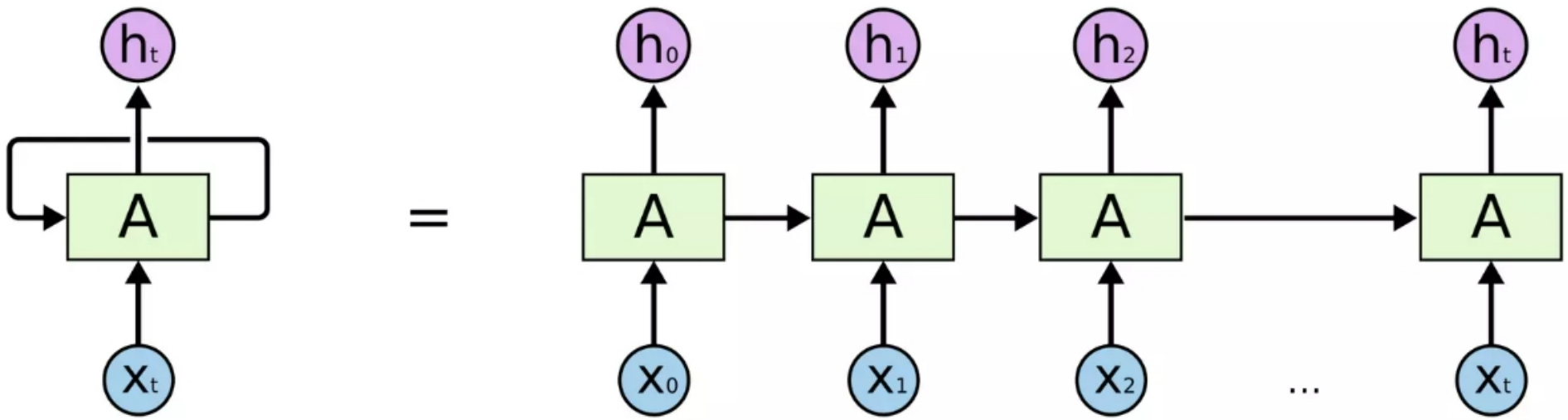


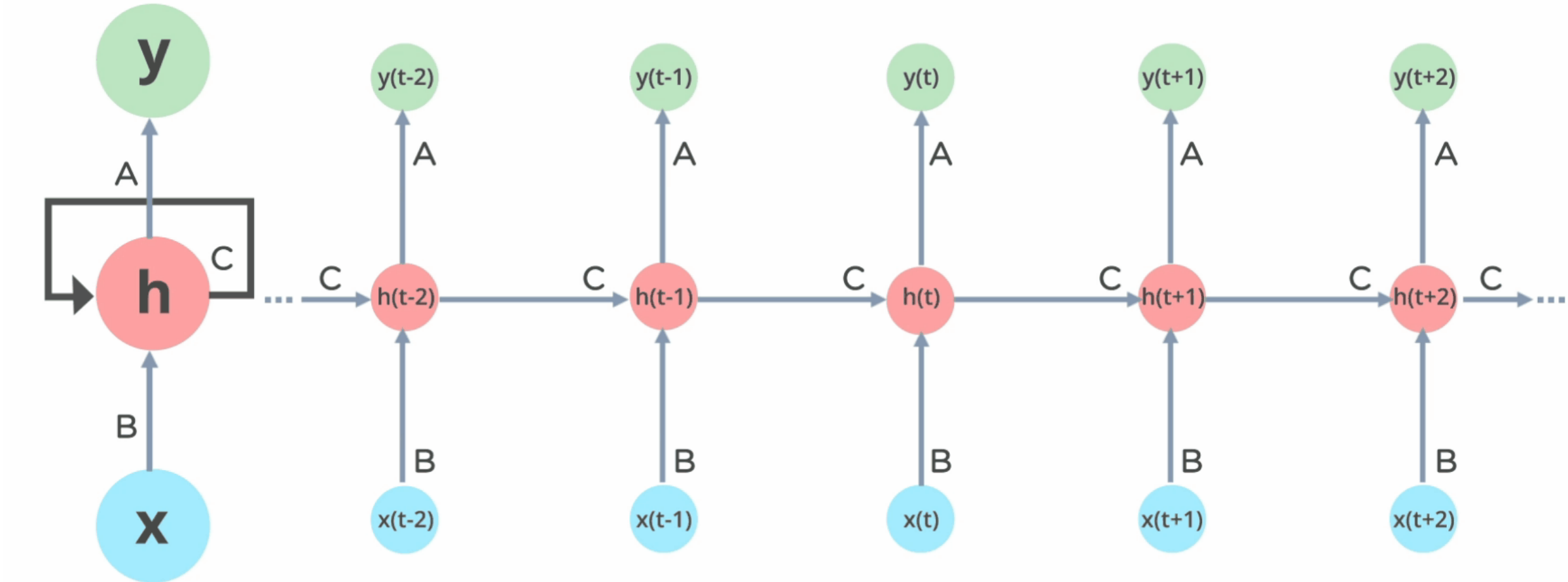


RNN Architecture



RNN Architecture Unfolded




y
A
h
C
B
x
y(t-2)
y(t-1)
y(t)
y(t+1)
y(t+2)
h(t-2)
h(t-1)
h(t)
h(t+1)
h(t+2)
x(t-2)
x(t-1)
x(t)
x(t+1)
x(t+2)

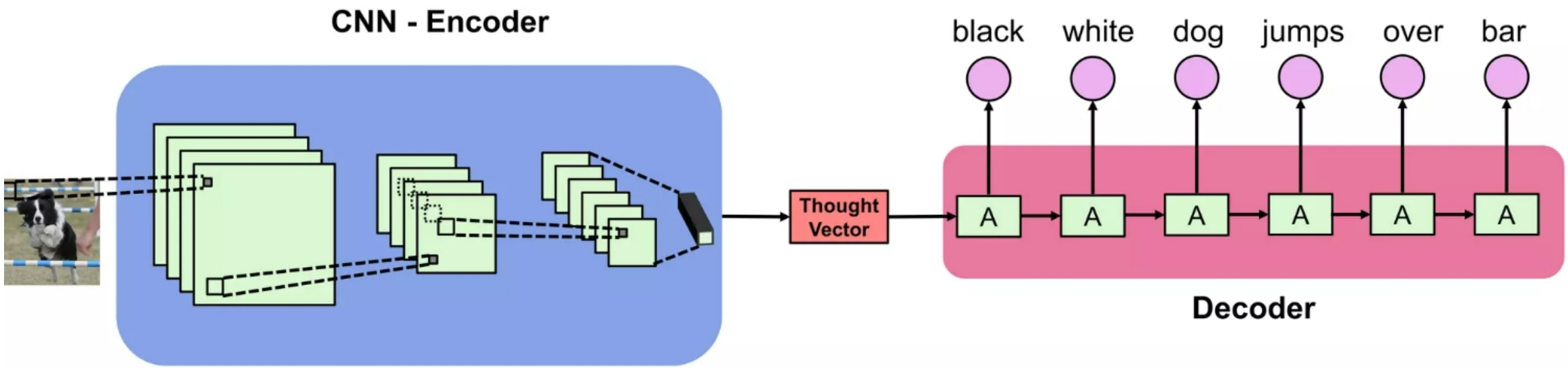
- The middle layer ‘h’ can consist of multiple hidden layers, each with its own activation functions and weights and biases. If you have a neural network where the various parameters of different hidden layers are not affected by the previous layer, ie: the neural network does not have memory, then you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

Applications of Recurrent Neural Networks

Image Captioning

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

Image Captioning

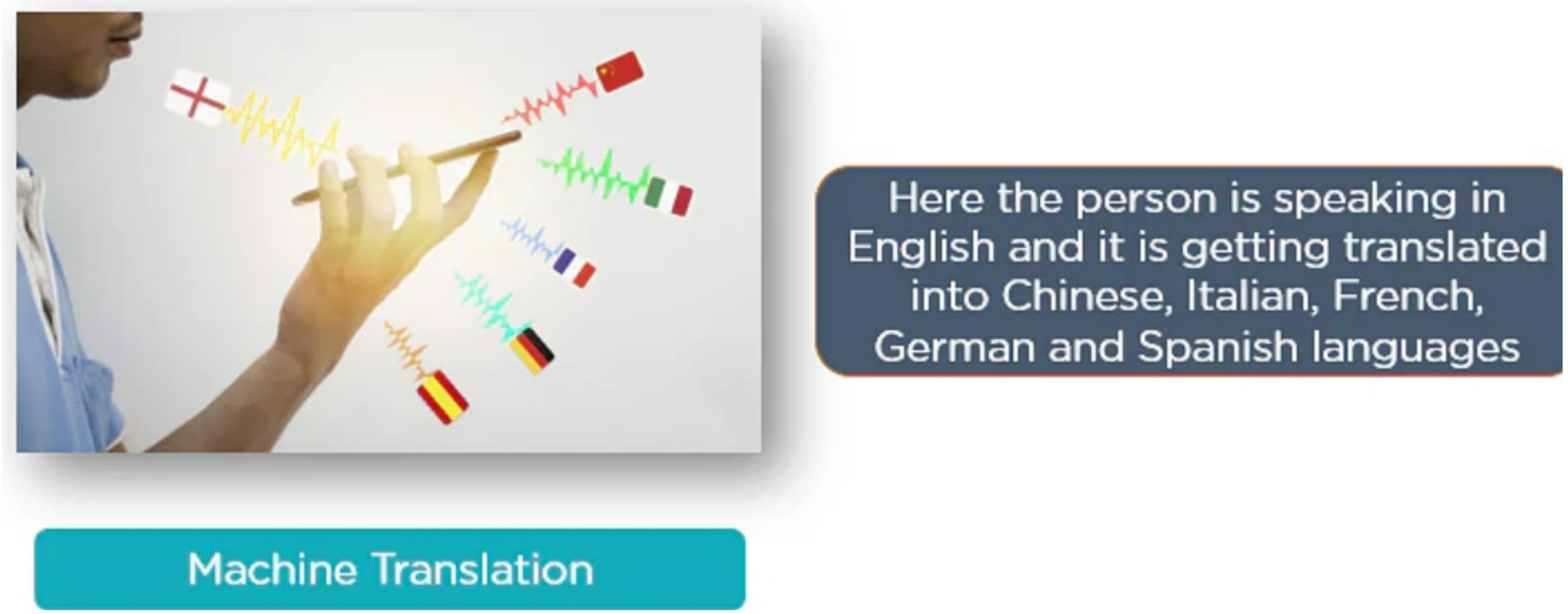


- **Time Series Prediction**
  - Any time series problem, like predicting the prices of stocks in a particular month, can be solved using an RNN.

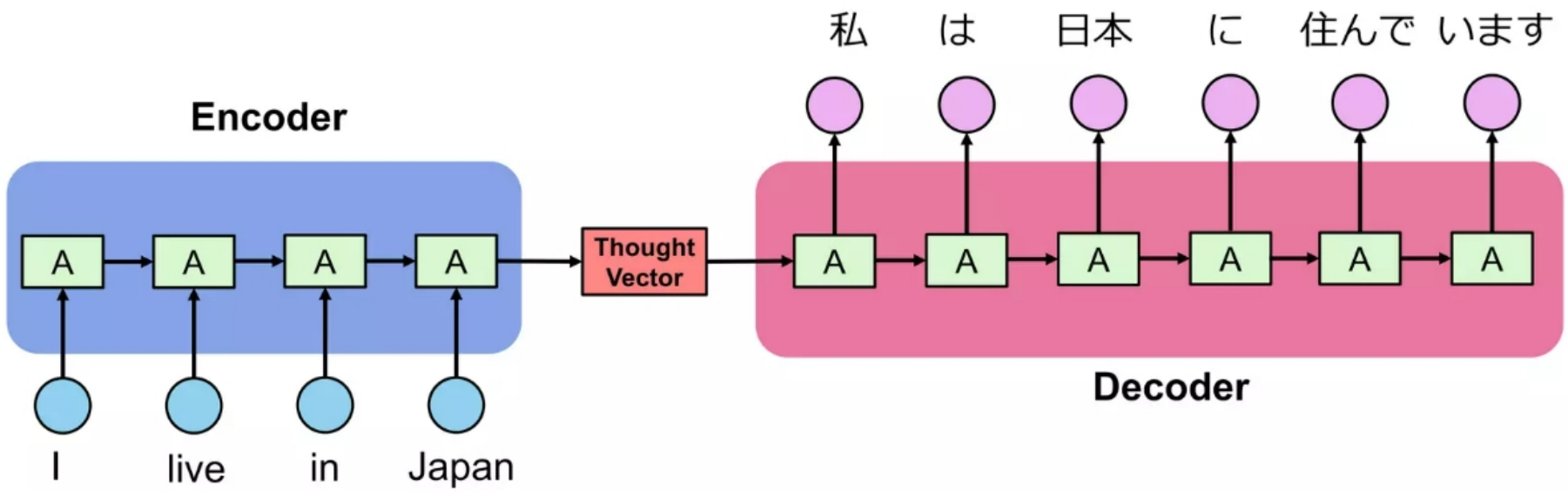
- **Natural Language Processing**
  - Text mining and Sentiment analysis can be carried out using an RNN for Natural Language Processing (NLP).

- **Machine Translation**

- Given an input in one language, RNNs can be used to translate the input into



Language Translation



Types of Recurrent Neural Networks

- There are four types of Recurrent Neural Networks:

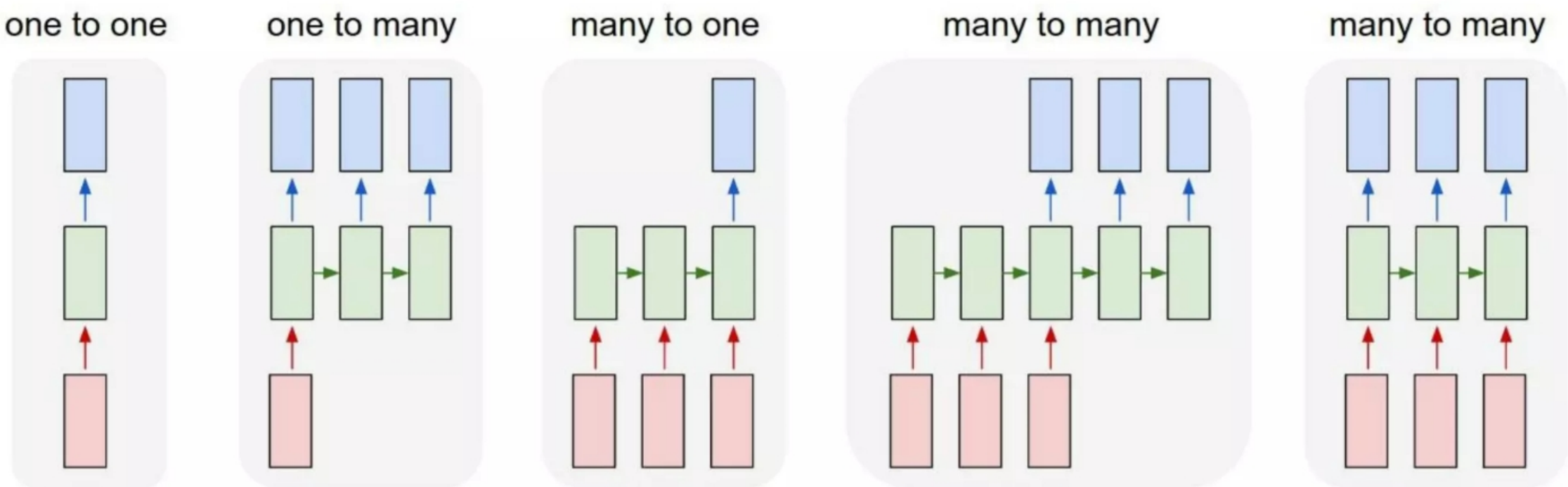
1.One to One

2.One to Many

3.Many to One

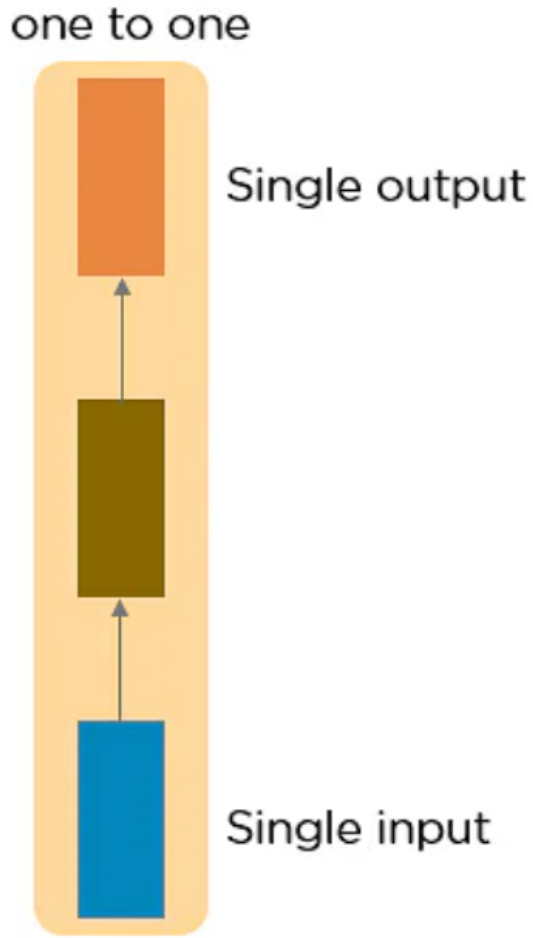
4.Many to Many

Sequences

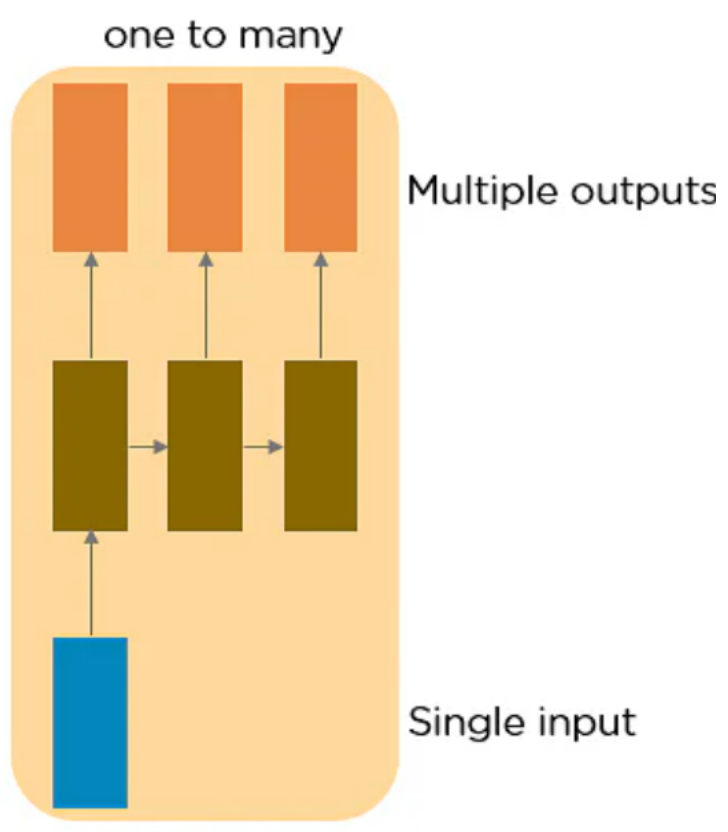


- **One to One RNN**

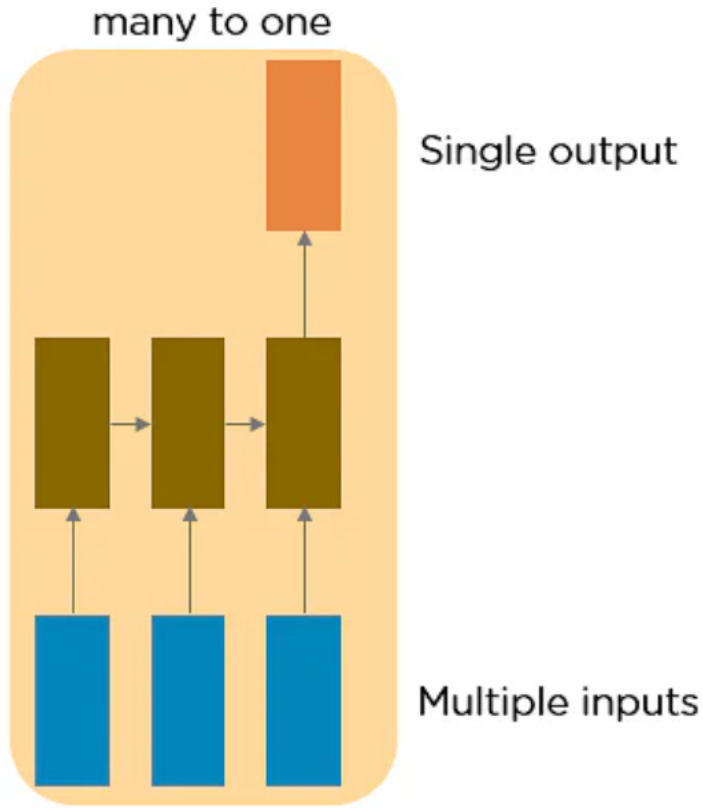
  - This type of neural network is known as the Vanilla
  - Neural Network.
  - It's used for general machine learning problems, whic has a single input and a single output.



- **One to Many RNN**
  - This type of neural network has a single input and
  - multiple outputs.
  - An example of this is the image caption.

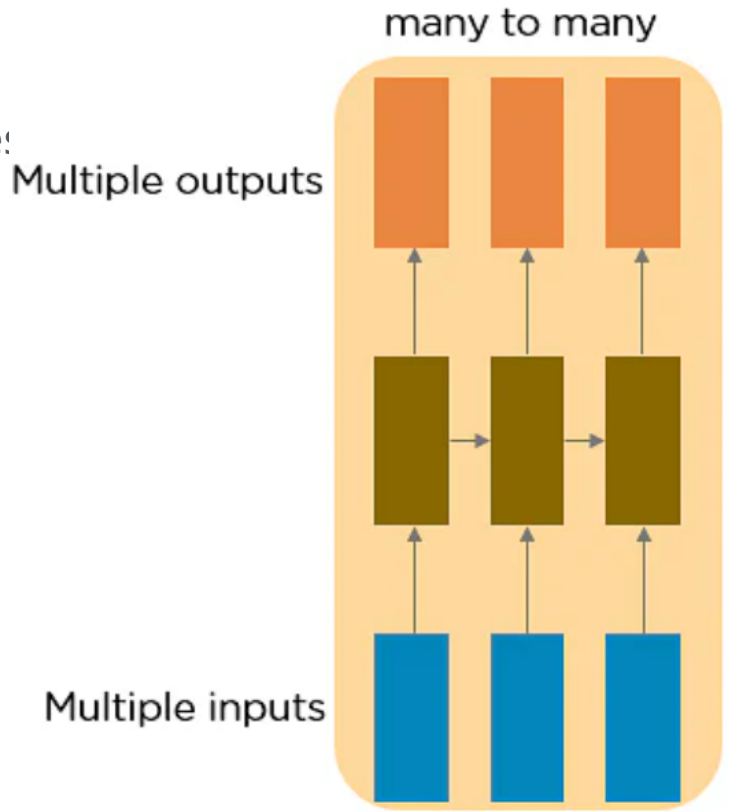


- **Many to One RNN**
  - This RNN takes a sequence of inputs and generate a single output.
  - Sentiment analysis is a good example of this kind network where a given sentence can be classifie expressing positive or negative sentiments.



- **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.