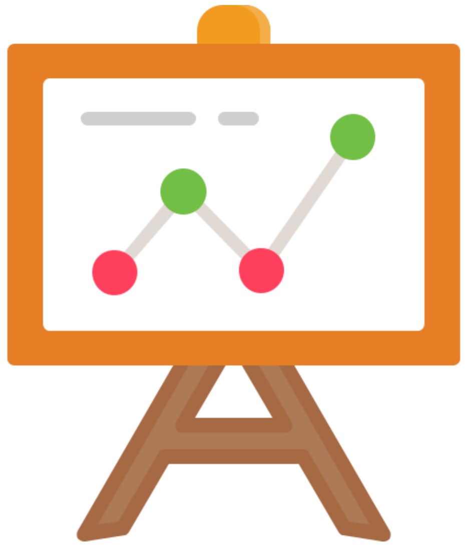
UNIVERSITÀ DEL SALENTO

Time Series

Concepts, Classification and Forecasting



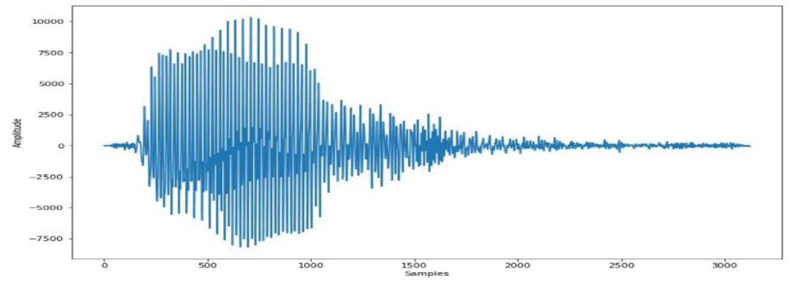
Time Series Definition

A time series is a *sequential set* of *data points*, typically measured over ***successive times***

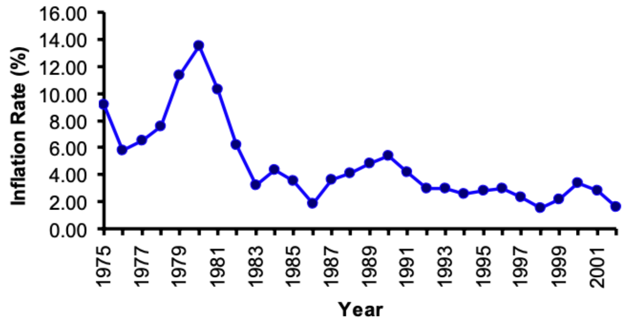
A time series that includes data from only **one variable** is called **univariate**, whereas one that includes data from **multiple variables** is referred to as **multivariate**.

A time series can be continuous or discrete.

- Continuous time series: observations are measured at every instance of time
  - e.g., temperature readings, audio signal



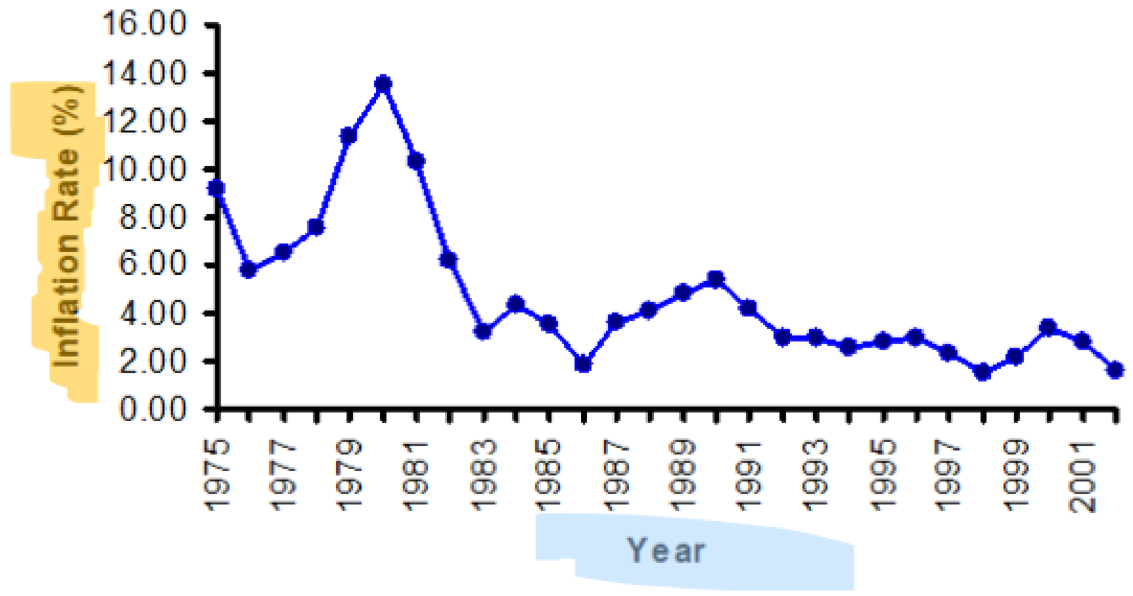
- Discrete time series: observations are measured at discrete points of time
  - e.g., city population, production of a company, exchange rates



Plotting time series

A time-series plot (time plot) is a two-dimensional plot of time series data

- **vertical axis** measures the variable of interest
- **horizontal axis** corresponds to the time periods



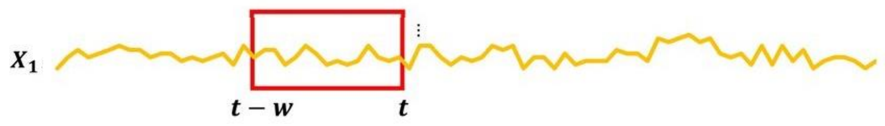
Univaraite and Multivariate

Univariate Time Series consists of observations of a **single variable** recorded sequentially over time. The analysis focuses solely on the past values of that one variable to understand its behavior and make forecasts.

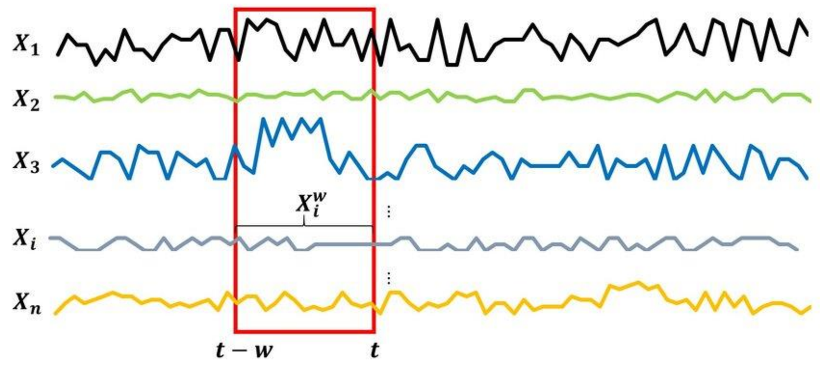
- **Example**: Daily temperature readings at a specific location.

Multivariate Time Series involves **two or more variables** recorded over time. These variables may influence each other, and the analysis often explores their interdependencies to improve forecasting accuracy.

- **Example**: Daily temperature, humidity, and wind speed recorded together over time.



Univariate Time Series

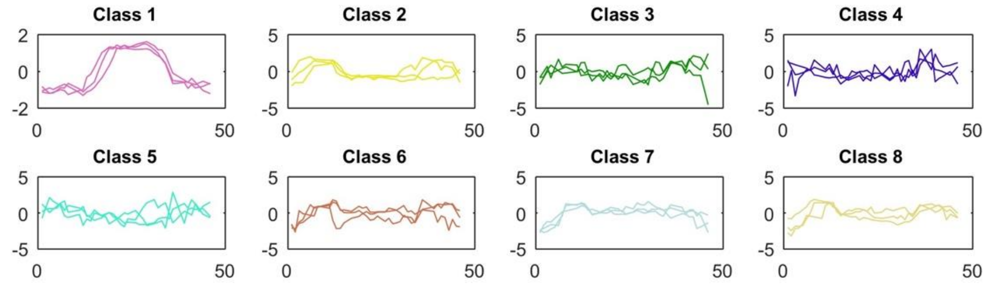


Multivariate Time Series

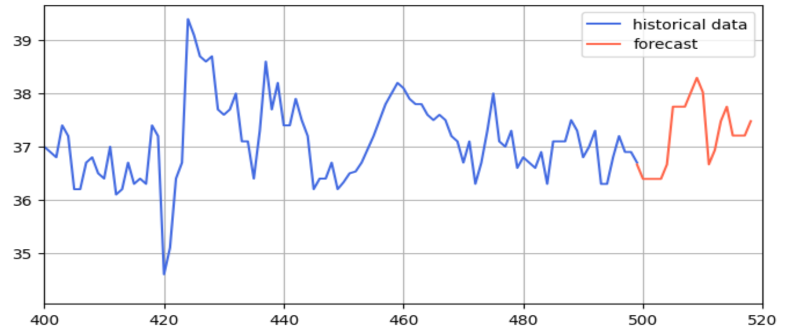
Classification vs Forecasting

Two main kinds of analysis can be performed on time series

- **Classification**
  - speech recognition
  - classification of machine failures



- **Forecasting future values**
  - energy demand prediction
  - weather forecasting
  - traffic prediction



Time Series Components

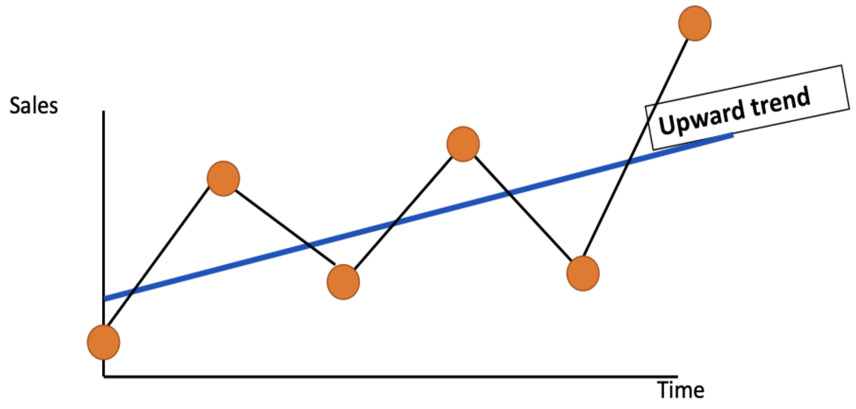
Time series can be characterized by 4 main **components** (none or more than one can occur in a single time series)

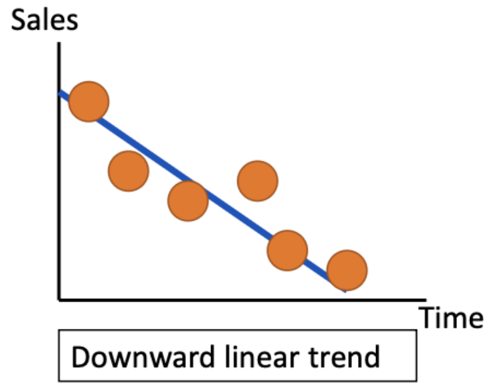
- **Trend**: The long-term direction or movement in the data over time (upward, downward, or flat).
- **Seasonal**: Regular, repeating patterns or fluctuations that occur within fixed periods (e.g., daily, monthly, yearly).
- **Cyclical**: Long-term oscillations or patterns that occur over irregular time intervals, often related to economic or business cycles.
- **Irregular (or Random)**: Random, unpredictable variations in the data that do not follow a pattern and are not explained by the other components. This component usually represents "noise" in the time series

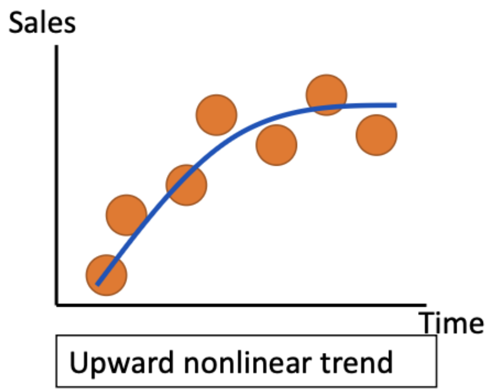
Trend Component

Trend Component describes the long-term direction or movement in the data over time (overall, persistent, long-term movement)

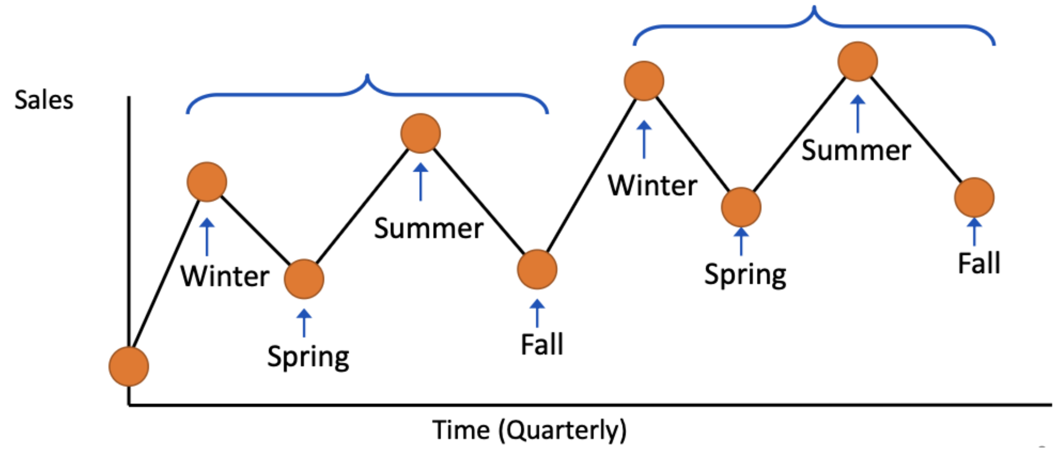
- Increasing or decreasing over time (upward or downward movement)
- Trend can be upward or downward
- Trend can be linear or non-linear





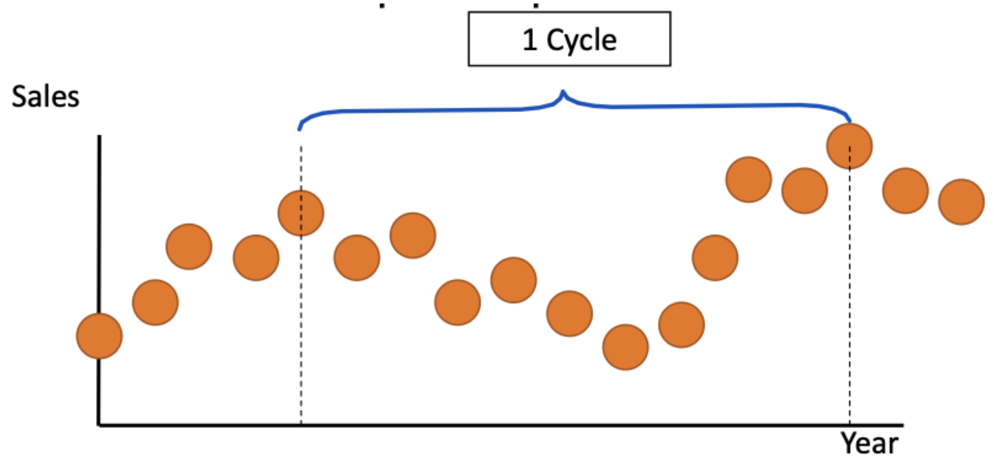


Seasonal components regards all the regular, repeating patterns or fluctuations that occur within fixed periods (e.g., daily, monthly, yearly).



Cyclical components are the long-term oscillations or patterns that occur over regular time intervals, often related to economic or business cycles.

Often measured peak to peak



Classification vs Forecasting

Many algorithms can be applied to address **classification** and **forecasting** tasks

- Machine learning algorithms such as
  - Random forest classifier/regressor
  - SVM
- Neural networks, etc.
- Statistical approaches: e.g., ARIMA (Autoregressive integrated moving average) models

Given an **analytics goal** different methods can be exploited

The algorithm selection is driven by

- Application requirements: accuracy, human-readable model, scalability, noise and outlier management
- The complexity of the analytics task

CNNs in Time Series Analysis

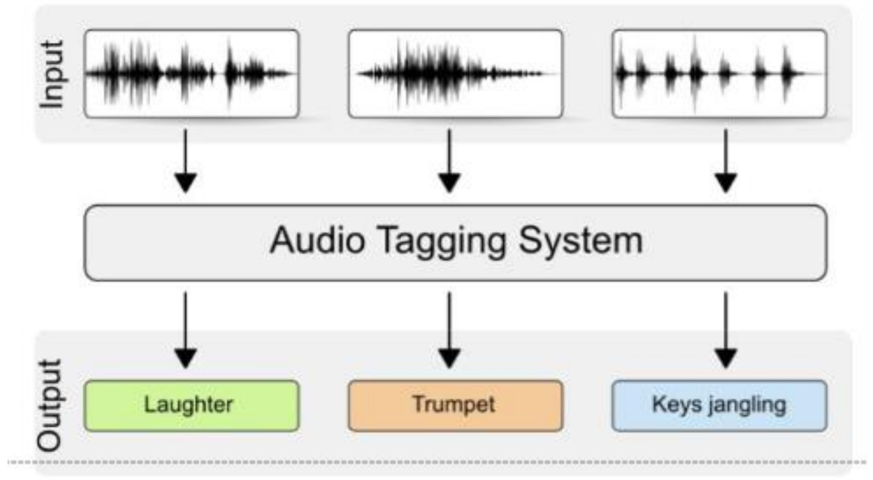
Convolutional Neural Networks (CNNs), originally developed for image processing, have proven effective for **time series analysis classification**

CNNs automatically learn to extract **local** and **hierarchical features** from time series data. For example, they can detect patterns such as *spikes*, *trends*, or *repeating sequences* without manual feature engineering.

- **Univariate Time Series**: Distinguishing types of signals or behaviors (e.g., anomaly detection, ECG signal classification).
- **Multivariate Time Series**: CNNs can process multiple variables simultaneously using multi-channel inputs, similar to RGB channels in images.

Due to their ability to use shared weights and local connections, CNNs are **computationally efficient** and can handle long time series by focusing on local temporal patterns.

In time series, **1D convolutions** are applied across the **time axis**, treating the time series as a one-dimensional sequence rather than a 2D image.

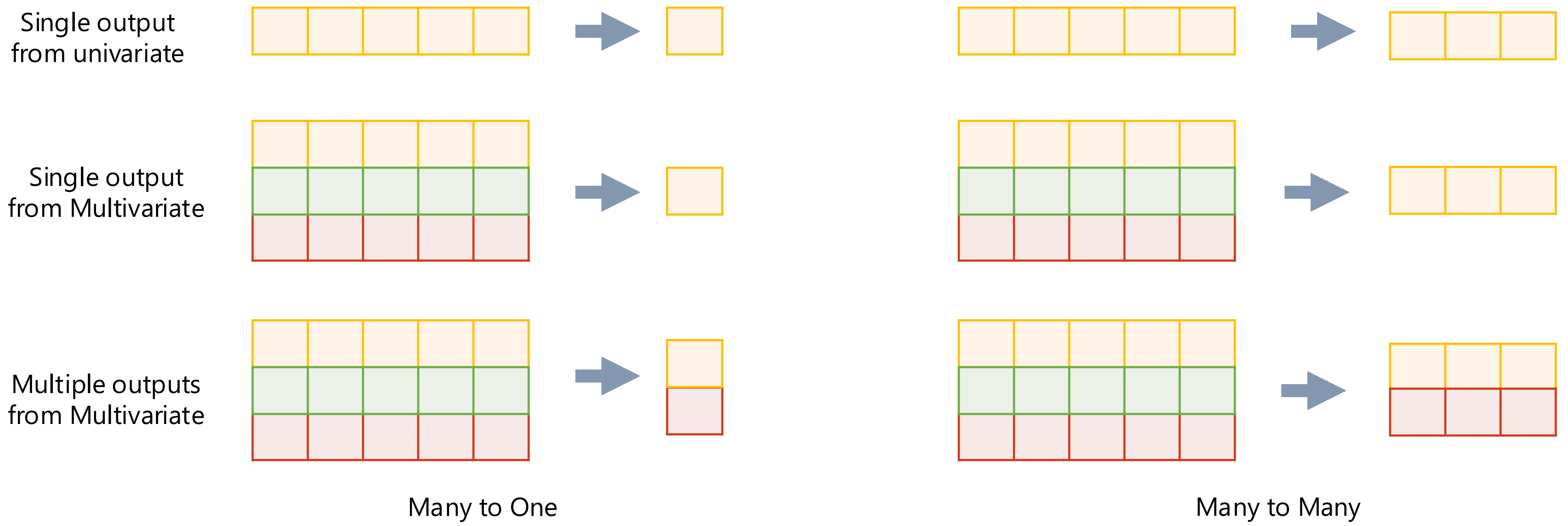


Time Series Forecasting

Time series forecasting is the process of **predicting future values** using **historical time-ordered** data.

- **Patterns**: Can include trends, seasonality, and cycles.
- **Goal**: To make informed predictions that support decision-making in areas like finance, weather, sales, and inventory management.

There exist several combination of input and output


Single output from univariate
Single output from Multivariate
Multiple outputs from Multivariate
Many to One
Many to Many

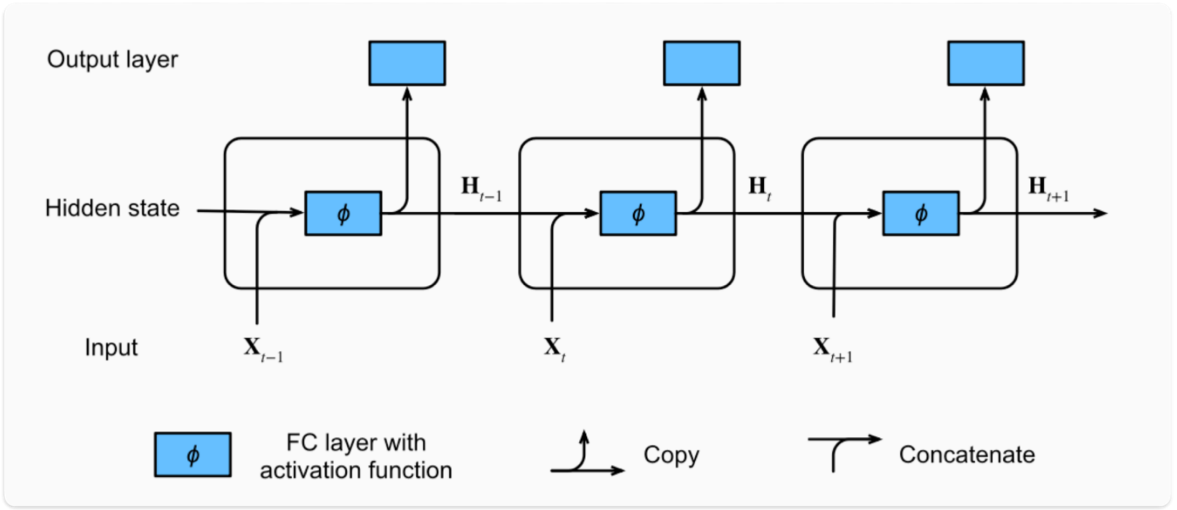
Recurrent Neural Networks

Recurrent Neural Network (RNN) is a type of neural network that contains **memory** and is best suited for *sequential data*.

Recurrent Neural Network is a generalization of a feed-forward neural network that has an **internal memory**

RNN is recurrent in nature as it performs the same function for every input of data while the output of the current input depends on the past one computation.

It considers the current input and the output that it has learned from the previous input.



Recurrent Neural Networks

The internal equation of a **Recurrent Neural Network (RNN)** cell describes how it updates its **hidden state** at each time step based on the **current input** and the **previous hidden state**.

$$h_t = f(W_{ih}\underline{x_t} + W_{hh}\underline{h_{t-1}} + b_h)$$

Where:

$\boldsymbol{h_t}$: Hidden state at time step t

$\boldsymbol{x_t}$: Input at time step t

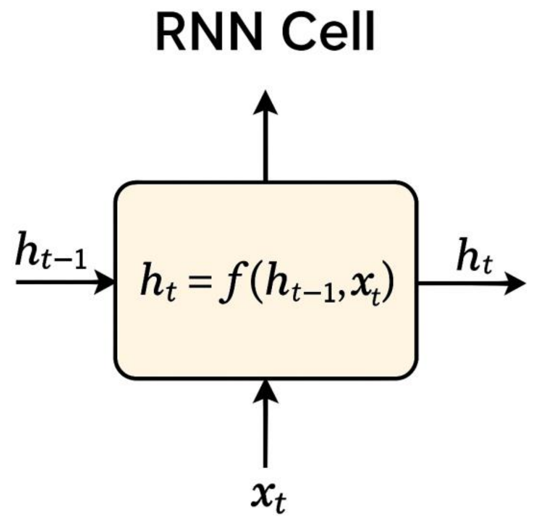
$\boldsymbol{h_{t-1}}$: Hidden state from the previous time step

$\boldsymbol{W_{ih}}$: Weight matrix for the input

$\boldsymbol{W_{hh}}$: Weight matrix for the previous hidden state

$\boldsymbol{b_h}$: Bias term

f : Usually a $tanh$ activation function (commonly used, but others like ReLU or sigmoid can be used)



Recurrent Neural Networks

One to One RNN. One to One RNN (Tx=Ty=1) is the most basic and traditional type of Neural network giving a single output for a single input

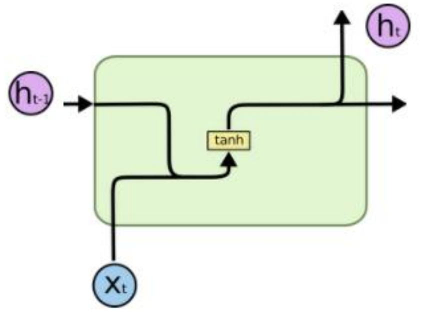
One to Many (Tx=1,Ty>1) is a kind of RNN architecture is applied in situations that give multiple output for a single input. In music generation, RNN models are used to generate a music piece (multiple output) from a single musical note (single input).

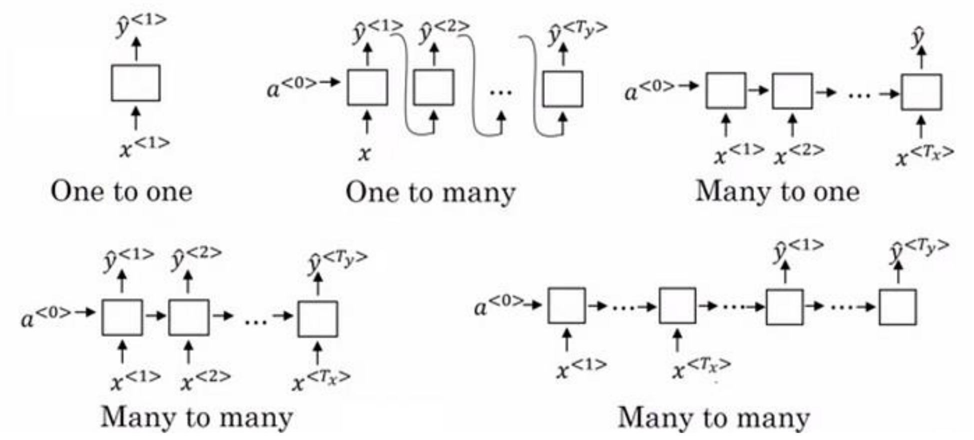
Many to One RNN architecture (Tx>1,Ty=1) is usually seen for sentiment analysis models as a common example. This model is used when multiple inputs are required to give a single output.

Many-to-Many: RNN (Tx>1,Ty>1) architecture takes multiple inputs and gives multiple outputs. Many-to-Many models can be two kinds:

1.Tx=Ty: when input and output layers have the same size (every input having an output). A common application is Named entity Recognition.

2.Tx!=Ty: where input and output layers are of different size. The most common application is seen in Machine Translation.





RNN Limitation

Gradient: is a partial derivative with respect to its inputs. A gradient measures how much the output of a function changes, if you change the inputs a little bit.

- A gradient simply measures the change in all weights with regard to the change in error.
- The higher the gradient, the steeper the slope, and the faster a model can learn.
- If the slope is almost zero, the model stops to learn.

Sometimes the gradient can become too small (**Vanishing Gradient**) or too large (**Exploding Gradient**) leading to

- Poor Performance
- Low Accuracy
- Long Training Period

RNN are **not** able memorize data for long time and begins to forget its previous inputs

RNN is **not** able to distinguish between important or not important information

Long-Short Term Memory

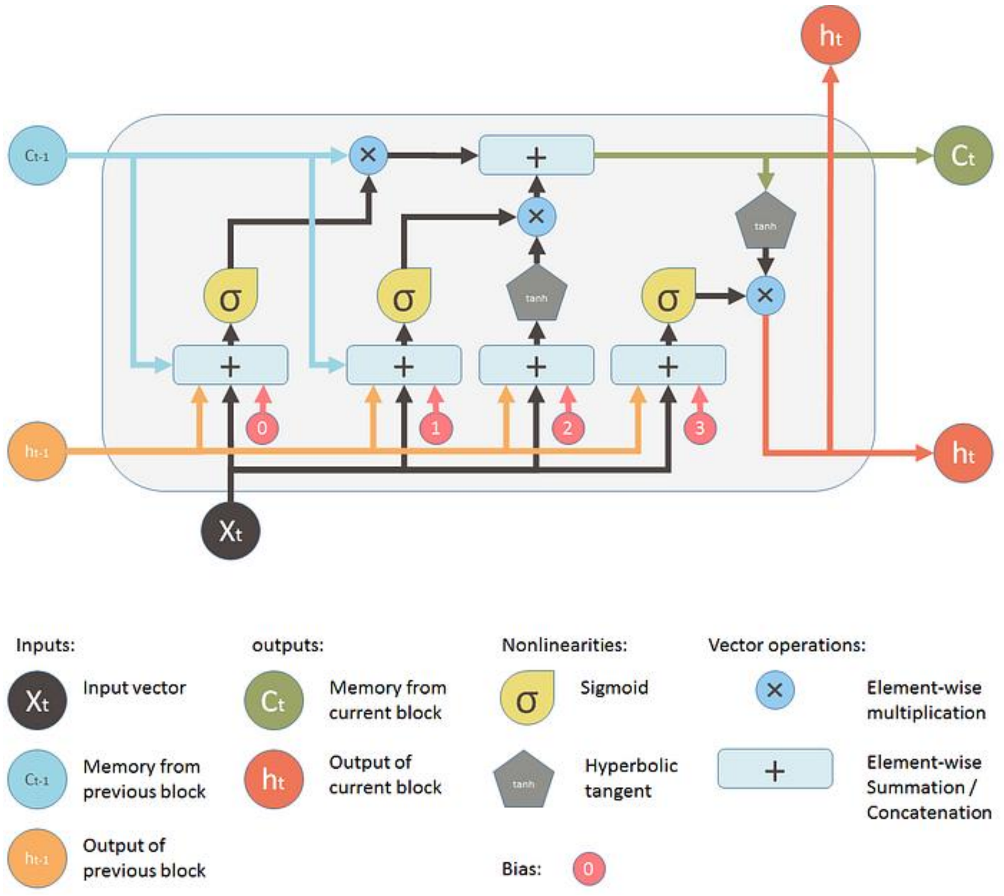
LSTM is a type of **Recurrent Neural Network (RNN)** specifically designed to **learn and remember long-term dependencies** in sequential data. Unlike standard RNNs, LSTMs can effectively capture patterns over longer sequences without suffering from the **vanishing** or **exploding gradient** problem.

Key Features:

- Contains **memory cells** that maintain information over time.
- Uses **gates** (input, forget, and output gates) to control the flow of information.

LSTMs learn ***what to remember***, ***what to forget***

LSTM is a memory-enhanced RNN that can model both short- and long-term dependencies in sequential data

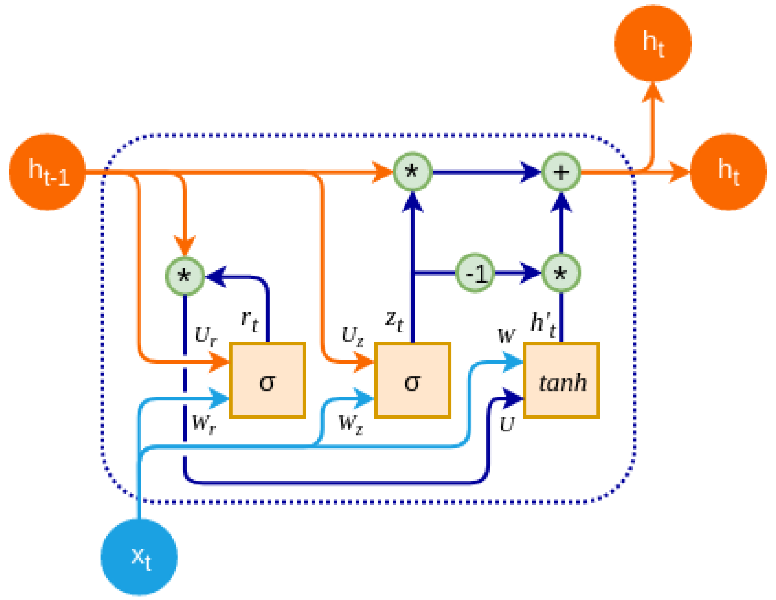


Gated Recurrent Unit (GRU)

GRU stands for **Gated Recurrent Unit**, which is a type of recurrent neural network (RNN) architecture that is similar to LSTM (Long Short-Term Memory).

GRU has a simpler architecture than LSTM, with fewer parameters

- easier to train
- computationally efficient



In GRU, the memory cell state is replaced with a "**candidate activation vector,**" which is updated using two gates: the reset gate and update gate.

Metrics for time series forecasting performance,

Mean Absolute Error (MAE)

- Measures average absolute error
- Easy to interpret
- Not sensitive to outliers

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|y_t - \widehat{y_t}|$$

Mean Squared Error (MSE)

- Penalizes large errors more than MAE
- Good for optimization
- Type equation here.

$$MSE = \frac{1}{n}\sum_{t=1}^{n}(y_t - \widehat{y_t})^2$$

Root Mean Squared Error (RMSE)

- Same units as the data
- More interpretable than MSE

$$MSE = \sqrt{MSE}$$