CHAPTER
6

TIME SERIES FORECASTING MODELS: CLASSICAL TO CURRENT

G. Arun Prasath[1] and S. Vishnu Shankar[2]
[1] *School of Agriculture and Animal Sciences, The Gandhigram Rural Institute, Dindigul, Tamil Nadu*
[2]*Department of Physical Sciences & Information Technology*
Tamil Nadu Agricultural University, Coimbatore
E-mail: arunprasathgtirupur@gmail.com & s.vishnushankar55@gmail.com

DOI: https://doi.org/10.34293/blp.9789395659581.ch006

Abstract
A time series data contains series of recorded observations or values or readings of a particular parameter or variable over different time. It may even be within a day as an hourly observation, or daily observations, weekly, monthly, annually, etc. The primary purpose of time series analysis is forecasting. Forecasting methods have evolved significantly over the decades. Classical approaches such as Exponential Smoothing and ARIMA models dominated earlier forecasting research due to their interpretability and strong statistical foundations. With advancements in computational power, machine learning methods such as Random Forest, Support Vector Regression, and Gradient Boosting emerged. The next major shift came with deep learning, particularly Recurrent Neural Networks (RNNs), LSTM, GRU, and 1D-CNN architectures, which are designed to learn temporal patterns automatically. However, all these models possess some kind of limitations, which made researchers to use hybrid models. Though there are several models, the nature and structure of the data, decides the model to be used for a particular dataset. This chapter gives the overview of different forecasting models.
Keywords*: Time series, stationarity, forecasting, heteroscedasticity, autoregressive, hybrid modelling.*

Introduction

Time series data consist of observations collected sequentially over time to describe the evolution of a particular variable, phenomenon, or process. These observations may be recorded at different frequencies like hourly, daily, monthly, or annually, but a fundamental requirement is that they must be captured at regular and consistent intervals to preserve the temporal structure. Time series data provide a unique window into how a system behaves through time, offering insights into long-term trends, recurring seasonal movements, cyclical fluctuations, and short-term irregularities. Because of this temporal richness, time series datasets are indispensable in fields such as economics, agriculture, climatology, finance, energy, and environmental sciences. The process of collecting time series data often depends on the nature of the parameter being monitored (Gujarati and Porter 2009). For example, rainfall records are gathered monthly, stock prices are recorded minute-wise, and temperature readings may be collected hourly. Regardless of the source, the value of such

data lies in their ability to reveal how a variable change, what factors influence it, and what might happen in the future. However, time series data also come with inherent challenges: missing observations, irregular sampling, structural breaks, measurement errors, and non-stationary behaviour. These issues can significantly affect the reliability of the analysis and forecasting.

The primary purpose of time series analysis is forecasting. Statisticians and economists rely on time-based forecasts to understand two fundamental aspects:

1. How one variable affects another over time, and
2. How the same variable is expected to behave in the future.

While regression models help quantify the relationship between variables, forecasting future values demands dedicated time series techniques. Producing accurate and reliable forecasts is not trivial as it requires careful data preparation, understanding the structure of the series, and selecting a model that captures its unique characteristics.

Evolution of Forecasting Techniques

Forecasting methods have evolved significantly over the decades. Classical approaches such as Exponential Smoothing and ARIMA models dominated earlier forecasting research due to their interpretability and strong statistical foundations. These models perform well when data exhibit clear linear patterns but are often limited when dealing with non-linear, highly volatile, or complex behaviours.

With advancements in computational power, machine learning methods such as Random Forest, Support Vector Regression, and Gradient Boosting emerged. These models are capable of capturing non-linear relationships and interactions that traditional models cannot. However, they often ignore temporal dependencies unless specifically engineered into the features.

The next major shift came with deep learning, particularly Recurrent Neural Networks (RNNs), LSTM, GRU, and 1D-CNN architectures, which are designed to learn temporal patterns automatically. These models can capture both short-term and long-term dependencies and handle non-linearities effectively. Despite their power, deep learning models require large datasets, high computational resources, and careful tuning, and they may lack interpretability.

To address the limitations of individual techniques, researchers increasingly use hybrid models that has the combinations of statistical, machine learning, or deep learning methods to leverage the strengths of each. Hybrid approaches aim to model linear and non-linear components separately, improving forecasting accuracy in complex real-world systems such as climate variables or financial time series.

AR Model

AR model – Auto Regressive includes the regressing the selected variable on its lagged variables. Let Y be selected variable, then Y has to regressed on different lag times of Y upto *p* times, where p can be selected based on certain conditions, the most commonly followed being the Akaike Information Criteria (AIC). Thus, AR method includes

$Y = B_0 + B_1 Y_{t-1} + B_2 Y_{t-2} + ... + B_p Y_{t-p} + u_t$

Where B represents the Regression coefficients, Y the selected variable and lagged upto p times and u_t refers to disturbance term or noisy error and hence AR(p) is the model.

MA model

MA model – Moving Average model includes the average of current and past values. As like above let Y be the selected variable then MA includes

$Y_t = C_0 + C_1 u_t + C_2 u_{t-1} + ... C_q u_{t-q}$

Where Y_t represents the moving average of the noisy error terms upto q times and hence MA(q) is the model.

Arima Method of Forecasting

ARIMA, a forecasting methodology popularly called as Box-Jenkins methodology or BJ Methodology, as it was developed by Box and Jenkins. The basic philosophy behind BJ methodology is "Let the data speak for themselves", which Made this methodology a popular among the people in the field of forecasting. ARIMA is the abbreviated form of Auto Regressive Integrated Moving Average, where Auto Regressive (AR) is a separate component and Moving Average (MA) is a separate component. The most important thing here is stationarity of timeseries.

The concept of stationarity in time series refers that the mean and variance are constant over time and the value of covariance between the two time periods depends only on the distance between the two time periods and not the actual time at which the covariance is computed (Dickey 2015). The condition of stationarity is one of the important concepts in econometrics and in timeseries modelling as most of the models we use for forecasting requires the time series data to be in stationary form (Manuca and Savit 1996).

The importance of stationarity is, one can use the stationary time series data to study the behaviour of a particular variable at any point of time, as the mean and variance of the entire data is almost same (Horvath et al 2014). If it is non-stationary, then the data can be used to study only the specified time period and the same cannot be used for forecasting of future values. It can also be said that a non-stationary time series will have time varying mean and the stationary time series will have same mean over the entire period. What happens if the time series is non-stationary? According to Enders, (2008), if non-stationary time series is used for modelling, the resulting regression becomes spurious regression, it means, even though the obtained value of R^2 is high with significant coefficients, the results will not be reliable.

If the data is non-stationary then it has to be differentiated until it becomes stationary. The differentiation may take one or two times depending upon the type of time, but the end it must become stationary to proceed for forecasting.

Combining the above two models AR(p) and MA(q) represent ARMA (p.q). ARMA – Auto Regressive Moving Average model, is a kind of modelling can be used if the data is already stationary. If the data is nonstationary then we need to differentiate it and go for ARIMA (p,d,q). Here d represents number of times the data is differentiated, usually one

differentiation is required and hence d takes value of 1 and in some cases, it takes value of 2, which differentiating the data set two times to get stationarity. Once the value of d is found, then it is essential to find 'p' and 'q'. Usually Akaike Information Criteria (AIC) and Schwartz Bayesian Criterion (SBC) will be used to find 'p' and 'q' (Mila and Parvin 2019). It is selected based on lowest value of either AIC or SBC. Also, the most important thing to be noted in ARIMA is, it is of the assumption that the residuals is homoscedastic in nature (Hipel et al 1977).

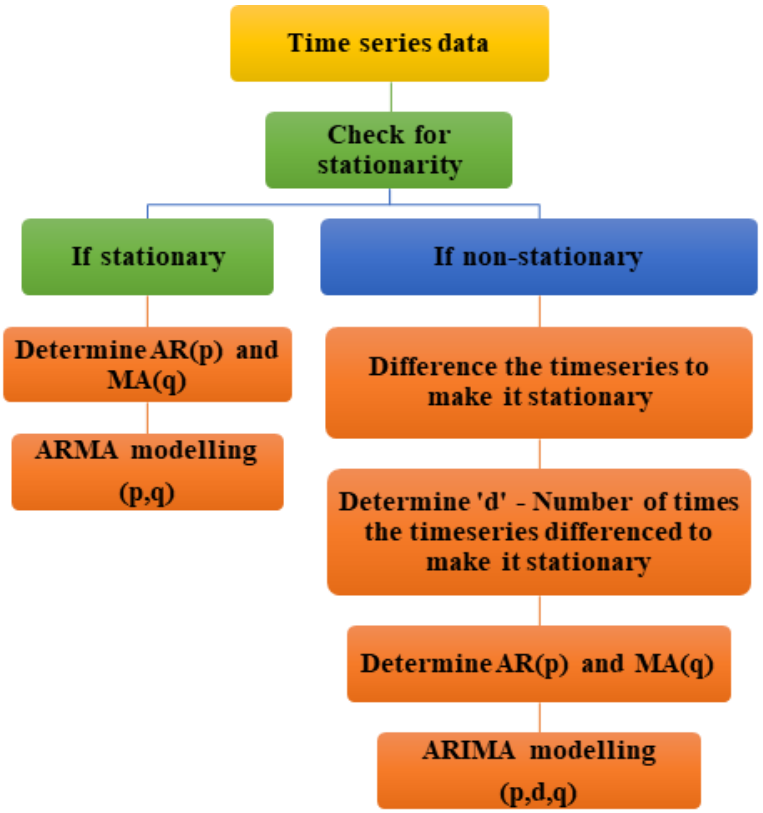

Fig 6.1. Flowchart to determine ARIMA/ARMA process

This ARIMA is most commonly used, easy and reliable method of forecasting as it contains simple four steps; Identification, Estimation, Diagnostic checking and forecasting (Jathav et al 2017). ARIMA is used particularly in forecasting the prices, production and productivity of agricultural crops. Praveen and sharma (2019) assessed the impact of climate vulnerability on agricultural production in India and forecasted the productivity of wheat, rice, cotton, gram, arhar, groundnut etc. Similarly, Tobacco production was forecasted by Mapuwei et al (2022) using ARIMA model with more than 90 per cent accuracy. A short flowchart description showing the method to determine ARIMA/ARMA process is given in **Fig.6.1**.

ARIMAX modelling

ARIMAX modelling is an extension of ARIMA model, wherein all the components of ARIMA along with exogenous variable or explanatory variable (X) that might affect the timeseries, can also be included in the model (Wangdi 2010). For example, in modelling the area of paddy crop in a country, exogenous variables or explanatory variables like climatic factors, rainfall, etc. may be included along with ARIMA modelling to make it as ARIMAX forecasting. The variables included will be one of the factors influencing Y and hence selected as X. Chadsuthi et al (2012) forecasted leptospirosis transmission in Thailand using ARIMAX model. The study used rainfall and temperature as exogenous variable in several combinations and reported ARIMAX model with rainfall and temperature has better prediction than ARIMA. The model had been extensively used in several fields in order to forecast, as it is more reliable and accurate because of the exogenous variable addition to the ARIMA model.

SARIMA modelling

SARIMA is another form of extending ARIMA, which includes seasonality factor (Sirisha et al 2022). According to Hipel and McLeod (1994) SARIMA model consists of two components. Mathematically it can be written as $(p,d,q) \times (P,D,Q)_S$. Here (p,d,q) refers to the components similar to ARIMA and it is non-seasonal in nature, whereas P refers to seasonal autoregression, D refers to seasonal differencing, Q refers to seasonal moving average and S refers to seasonal length (Bockwell and Davis 1991). The same methodology was used by Dabral and Murry (2017) to forecast the rainfall in Arunachal Pradesh, India. The study used different seasonal length like monthly, weekly and daily to forecast the rainfall for almost 14 years from 2014 to 2027 using SARIMA model.

SARIMAX

SARIMAX is an extended from of SARIMA, which includes seasonal factor in ARIMA along with exogenous variable X (Manigandan et al 2021 and Naveena 2017). Dutta and Roy (2021) used SARIMAX model to forecast the level of indoor air pollutants like CO_2A, Particulate matter, Total volatile organic compounds etc. The study includes number of humans in the room, holiday, temperature, etc, as exogenous variables and compared the results of SARIMA and SARIMAX models. The results revealed that SARIMAX model with exogenous variables outperformed SARIMA and has the accuracy of 97 per cent. According to Alharbi and Csala (2022), ARIMA model with seasonality and exogenous variable i.e. SARIMAX reduces prediction error, autocorrelation problems, improves prediction accuracy and performed better than others. They added the addition of exogenous variables improves efficiency and performance of the model.

ARCH

ARCH refers to Auto Regressive Conditional Heteroscedasticity. In previous forecasting models like ARIMA, ARIMAX, etc, we assume the residuals to be homoscedastic, but here ARCH made an attempt to model the variance and hence, volatility in the data can be

modelled, allowing the residuals to be heteroscedastic in nature (Engle 2001). It can also be stated that, time series data with heteroscedastic in nature can be modelled using ARCH. Unlike other models, here the condition is residuals of the time series data should be heteroscedastic in nature (Jordaan et al 2007). In order to carryon ARCH, first fit the mean equation may be using ARIMA, then it is essential to check the conditional heteroscedasticity in residuals. If exists, then ARCH can be applied and LM test will be done for diagnostic checking and validation. However ARCH model has some drawbacks, they are as follows. Since ARCH uses square value of error, the positive and negative values have no effect on volatility (Thiyagarajan et al 2015). In addition, if the order of ARCH is large, then large number of parameters are to be estimated (Lama et al 2015). Different forecasting models and their relationship was given below in **Fig 6.2.**

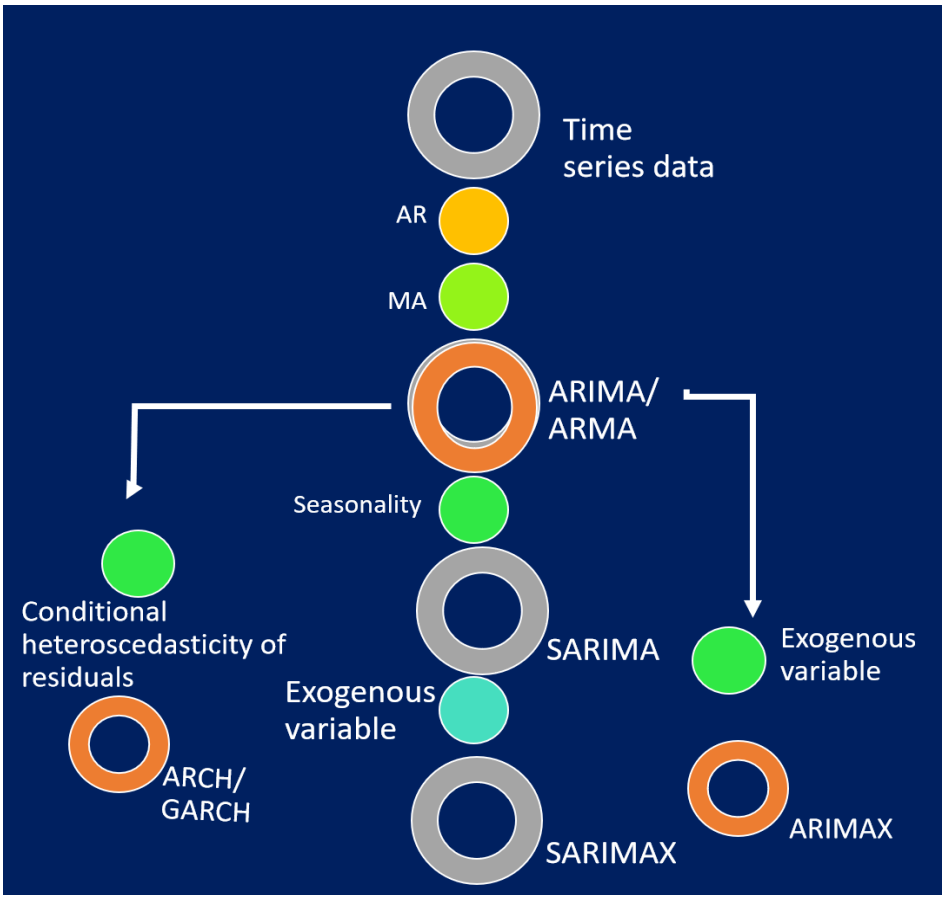

Fig. 6.2. Different timeseries forecasting models

Garch

The defects of ARCH model are rectified in GARCH, which is abbreviated form of Generalized Auto Regressive and Conditional Heteroscedasticity, generally it is said to be a parsimonious form of ARCH.

Exponential Smoothing

Exponential smoothing is a kind of tool used to forecast time series data. In this method, weights are assigned to the data based on principle that weights decline based on time, as the time increases weights decreases (Sendhil 2024). Let Y_t be the forecasted time series and X_t be the actual series, then the forecast for the next day (t+1 day) will be in the form

$Y_{t+1} = a X_t + (1\text{-}a) Y_t$,

where a refers to smoothing coefficient and it takes value between 1 and 0. Exponential smoothing may be generally classified into Single, double and Triple exponential smoothing and its use depends upon the properties of timeseries data. If the data has no trend and seasonality then Single exponential smoothing can be followed. If the data has trend but without seasonality then Double exponential smoothing has to be used for forecast and if the data has both trend and seasonality then forecasting can be done using Triple exponential smoothing (Areef et al 2020). Depending upon the type of timeseries data, by using the table below the method exponential smoothing can be adopted for forecasting.

Table – 6.1. A guide to choose exponential smoothing method based on time series data

Time series data containing		Type of exponential smoothing can be adopted
Trend	**Seasonality**	
No	No	Single exponential smoothing
Yes	No	Double exponential smoothing
Yes	Yes	Triple exponential smoothing

Neural Netwroks

Artificial Neural Networks

ANNs are machine learning models designed to identify complex and nonlinear patterns in data. They are highly effective in prediction and classification tasks which is inspired by the architecture of biological neural networks. ANN is made up of three primary components such as input layer, hidden layers, and output layer. Each layer consists of nodes that are interconnected through weights, which play a key role in the network's learning process. Initially, the input values are assigned weights, and these weighted inputs are processed through the hidden layers. The structure of ANN is illustrated in **Fig. 6.3**. Each neuron computes an output based on the weighted sum of the inputs and the applied activation function. The weights of the connections between the neurons are adjusted during training to minimize the error in predictions. This optimization is typically performed using the methods such as backpropagation and gradient descent. Mathematically, an ANN model can be represented as follows:

$$y_t = f\left(\sum_{j=1}^{q} \omega_j g\left(\sum_{i=1}^{p} \omega_{ij} y_{t-i}\right)\right)$$

where y_t is the observation at time t, ω_j(j=1,2,…q) and ω_{ij}(i=1,2…p, j=1,2,…q) represent the connection weights, which are the parameters of the model. Here, p refers to the number of input nodes, while q indicates the number of hidden nodes. The functions g and f correspond

to the activation functions utilized in the hidden and output layers, respectively which help introduce non-linearity into the model.

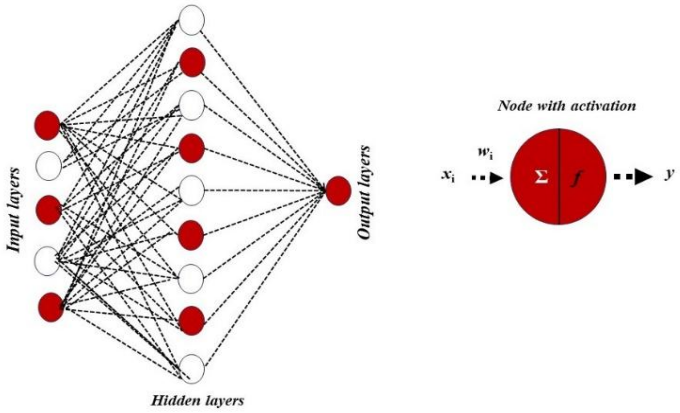

Fig. 6.3 Artificial Neural Networks

Support Vector Regression

SVR is an effective machine learning technique used for regression tasks. It is particularly well-suited for capturing non-linear patterns in data by the use of kernel functions (Su et al. 2022). It works by transforming input data into a high-dimensional feature space using kernel functions, enabling it to address non-linearity in the data. The primary objective of SVR is to identify a function that approximates the target values while ensuring that the deviation from the true values does not exceed a specified threshold (ε). This is achieved through an optimization process that strikes a balance between the complexity of the model and the training error (Bargam et al. 2024). Support Vector Regression model is represented by the equation:

$$f(x)=w^T\phi(x) + b$$

where w is the weight vector, x is the input feature vector, b is the bias term, and $\phi(x)$ is the non-linear mapping function, also referred to as the kernel function. The kernel function allows the SVR model to handle non-linear relationships by transforming the input space into a higher-dimensional feature space.

Random Forest

Random Forest is a versatile ensemble learning method widely used for classification and regression problems in machine learning (Portoles et al. 2018). It works by constructing multiple decision trees during training, each built using a randomly selected subset of the dataset and features (Couronne et al. 2018). This randomness in feature selection reduces the correlation between trees and enhances the model generalization. The RF algorithm employs a technique known as bagging where several subsets of the training data are created, and a decision tree is trained on each subset (Biau and Fr 2012). The final output for regression tasks is obtained by averaging the predictions of all trees. Mathematically, the prediction $\hat{y}$ in a Random Forest can be represented as:

$$\hat{y} = \frac{1}{m} \sum_{j=1}^{m} T_j(x)$$

where m is the total number of trees in the forest and $T_j(x)$ is the prediction made by j^{th} tree for input x. In most research studies, it is widely accepted that increasing the number of trees generally does not decrease predictive performance, although it does lead to higher computational costs (Tyralis and Papacharalampous 2017) . This approach enhances prediction accuracy while minimizing the risk of overfitting, making Random Forest a widely used

Convolutional Neural Networks

CNNs are a specialized class of deep learning algorithms designed for processing structured grid data, such as images. CNNs excel at automatically learning spatial hierarchies of features, making them ideal for tasks like image classification, object detection, and video analysis. The core components of a CNN include the input layer, which receives the time series data, convolutional layers that apply kernels to capture patterns within the data, activation functions such as ReLU or Tanh that introduce non-linearity, pooling layers that reduce the data dimensions to minimize computational load and prevent overfitting, and fully connected layers that combine the extracted features to generate final predictions for forecasting (Mehtab and Sen 2020). It is visually represented in **Fig. 6.4**.

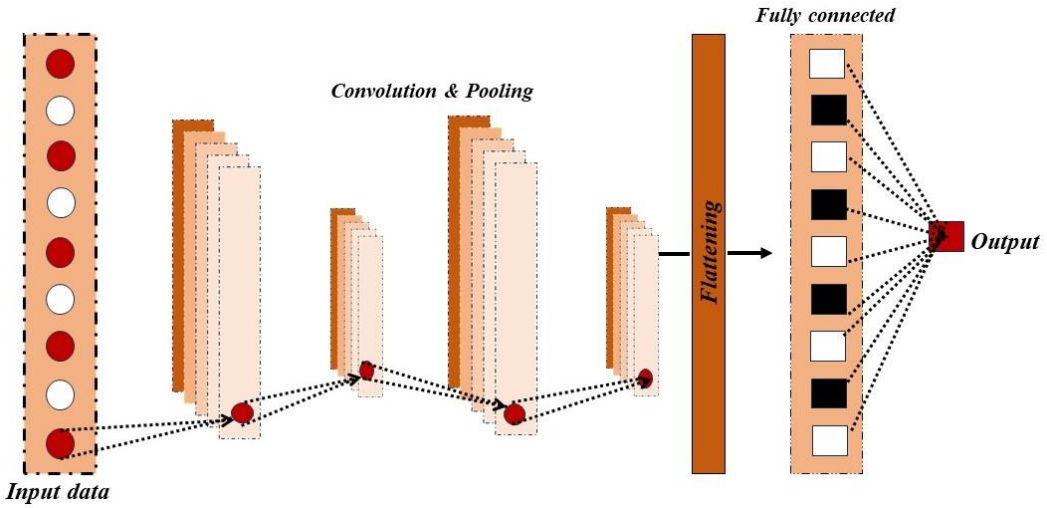

Fig. 6.4 Convolutional Neural Network

The convolution operation in CNNs is mathematically expressed as

$O_t = \tanh(x_t * k_t + b_t)$

where O_t is the convolved output, x_t is the input vector, k_t represents the weights of the convolutional kernel, and b_t is the bias. By using these components, CNNs efficiently detect patterns, reduce dimensionality, and extract essential features to perform complex tasks.

Recurrent Neural Networks

RNNs are specialized neural networks that are designed to handle sequential data by maintaining a hidden state that acts as a form of memory, capturing the temporal dependencies present in the input data (Kurumatani 2020). With each time step, the hidden state is updated using the current input and the previous hidden state, allowing the network to store past information and influence future predictions (Ugurlu et al. 2018). The schematic representation of RNN is illustrated in **Fig. 6.5.** To overcome challenges like vanishing gradients, advanced RNN architectures such as LSTM and GRUs have been developed. These models are well-suited for tasks requiring long-term memory, such as language modeling, speech recognition, and time series forecasting.

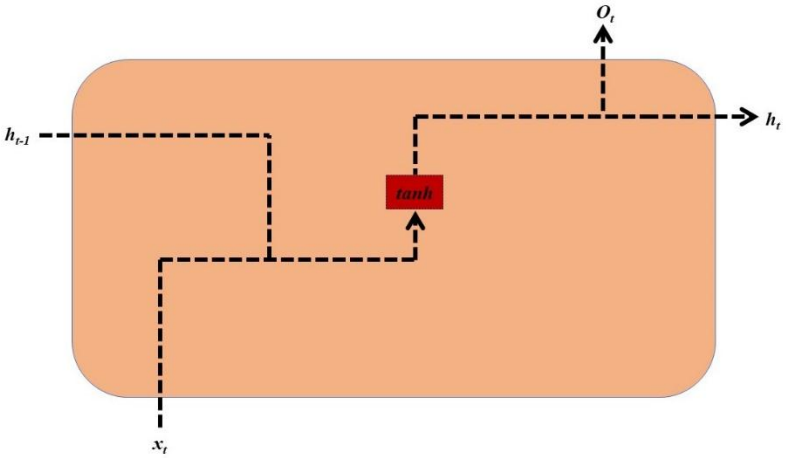

Fig. 6.5 Recurrent Neural Network

Key components of RNNs include recurrent units that maintain the hidden state, which is updated by combining the previous hidden state with the current input. The network also has input and output layers, where the input layers receive sequential data, and the output layers generate predictions. RNN architectures can vary, supporting different mappings such as one-to-one, one-to-many, many-to-one, or many-to-many, based on the specific task. Advanced architectures like LSTMs and GRUs introduce gating mechanisms such as input, forget, and output gates that help control the flow of information, enhancing the model's ability to learn long-term dependencies. Furthermore, RNNs employ common parameters across time steps within the same layer, enabling better generalization and ensuring consistent learning through backpropagation (Ismailova et al. 2024). The mathematical representation of RNN is

$h^t = f(h^{t-1}, x^t ; \theta)$

where h^t is the hidden state time step t, x^t is the input at time step t , f is the activation function and θ represents the parameters of RNN.

Long Short-Term Memory

LSTM networks are specialized type of Recurrent Neural Network that are designed to effectively capture and use temporal dependencies in sequential data (Malashin et al. 2024).

LSTMs address significant challenges faced by traditional RNNs, including the vanishing and exploding gradient issues, which restrict their capacity to learn long-range dependencies in time series data. One of the key characteristics of LSTMs is their use of memory cells, which enable the network to retain information over long durations. This is facilitated by the cell state, which acts as a pathway for the network to control the flow of data, allowing it to decide when to retain or discard the information (Sabbu and Ganesan 2024). The key components of an LSTM include three main gates namely the input gate (i_t), which regulates the addition of new information to the cell state using a sigmoid function, the forget gate (f_t), which manages the removal of irrelevant information and the output gate (o_t), which determines which part of the cell state should be output at each time step, influencing the hidden state used in predictions (Ji et al. 2022). It is visually represented in **Fig. 6.6.**

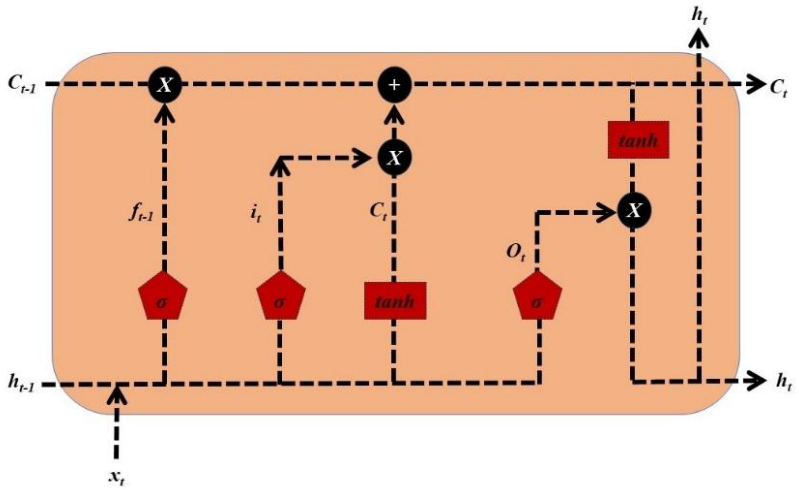

Fig. 6.6 Long Short-Term Memory

The operations of LSTM network are represented as follows

$f_t = \sigma [W_f * (h_{t-1} , x_t) + b_f]$

$i_t = \sigma [W_i * (h_{t-1} , x_t) + b_i]$

$o_t = \sigma [W_o * (h_{t-1} , x_t) + b_o]$

$c_t = \tanh [W_c * (h_{t-1} , x_t) + b_c]$

$h_t = o_t * \tanh (c_t)$

where x_t represents the current input, h_{t-1} is the previous hidden state, σ denotes the sigmoid activation function. The terms b_f, b_i, b_o correspond to the biases for the forget, input, and output gates, respectively. The weights associated with the forget, input, and output gates are denoted by W_f, W_i, W_o respectively. c_t is the cell state vector and h_t represents the output vector. These features enable LSTMs to overcome the limitations of traditional RNNs and excel at tasks that require learning long-term dependencies, such as language modeling, speech recognition, and time series forecasting.

Gated Recurrent Unit

GRU is a type of recurrent neural network designed to that is designed to handle sequential data effectively. It addresses common challenges faced by traditional RNNs, such as gradient explosion and vanishing gradients, which hinder the learning of long-term dependencies (Yu et al. 2023). GRUs are popular in time series analysis due to their efficiency and strong performance. They utilize two main gates namely the reset gate which determines the amount of past information to discard, and the update gate, which controls how much new information should be added to the current state (Chen et al. 2023). Both GRU and LSTM work in a similar way, but the GRU simplifies the structure by combining the forget and input gates into update gate. It uses a single state that includes both the hidden and cell states, making the model simpler and speeds up training. GRUs maintain a memory state that captures dependencies in sequential data, making them ideal for applications such as time series forecasting, stock price prediction, and natural language processing (Sari et al. 2022) . Their ability to efficiently model long-term dependencies has led to widespread use in various fields, including finance and weather forecasting. GRU architecture is illustrated in **Fig.6.7.**

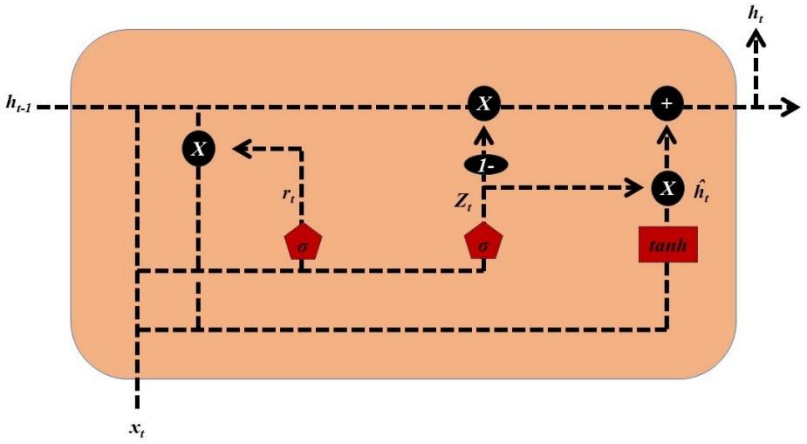

Fig. 6.7 Gated Recurrent Unit

Mathematically, the GRU can be represented as follows:

$z_t = \sigma [W_z * x_t + U_z * h_{(t-1)} + b_z]$

$r_t = \sigma [W_r * x_t + U_r * h_{(t-1)} + b_r]$

$\tilde{h}_t = \tanh [W_h * x_t + (r_t . U_h * h_{(t-1)} + b_h)$

$h_t = z_t * h_{(t-1)} + (1-z_t) * \tilde{h}_t$

where x_t refers to the input vector at time step t $h_{(t-1)}$ represents the previous vector cell value state, z_t is the update gate, r_t is reset gate, σ represents the sigmoid activation function. $\tilde{h}_t$ is the output candidate from cell state vector, W_z is update gate weight, U_z is hidden state weight, W_z is update gate weight, U_z is hidden state weight, Wr is reset gate weight, U_r is hidden state weight, W_h is cell state vector weight, U_h is hidden state weight, b_z,b_r, b_h are biases associated with update gate, reset gate, and cell state vector respectively. h_t is the cell state vector at time step t algorithm in research and real-world applications.

Hybrid Methodology

Time series data are characterized by their complex structures which could possess the properties of linearity, non-linearity, or a combination of both at a time, including the volatility in their residuals. Combining two or more time series models can help in addressing the limitations of individual approaches and capture the different aspects of the data. The hybridization of time series techniques is a sophisticated process that necessitates careful consideration of the strengths and limitations of each technique. By combining complementary methods, the hybrid approach can produce more accurate and robust forecasts, leading to better-informed decision-making.

Arma-Garch

ARCH (Autoregressive Conditional Heteroskedasticity) and GARCH (Generalized Autoregressive Conditional Heteroskedasticity) are two popular models used for modeling and forecasting volatility in time series (Andersen *et al*. 2014). They are designed to capture the heteroscedasticity or time-varying variance in the residuals. ARCH models assume that the conditional variance of the time series is a function of its past squared residuals or errors. In other words, the volatility of the series depends on its history. The GARCH model is the extended version of the ARCH framework that incorporates lagged conditional variances in the model. This allows for more flexible and accurate modeling of volatility in the data. GARCH models are specified by two parameters: the order of the Autoregressive process (p) and the order of the Moving Average process (q). The GARCH model with mean and variance equation can be represented as follows:

Mean equation: $y_t = \mu + \varepsilon_t$

Variance equation: $\sigma_t^2 = \alpha_0 + \alpha_1 \ \varepsilon_{t-1}^2 + \beta_1 \ \sigma_{t-1}^2$

Where y_t is the observed value at time t,

μ is the mean of the time series,

ε_t is the error term at time t, and

σ_t^2 is the conditional variance of the error term at a time with non-negative α and β terms.

GARCH model assumes that the conditional variance of the error term at time t depends on both the squared error term at time t-1 and the conditional variance of the error term at time t-1. Checking the heteroscedasticity in the squared residuals of the ARIMA or mean model is the first and most important step before proceeding with volatility model fitting. The GARCH model is fitted only if heteroscedasticity is confirmed. The ARCH-LM test is a Lagrange Multiplier test used to determine the presence of ARCH effects in a time series. The test is based on regressing the squared residuals from a given model onto their lagged values. The null hypothesis of the ARCH-LM test is that there is no ARCH effect in the residuals. The formula for the ARCH-LM test statistic is:

LM test = nR^2

Where n is the sample size. R^2 is the coefficient of determination from the auxiliary regression of the squared residuals onto their p-lagged values.

ARMA-GARCH is a combination of Autoregressive Moving Average and GARCH models. ARMA models are used for modeling the mean of the time series, while GARCH

models are used for modeling the volatility (Bawa *et al*. 2021). ARMA-GARCH models are specified by four parameters: the orders of the Autoregressive process (p) and Moving Average process (q) for the mean component, and the orders of the Autoregressive process (r) and the Moving Average process (s) for the volatility component. The mean equation of the ARMA-GARCH model is,

$y_t = \emptyset_1 \, y_{t-1} + \ldots + \emptyset_p * y_{t-p} + \theta_1 \; \varepsilon_{t-1} + \ldots + \theta_q \, \varepsilon_{t-q}$

Where $\emptyset_p$ and θ_q are the coefficients of the AR and MA terms.

The variance equation will be the same as the GARCH model. The best-fitted model was selected based on the least values of AIC and BIC.

ARIMA-ANN

When the data under consideration comprises both linear and nonlinear information, applying either ARIMA or ANN is not advisable. In this case, the hybrid methodology of combining ARIMA and ANN is appreciable (Rathod and Mishra, 2018). The general form of hybrid methodology is given by,

$y_t = L_t + N_t$

Where L_t and N_t are the linear and non-linear components present in the time series y_t. Initially, the linearity is accounted for using the ARIMA model. Let $\hat{L}_t$ be the predicted values of the linear model and the residual at time t is given by $e_t = y_t - \hat{L}_t$. The non-linearity left in the residuals of the fitted model was checked by conducting a two-tailed Brock-Dechert-Scheinkman (BDS) test. After confirming the presence of non-linearity in the residuals of the fitted ARIMA model, the residuals are fitted using the ANN model (Paul *et al*. 2020). If $\hat{N}_t$ is predicted series from ANN and then the ARIMA-ANN model at time t is obtained by,

$\hat{y}_t = \hat{L}_t + \hat{N}_t$

Similarly, the forecast can be also attained. The working flow chart of the hybrid model is given in **Fig..6.8**. Similarly, many hybrid models can be developed with ML and DL models.

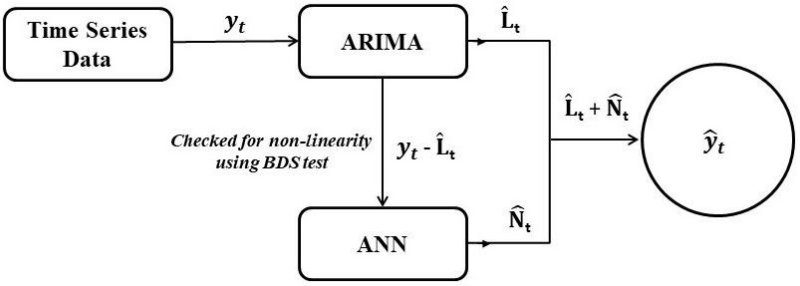

Fig 6.8. Working flow chart of the hybrid model

Evaluation Metrics

The accuracy of the models is assessed by comparing the actual values with the predicted values. To evaluate model performance, several metrics such as Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), Root Mean

Square Percentage Error (RMSPE) and Median Absolute Percentage Error (Median APE) are employed. Models that result in lower values for these metrics are considered the most accurate, as they more closely align with the observed data. The error metrics used are defined as follows:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2}$$

$$\text{MAPE} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right| \times 100$$

$$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n}|y_t - \hat{y}_t|$$

$$\text{RMSPE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{y_i}\right)^2}$$

$$\text{Median APE} = \text{median}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100$$

where y_t is the actual values, $\hat{y}_t$ is the forecasted values and n is the number of observations in the time series.

Conclusion

Despite substantial progress, forecasting continues to be a challenging task. Every modelling approach whether statistical, machine learning, deep learning, or hybrid that comes with its own strengths and limitations. Some models excel at capturing linear patterns, others perform better with nonlinear behaviour, and some require large datasets or strict assumptions. Because of these inherent trade-offs, no single model consistently outperforms others across all types of time series. Ultimately, it is the nature and structure of the data, its variability, trend, seasonality, noise level, and non-stationarity, that determines which model is most suitable. This reality drives the constant development of new modelling strategies and improved frameworks aimed at enhancing accuracy, robustness, and adaptability.

Disclosure Statement

The authors reported no potential conflict of interest.

References

1. Alharbi, F. R., & Csala, D. (2022). A seasonal autoregressive integrated moving average with exogenous factors (SARIMAX) forecasting model-based time series approach. *Inventions, 7*(4), 94.
2. Areef, M., Rajeswari, S., Vani, N., & Naidu, G. M. (2020). Forecasting of onion prices in Bangalore market: An application of time series models. *Indian Journal of Agricultural Economics, 75*(2), 217–227.
3. Bargam, B., Boudhar, A., Kinnard, C., Amazirh, A., & Maanan, M. (2024). Evaluation of the support vector regression (SVR) and the random forest (RF) models accuracy for streamflow prediction under a data-scarce basin in Morocco. *Discover Applied Sciences, 6*, 1–17. https://doi.org/10.1007/s42452-024-05994-z

4. Bawa, M. U., Dikko, H. G., Shabri, A., Garba, J., & Sadiku, S. (2021). Forecasting performance of hybrid ARIMA-FIGARCH model and hybrid of ARIMA-GARCH model: A comparative study. *Journal of Mathematical Problems Equations and Statistics, 2*(2), 48–58.
5. Biau, G. (2012). Analysis of a random forests model. *The Journal of Machine Learning Research, 13,* 1063–1095.
6. Boxwell, P. J., & Davis, R. A. (1991). *Time series: Theory and methods* (2nd ed.). Springer.
7. Chadsuthi, S., Modchang, C., Lenbury, Y., Iamsirithaworn, S., & Triampo, W. (2012). Modeling seasonal leptospirosis transmission and its association with rainfall and temperature in Thailand using time–series and ARIMAX analyses. *Asian Pacific Journal of Tropical Medicine, 5*(7), 539–546.
8. Chen, C., Xue, L., & Xing, W. (2023). Research on improved GRU-based stock price prediction method. *Applied Sciences, 13*(15), 8813. https://doi.org/10.3390/app13158813
9. Couronné, R., Probst, P., & Boulesteix, A.-L. (2018). Random forest versus logistic regression: A large-scale benchmark experiment. *BMC Bioinformatics, 19,* 1–14. https://doi.org/10.1186/s12859-018-2264-5
10. Dabral, P. P., & Murry, M. Z. (2017). Modelling and forecasting of rainfall time series using SARIMA. *Environmental Processes, 4*(2), 399–419.
11. Dickey, D. A. (2015). Stationarity issues in time series models. *SAS Users Group International, 30,* 19.
12. Dutta, J., & Roy, S. (2021). IndoorSense: Context based indoor pollutant prediction using SARIMAX model. *Multimedia Tools and Applications, 80*(13), 19989–20018.
13. Enders, W. (2008). *Applied econometric time series.* John Wiley & Sons.
14. Gujarati, D. N., & Porter, D. C. (2009). *Basic econometrics.* McGraw-Hill/Irwin.
15. Hipel, K. W., & McLeod, A. I. (1994). *Time series modelling of water resources and environmental systems.* Elsevier Science Publishers B.V.
16. Hipel, K. W., McLeod, A. I., & Lennox, W. C. (1977). Advances in Box-Jenkins modeling: 1. Model construction. *Water Resources Research, 13*(3), 567–575.
17. Horvath, L., Kokoszka, P., & Rice, G. (2014). Testing stationarity of functional time series. *Journal of Econometrics, 179*(1), 66–82.
18. Ismailova, A., Beldeubayeva, Z., Kadirkulov, K., & Turlybekova, A. (2024). Forecasting stock market prices using deep learning methods. *International Journal of Electrical & Computer Engineering, 14*(3), 3290–3298.
19. Jadhav, V., Chinnappa Reddy, B. V., & Gaddi, G. M. (2017). Application of ARIMA model for forecasting agricultural prices. *Journal of Agricultural Science and Technology, 19*(5), 981–992.
20. Ji, H., Lou, Y., Cheng, S., Ren, W., Jin, S., & Li, C. (2023). An advanced long short-term memory (LSTM) neural network method for predicting rate of penetration (ROP). *ACS Omega, 8*(1), 934–945. https://doi.org/10.1021/acsomega.2c06308
21. Jordaan, H., Grove, B., Jooste, A., & Alemu, Z. G. (2007). Measuring the price volatility of certain field crops in South Africa using the ARCH/GARCH approach. *Agrekon, 46*(3), 306–322.

22. Kurumatani, K. (2020). Time series forecasting of agricultural product prices based on recurrent neural networks and its evaluation method. *SN Applied Sciences, 2,* 1434. https://doi.org/10.1007/s42452-020-03225-9
23. Lim, B., & Zohren, S. (2021). Time-series forecasting with deep learning: A survey. *Philosophical Transactions of the Royal Society A, 379*(2194), 20200209.
24. Malashin, I., Tynchenko, V., Gantimurov, A., & Tynchenko, S. (2024). Applications of long short-term memory (LSTM) networks in polymeric sciences: A review. *Polymers, 16*(18), 2607.
25. Manigandan, P., Alam, M. S., Alharthi, M., Khan, U., Alagirisamy, K., Pachiyappan, D., & Rehman, A. (2021). Forecasting natural gas production and consumption in United States—Evidence from SARIMA and SARIMAX models. *Energies, 14*(19), 6021.
26. Manuca, R., & Savit, R. (1996). Stationarity and nonstationarity in time series analysis. *Physica D: Nonlinear Phenomena, 99*(2–3), 134–161.
27. Mapuwei, T. W., Ndava, J., Kachaka, M., & Kusotera, B. (2022). An application of time series ARIMA forecasting model for predicting tobacco production in Zimbabwe. *American Journal of Modeling and Optimization, 9*(1), 15–22.
28. Mehtab, S., & Sen, J. (2020, October). Stock price prediction using CNN and LSTM-based deep learning models. In *2020 International Conference on Decision Aid Sciences and Application (DASA)* (pp. 447–453). IEEE.
29. Mila, F. A., & Parvin, M. (2019). Forecasting area, production and yield of onion in Bangladesh by using ARIMA model. *Asian Journal of Agricultural Extension, Economics & Sociology, 37*(4), 1–12.
30. Naveena, K., & Subedar, S. (2017). Hybrid time series modelling for forecasting the price of washed coffee (Arabica plantation coffee) in India. *International Journal of Agriculture Sciences.*
31. Paul, R. K., Vennila, S., Yadav, S. K., Bhat, M. N., Kumar, M., Chandra, P., Paul, A. K., & Prabhakar, M. (2020). Weather based forecasting of sterility mosaic disease in pigeon pea *Cajanus cajan* using machine learning techniques and hybrid models. *Indian Journal of Agricultural Sciences, 90*(10), 1952–1958.
32. Pórtoles, J., González, C., & Moguerza, J. M. (2018). Electricity price forecasting with dynamic trees: A benchmark against the random forest approach. *Energies, 11*(6), 1588. https://doi.org/10.3390/en11061588
33. Praveen, B., & Sharma, P. (2020). Climate variability and its impacts on agriculture production and future prediction using autoregressive integrated moving average method (ARIMA). *Journal of Public Affairs, 20*(2), e2016.
34. Rathod, S., & Mishra, G. C. (2018). Statistical models for forecasting mango and banana yield of Karnataka India. *Journal of Agricultural Science and Technology, 20*(4), 803–816.
35. Sabbu, S., & Ganesan, V. (2024). A robust LSTM model for video detection and classification: An optimization procedure. *International Journal of Intelligent Systems and Applications in Engineering, 12*(7s), 505–511.

36. Sari, Y., Firmanul Arifin, Y., Novitasari, N., & Faisal, M. R. (2022). Deep learning approach using the GRU-LSTM hybrid model for air temperature prediction on daily basis. International Journal of Intelligent Systems and Applications in Engineering, 10(3), 430–436.
37. Sendhil, R., Chandrasekar, V., Sang, L. L. M., & Joseph, J. (2024). Forecasting methods–An overview. In Basic econometric tools and techniques in data analytics (p. 139). Notion Press.
38. Sirisha, U. M., Belavagi, M. C., & Attigeri, G. (2022). Profit prediction using ARIMA, SARIMA and LSTM models in time series forecasting: A comparison. IEEE Access, 10, 124715–124727.
39. Su, X., He, X., Zhang, G., Zhang, Y., & Wei, M. (2022). Research on SVR water quality prediction model based on improved sparrow search algorithm. Computational Intelligence and Neuroscience, 2022, 7327072. https://doi.org/10.1155/2022/7327072
40. Tealab, A. (2018). Time series forecasting using artificial neural networks methodologies: A systematic review. *Future Computing and Informatics Journal, 3*(2), 334–340.
41. Thiyagarajan, S., Naresh, G., & Mahalakshmi, S. (2015). Forecasting volatility in Indian agri-commodities market. *Global Business & Finance Review, 20*(1), 95–104.
42. Tyralis, H., & Papacharalampous, G. (2017). Variable selection in time series forecasting using random forests. Algorithms, 10(4), 114. https://doi.org/10.3390/a10040114
43. Ugurlu, U., Oksuz, I., & Tas, O. (2018). Electricity price forecasting using recurrent neural networks. Energies, 11(5), 1255. https://doi.org/10.3390/en11051255
44. Wangdi, K., Singhasivanon, P., Silawan, T., Lawpoolsri, S., White, N. J., & Kaewkungwal, J. (2010). Development of temporal modelling for forecasting and prediction of malaria infections using time-series and ARIMAX analyses: A case study in endemic districts of Bhutan. Malaria Journal, 9, 251.
45. Yu, Z., Sun, Y., Zhang, J., & Huang, J. (2023). Gated recurrent unit neural network (GRU) based on quantile regression (QR) predicts reservoir parameters through well logging data. Frontiers in Earth Science, 11: 1087385.