Comparative Study of Time-Series Forecasting Models for Wind Power Generation in Gujarat, India

Sulagna Mahata 1, Piyush Harsh 2, and Vineet Shekher 2

1BIT Sindri
2Affiliation not available

December 7, 2023

Abstract

The rapid rate of transformation in the power sector of India has placed a significant emphasis on robust grids and distributed generation units. The observable shift in the energy sector, especially in wind and solar energy, also requires smooth integration of Distributed Generation units with the existing power grid. Precise wind power generation forecast, therefore, becomes an important and complex task for the strategic planning and management of the systems. We, thus, aim towards a system that can actually provide precise wind power forecasts by applying machine learning techniques. This work proposes a comparative and comprehensive analysis of Artificial Neural Network (ANN), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Autoregressive Integrated Moving Average (ARIMA) model. The experimentations and modelling are performed considering meteorological and historical power generation data. The study is concentrated in Kutch, Gujarat and is validated on the data collected from the Central Electricity Authority (CEA), India for power generation data and weather data collected from regional weather centres. The findings show that ARIMA outperforms the other models for non-linear data in multivariate analysis, with a MAPE score of 5.87 on the prediction dataset.
Comparative Study of Time-Series Forecasting Models for Wind Power Generation in Gujarat, India

Piyush Harsh\textsuperscript{a} *, Sulagna Mahata\textsuperscript{a}, Vineet Shekher\textsuperscript{a}

\textsuperscript{a}Department of Electrical Engineering, BIT Sindri, Dhanbad, Jharkhand, India

Abstract

The rapid rate of transformation in the power sector of India has placed a significant emphasis on robust grids and distributed generation units. The observable shift in the energy sector, especially in wind and solar energy, also requires smooth integration of Distributed Generation units with the existing power grid. Precise wind power generation forecast, therefore, becomes an important and complex task for the strategic planning and management of the systems. We, thus, aim towards a system that can actually provide precise wind power forecasts by applying machine learning techniques. This work proposes a comparative and comprehensive analysis of Artificial Neural Network (ANN), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Autoregressive Integrated Moving Average (ARIMA) model. The experimentations and modelling are performed considering meteorological and historical power generation data. The study is concentrated in Kutch, Gujarat and is validated on the data collected from the Central Electricity Authority (CEA), India for power generation data and weather data collected from regional weather centres. The findings show that ARIMA outperforms the other models for non-linear data in multivariate analysis, with a MAPE score of 5.87 on the prediction dataset.

Keywords: Wind Energy; Wind Power Forecasting; Artificial Neural Network; Recurrent Neural Network; Long Short-Term Memory; Gated Recurrent Unit; Autoregressive Integrated Moving Averages; MAPE; Power Systems

1. Introduction

Increasing levels of pollution and depleting natural non-renewable energy sources have brought the attention of humankind towards renewable energy sources. Fossil fuels, since the Industrial Revolution have pumped the human economy and society. But with the population explosion, the energy demand has surged exponentially, and renewable energy provides a sustainable alternative to fossil fuels. In a developing nation like India, energy is crucial and can be game-changing in the years to come. Solar, wind, geothermal, and nuclear energy can be developed as the backbone of India's power and energy sectors.

The wind energy sector has witnessed a rapid advancement in India in recent times. India ranks fourth in installed wind power capacity, after China, and the USA followed by Germany with 43.94 GW of wind power installed [1]. Wind energy accounts for about 10% of India’s total installed energy generation capacity and is the oldest developed renewable energy technology [2]. According to the National Institute of Wind Energy's evaluation of the nation's wind resources, there is a potential for wind energy of roughly 302 GW at 100 metres, 695.5 GW at 120 metres and 1,164 GW at 150 metres above ground [3].

The state of Gujarat has a potential of about 142.56 GW of wind power potential at 120 metres AGL, preceded by Rajasthan, Karnataka, and Maharashtra. While Tamil Nadu ranks first in the nation with the highest installed capacity, Gujarat follows the second rank in terms of installed capacity [4].

Renewable energy (RE) resources, especially wind and solar energy, are rapidly rising in India to meet the country's growing energy demands. With around 423.36 GW of installed capacity [5], the Indian power grid is one of the world's largest synchronised networks. India is an emerging economy with 17.76% world population which implies there's a huge energy demand in all sectors. To meet the energy requirements, the nation is fostering the RE industry, especially Solar and Wind energy sources. The large coastline and windy terrains offer an ideal spot for wind power generation. Moreover, with advancing technology there has been a decline in the cost of wind power, making it more competitive and affordable. Wind power will not only reduce India’s reliance on fossil fuels but also help generate new jobs and improve energy security.

Being a variable energy source, wind energy is surrounded by a question of its reliability. Thus, to incorporate wind energy sources into the existing grid, it is critical to be able to forecast the power generation. If not forecasted
properly, the power grid may face problems like blackouts or brownouts. Thus, precise forecasting is required for a healthy grid integration. Load forecasting helps power companies estimate and plan for new generation and transmission capacity to meet the demands. Load forecasting is useful for controlling power grid operations and scheduling power plants, which reduces emissions and operational costs.

Short-Term Load Forecasting (STLF), Medium-Term Load Forecasting (MTLF), and Long-Term Load Forecasting (LTLF) are classified according to the time horizon. The forecasting of load and energy has become even more complex and has managed to predict more accurately over time. Popularly used prediction models are distinguished as statistical models which include Autoregressive (AR), Autoregressive Integrated Moving Average (ARIMA), exponential models, machine learning models and hybrid models [6].

Fig. 1: Map of India highlighting Gujarat and the Kutch region.

India's rich geography and large coastline add to the advantage of India's future energy reliance. With a promising potential in Gujarat, this study is concentrated on forecasting wind power generation in this region. The weather factors like air density, ambient temperature, humidity, and wind speed in the region favour the establishment of wind farms. In Gujarat, the administrative district of Kutch records an average wind speed of more than 8 m/s according to the National Renewable Energy Laboratory (NREL) [7]. The geographical features like the Arabian Sea and the Gulf of Kutch also influence the generation of wind energy.

2. Literature Survey

Time series forecasting techniques are based on analysis of historical data, to forecast future patterns [8]. Load forecasting is an essential stage in the management and planning of utilities in power systems. Numerous techniques have been developed and put into use for predicting. AMF models (Average Meteorological Forecast) are numerical weather prediction models that are used to calculate wind power after estimating wind speed. Correlation in Space Models relies on the spatial connections of wind speed at various locations [9]. Auto-Regressive (AR), Auto-Regressive Moving Average (ARMA), and Auto-Regressive Integrated Moving Average (ARIMA) statistical models are used to find association between online measured power data and offline data [8].

Intelligent forecasting models are receiving more and more attention as machine learning and artificial intelligence advance. These models offer enormous potential for handling nonstationary wind energy series because of the robustness and efficacy of intelligent techniques. When dealing with issues that cannot be
characterised analytically, the intelligent forecasting model can accomplish higher forecasting steps and get superior accuracy compared to physical and statistical approaches [10].

Adrian-Nicolae Buturache et al. provided a fine-tuned ANN model for properly forecasting wind energy projections, but it took time and processing power to identify the best hyperparameter combination [11]. In contrast to ANN, which required optimisation and fine-tuning to produce the best results, regression tree models, like SVR, gave simplicity. These tree-based models performed best when just one statistic was used [11]. Since weather conditions have a big impact on energy output, wind energy is a rather unpredictable alternative. As a result, single metric models cannot be used for accurate estimation. For the purpose of forecasting wind speed and wind power, Jung et al. analysed the physical, statistical, spatial correlation, and regional forecasting models [12].

ARIMA model was utilised by Ernesta Grigonyte et al. to forecast wind speed in the Baltic region for a brief period based on previous observations. Despite the authors’ best efforts, a season-specific framework could not be achieved [13]. Based on information from Western Australia, Eddie Yatiyana et al. projected wind speed and direction using ARIMA independently. Ilham Tyass et al. obtained a forecast error of about 16% on an hour ahead forecast, based on the Seasonal ARIMA model for the Morocco region [14]. Similarly, Bri-Mathias Hodge et al. [15], Jing Shi et al. [16], and Pei Du et al. [17] used statistical models to anticipate short-term wind power, finding comparable results with roughly 10% error percentage.

<table>
<thead>
<tr>
<th>Forecasting Technique</th>
<th>Remarks</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deep Learning Models (RNN, LSTM, GRU)</td>
<td>Handle long temporal dependencies well, GRU and LSTM perform well when fine-tuned and optimised</td>
<td>[28]</td>
</tr>
<tr>
<td>ANN – based method</td>
<td>In comparison to other machine learning like SVR, Random Forest, ANN is complex and attains a better accuracy in forecasting</td>
<td>[11], [27]</td>
</tr>
<tr>
<td>Statistical Models (ARIMA, SARIMA)</td>
<td>Statistical models like ARIMA provide a reliable output, utilise the wind data for forecasting</td>
<td>[13] – [16]</td>
</tr>
<tr>
<td>Hybrid Models</td>
<td>Provides a robust prediction combining the features of the models, complexity and computation is different model to model</td>
<td>[17] – [18]</td>
</tr>
</tbody>
</table>

A hybrid LSTM-CNN model was put forth by Xiaojiao Chen et al., who used data from a Chinese offshore wind farm to validate it. Their team recorded lower MSE and MAE scores in comparison to LSTM, CNN and SVM models [18]. To address the complex problem of wind energy forecasting, numerous hybrid forecasting techniques, notably models based on artificial forecasters, have been created. Hybrid models are a concept that is not entirely clear. In general, a hybrid method is a blend of two different techniques or methodologies. Hybrid models integrate the benefits and characteristics of various approaches, and their total performance outperforms single models as a whole [19].

2.1. Organisation of the paper

The authors have concentrated on a thorough examination of time-series forecasting models since the most efficient model could help to develop a system to strategize and energy management. The topic of interest and the current energy situation in India are introduced in Section 1, and the remainder of the paper is organised as follows: Section 2 discusses the recent publications in this area while Section 3 lays out the methodology and sets on the theoretical backgrounds for the various models used. In Section 4, the data has been carefully examined and graphically represented. Section 5 summarises the experiment's findings, and Section 6 wraps up the study and examines the findings.

3. Wind Forecasting Techniques

In our work, we propose a comprehensive study of time-series models, namely Long Short-Term Memory (LSTM) networks, Artificial Neural Networks (ANN), Recurrent Neural Networks (RNN), Gated Recurrent Units
Deep learning techniques have recently been applied for wind energy prediction after a rapid development in recent years. Historically, single-variate time series models have shown statistical models like ARIMA to be stable and perform well. Deep learning and statistical models outperform more conventional machine learning techniques in terms of feature extraction and model generalisation. Among typical deep learning methods, Recurrent Neural Networks (RNN) show excellent performance when dealing with time series problems, while Long Short-Term Memory Networks (LSTM) [22] and Gated Recurrent Units (GRU) are advanced methods of RNN that better deal with the vanishing gradient issue of RNN. ANN can be used to model complex nonlinear patterns and prediction problems [23] and ARIMA has been a widely used forecasting model due to its simplicity and its ability to generalize for non-stationary series.

Fig. 2 is used to visualize the overall flow for the wind power generation model. The workflow can be divided into following phases - the data preparation phase, the training phase, and the testing phase. In the data preparatory phase, the data is extracted and collected to form a dataset wherein different features and parameters are organized on a daily basis. This data is used for analysing the trend, generating meaningful graphs, and highlighting key points from the data. The data is eventually normalised and prepared for the training phase in this stage. In the training phase, the model is created, fine-tuned, and trained. The dataset is split into training, validation, and testing subsets before the model is trained. During the training phase, training and validation sets are used and model performance is evaluated. Subsequently, in the testing phase, the predictions are made, and the actual forecast is made.

3.1. Autoregressive Integrated Moving Averages (ARIMA)

ARIMA is a non-stationary time series statistical model that is commonly used for short-term forecasting. George Box and Gwilym Jenkins proposed the ARIMA model as an extension of the autoregressive moving average (ARMA) model in their 1970 book "Time Series Analysis: Forecasting and Control" [24]. Fig. 3 shows the algorithm of the ARIMA model.

ARIMA models are built on the ideas of autoregressive (AR), moving average (MA), and differencing (I). The AR component simulates the linear relationship between the current and prior values. The MA component models
the error term as a linear combination of prior error terms, but the I component handles the non-stationary character by differencing it until it becomes stationary. The order of the autoregressive model (the number of time lags) is denoted by p, the degree of differencing is denoted by d, and the order of the moving average portion is denoted by q.

Mathematically, Autoregressive (AR) models are formulated as:

\[ \gamma_t' = C + \phi_1 \gamma_{t-1}' + \phi_2 \gamma_{t-2}' + \ldots + \phi_p \gamma_{t-p}' + e_t \]  

(1)

And Moving Average (MA) components are formulated as follows:

\[ \gamma_t' = C + \psi_1 \epsilon_{t-1} + \psi_2 \epsilon_{t-2} + \ldots + \psi_q \epsilon_{t-q} + e_t \]  

(2)

Where,

\( \gamma_t' \) where is the differenced series (it may have been differenced more than once),

‘p’ is the auto-regressive component,

‘q’ is the moving-average component,

\( e_t \) is white noise,

\( \gamma_{t-1}', \gamma_{t-2}', \ldots, \gamma_{t-p}' \) denote the value of the variable at previous time periods, and

\( \epsilon_{t-1}, \epsilon_{t-2}, \ldots, \epsilon_{t-q} \) denote the error terms at previous time periods.

A typical ARIMA model, designated as ARIMA (p, d, q), is theoretically expressed as follows after combining both AR and MA parts:

\[ \gamma_t' = C + \phi_1 \gamma_{t-1}' + \ldots + \phi_p \gamma_{t-p}' + \psi_1 \epsilon_{t-1} + \ldots + \psi_q \epsilon_{t-q} + e_t \]  

(3)

Fig. 3: ARIMA model for wind power forecasting
3.2. Artificial Neural Network (ANN)

Artificial Neural Network is a replica of the human brain and is capable of recognizing relations in a set of data. A simple artificial neural network consists of three layers: input, hidden and output layers. ANN is a subset of machine learning which builds the foundation for deep learning algorithms [25]. The generalised structure of an Artificial Neural Network (ANN) is illustrated through Fig. 4.

![Generalised Structure of Artificial Neural Network (ANN)](image)

In the neural network, the computation unit is called a neuron while other components include inputs, weights, bias, and activation functions. The Backpropagation approach was employed in this model to change the weights in each layer. The error is determined by comparing the forward-propagated outputs from the network that are desired and estimated. These errors are then processed throughout the network from the output layer to the hidden layer. The Stochastic Gradient Descent Algorithm is used to compute and update the new weights as we backpropagate across the network.

A neural net can be simply formulated as:

\[ y = f(W \times x + b) \]  

where,

- \( y \) is the output of the ANN,
- \( x \) is the input to the ANN,
- \( W \) is the weight matrix,
- \( b \) is the bias term,
- \( f \) is the activation function.

3.3. Recurrent Neural Network (RNN)

The recurrent neural network is a form of neural networks with the ability to resolve time-dependent issues is recurrent neural networks. The capacity of RNN models to memorise data from prior experiments is one of their key characteristics. Fig. 5 depicts the basic architecture of RNNs.

![Architecture of Recurrent Neural Network (RNN)](image)

RNNs are equipped with a "memory" that can recall all data related to the previous calculations. It uses the same parameters for each input since it performs the same work on all inputs or hidden layers to create the result.
It reduces parameter complexity in contrast to other neural networks. RNNs can learn the temporal relationships between the data points in a sequence and use this information to make predictions about future data points. RNNs are built on the concept of a feedback loop. The output of each neuron in a feedforward neural network is only transferred to neurons in the following layer. In an RNN, however, each neuron's output is also transmitted back to itself as well as the neurons in the previous layer. This feedback loop enables the RNN to learn the correlation between data points in a sequence and utilise that knowledge to predict future data points.

RNNs can be formulated as follows:

\[ h_t = f(x_t, h_{t-1}) \]  
\[ y_t = g(h_t) \]

Where,
- \( h_t \) is the hidden state of the RNN at time step \( t \),
- \( x_t \) is the input to the RNN at time step \( t \),
- \( f \) is the hidden layer activation function,
- \( g \) is the output layer activation function.

The vector \( h_t \) represents the RNN's hidden state and stores the information about the sequence up to time step \( t \). The input, \( x_t \), is the data point at time \( t \). The hidden layer activation function, \( f \), takes the hidden state and the input and produces a new hidden state. While the output layer activation function, \( g \), takes the hidden state and generates an output prediction.

Though the RNNs were specifically designed to handle sequential data, the vanishing gradient problem obstructed the precision of the model. When the gradient values are too small, disappearing gradients happen, which stops learning or makes learning take too long. This provided room for improvement and thus, GRU and LSTM were developed to tackle its shortcomings.

3.4. Gated Recurrent Unit (GRU)

Gated Recurrent Units (GRUs) are a type of recurrent neural network (RNN) that was introduced in 2014 by Kyunghyun Cho et al [26]. GRUs are a simplified version of LSTMs, which are another type of RNN. GRUs can adapt to long-term dependencies in sequential data while being computationally efficient. Fig. 6 represents the basic structure of a GRU network.

![Architecture of Gated Recurrent Unit (GRU)](image)

The concept of GRUs is based on the idea of gating. Gating is a way of controlling the flow of information through a neural network. In GRUs, there are two gates: the reset gate and the update gate. The reset gate controls how much of the previous hidden state is kept, and the update gate controls how much of the new input is incorporated into the hidden state.

GRUs can be formulated as follows:
\[ h_t = z_t \times h_{t-1} + (1 - z_t) \times g_t \]  \hspace{1cm} (7)
\[ z_t = \sigma(W_z[h_{t-1}, x_t]) \]  \hspace{1cm} (8)
\[ g_t = \sigma(W_g[h_{t-1}, x_t]) \]  \hspace{1cm} (9)

Where,
- \( h_t \) is the hidden state of the GRU at time step \( t \),
- \( x_t \) is the input to the GRU at time step \( t \),
- \( z_t \) is the reset gate at time step \( t \),
- \( g_t \) is the update gate at time step \( t \),
- \( W_z \) and \( W_g \) are the weight matrices for the reset gate and update gate, respectively,
- \( \sigma \) is the sigmoid function.

In conclusion, GRU networks are a subclass of RNNs that efficiently describe sequential data by selectively updating the hidden state at each time step via gating methods.

### 3.5. Long Short-Term Memory (LSTM)

Long Short-Term Memory Networks or simply LSTM is a part of the recurrent neural network family, which eliminates the vanishing gradient problem of standard RNNs. Introduced in 1997 by Sepp Hochreiter and Jürgen Schmidhuber [22], LSTMs are able to learn long-term dependencies in sequential data.

![Architecture of Long Short-Term Memory (LSTM).](image)

The underlying principle of LSTM is the concept of "Gates". The three gates that make up the LSTM network, as shown in Fig. 7, are the forget gate, input gate, and output gate. The amount of the prior hidden state data that is retained depends on the forget gate. The input gate controls how much of the new input is incorporated into the hidden state while the output gate controls how much of the hidden state is output. This network comprises simple units called memory cells. The network is trained using a back-propagation algorithm.

\[ i_t = \sigma(W_i, [h_{t-1}, x_t]) + b_i \]  \hspace{1cm} (10)
\[ C_t = \tanh(W_c, [h_{t-1}, x_t]) + b_c \]  \hspace{1cm} (11)
\[ f_t = \sigma(W_f, [h_{t-1}, x_t]) + b_f \]  \hspace{1cm} (12)
\[ O_t = \sigma(W_o, [h_{t-1}, x_t]) + b_o \]  \hspace{1cm} (13)
\[ h_t = O_t \times \tanh(C_t) \]  \hspace{1cm} (14)

Where,
- \( f_t \) is the forget gate,
- \( i_t \) is the input gate,
\( \alpha_t \) is the output gate, \\
\( c_t \) is the cell state, \\
\( h_t \) is the hidden state, and \\
\( \sigma \) is the sigmoid function.

4. Data Description

The wind power generation dataset is real electricity power generation data records collected from the Central Electricity Authority (CEA), India [20] while the weather collected from different weather stations gave the required weather data for two years of the Kutch region of Gujarat. A collective data set was further prepared for proper trend analysis and as per the needs of the models, modifications were made. The dataset consisted of a total of 725 data records with date/time and factors like wind speed (m/s), wind direction (degrees), temperature (°C), air density, and humidity (%) were considered along with the historical power generation data (MW). Initial data cleaning was performed using Microsoft Excel and further analysis and visualisation were made using Python libraries like Matplotlib, Seaborn, Plotly and others. Table II presents the correlation between the features, scaled in the range of -1 and 1 while Fig. 9 represents the dataset relationship visually. The data is from the Kutch region, but the results of the study may apply to other regions with similar weather conditions.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Western Wind Power Gen</td>
<td>1.00</td>
<td>0.94</td>
<td>0.72</td>
<td>0.60</td>
<td>0.36</td>
<td>-0.32</td>
<td>0.27</td>
<td>0.20</td>
</tr>
<tr>
<td>Gujarat Wind Power Gen.</td>
<td>0.94</td>
<td>1.00</td>
<td>0.70</td>
<td>0.62</td>
<td>0.25</td>
<td>-0.22</td>
<td>0.18</td>
<td>0.16</td>
</tr>
<tr>
<td>Avg. Wind Speed</td>
<td>0.72</td>
<td>0.70</td>
<td>1.00</td>
<td>0.76</td>
<td>0.29</td>
<td>-0.45</td>
<td>0.43</td>
<td>0.34</td>
</tr>
<tr>
<td>Max. Wind Speed</td>
<td>0.60</td>
<td>0.62</td>
<td>0.76</td>
<td>1.00</td>
<td>0.18</td>
<td>-0.22</td>
<td>0.20</td>
<td>0.16</td>
</tr>
<tr>
<td>Avg. Humidity</td>
<td>0.36</td>
<td>0.25</td>
<td>0.29</td>
<td>0.18</td>
<td>1.00</td>
<td>-0.30</td>
<td>0.17</td>
<td>0.32</td>
</tr>
<tr>
<td>Avg. Air Density</td>
<td>-0.32</td>
<td>-0.22</td>
<td>-0.45</td>
<td>-0.22</td>
<td>-0.30</td>
<td>1.00</td>
<td>-0.86</td>
<td>-0.47</td>
</tr>
<tr>
<td>Avg. Temp.</td>
<td>0.27</td>
<td>0.18</td>
<td>0.43</td>
<td>0.20</td>
<td>0.17</td>
<td>-0.86</td>
<td>1.00</td>
<td>0.50</td>
</tr>
<tr>
<td>Wind Direction</td>
<td>0.20</td>
<td>0.16</td>
<td>0.34</td>
<td>0.16</td>
<td>0.32</td>
<td>-0.47</td>
<td>0.50</td>
<td>1.00</td>
</tr>
</tbody>
</table>
4.1. Data Preprocessing

We combined the wind power generation data from CEA with the corresponding weather data based on timestamp alignment. This integration enables us to correlate wind power output with relevant meteorological parameters. Further, the dataset was filtered and cleaned for the model training. The data may contain unwanted measures or variable measurements or mistaken values which could be studied through analysis and trends. The wind power generation data was pre-processed in order to remove any missing or NAN values and outliers. The weather data was also pre-processed to convert the values to a common scale. Normalisation was done and the data was scaled between the range of 0 and 1. To ensure consistent scaling across different variables, we performed data normalization, typically using techniques like Standard scaling. The stationarity of the data was checked with the help of the ADF test and seasonality was checked with ACF and PACF for the ARIMA model.

Fig. 8: Subplots for Wind Power Generation in Gujarat, Average Temperature, Average Wind Speed and Wind direction.
With the help of Microsoft Excel, initial cleaning and analysis were performed. Further using Python libraries like matplotlib and seaborn, trends were studied. Average wind speed was observed to be more than 8 m/s and winds from the South-West direction were noticed in the area due to the vicinity to the Arabian Sea. Fig. 9 illustrates the dynamic interplay between wind power generation and weather factors, including temperature, wind speed, and wind direction (in degrees), providing valuable insights into the connection between weather variables and wind energy generation in Gujarat’s Kutch region.
Boxplots were created to quickly summarize the annual seasonality of average wind speed and wind power generation. The plots also showed outliers and median values for the parameters and summarized the trends. As seen in Fig. 11, wind power generation was higher in May and August, when demand is typically high during the summer. The outliers in Fig. 10 (average wind speed) represent the maximum wind speeds during the time period. May and July are the months with the highest average wind speed, as seen in Fig. 10.

5. Results

In this research work, a comparative study between different machine learning models - Long Short-Term Memory (LSTM) networks, Artificial Neural Networks (ANN), Recurrent Neural Networks (RNN), Gated Recurrent Units (GRU), and Autoregressive Integrated Moving Average (ARIMA), were used. A machine running Windows 11 with an Intel Core i5-10210U CPU, 8 GB of RAM, and a 64-bit operating system was used to test and train the proposed models. Throughout, Python 3.10 was used for model creation, training, and data processing. The dataset contains 725 data records, of which 80% are used for training, 15% for validation, and 5% for testing. The models were carefully tested and hyper-tuned using different parameters like activation function, batch size, and learning rate to optimise the performance.

We conducted training with varying batch sizes, namely 7, 14, and 21, to explore their impact on model convergence and generalization. Additionally, we trained the model over 1000 epochs to ensure adequate learning, while employing stochastic gradient descent (SGD) served as the optimizer with a momentum of 0.99. A decay rate of 0.001 was applied to regulate learning rates throughout training, promoting model stability. To prevent overfitting and enhance model robustness, we implemented early stopping with patience of 20 epochs and a minimum delta of 0.00001 in the validation loss. This mechanism allowed us to halt training when the model's performance deteriorated, thus preventing unnecessary training iterations and potential overfitting. Furthermore, we tested with different learning rates: 0.01, 0.001, and 0.0001, to gauge their influence on convergence speed and accuracy.

These parameter configurations were meticulously chosen in order to balance model complexity and generalization, ultimately enabling us to extract meaningful insights and accurate wind energy forecasts in the Kutch.

To comprehensively assess the efficacy of our models, we employed a set of fundamental performance metrics. These metrics, namely the Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE), served as crucial indicators of our model's predictive accuracy. The MAE serves as a measure of the magnitude of predictive errors, helping us understand the typical size of deviations between actual and predicted values. On the other hand, MSE quantifies the square of the differences between actual and predicted values, providing insights into the overall error distribution. RMSE complements MSE by offering a clear indication of the severity of discrepancies between the actual and forecasted values, taking the square root of the MSE for interpretability. Lastly, the Mean Absolute Percentage Error plays
a vital role in capturing prediction accuracy. Unlike other metrics, MAPE prevents the problem of errors cancelling each other out and provides a more accurate reflection of real-world data discrepancies.

\[
\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} | y_i - \hat{y}_i | \tag{15}
\]

\[
\text{MAPE} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100\% \tag{16}
\]

\[
\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{17}
\]

\[
\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \tag{18}
\]

Where,

- \( y_i \) - actual value,
- \( \hat{y}_i \) - predicted value, and
- \( n \) - sample size

![Fig. 12. Comparison between ARIMA vs Deep Learning Models (Number of Neurons: 21).](image1.png)

![Fig. 13. Comparison between ARIMA vs Deep Learning Models (Number of Neurons: 63).](image2.png)
In the pursuit of developing robust forecasting models, a meticulous hyperparameter tuning process was undertaken for each model considered in our study. For the ARIMA model, an automated approach, AutoARIMA, was employed to identify the optimal configuration, resulting in the selection of ARIMA (1,2,2), results depicted in Table V. In parallel, the other models were also rigorously optimised and hyperparameter tuning was performed as mentioned earlier in the section. Table III and Table IV have been incorporated into our analysis to provide a concise yet comprehensive comparison of the performance metrics for our forecasting models. These tables serve as valuable visual aids in the results section, offering a clear and structured overview of how each model performs, in terms of MAE, MSE, MAPE and RMSE.

Table III presents the performance metrics for the forecasting models with an architecture employing 21 neurons, allowing for a detailed assessment of their accuracy and predictive capabilities. In contrast, Table IV provides a comparative analysis of the same models but with an increased neural network complexity, featuring 63 neurons. These tables offer insights into how varying neural network sizes impact the models' forecasting accuracy with respect to wind power generation in the Kutch region. Table V presents the performance of ARIMA model on different configurations, with an optimal result of 5.87 MAPE score.

<table>
<thead>
<tr>
<th>METRICS</th>
<th>BATCH SIZE</th>
<th>LEARNING RATE</th>
<th>TEST DATA</th>
<th>PREDICTION DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>MAE</td>
<td>MAPE</td>
</tr>
<tr>
<td>ANN</td>
<td>21</td>
<td>0.001</td>
<td>6.75</td>
<td>12.86</td>
</tr>
<tr>
<td>RNN</td>
<td>21</td>
<td>0.001</td>
<td>7.59</td>
<td>14.49</td>
</tr>
<tr>
<td>GRU</td>
<td>21</td>
<td>0.01</td>
<td>7.37</td>
<td>14.28</td>
</tr>
<tr>
<td>LSTM</td>
<td>7</td>
<td>0.0001</td>
<td>7.23</td>
<td>14.83</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>METRICS</th>
<th>BATCH SIZE</th>
<th>LEARNING RATE</th>
<th>TEST DATA</th>
<th>PREDICTION DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>MAE</td>
<td>MAPE</td>
</tr>
<tr>
<td>ANN</td>
<td>21</td>
<td>0.001</td>
<td>7.80</td>
<td>14.74</td>
</tr>
<tr>
<td>RNN</td>
<td>21</td>
<td>0.01</td>
<td>7.01</td>
<td>13.38</td>
</tr>
<tr>
<td>GRU</td>
<td>14</td>
<td>0.001</td>
<td>7.24</td>
<td>13.96</td>
</tr>
<tr>
<td>LSTM</td>
<td>14</td>
<td>0.001</td>
<td>6.99</td>
<td>14.42</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TRAIN SIZE</th>
<th>TEST SIZE</th>
<th>MODEL</th>
<th>TEST DATA</th>
<th>PREDICTION DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>95%</td>
<td>5%</td>
<td>(1,2,2)</td>
<td>6.24</td>
<td>11.51</td>
</tr>
<tr>
<td>97%</td>
<td>3%</td>
<td>(1,2,2)</td>
<td>7.88</td>
<td>13.17</td>
</tr>
</tbody>
</table>

6. Conclusion

Contemporary wind energy planning techniques are employed to anticipate difficult problems in operational loads, mitigate risks, and optimise performance. In order to balance the energy supply and demand, which is essential for both economic stability and the dependability of the power grid, complete power system planning must include wind power forecasts. Because supply and demand must be balanced for the economy and for a secure and stable power system, it is important to manage energy in a sustainable way. Deep learning algorithms have recently in advanced prediction models as effective methods for future power generation predictions. These techniques can be key in planning for potential modification and optimisation of power production.
Our comparative study underscores the prowess of the ARIMA model when properly fine-tuned and optimised generates more accurate forecasts, however, other deep learning forecasting models also yield remarkable accurate forecasts. The key finding, in the analysed time frame, forecasts the energy generation in the Kutch presents a month's forecast with precision and accuracy. With a goal towards sustainable development and to meet India's escalating energy demands, the prominence of renewable energy sources, especially in regions like Gujarat with abundant wind energy potential, becomes increasingly evident.

Nevertheless, wind energy generation forecasting is directly impacted by the ever-changing weather conditions, the subject remains open despite significant results and existing works. The inconsistency in forecasting during peak hours and uncertainty of weather conditions hinder the precision of the forecasting models. In light of these challenges, further study and innovation in the field of wind power forecasting are crucial. Future endeavours should aim to enhance the reliability of forecasting models, in time contributing to the continued growth of the energy sector in regions like Gujarat and advancing the broader goals of sustainable energy production and resource management.

Acknowledgements

The authors declare that the results of the research discussed in this paper were not influenced by any known conflicting interests or close connections.

References


