TRIBHUVAN UNIVERSITY INSTITUTE OF ENGINEERING PULCHOWK CAMPUS, PULCHOWK

Thesis No.: 075/MSCCD/018

"Estimation of Global Solar Radiation Potential using Hybrid Models : A Case Study of Nepal"

by

Sushant Chalise

A THESIS REPORT

SUBMITTED TO THE DEPARTMENT OF APPLIED SCIENCES AND CHEMICAL ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTERS OF SCIENCE IN CLIMATE CHANGE AND SUSTAINABLE DEVELOPMENT PROGRAMME

DEPARTMENT OF APPLIED SCIENCES AND CHEMICAL ENGINEERING

LALITPUR, NEPAL

SEPTEMBER, 2021

COPYRIGHT

TRIBHUVAN UNIVERSITY INSTITUTE OF ENGINEERING, PULCHOWK CAMPUS DEPARTMENT OF APPLIED SCIENCES AND CHEMICAL ENGINEERING

The undersigned certify that they have read, and recommended to the Institute of Engineering for acceptance, thesis entitled "**Estimation of Global Solar Radiation Potential using Hybrid Models : A Case Study of Nepal**" submitted by **Sushant Chalise (075/MSCCD/018)** in partial fulfilment of the requirements for the degree of M.Sc. in Climate Change and Sustainable Development Programme.

Supervisors:

……………………………………

Prof. Dr. Khem Narayan Poudyal

Department of Applied Sciences and Chemical Engineering

IOE, Pulchowk Campus

……………………………………

Asst. Prof. Aayush Bhattarai

Department of Mechanical and Aerospace Engineering

IOE, Pulchowk Campus

Committee Chairperson:

…………………………………….

Prof. Dr. Ram Kumar Sharma

Department of Applied Sciences and Chemical Engineering

IOE, Pulchowk Campus

Program Coordinator:

……………………………………

Prof. Dr. Khem Narayan Poudyal

Department of Applied Sciences and Chemical Engineering

IOE, Pulchowk Campus

External Examiner:

……………………………………

Asst. Prof. Nawraj Bhattarai

Department of Mechanical and Aerospace Engineering

IOE, Pulchowk Campus

Date: Sept, 2021

ACKNOWLEDGEMENT

First of all, I would like to thank my thesis supervisor Prof. Dr. Khem Narayan Poudyal and Asst. Prof. Aayush Bhattarai for their constant guidance, supervision, comments and inspiration throughout the study. I'd also like to express my gratitude to the Institute of Engineering's Department of Applied Sciences and Chemical Engineering for allowing me to complete thesis work as part of my Master of Science in Climate Change and Development. I am also grateful to Asst. Prof. Neeraj Adhikari for his guidance, encouragement and continuous support throughout the project. Also, I am indebted to Asst. Prof. Sanjaya Neupane, for his constructive feedback. I would like to acknowledge the help from Er. Ashok Chalise, Mr. Anil Kumar Singh for helping me with the study material. Lastly, I would like to extend my appreciation to all my friends and family for their constant support to bring this research together.

Sincerely,
Sushant Chalise
075MSCCD018

ABSTRACT

Solar energy has immense promise as a source of renewable energy. It is abundant throughout the year, although it is subject to uncertainty due to various parameters. Sun energy sources' affectivity and productivity can be improved by accurate forecasting of solar radiation. Forecasting Global Solar Radiation (GSR) in the field of research has attracted widespread attention from the research community in many practical fields including energy. Different models for predicting GSR potential have been used in the literature. One of the most prominent linear models for time series forecasting is the Autoregressive Integrated Moving Average (ARIMA). There are also different machine learning models which show promising forecasting results. To take advantage of the unique benefits of ARIMA and machine learning models in linear and nonlinear modeling the data of solar radiation potential, we propose a hybrid method combining ARIMA and machine learning models ANN(Artificial Neural Network) and LSTM(Long Short Term Memory) models in this study. The dataset was obtained for the location of Kushma, Parbat for duration between 1990 to 2014. For the supplied data sets, the ARIMA plus ANN hybrid model was seen to be the best method for predicting solar radiation potential. The correlation coefficient (R square) is calculated 0.847. The error values for this model are accessed as RMSE, MAPE and MAE of 1.719, 6.456 and 1.330 respectively. The experimental results of real data sets show that the combined model can effectively improve the prediction accuracy achieved by any model used alone. TThe acquired results also demonstrated that the created model could be utilized to calculate the solar radiation potential of any geographic region with known climatic parameters.

CONTENTS

LIST OF ABBREVIATION

ADF :	Augmented Dickey Fuller
AIC :	Akalkes Information Criterion
$ARIMA$:	AutoRegressive Integraged Moving Average
ANN :	Artificial Neural Network
$GSCI$:	Goldman Sachs Commodity Index
GSR :	Global Solar Radiation
$LSTM$:	Long Short Term Memory
MAE :	Mean Absolute Error
$MAPE$:	Mean Absolute Percentage Error
ML :	Machine Learning
$RMSE.$:	Root Mean Square Error
RNN :	Recurrent Neural Network
$WECS$:	Water and Energy Commission Secretariat

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1.1 Background

One of the most important energy sources that may be received directly from the sun is solar energy. Sun is rich in energy and helps the survival of human beings and all the living things on the earth. Solar radiation is significantly affected when it passes through the earths atmosphere (WECS, 2010). Solar energy is the energy emitted by the sun in the form of radiation. This energy is obtained through the thermonuclear reaction process (Iqbal, 2012). Because Nepal is located at a favorable latitude, it receives roughly 300 days of sunshine each year, with an annual average solar radiation of 3.66.2 kWh/m^2/day per year. In addition, there are approximately 6.8 hours of sunshine per day (Tarpley, 1979). Solar energy is used as an alternative energy source not only in remote areas, but also in urban areas of Nepal. Solar energy is crucial for heating systems, pumping water to construct contemporary agricultural greenhouses, and solar photovoltaic lighting (Awasthi and Poudyal, 2018). Nepal is in the monsoon climate zone, with annual rainfall ranging from 1,000 to 2,500 millimeters, with 80 percent to 90 percent of that falling during the monsoon season through June through September. In the summertime, maximum mean temperature varies from over 40^oC in the lowlands to 20^oC in the central plateau hills, and below 16^oC in the highlands at 2,000 m to 4,000 m. Temperatures at higher altitudes are substantially colder in the winter (International Renewal Energy, 2018).

Recent climate change and high demand for electricity have led to the need for electricity from green and renewable sources, including solar energy. Solar energy, an abundant source of sustainable energy, has the least environmental impact and makes the sun an important source of energy (Heng et al., 2017).Using autoregressive integrated moving average (ARIMA), artificial neural network (ANN), and long-term short-term memory (LSTM) models, we offer a mixed algorithm to time series analysis. The following viewpoints provide motivation for such hybrid approach. First, determining whether such a time series undergoing consideration is produced by an underpinning linear or nonlinear mechanism, and whether one technique is far more efficient than another, is sometimes challenging in reality. As a result, selecting the appropriate method for a given situation is challenging for tippers. Various models are usually examined, and the one with the best accurate result is chosen. The final model chosen, however, is not exactly the right for future developments due to various possible factors involved such as sample variation, modelling ambiguity, and systemic transformation. The challenge of model selection can be solved with little extra effort by combining multiple strategies. Secondly, actual data are infrequently linear or non-linear in nature. They frequently include both linear and non - linear patterns. If this is really the situation, neither ARIMA or hardly machine learning (ML) methods could be used to analyze and simulate time series since this ARIMA model cant solve non-linear interactions and the neural network model can't either handle linear and nonlinear patterns alike. Complex autocorrelation patterns can thus be more properly described in the dataset by merging ARIMA and ML models. Third, the forecasting research nearly universally agrees that no strategy is the best throughout every scenario, as presented in the articles by (Chatfield, 1988) and (Jenkins, 1982). This is mostly owing to the fact that real-world problems are frequently complicated, and an approach may not always be capable of capturing multiple trends with equal effectiveness. The majority of studies on time series prediction utilizing neural network models are found in the literature such as (Tang and Fishwick, 1993), (Hwarng, 2001), Given mixed results, employ ARIMA models as standards comparasionăto evaluate the success of the ANN model. Numerous empirical investigations, involving multiple large-scale predicting competitions, show that integrating numerous new variations can sometimes enhance prediction accuracy when compared to the standaloneămodel, without the need to determine the "real" or "perfect" model (Makridakis et al., 1982). As a result, combining several models can enhance prediction productivity by improving the potential to gather distinct patterns within the data. Numerous empirical investigations have also shown that integrating several new variations improves forecasting accuracy when compared to using each model separately. Furthermore, the integrated model seems to be more resistant to potential data structure changes. Solar radiation forecast is based on online data.

Time series analysis is a valuable research subject that has gotten a lot of interest from the academic and scientificăresearchers in a lot of different fields, including finance ((Wei, 2016), (Adhikari and Agrawal, 2014)), agriculture ((Ezzine et al., 2014), (Garrett et al., 2013)), energy ((Sadaei et al., 2014), (Bahrami et al., 2014)) and transport ((Xiao et al., 2012), (Zhang et al., 2013)), environment ((Deng et al., 2015), (Feng et al., 2015)), etc. In recent decades,Numerous efforts have been made by researchers to construct effective prediction models in order to enhance predictive performance, but the research and development of new predictive models to improve forecast accuracy has not stopped (De Gooijer and Hyndman, 2006). A neural network is a highly distributed simultaneous processor made up of simple computationalăelements that seem to have a natural inclination to accumulate and grant access empirical evidences. Artificial intelligence that replicates the activity of the human brain is known as ANN (Haykin, 2010). ANNs can analyze linear and nonlinear phenomena without making underlying assumptions, which is the case with many of the conventional analytical techniques. They were employed in a variety of scientific and technological domains ((Yang et al., 2005); (Chantasut et al., 2004)).

1.2 Problem Statement

A hybrid forecasting model has recently appeared, called a hybrid model. The hybrid model can be a combination of the same or different models. One of the most popular categories of mixed models is linear and nonlinear mixed models. For example, To take advantage of the unique benefits of ARIMA and ANN in linear and nonlinear analysis, Zhang (2003) devised a mix of regression integral moving average and artificial neural network. The simulation results indicated that, instead of employing ARIMA or ANN alone, the Zhang approach could be a more effective way to increase predictive performance. However, a disadvantage of the Zhang model is that it assumes that the linear and nonlinear elements of the time - series data have an additive connection. Therefore, in some cases, the Zhang model does not perform as well as its components (ie, ARIMA and ANN) (Taskaya-Temizel and Casey, 2005). Furthermore, with respect to the mixed results of the Zhang model, Khashei and Bijari (Khashei and Bijari, 2011) extended the Zhang model by defining time series as a function of linear and non-linear components. In the Khashei and Bijari model, ARIMA first extracts the linear component of the time series. Then, they assumed that the non-linear component remained in the ARIMA residuals and in the time series data. Secondly, ANN is used to examine the function of ARIMA results, the lag of ARIMA residuals, and the lag value of time series. In fact, these two hybrid models (Zhang model and Khashei and Bijari model) are applied to several real-world applications, such as stock index (Wang et al., 2012), agricultural product prices (Shahwan and Odening, 2007), moisture content soil (Liu et al., 2008), agricultural imports Value (Lee and Liu, 2014), irrigation water demand (Pulido-Calvo and Gutierrez-Estrada, 2009), sugar and alcohol (Ribeiro and Oliveira, 2011), and Goldman Sachs Commodity Index (GSCI) futures prices (Bo et al., 2007). Usually, these hybrid models promise better predictions, but are only a measure of average accuracy. Therefore, in certain forecast periods, a single model can provide higher accuracy. In addition, the input variable is just the lag value of the time series. In accordance with these limitations, it is possible to improve the accuracy of the prediction by suitably combining a single model and a mixed model and input preprocessing. As far as we know, the study of GSR forecasts is limited to the use of the ARIMA model, emperical models and ANN models. Although GSR is one of the most important sources of renewable energy for the world there is no research on the hybrid model of GSR from Nepal. Nepal is also enlisted as the fourth vulnerable county in terms of climate change. These natural calamities can directly affect the production of hydroelectricity. There is a high need of sustainable energy supply and it can be performed with the mix of solar energy. Therefore, GSR prediction can help decision makers involved in energy demand supply chain improve production plans, assist the ministries in policy formulation and generate profits in the energy markets. For these reasons, it seems interesting to use the prediction of GSR in Parbat district as a case study in this paper. Therefore, we propose a new mixed forecast model for daily GSR. By involving a ARIMA and ML, the proposed model has been significantly expanded from the Zhang model and the Khashei and Bijari models. Furthermore, the proposed model not only includes the lag value of the time series as an input variable, but also includes additional variables such as the moving average and the annual seasonal index. The experiment comprehensively carried out various hybridization schemes in terms of structure and variables to find the most suitable prediction model for the export of cassava. Building dependable solar energy systems necessitates knowledge of the GSR in the area where the system will be installed. Several countries built GSR method to predict sun radiation and aid in the construction of solar power generation. We still don't have any such modeling for GSR in Nepal, thus this is the first effort to create one for Kaski district, which will also be expanded to other regions afterwards. Moreover, for the past few decades, numerous researches have been carried out for effective solar irradiance prediction. Usually, a time series forecasting has been focused using a single model (Patil, 1990); (Tang and Fishwick, 1993). In this work, three different models are implemented and also a hybrid model is purposed.

1.3 Objectives

The objectives of this study are given below.
Main Objectives

- To create a hybrid model using ARIMA and machine learning model (ANN and LSTM) to predict GSR in Kusuma, Parbat.

Specific Objectives

- To analyze the accuracy of different standalone models.
- To identify the best effective model for prediction of GSR in Kusuma, Parbat.
- To recommend the best model for the study site.

1.4 Rationale of the Study

Many researchers have found that prediction methods are so modern that they overshadow traditional methods when applied to various data sets. However, modern forecasting methods have not been widely used to estimate global solar radiation. Many research is mainly focused on the performance of hybrid model in a generalized set of time series data, as climate change is found to be unequivocal it is equally important to analyze the model efficiency for climatic data as well. The research is mainly focused on predicting the global solar radiation potential and calculating the efficiency of such models for climatic data. It‘s more like an exploration of how data analytics also supports achievability of renewable energy.

1.5 Scope and Limitation of Research

The scope of the study is to identify the errors in forecasting the time series data for solar radiation potential. This shall be identified by studying different statistical error measurement factors. Three different types of models along with the hybrid will be used for the study. Finally, alternative methods and directions are proposed for future research. The data used in this study is acquired from a website and hence can be faulty in different way. Also, some assumptions are made to create a hybrid model.

First, the most fundamental restriction must be reported in carrying out the construction of the investigation in general. This thesis is subject to restrictions with respect to the time and time required for different processes. In particular, the computational load of problems increases in parallel with the needs of time. Of course, these restrictions lead to a need for simplifications through a multitude of thesis elements. One of these simplifications is related to data collection. Lack of required format and sufficient data from ground radiant sources, the data was collected from the satellite station. only the historical data of solar radiation were used, limiting our model and the reference points for the architecture of the univariate Time Series. A multivariate series model with additional explanatory variables would have resulted in a stronger base to attract empirical conclusions. Therefore, the structure of the univariate time series limits the validity of the generalization of the hybrid model. This could mean that the application in different periods of time, or in another solar radiation, would have led to less accurate forecasts.

Another branch of the all-encompassing time constraint is the simplification applied to our data size. The fundamental advantage of neural networks is that they can handle large amounts of data, which means that this work is limited in checking such learning algorithms. As mentioned in the previous paragraph, including additional data points by covering a longer period of time or additional variables will increase the utility of the neural network. Considering the conversion from daily data to monthly data, realizing that this limitation may be seen as a contradiction. It is undeniable that this conversion reduces the number of data points input to the model. However, this is viewed as a compromise between the time span of coverage and computational feasibility. The use of monthly data can cover a longer period of time without the computational effort becoming unmanageable. The desire to provide a sufficient amount of data for the neural network leads in this work to rely on the simplification of the data verification division. Usually, time series cross-validation involves breaking up a large amount of data in order to verify it over many time steps. However, the lack of data in neural networks means that the window size of the time series cross-validation remains large, which reduces the number of possible evaluation steps. However, it should be refuted that every time step in our data breakdown involves the forecasting and calculation of performance indicators made up of 289 solar radiation data. Against this background, the creation of a general model from the initial training data and its evaluation on a limited number of test sets was determined as an acceptable compromise.

The restrictions relating to time and calculation efforts also induce the need for limitations in model applications. In this thesis, automatic paintings were used to reduce time consumption of certain methodological passages. These automated frames are excellent for this reason; However, they can also cause sub-optimal solutions or a potentially hindered learning, as it reduces the investigator's participation. In other cases, for example, in the inclusion of the reference models, the setting of the parameters is limited, which will have softens on the validity of the results. Limited time has also conducted this thesis that makes no effort to look inside the black box of the LSTM Beizer (1995). Traditionally, the applications of advanced neuronal networks for decision-making have received criticisms to be used in favor of interpretable models. This criticism has activated an answer in which the researchers have developed methods to support the transfer through neuronal networks, which allow mechanisms to cause learning. On the other hand, this thesis presented has no effort to explain the causes of changes in the potential of solar radiation, but discuss the potential value of modern learning applications in energy forecasts. These limitations presented are the starting point for a discussion on how this specific method can be embroidered in further research, and therefore contribute to improving the predictive power and implicit learning through greater interpretability.

CHAPTER 2: LITERATURE REVIEW

2.1 Global Solar Radiation

The sun is the greatest source of energy. It is considered the most important renewable energy source. Furthermore, the most plentiful source of energy on the planet is solar radiation. But we humans, especially in Nepal, are using fossil fuel for daily use. As we are moving towards the new era and the interest in solar power application areas are increasing everyday. Firewood was the most dependent source of energy for household cooking with 84 percent dependent in Nepal. There is an average of 2550 sunshine hours per year with an average of 6.59 sunshine hours per day. The use of solar energy is now a necessity to satisfy an important part of the country's energy needs. The need of renewable energy sources and difficulty in assessing the potential of production renewable energy come together with great challenges, physically and economically. The Sun, as the ultimate source of energy could be decisive in producing renewable energy for daily usage. Solar radiation values are only available for a limited number of locations and therefore need to be interpolated to find the value of other locations in order to obtain the best possible location for a solar park (International Renewal Energy, 2018). The artificial neural network is an efficient artificial intelligence model for making predictions by training the model with appropriate data variables. Various literatures show that many researchers outside of Nepal, such as (Widrow, 1960); (Rosenblatt, 1958) and (Ghosh and Deuser, 1995) estimate total global solar radiation using the ANN model which is based on meteorological factors. These models cannot be used efficiently in Nepal due to seasonal variations, different weather and geographic conditions, as stated by (Carpenter and Grossberg, 2010). The uncertainty of solar radiation and the modeling capabilities of artificial neural networks (ANN) have inspired the application of ANN techniques to predict solar radiation.

2.2 Autoregressive Integrated Moving Average

In an ARIMA model, the forecasted value of a variable is believed to be a linear mixture of many prior observations and random errors. That is, the core principle that generates the time series has the form, as given by BoxJenkins,

$$Y_t = \theta_0 + \phi_1 y_{t1} + \phi_2 y_{t2} + \cdots + \phi_p y_{tp} + \epsilon_t - \phi_1 \epsilon_{t-1} - \phi 2 \epsilon_{t-2} - \cdots - \phi q \epsilon_{t-q} \tag{2.1}$$

where y_t and $epsilon_t$ are the true value and random error in time t, respectively. phi_i (i = 1,2,......, p) and $beta_j$ (j = 0,1,2,..., q) are model parameters. The integers p and q are known as model orders. The random errors $epsilon_t$ are assumed to be independent and evenly dispersed, with a mean of zero and a variance of $sigma^2$. Eq. (2.1) includes numerous key ARIMA model group specific instances. When q = 0, (2.1) transforms into an AR model of order p. The model simplifies to an MA model of order q when p = 0. One of the most important aspects of ARIMA modeling is determining the model's right order (p, q). Developed from prior work by (Yule, 1926) and (Wold, 1938), (Box and Jenkins, 1970) has discovered a realistic method for building ARIMA models that has far-reaching consequences for time series research and prediction. Model development, parameter estimates, and diagnostic imaging are all iterative processes in the Box-Jenkins approach. The model estimation principle states that a time series created by an ARIMA process must exhibit theoretical autocorrelation features. It is sometimes possible to detect one or more plausible explanations for a given time series by examining experimental and predicted autocorrelation characteristics. (Box and Jenkins, 1970) suggested identifying the ordering of the ARIMA model that used the autocorrelation function and partial autocorrelation function of the data set as essential tools A data processing is frequently necessary in the recognize phase to create the trend analysis stationary. Stationarity is a requirement for building an ARIMA model that can be used for prediction. The statistical features of a stationary time series, including the average mean and autocorrelation architecture, remain relatively stable. When time series exhibit pattern and homogeneity of variance, differentiation and power modifications are frequently used to eliminate the pattern and stabilize the variance before fitting an ARIMA structure. Determining the parameters of the model is simple when one basic model has been defined. To reduce the total degree of mistakes, the parameters are estimated. A non-linear optimization method can be used to accomplish this. The diagnostic evaluation of the model's suitability is the final step in the modeling process. This is used to see if the model's hypotheses concerning mistakes $epsilon_t$ are correct. The goodness of fit of the preliminary model to the past records could be assessed using a variety of diagnostic statistics and residuals diagrams. If the model is insufficient, a new preliminary model must be identified, followed by the parameter estimates and model verification procedures. Alternative models can be suggested using diagnostic data. This three-step modeling procedure is usually conducted multiple times before a successful model is chosen. The final model chosen can be utilized to make predictions.

2.3 Artificial Neural Network

The proportion of non-linear architectures that may be used to explain and forecast a time series increases considerably when the linear restriction in the model form is removed. According to (De Gooijer and Kumar, 1992), a successful nonlinear model should be general enough to capture some of the nonlinear processes in the data. The availability of historical data in meteorological supply databases and the fact that ANNs are data-driven methods that can make a non-linear association between sets of input and output variables make this modeling software tool be very attractive (Notton et al., 2018). Since the costs of solar radiation meters are very high, it is not possible to install these meters everywhere (Fouilloy et al., 2018). Artificial neural networks are one type of model that can approximate diverse data nonlinearities. ANNs are versatile computer frameworks that may be used to model a wide range of nonlinear issues.

ANN models have a significant advantage over other types of nonlinear models in that they are universal approximators that can accurately approximate a wide range of functions. Their power is derived from the parallel processing of digital information. The model building method does not necessitate any prior assumptions about the model shape. Instead, the network model is heavily influenced by the data's features. There are two types of artificial neural networks: feedforward and feedback networks (loops). The network connection in the first network does not form a loop, whereas the latter network can have one or more loops. The most common feedforward network sequence is hierarchical networks, wherein neurons are organized onto levels with strict connections in one direction via one layer to the next (Jain et al., 1996). According to (Zhang et al., 2001), For time series analysis and prediction, the hidden neuron layer feedforward system is by far the most widely used model structure. The architecture is made up of three layers of simple processing elements coupled by acyclic links. The mathematical description of the association between output y_t and the inputs $y_{t-1}, y_{t-2}, ..., y_{t-p}$ is as follows:

$$y_t = \alpha_0 + \sum_{j=0}^{q} \alpha_j g(\beta_{0j} + \sum_{i=0}^{p} (\beta_{ij}) y_{t-1}) \tag{2.2}$$

where $alpha_j$ (j = 0,1,2,...,q) and $beta_{ij}$ i = 0,1,2,...,p; j = 1,2,...,q) are model parameters known as weight vectors, where p describes the amount of input nodes and q represents the number of hidden layers. As a hidden layer function f(), the logistic function is commonly used. it is utilized to transport data across layers, it is given by

$$g(x) = \frac{1}{1 + exp(-x)} \tag{2.3}$$

As a result, the ANN model in (2.2) conducts a nonlinear functional mapping from past data $y_{t1}, y_{t2}, , y_{tp}$ to the future data y_t, i.e.,

$$y_t = f(y_{t1}, y_{t2}, , y_{tp}, w) + \epsilon_t \tag{2.4}$$

where w is a vector comprising all parameters and f is a framework and connection weights-based function. The neural network becomes a nonlinear autoregressive framework as a result. Expression implies single output node in the output layer (2.2), which is commonly used for one-step-ahead forecasting. The simple network presented in (2.2) is remarkably powerful in that it can approximate any function as long as the number of hidden nodes q is large enough. (Hornik et al., 1990). In practice, in out-of-sample prediction, a simple network layout with a modest number of hidden nodes typically works effectively. This could be related to the overfitting effect that occurs frequently in neural network models. Although an overfitted model has a perfect match to the dataset used to construct it, it has poor expressive capability for variables beyond the samples. The factor q is dependent on the inputs, and there is no set formula for calculating it. The selection of the quantity of lagged data, p, the size of the input sequence is yet another important issue of ANN modeling of time-series data, in addition to determining an adequate number of hidden nodes. Because it determines the autocorrelation (non-linear) architecture of the data series, this is likely its most crucial parameter to predict in an ANN model. There is, nevertheless, no concept which can be used to govern the choice of p. As a result, studies are frequently conducted to determine both an adequate p and an appropriate q. The network is ready to be analyzed, which is a process of parameter estimation, after a network structure (p, q) is defined. The coefficients are determined in such a manner that a generic accuracy requirement, like mean square error, is reduced, just like in ARIMA analysis. Other than the basic backpropagation training procedure, this is done with various efficient nonlinear optimization algorithms. In most cases, the estimated model is tested on a distinct allocated dataset that has not been subjected to that same training phase. This procedure differs from the one used in the construction of ARIMA models, which normally uses a sample for the identification, estimation and evaluation of the model. The reason for this is that the ARIMA model's general (linear) shape is given first, and then the model's order is inferred from the data. The conventional statistical approach believes that the best model for fitting past data is likewise the best model for predicting the future. (Fildes and Makridakis, 1995) under stationary conditions. In the case of ANNs, both the shape of the model (non-linear) and the model's order should be calculated from the data. As a result, an ANN model has a higher chance of overfitting the dataset. The ARIMA and ANN models share several characteristics. Each encompass a diverse range of models with varying model orders. To get the best outcomes, data transformation is frequently required. To create an effective model, you'll need a high sample size. Your modeling procedures are continuous and exploratory, and individual judgment is sometimes necessary to implement the model. Saving is typically a guiding criterion when picking a good time series forecasting, due to the possibility of overfitting with both models.

2.4 Long-Short Term Memory

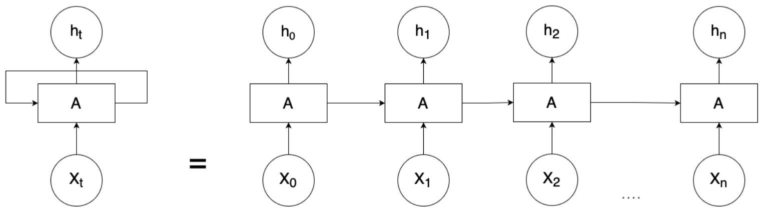


Figure 2.1: LSTM architecture in its most basic form

The LSTM model developed by Hochreiter and Schmidhuber (1997) in 1997 is indeed a subset of the recurrent neural network (RNN) architecture. The LSTM model was created with the intention of learning through long-term relationships. It is made up of the LSTM structure, which is a complicated structure buried within the layers. LSTM is a popular and commonly used deep learning model today, with applications in a variety of fields, as mentioned by Atienza (2018). The underlying LSTM architecture is shown in Fig. 2.1.

The memory-based RNN cell is at the heart of the LSTM's basic structure. This memory cell is useful for searching and retrieving from the past. This memory cell facilitates the transmission of previous data to the next level. Based on its training needs, the model selects past data. It is a frequent exercise for the LSTM network to retain valuable information over just a significant period of time, but it is a crucial behavior of the LSTM network ((Zhao et al., 2017)). In Figure 2.2, the fundamental LSTM structure is shown. Here, x_t symbolizes the preceding unit's input

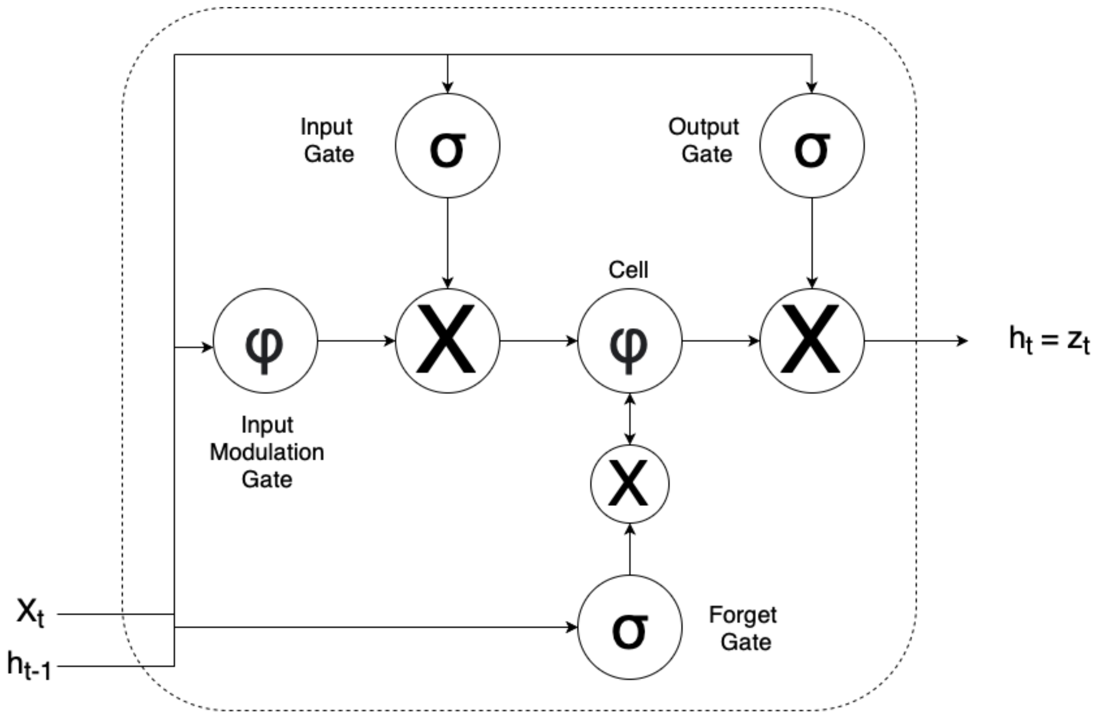


Figure 2.2: Basic Cell structure of LSTM

data or output at time step t, ht indicates the hidden output value, and ht1 refers the prior or past output. The LSTM model contains gates such as input gate, output gate, forget gate, and inputs modulation gate. Eqs. (5), (6), and (7) are used to compute the input gate I_{tj}, forget gate F_{tj}, and output gate O_{tj} of the LSTM model.

$$I_{tj} = \sigma(W_I x_t + W_I h_{t-1} + b_I)^j \tag{2.5}$$

$$F_{tj} = \sigma(W_F x_t + W_F h_{t-1} + b_F)^j \tag{2.6}$$

$$O_{tj} = \sigma(W_O x_t + W_O h_{t-1} + b_O)^j \tag{2.7}$$

The sigmoid activation function is represented by $sigma$, the voltage vectors are represented by b, and the weight matrices are represented by W. The memory is maintained with in LSTM model at time t, and afterwards the modified memory function c_{jt} is computed using Eq (2.8),

$$c_{tj} = F_{tj} c_{jt-1} + I_{tj} c_{jt} \tag{2.8}$$

Now, using Eq. (2.9), the upgraded memory subject matter is estimated, and then Eq. (2.10) is used to predict the LSTM model's result.

$$c_{tj} = tanh(W_C x_t + W_C h_{t-1} + b_C)^j \tag{2.9}$$

$$h_j = O_j tanh(c_j) \tag{2.10}$$

The epoch, like certain other ANNs, is in charge of LSTM training phase. This epoch is responsible for calculating the network load W. The epoch, that is related to the number of iterations upon that particular dataset, determines the network weight. With deep learning models, the problem of enhancing the network via changing the weights is crucial. As a consequence, transmitting all the data across the same network multiple times is a good idea, and we may aim for an even more accurate and precise forecasting model using it. However, because each dataset may contain different behaviors, the number of epochs required to achieve optimal weights is unknown. As a result, various numbers of epochs might well be necessary for the optimum train network.

2.5 Hybrid Model

In their respective linear and non-linear domains, both of the ARIMA and ANN modeling have been successful. However, none of them is a universal model suitable for all circumstances. The ARIMA models approach to complex nonlinear problems may not be adequate. On the other hand, the use of RNA to model linear problems has produced mixed results. For example, On the basis of simulation results, (Denton, 1995) shown that artificial neural surpass linear regression models in the presence of outliers or cointegration with in data. The effectiveness of RNA for linear regression issues was also discovered to be dependent on sample size and noise level, according to (Markham and Rakes, 1998). As a result, applying ANNs to any form of data is not recommended. Because it's difficult to fully comprehend the qualities of data in a real-world scenario, a hybrid method that combines linear and non-linear model - based powers can be a useful solution. Multiple features of the underlying patterns can be conveyed by merging different models. A time series can be thought of as having a linear autocorrelation architecture and a nonlinear element. In other words,

$$y_t = L_t + N_t \tag{2.11}$$

, where y_t is the time series data for time t, and L_t and N_t are the linear and nonlinear parts of the time series data for time t, respectively. These two components must be calculated based on the data. We first let ARIMA model the development pipeline, and then the linear model's residuals will only include the non - linear relation. Let e_t be the linear model's remainder at time t, and then

$$e_t = y_t - L_t^{'} \tag{2.12}$$

where $L_t^{'}$ is the estimated relationship's predicted value for time t given by equation (2.2). When establishing whether linear models are weak, residuals are critical. If there are still linear correlation structures in the residuals, a linear regression model is insufficient. In contrast, residual analysis is unable to uncover any nonlinear patterns within the data. In addition, there are currently no generic diagnostic metrics for nonlinear autocorrelation links. As a result, even if a model has passed diagnostic testing, it may still be insufficient since nonlinear interactions have not been adequately modeled. The ARIMA's limitation will be revealed if the residuals show a large nonlinear trend. By using ANNs to model residuals, nonlinear relationships can be uncovered. With n input neurons, the ANN model for residuals can be expressed as

$$e_t = f(e_{t-1}, e_{t-2}, , e_{t-n}) + \epsilon_t \tag{2.13}$$

where $epsilon_t$ is the random error and f() is the neural network-determined nonlinear function. It's worth noting that the error term isn't always random if the model f isn't acceptable. As a result, proper model identification is crucial. The forecast from (2.13) will be denoted as $\mathcal{N}_t$, and the combined forecast will be

$$y_t^{'} = L_t^{'} + \mathcal{N}_t \tag{2.14}$$

, where $y_t^{'}$ is the prediction at time t.

In summary, the suggested hybrid system technique comprises of two parts. An ARIMA method is used to assess the linear aspects of the issue in the first stage. The residuals from the ARIMA model are modeled using a neural network model in the second stage. The residuals of the linear model will contain information on the nonlinearity because the ARIMA model cannot represent the nonlinear nature of the data. The findings of the neural network can be utilized to anticipate the ARIMA model's error terms. In determining diverse patterns, the hybrid model takes advantage of the unique features and strengths of both the ARIMA and ANN models. To improve overall modeling and forecasting performance, it may be suitable for modelling linear and nonlinear patterns independently using various models and then integrate the forecasts. As previously indicated, subjective judgment of model order and model soundness is frequently required while developing ARIMA and ANN models. It's possible that the hybrid technique will use poor models. The current BoxJenkins technique, for example, focuses on low order autocorrelation. Even if strong higher-order autocorrelations exist, a model can be considered acceptable if lower-order autocorrelations aren't significant. This suboptimality should not undermine the usefulness of the hybrid model. Many author has pointed out that for a hybrid model to produce superior predictions, the component model must be suboptimal.

CHAPTER 3: RESEARCH METHODOLOGY

3.1 Overview

The research methods outlines what research is, how it is conducted, what type of data is needed for a study, and which data gathering instruments are most suited to the study's objectives. The suggested study employs an inductive research strategy. Moving from individual findings to bigger generalizations is how inductive inquiry works. This study uses only quantitative methods of data collection for the study. The research area will be observed and interpreted using case study research strategy. Case study methodology helps to study in depth and explore the reality. The area under research will be studied to obtain detail information in context of various physical, social, economic, institutional and natural components. This study will also identify adaptation practices by the community, which aids in making the post-earthquake settlement more climate resilient.

3.2 Data Collection

Lack of required format and sufficient data from the ground-based sources, the data were collected from satellite-based station. The satellite data were collected from the web portal https://globalweather.tamu.edu/. Specifically, the data were selected for the location with latitude 28.25 and longitude 83.75, whose co-ordinate locate to Kusma, Parbat. The data we collected is in the timeframe between 1990-2014. The 18 years data between 1990 to 2008 period were used for training purpose and the remaining 6 years data between 2009 to 2014 were used for testing and plotting (validation) purposes.

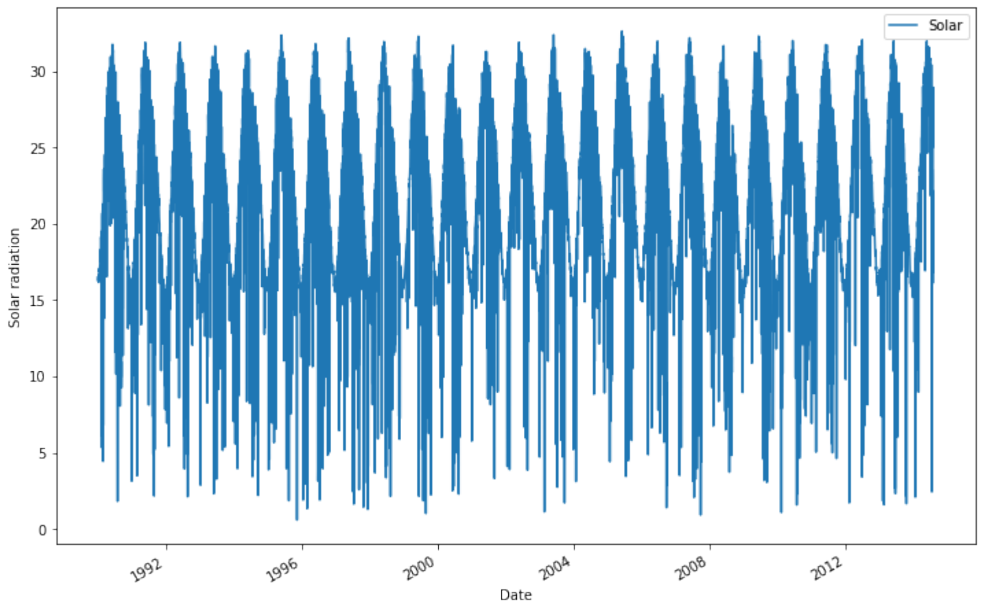


Figure 3.1: Solar radiation potential from 1990 to 2014

3.3 Methods and Materials

There are three basic steps to the research methodology. The first part consists primarily of a desk study, which includes a review of a variety of publications via academic journals, conference proceedings, and books in order to provide a solid foundation for the research findings. Likewise, second phase requires research of the study area and collection of data. Finally, Python 3.8.8 was used to create models and perform different calculations. Lack of required format and sufficient data from the ground-based sources, the data were collected from online mediums. The data was collected from the https://globalweather.tamu.edu/. Specifically, the data were selected for the location with latitude 28.25 and longitude 83.75, whose coordinates lie to Kusma, Parbat. The data were cleaned to only contain dates and Solar potential in MJ/m2. The data ranges from 1990-01-01 to 2014-07-31. The plot of overall dataset is as shown in the figure 3.1. Figure 3.2 is the histogram plot of the solar radiation data set.

The data were cleaned to only contain dates and Solar potential in MJ/m2. The data ranges from 1990-01-01 to 2014-07-31. Total of 8000 daily data points were converted to a total of 294 monthly data. The datas are split into 75-25% i.e 220 and 74 respectively for training and testing.

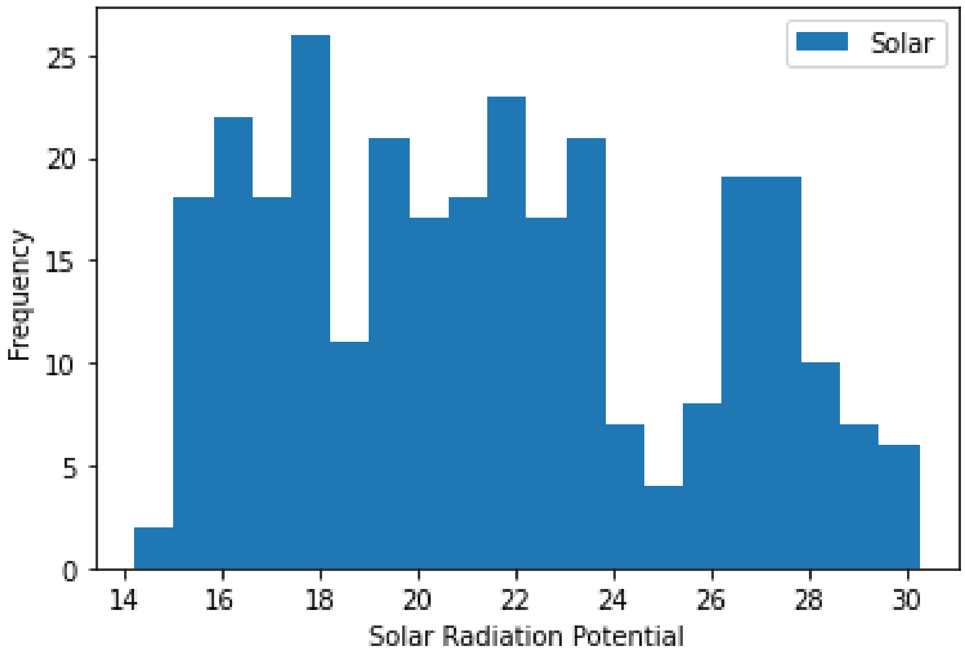


Figure 3.2: Histogram of GSR

3.4 Autoregressive Integrated Moving Average Model

To begin, we must assess whether or not the time series is stationary. For this, we use the Augmented Dickey-Fuller (ADF) analysis. It's a kind of test known as a unit root test. A unit root test, by definition, determines how severely a time series is challenged by a trend. ADF optimizes an information criteria over many lag values using an autoregressive model. The test's null hypothesis would be that the time series isn't stationary and can be expressed by a unit root. Another hypothesis would be that the data series is stationary. We perform this by importing adfuller from statsmodels.tsa.stattools in python. The result is interpreted using the p-value from the test. From the test it is found that the p-value is 0.03 (<0.05) hence we reject the null hypothesis and finalize that the time series of solar radiation potential is stationary. From the result of ADF test we can interpret that the order 'd' which is the differencing term of

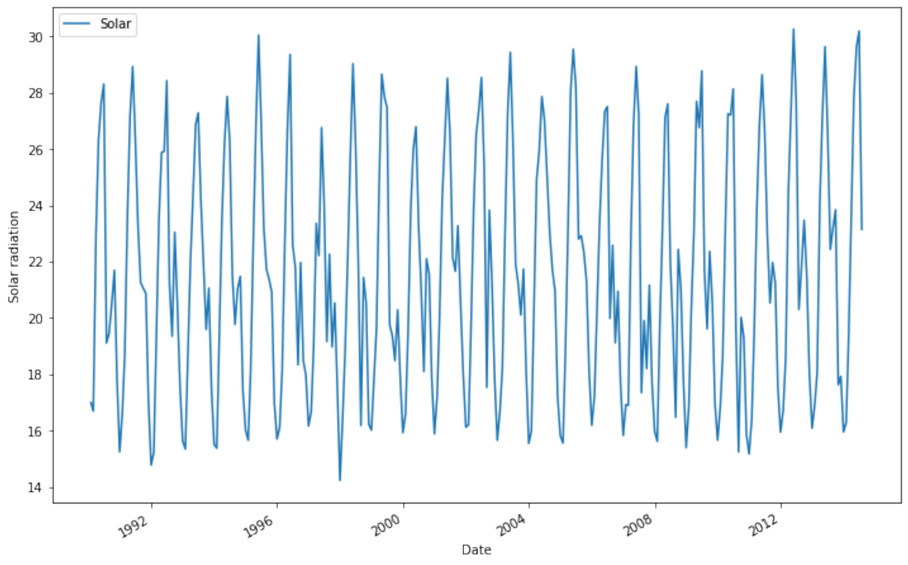


Figure 3.3: Monthly Solar Radiation

the ARIMA is zero. Now next step is to determine the ARIMA model's additional parameters. For this case we used auto_arima function form pmdarima library. This function uses the AIC (Akalke's Information Criterion) score to determine the quality of a given model order. It just

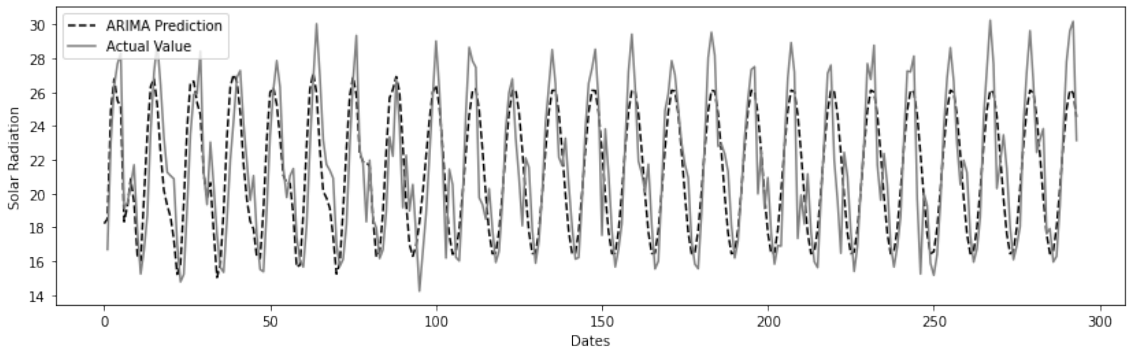
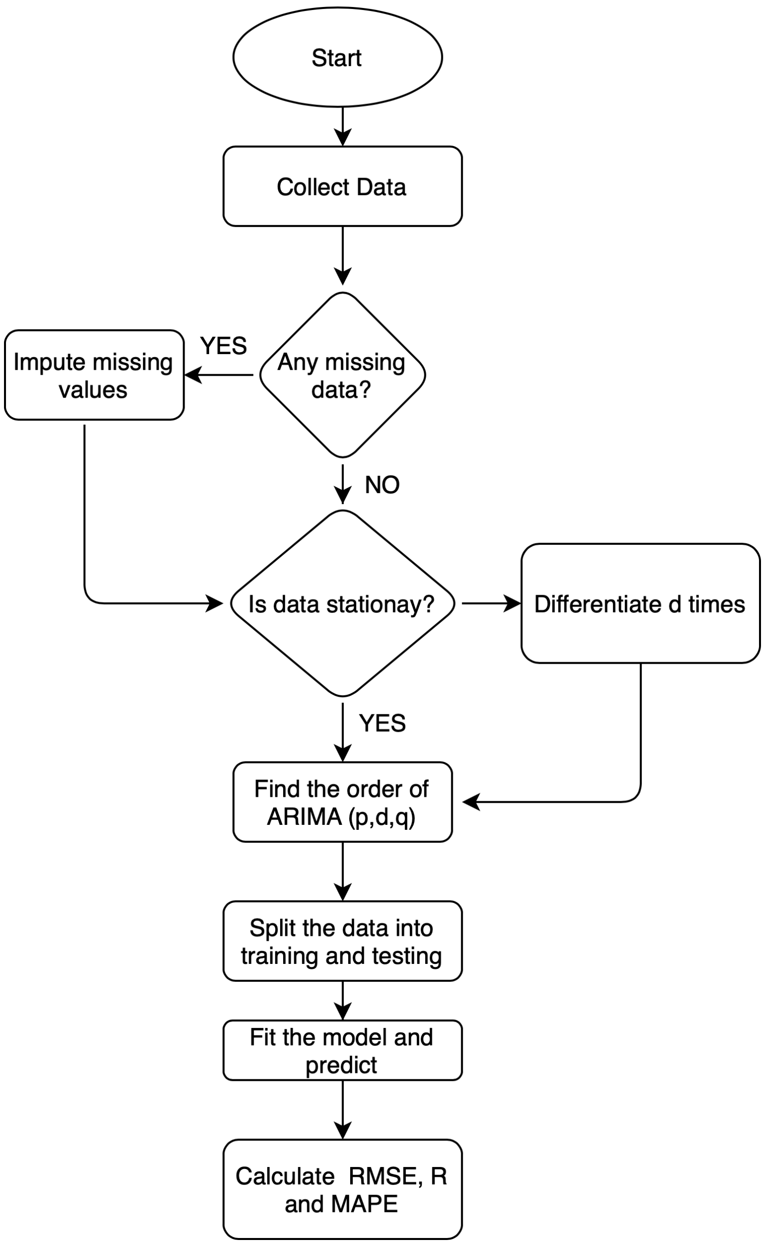


Figure 3.4: Prediction of ARIMA model

tries to reduce the AIC score as much as possible. The lowest AIC score was 315.876 for ARIMA model of order (7, 0, 2). The overall flow of ARIMA model is as shown in the figure 3. After the order is fixed, we trained the data. The content is divided into 2 parts, one as training and one for the testing. The last 74 days data were reserved for testing purpose. This is because we train the system just on dataset first and then hide the testing component of the model. We execute prediction on the test data as once model is built to assess how well it works. To train the model, we simply call the ARIMA method, pass it our collection of data, and specify the ARIMA order we would like to train. The actual data is then compared with the predicted reuslt. The figure below shows the plot between ARIMA prediction and actual data. The residuals



Start
Collect Data
Any missing data?
YES
Impute missing values
NO
Is data stationay?
Differentiate d times
YES
Find the order of ARIMA (p,d,q)
Split the data into training and testing
Fit the model and predict
Calculate RMSE, R and MAPE

Figure 3.5: Flow diagram for ARIMA model

were calculated differentiating the actual and predicted value as described in equation 2.12.

3.5 Machine Learning Model

To develop an ANN model, we used Keras, which is a powerful and open-source library of Python for developing and evaluating deep learning models. The starts by loading the dataset. The data loaded is same as shown in figure (2). We divided the dataset into training and testing, same as we did with the ARIMA model. The data is separated in this way: the last 74 monthly data points are utilized for testing, while the rest of the data is used to train the model. After that, we design a Sequential model and gradually add layers till we achieve satisfactory results. The challenge of increasing the number of input neurons and layers is complex. A method of trial and error experimentation is used to find the ideal network structure. The Dense class' is used to define final fully connected layers. The activation function can be supplied as the second input, and the number of interconnected neurons with in layers can be specified as the first argument. Figure 3.6 depicts the flow diagram of an ANN model.

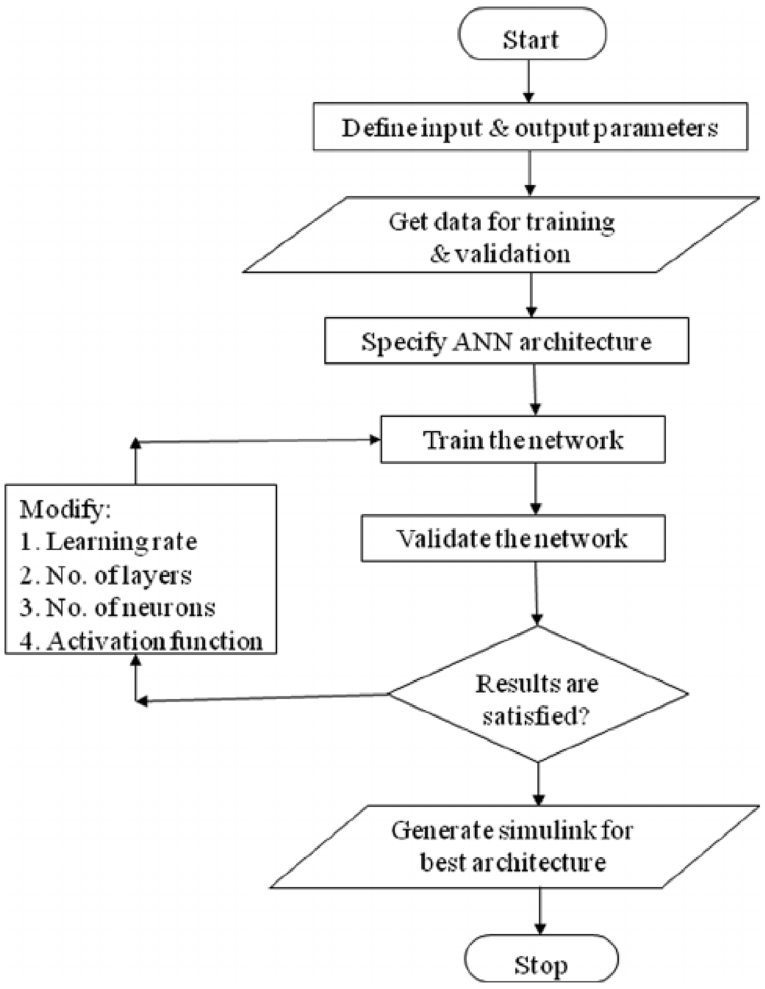


Figure 3.6: Flow diagram for ANN model

Here we have used 50 x 25 x 1 network. In both the input and hidden layers, the activation function is tanh,' whereas in the output nodes, a linear activation function is utilized. Finally after creating the model, we compiled by adding additional properties. The additional properties are added so that while training the network. In our sample dataset, the model determines the best weights to map inputs to outputs. The optimizer was used to look for potential network weights, and the loss function was utilized to examine a set of weights. We utilized the loss function mean squared error' and the optimizer adam' in this scenario. The adam' variant of the gradient descent method is popular because it self-tunes and produces good results in a wide range of problems. We next used fit()' routines to train the imported data. The training is done in epochs, with each epoch divided into batches. The model can be applied to a variety of epochs, with each epoch consisting of one or more batches, depending on the batch size selected. The training process will go through the dataset for a specified number of iterations

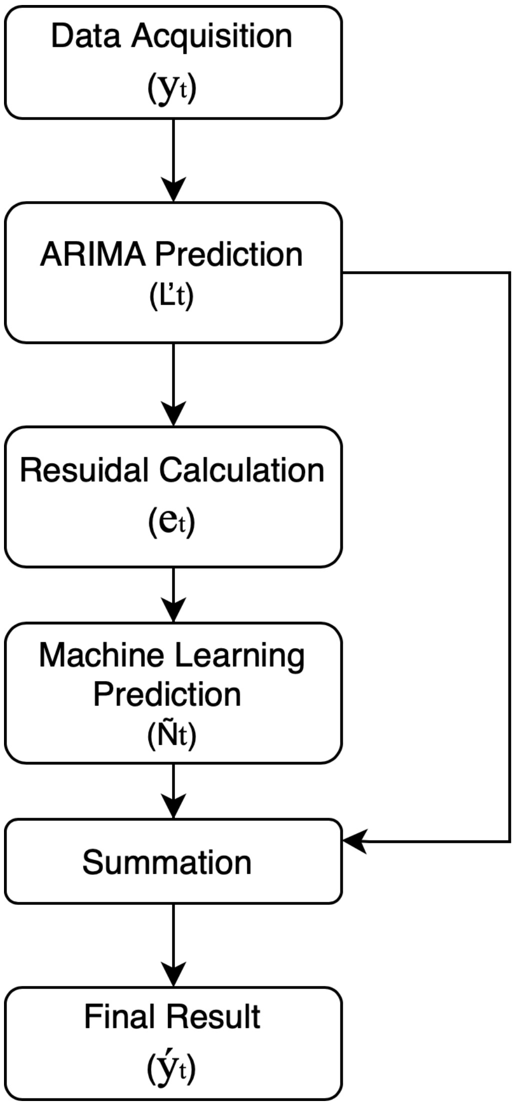


Figure 3.7: Flow diagram for Hybrid ARIMA-ANN model

called epochs, which we must provide with the epochs option. The batch size, which is set via the batch size option, is the number of dataset rows that are evaluated before the model weights are modified inside each epoch. We utilized 2000 epochs and a batch size of 512 in this example. Experimentation and trial and error are also used to select these setups. Finally, we put the model to the test with the testing dataset.

3.6 Hybrid Model

The purposed model makes use of ARIMA along with the machine learning model. Firstly, making use of ARIMA model, residuals is calculated. The residuals are presumed to be nonlinear because the ARIMA model captures the linear element of the time series. These data are fed to the machine learning models ANN and LSTM. The results from these models are summed with the result from the ARIMA model making it the final prediction. The flow of hybrid model is as shown in figure 3.6.

CHAPTER 4: STUDY AREA

We have selected Parbat, Nepal as the research site for this project. Parbat District, is one of the eleven directly constituent districts of the Gandaki province of Nepal. This district has Parbat as its headquarters, with an area of 494 square kilometres. It is Nepal's fourth smallest district. It spans the latitudes of 28° 00' 19" N and 28° 23' 59" N, as well as the longitudes of 830 33' 40" E and 83° 49' 30" E. 66.9% of the area lies in subtropical climatic zones (Karmacharya, 2013). The Kushma Parbat is Hill station is beautiful and diversified geographically. The climate zone includes upper tropical, subtropical, temperature and subalpine in between the elevation ranging from 300m to 3000m where rocky hills surround the place (Subedi and Subedi, 2019). Parbat District, is one of the eleven directly constituent districts of the Gandaki province of Nepal. This district has Parbat as its headquarters, with an area of 494 square kilometres.

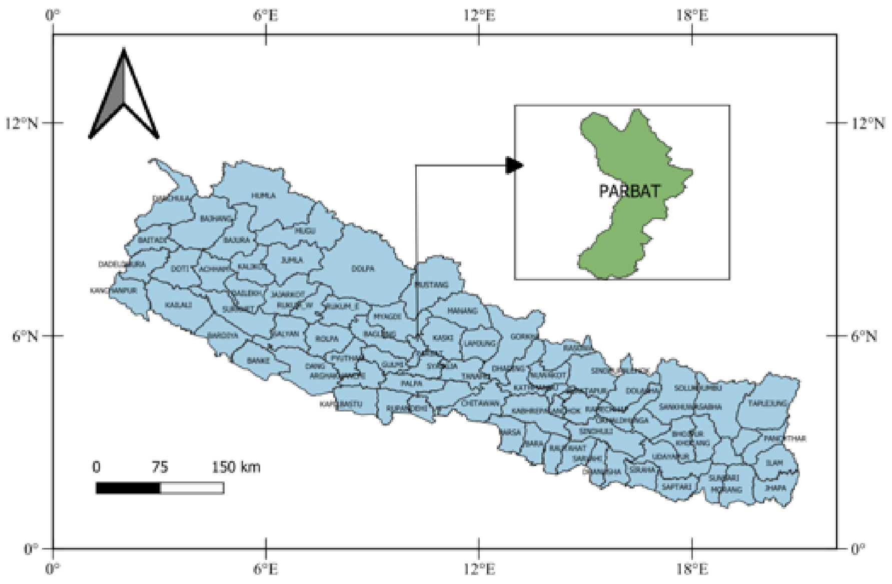


Figure 4.1: Map of Nepal,
Source: Department of Survey, GON, 2020

4.1 Data Set

The solar radiation potential of the Parbat region is represented in this data set. Predicting solar radiation is an important yet difficult task. There have been numerous linear and nonlinear theoretical models developed, however basic random walk models are the most effective in out-of-sample forecasts. Recent neural network implementations in this discipline have yielded mixed results. The information in this article is based on monthly observations from 1990 to 2014, which results in 298 data points in the time series. Each data set is divided into two training and testing samples in order to evaluate the predictive performance of different models. The training set is being used to create a model, and also the test mode is used to analyze it once it's been built. Table 4.1 shows the data composition of the three data sets. Table 4.2 also displays the statistical features of the dataset used to train and test the model.

Table 4.1: Composition of data set

S.N	Series	Sample size	Training set	Testing set
1	Solar Radiation Potential	294	220	74

Table 4.2: Statistical properties of the data

S.N	Parameter	Solar Radiation Potential
1	Unit	MJ/m^2
2	Count	294
3	Mean	21.498
4	Standard Error	0.243
5	Mode	16.998
6	Standard deviation	4.177
7	Sample Variance	17.450
8	Kurtosis	-1.034
9	Skewness	0.282
10	Minimum	14.233
11	Maximum	30.255
12	Sum	6320.570
13	Confidence Level (95.0%) Interval	(21.021, 21.976)

CHAPTER 5: RESULTS AND DISCUSSION

5.1 Results

In this chapter our results are presented, which serve as the basis for answering our research questions on how modern forecasting methods can improve the GSR forecast. The evaluation of the change in results is then introduced across development and test sets. This solves the research problem by allowing us to examine more closely the stability of the model over time. Finally, we compare our results with the existing literature.

While modeling the ARIMA model ADF test resulted in the p-value to be 0.03 confirming the data to be stationary. The best ARIMA model was found for the order (7,0,2) with AIC of 315.876. Plot of the residuals from the ARIMA model is as shown in figure below. The residuals

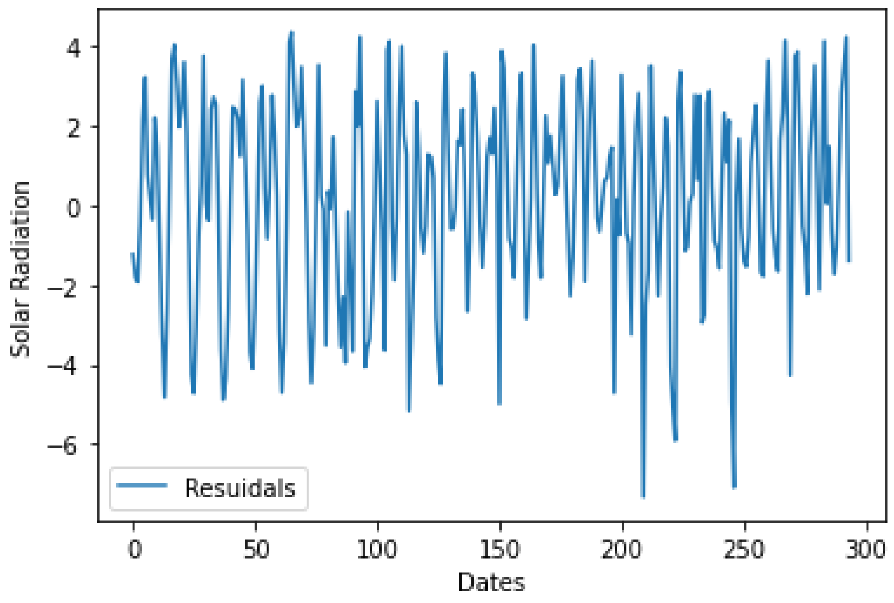


Figure 5.1: Residuals from ARIMA model

are normalized using the min_max_transform function and fed into the neural network. Keras, a strong and open-source Python library for constructing and assessing deep learning models, was used. The dataset was divided into two parts: training and testing. The data is separated in this way: the last 74 monthly data points are utilized for testing, while the rest of the data is used to train the model. After that, we design a Sequential model and gradually add layers till we achieve satisfactory results. The challenge of increasing the number of input neurons and layers is complex. A method of trial and error experimentation is used to find the ideal network structure. The Dense class' is used to define a completely connected layer. The first argument can specify the number of neurons or nodes in the layers, and the second argument can specify the activation function.

The best model for ANN is found using an 50 x 25 x 1 network. In both the input and hidden layers, the activation function is tanh,' whereas in the output layer, a linear activation function is utilized. Finally after creating the model, we compiled by adding additional properties. The other attributes are added so that the model can discover the appropriate weights to map inputs to outputs in our dataset when training the network. The optimizer was used to look for potential network weights, and the loss function was utilized to evaluate a set of weights. We utilized the loss function mean squared error' and the optimizer adam' in this scenario. The gradient descent algorithm adam' is a popular variant since it automatically tunes itself and produces good results in a wide range of tasks. We next used fit()' routines to train the imported data. The training is done in epochs, with each epoch divided into batches. The model can be used for numerous epochs, and each epoch is made up of one or more batches, based on the batch size selected. The training process will go through the dataset for a specified number of iterations called epochs, which we must provide with the epochs option. The batch size is the number of dataset rows that are examined before the model weights are adjusted inside each epoch, as specified by the batch size parameter. We utilized 2000 epochs and a batch size of 512 in this example.

Similar to ANN data are fed into the LSTM model after performing minmaxscaler. This is done because varying time intervals of data have distinct value ranges, then if the data isn't normalized, data near to 0 won't add anything at all to the process of learning. LSTM also expects the data to be in a specific format. We start by defining the Time Series Generator. Here we create a batch of 12 data because the model will predict the next month data by using the previous 12 months data. So the Time Series Generator will have 12 inputs and 1 output feature. The LSTM model is created with 100 neurons, relu is used as an activation function with optimizer adam' and mse as a loss function. After creating the model, we have used 50 epochs to fit the model. After the model is fitted, we predicted the last 30 months of global solar radiation potential.

The proposed model was tested on solar radiation data (Table 4.1) to see how well it predicted the future (Figure 4.4). Three measures were used in this study to predict performance: mean square error (MSE), mean absolute error (MAE), and mean absolute percentage error (MAPE) to compare the suggested model's performance against that of ARIMA, ANN, and LSTM. For solar radiation data, there are monthly records on average mediated 294 (1990-2014). As a training set, the first 206 records have been used. The test set consists of the remaining 74 records. To begin, the ARIMA was used to construct forecast and residuals from a temporal sequence of sun spots. The best well-fitting model is ARIMA (7,0,2). Second, the ARIMA residues are calculated and examined using the ANN and the LSTM. The best ANN and LSTM for approach residues and details are ANN (50x25x1) and LSTM (100x1), respectively. The performance measures are examined after the final forecast (Table 4.2). The ARIMA-ANN model has the lowest error in the specified data set, according to the comparison. In the ARIMA-ANN hybrid forecast, RMSE, MAE and MAPE are 1.719, 1.330 and 6.456% respectively. For the forecast of ARIMA-LSTM, RMSE, MAE and MAPE is 2.029, 1.638 and 7.580%, which are higher than the provision of ARIMA-ANN hybrid, since, in general, LSTM requires large data series for learning. The ARIMA-ANN analysis, on the other hand, probably is the best framework because of error measurements are low and the square value is high. As a result, the suggested model is the most effective for forecasting solar radiation time series.

The results of different models are as shown in the table below.

Table 5.1: Performance of each Model

S.N	Models	R^2	RMSE	MAPE	MAE
1	ARIMA	0.809	1.928	7.276	1.545
2	ANN	0.835	2.426	9.776	1.972
3	LSTM	0.836	1.892	6.216	1.365
4	ARIMA - ANN	0.847	1.719	6.456	1.330
5	ARIMA - LSTM	0.810	2.029	7.580	1.638

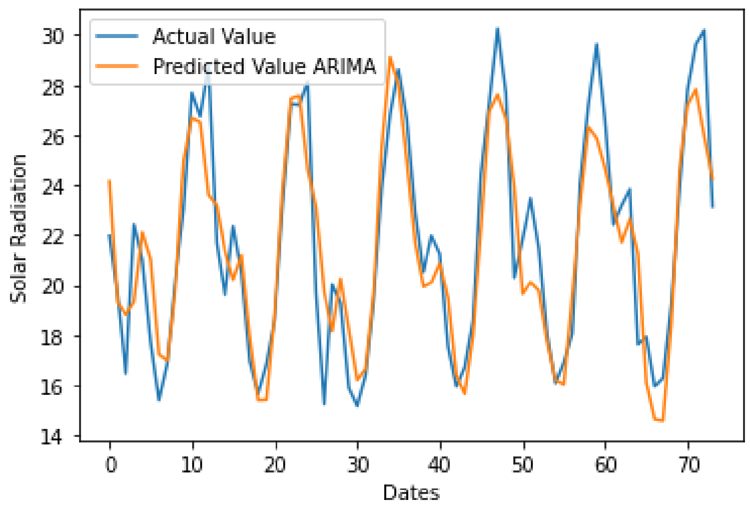


Figure 5.2: Actual and Predicted value of Solar radiation using ARIMA model

The plot of actual versus predicted solar radiation potential using ARIMA, ANN, LSTM, ARIMA-ANN and ARIMA-LSTM is shown in figure 5.1 - 5.6.

5.2 Discussions

The aforementioned elements led us to our proposed model consisting of an ARIMA component, an ANN component and a LSTM component, which was responsible for explaining the linear and non-linear tendencies in the data. Our experimental approach was populated by data on daily solar radiation potential converted to monthly solar radiation potential data. Summarizing,

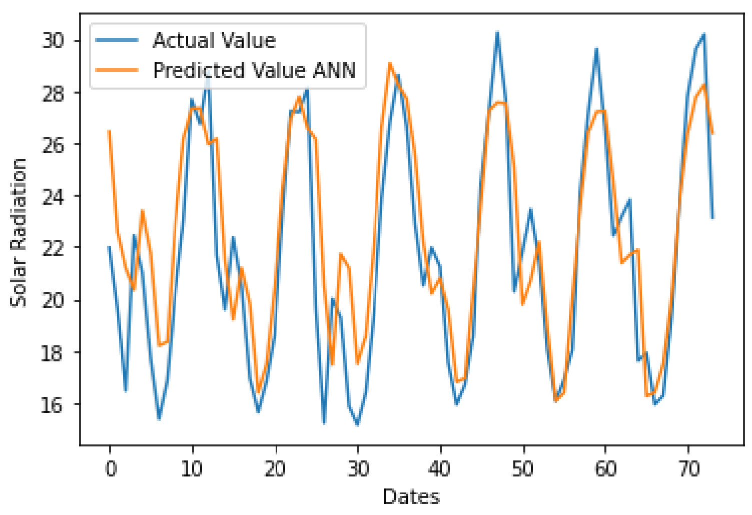


Figure 5.3: Actual and Predicted value of Solar radiation using ANN model

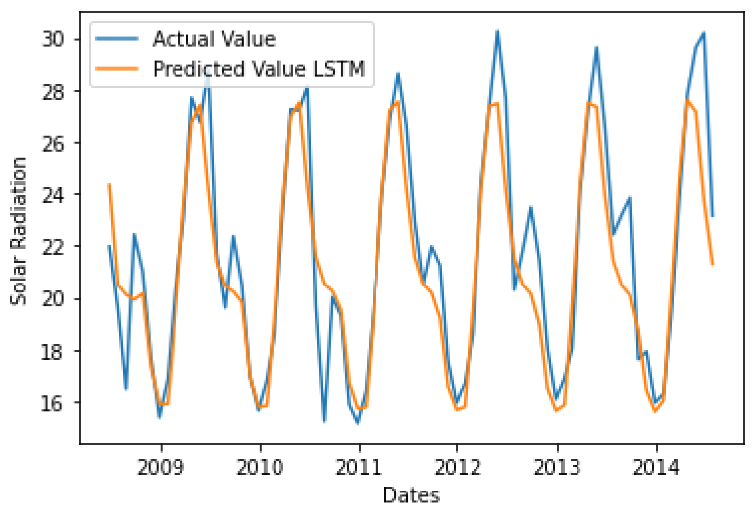


Figure 5.4: Actual and Predicted value of Solar radiation using LSTM model

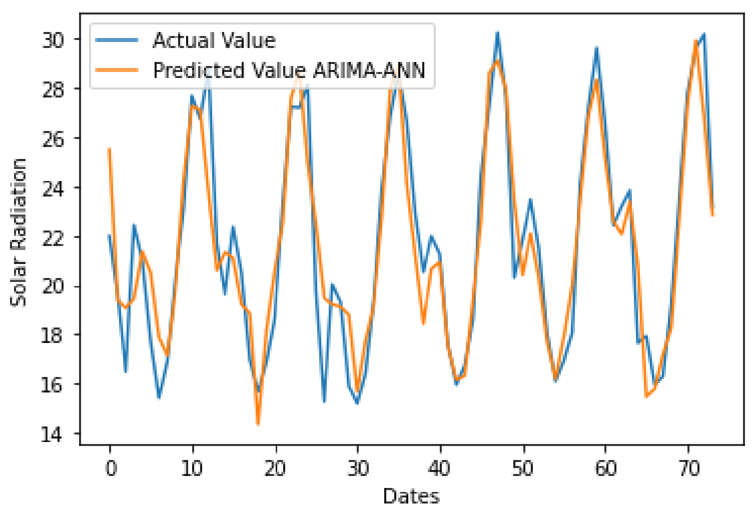


Figure 5.5: Actual and Predicted value of Solar radiation using ARIMA-ANN model

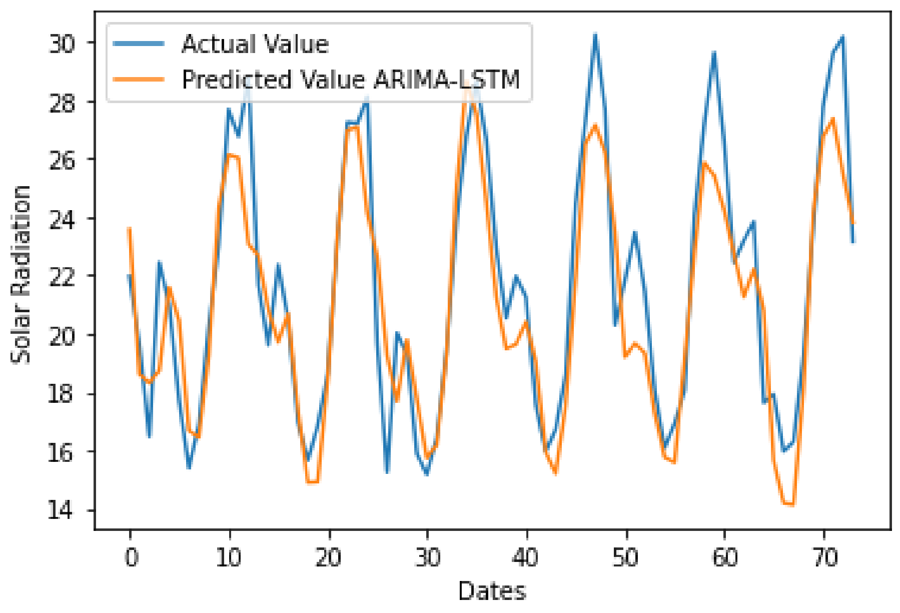


Figure 5.6: Actual and Predicted value of Solar radiation using ARIMA-LSTM model

this approach included a range of benchmarks consisting of conventional standalone methods for estimating GSR and an alternative hybrid machine learning method. This ensured a comparative design of the experiment which aimed to provide findings related to our research question. In chapter 3, different standalone models such as ARIMA, ANN and LSTM were used. In time series forecasting, a hybrid approach of both the ARIMA-ANN and ARIMA-LSTM has indeed been constructed to achieve significant advantages among some of the various model types and sophistication levels. The models' ability to forecast was validated using test data sets.

According to the findings, the ARIMA-ANN model has the best RMSE and MAPE performance for the provided data set. The advantage of integrating the ARIMA with the ANN in incorporating linear and nonlinear components of the approximations and precision without linear or nonlinear assumptions means an improvement in prediction performance. The proposed methodology for the purpose of achieving an effective and accurate implementation of solar radiation prediction has been outlined in this approach. The approach leverages the machine learning approaches to achieve the prescribed goals for the prediction. One of the most important aspects in greatly boosting the accuracy of the method has been the introduction of machine learning algorithms in the form of ANN and LSTM. The RMSE is utilized to achieve the performance metrics of the approach. Our method is filled with selected data and the results provided illustrate the untapped potential of modern forecasting methods for providing input accuracy for energy planning. The prediction method selected in our article has overall better predictive performance than conventional methods for all test sets.The RMSE achieved by our implementation indicates significant improvements in the accuracy of the prediction of the solar radiation. The RMSE achieved by using hybrid of ARIMA with the LSTM is 27.49 whereas with the ANN module achieves 36.55. LSTMs are unquestionably more intricate and difficult to train, and they rarely outperform simple ARIMA models or other machine learning models. This is why the hybrid of ARIMA-LSTM performed poorly for the given dataset.With an arbitrary number of inputs but a fixed number of outputs, neural networks provide the ability to learn potentially chaotic and nonlinear relationships. The application of robust machine learning algorithms through the use of high level libraries using the Keras approach has been one of the key features for such a drastic improvement in the performance metrics and the prediction through the proposed methodology.

CHAPTER 6: CONCLUSION AND RECOMMENDATIONS

6.1 Conclusion

This thesis aimed at investigating how modern approaches for forecasts can contribute to more stable energy forecast. The revision of the review and literature provided the reasoning of why this research should be made. The hybrid methodology has addressed the time series forecasting problem of the global solar radiation problem for energy planning. Furthermore, the review of results revealed a decision-making area resolved with hybrid methods, despite the developments of various standalone methods applicable to the problem.

Finally, the proposed hybrid models have outperformed standard single models in terms of forecasting abilities. Specifically, ARIMA-ANN hybrid can be used as alternative models for time series prediction. The engineers, scientist, planners and designers can apply the hybrid models specified for the estimation solar radiation potential to obtain more accurate prediction results in this study area. Also, similar prediction can be done using these models in other location of Nepal. Furthermore, the hybrid models for solar radiation forecasting can be applied to other climatic variables like temperature, wind, humidity, etc. sharing the similar characteristic with the solar radiation as well.

6.2 Future Research

Further research on these field may be because the higher added value of modern forecasting methods will lead to greater possibilities for actual energy adoption. The predictive ability can be improved by exploring different learning algorithms or replacing databases. Parameter optimization is expensive in time and computational work, which greatly limits the scope of this article. In addition, the inclusion of explanatory time series is an additional area of research that may yield interesting discoveries, and further strengthen the forecasting capabilities that modern forecasting methods can demonstrate to attract professionals. However, as mentioned above, we also want to point out that further research to improve the predictive capabilities of these new methods should be carried out at the same time as attempts to automate and simplify their implementation in real-world problems to better reduce barriers to energy mapping. Therefore, we will pay equal attention to the development of frameworks for implementing modern methods and the continuous expansion of model complexity. Machine learning as well as deep learning algorithms have yet to deliver the expected results for univariate time series prediction, and additional research is needed.

REFERENCES

Adhikari, R. and Agrawal, R. (2014). A combination of artificial neural network and random walk models for financial time series forecasting. *Neural Computing and Applications*, 24(6):1441–1449.

Atienza, R. (2018). *Advanced Deep Learning with Keras: Apply deep learning techniques, autoencoders, GANs, variational autoencoders, deep reinforcement learning, policy gradients, and more*. Packt Publishing Ltd.

Awasthi, J. and Poudyal, K. N. (2018). Estimation of global solar radiation using empirical model on meteorological parameters at simara airport, bara, nepal. *Journal of the Institute of Engineering*, 14(1):143–150.

Bahrami, S., Hooshmand, R.-A., and Parastegari, M. (2014). Short term electric load forecasting by wavelet transform and grey model improved by pso (particle swarm optimization) algorithm. *Energy*, 72:434–442.

Beizer, B. (1995). *Black-box testing: techniques for functional testing of software and systems*. John Wiley & Sons, Inc.

Bo, W., Shouyang, W., and Lai, K. (2007). A hybrid arch-m and bp neural network model for gsci futures price forecasting. In *International conference on computational science*, pages 917–924. Springer.

Box, G. and Jenkins, G. (1970). Control. *Halden-Day, San Francisco*.

Carpenter, G. A. and Grossberg, S. (2010). Adaptive resonance theory.

Chantasut, N., Charoenjit, C., and Tanprasert, C. (2004). Predictive mining of rainfall predictions using artificial neural networks for chao phraya river. In *4th International conference of the asian federation of information technology in agriculture and the 2nd world congress on computers in agriculture and natural resources*, pages 9–12.

Chatfield, C. (1988). What is the bestmethod of forecasting? *Journal of Applied Statistics*, 15(1):19–38.

De Gooijer, J. G. and Hyndman, R. J. (2006). 25 years of time series forecasting. *International journal of forecasting*, 22(3):443–473.

De Gooijer, J. G. and Kumar, K. (1992). Some recent developments in non-linear time series modelling, testing, and forecasting. *International Journal of Forecasting*, 8(2):135–156.

Deng, W., Wang, G., and Zhang, X. (2015). A novel hybrid water quality time series prediction method based on cloud model and fuzzy forecasting. *Chemometrics and Intelligent Laboratory Systems*, 149:39–49.

Denton, J. W. (1995). How good are neural networks for causal forecasting? *The Journal of Business Forecasting*, 14(2):17.

Ezzine, H., Bouziane, A., and Ouazar, D. (2014). Seasonal comparisons of meteorological and agricultural drought indices in morocco using open short time-series data. *International Journal of Applied Earth Observation and Geoinformation*, 26:36–48.

Feng, X., Li, Q., Zhu, Y., Hou, J., Jin, L., and Wang, J. (2015). Artificial neural networks forecasting of pm2. 5 pollution using air mass trajectory based geographic model and wavelet transformation. *Atmospheric Environment*, 107:118–128.

Fildes, R. and Makridakis, S. (1995). The impact of empirical accuracy studies on time series analysis and forecasting. *International Statistical Review/Revue Internationale de Statistique*, pages 289–308.

Fouilloy, A., Voyant, C., Notton, G., Motte, F., Paoli, C., Nivet, M.-L., Guillot, E., and Duchaud, J.-L. (2018). Solar irradiation prediction with machine learning: Forecasting models selection method depending on weather variability. *Energy*, 165:620–629.

Garrett, K. A., Dobson, A., Kroschel, J., Natarajan, B., Orlandini, S., Tonnang, H. E., and Valdivia, C. (2013). The effects of climate variability and the color of weather time series on agricultural diseases and pests, and on decisions for their management. *Agricultural and Forest Meteorology*, 170:216–227.

Ghosh, J. and Deuser, L. (1995). Classification of spatiotemporal patterns with applications to recognition of sonar sequences. In *Neural Representation of Temporal Patterns*, pages 227–250. Springer.

Haykin, S. (2010). *Neural networks and learning machines, 3/E.* Pearson Education India.

Heng, J., Wang, J., Xiao, L., and Lu, H. (2017). Research and application of a combined model based on frequent pattern growth algorithm and multi-objective optimization for solar radiation forecasting. *Applied Energy*, 208:845–866.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hornik, K., Stinchcombe, M., and White, H. (1990). Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural networks*, 3(5):551–560.

Hwarng, H. (2001). Insights into neural-network forecasting of time series corresponding to arma(p,q) structures. *Omega*, 29(3):273–289.

Iqbal, M. (2012). *An introduction to solar radiation*. Elsevier.

Jain, A. K., Mao, J., and Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, 29(3):31–44.

Jenkins, G. M. (1982). Some practical aspects of forecasting in organizations. *Journal of forecasting*, 1(1):3–21.

Karmacharya, S. (2013). *THE PROBLEMS AND PROSPECTS OF TOURISM IN NEPAL (A CASE STUDY OF PARBAT, DISTRICT, NEPAL)*. PhD thesis, Central Department of Economics Tribhuvan University, Kirtipur, Kathmandu, Nepal.

Khashei, M. and Bijari, M. (2011). A novel hybridization of artificial neural networks and arima models for time series forecasting. *Applied soft computing*, 11(2):2664–2675.

Lee, Y.-S. and Liu, W.-Y. (2014). Forecasting value of agricultural imports using a novel two-stage hybrid model. *Computers and electronics in agriculture*, 104:71–83.

Liu, H., Xie, D., and Wu, W. (2008). Soil water content forecasting by ann and svm hybrid architecture. *Environmental monitoring and assessment*, 143(1):187–193.

Makridakis, S., Andersen, A., Carbone, R., Fildes, R., Hibon, M., Lewandowski, R., Newton, J., Parzen, E., and Winkler, R. (1982). The accuracy of extrapolation (time series) methods: Results of a forecasting competition. *Journal of Forecasting*, 1(2):111–153.

Markham, I. S. and Rakes, T. R. (1998). The effect of sample size and variability of data on the comparative performance of artificial neural networks and regression. *Computers & operations research*, 25(4):251–263.

Notton, G., Nivet, M.-L., Voyant, C., Paoli, C., Darras, C., Motte, F., and Fouilloy, A. (2018). Intermittent and stochastic character of renewable energy sources: Consequences, cost of intermittence and benefit of forecasting. *Renewable and sustainable energy reviews*, 87:96–105.

Patil, R. B. (1990). *Neural networks as forecasting experts: test of dynamic modeling over time series data*. PhD thesis, Oklahoma State University.

Pulido-Calvo, I. and Gutierrez-Estrada, J. C. (2009). Improved irrigation water demand forecasting using a soft-computing hybrid model. *Biosystems engineering*, 102(2):202–218.

Ribeiro, C. O. and Oliveira, S. M. (2011). A hybrid commodity price-forecasting model applied to the sugar–alcohol sector. *Australian Journal of Agricultural and Resource Economics*, 55(2):180–198.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Sadaei, H. J., Enayatifar, R., Abdullah, A. H., and Gani, A. (2014). Short-term load forecasting using a hybrid model with a refined exponentially weighted fuzzy time series and an improved harmony search. *International Journal of Electrical Power & Energy Systems*, 62:118–129.

Shahwan, T. and Odening, M. (2007). Forecasting agricultural commodity prices using hybrid neural networks. In *Computational intelligence in economics and finance*, pages 63–74. Springer.

Subedi, N. and Subedi, I. P. (2019). Pollinator insects and their impact on crop yield of mustard in kusma, parbat, nepal. *Journal of Institute of Science and Technology*, 24(2):68–75.

Tang, Z. and Fishwick, P. A. (1993). Feedforward neural nets as models for time series forecasting. *ORSA journal on computing*, 5(4):374–385.

Tarpley, J. (1979). Estimating incident solar radiation at the surface from geostationary satellite data. *Journal of Applied Meteorology and Climatology*, 18(9):1172–1181.

Taskaya-Temizel, T. and Casey, M. C. (2005). A comparative study of autoregressive neural network hybrids. *Neural Networks*, 18(5-6):781–789.

Wang, J.-J., Wang, J.-Z., Zhang, Z.-G., and Guo, S.-P. (2012). Stock index forecasting based on a hybrid model. *Omega*, 40(6):758–766.

WECS, W. (2010). Energy commission secretariat. *Energy Synopsis Report, Government of Nepal*.

Wei, L.-Y. (2016). A hybrid anfis model based on empirical mode decomposition for stock time series forecasting. *Applied Soft Computing*, 42:368–376.

Widrow, B. (1960). An adaptive adaline neuron using chemical. *Stanford University*.

Wold, H. (1938). *A study in the analysis of stationary time series*. PhD thesis, Almqvist & Wiksell.

Xiao, Y., Xiao, J., and Wang, S. (2012). A hybrid forecasting model for non-stationary time series: An application to container throughput prediction. *International Journal of Knowledge and Systems Science (IJKSS)*, 3(2):67–82.

Yang, J., Rivard, H., and Zmeureanu, R. (2005). Building energy prediction with adaptive artificial neural networks. In *Ninth International IBPSA Conference Montréal*, pages 15–18. Citeseer.

Yule, G. U. (1926). Why do we sometimes get nonsense-correlations between time-series?–a study in sampling and the nature of time-series. *Journal of the royal statistical society*, 89(1):1–63.

Zhang, C., Huang, L., and Zhao, Z. (2013). Research on combination forecast of port cargo throughput based on time series and causality analysis. *Journal of Industrial Engineering and Management (JIEM)*, 6(1):124–134.

Zhang, G. P. (2003). Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50:159–175.

Zhang, G. P., Patuwo, B. E., and Hu, M. Y. (2001). A simulation study of artificial neural networks for nonlinear time-series forecasting. *Computers & Operations Research*, 28(4):381–396.

Zhao, Z., Chen, W., Wu, X., Chen, P. C., and Liu, J. (2017). Lstm network: a deep learning approach for short-term traffic forecast. *IET Intelligent Transport Systems*, 11(2):68–75.

Appendices

APPENDIX A: Importing Libraries in Python

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from statsmodels.tsa.arima_model import ARIMA
from statsmodels.graphics.tsaplots import plot_acf
from statsmodels.graphics.tsaplots import plot_pacf
from statsmodels.tsa.seasonal import seasonal_decompose
from sklearn.metrics import mean_squared_error, mean_absolute_error, mea
from statsmodels.tsa.stattools import adfuller
import torch
from math import sqrt
import torch.nn as nn
import tensorflow as tf
from itertools import product
from google.colab import drive
from scipy.stats import sem
from statistics import mode
from collections import Counter
from scipy.stats import kurtosis
import scipy.stats as st
from statsmodels.tsa.stattools import adfuller
from keras.preprocessing.sequence import TimeseriesGenerator
from keras.models import Sequential
from keras.layers import Dense,Activation,Dropout
from sklearn import preprocessing
from keras.wrappers.scikit\_learn import KerasRegressor
from keras.layers.recurrent import LSTM
```

APPENDIX B: Code Used

```
df=pd.read_csv('Mydata.csv',index_col='Date', parse_dates=True)

X = df['Solar']
result = adfuller(X)
d=-1
print('ADF␣Statistic:␣\%f' \% result[0])
print('p-value:␣\%f' \% result[1])
print('Critical␣Values:')
for key, value in result[4].items():
    print('\t\%s:␣\%.3f' \% (key, value))
if result[1]<=0.05:
    d=0
else:
for i in range(1,4):
        X=pd.DataFrame(X)
        X=X-X.shift(periods=1)
        X=X.dropna()
        X=np.array(X)
        result = adfuller(X)
        print('d=\%f' \% i)
        print('ADF␣Statistic:␣\%f' \% result[0])
        print('p-value:␣\%f' \% result[1])
        print('Critical␣Values:')
        for key, value in result[4].items():
            print('\t\%s:␣\%.3f' \% (key, value))
        if result[1]<=0.05:
            d=i
            break

q_arima = range(0, 3)
d_arima=d
p_arima = range(0, 8)
AIC_arima = []
ARIMAX_model = []
pdqs = [(x[0], d_arima, x[1]) for x in list(itertools.product(p_arima, q

for pdq in pdqs:
        try:
            mod = ARIMA(df['Solar'],order=pdq)

            results = mod.fit()
            print('ARIMAX{}␣-␣AIC:{}'.format(pdq, results.aic))
            AIC_arima.append(results.aic)
            ARIMAX_model.append([(1,1,0), pdq])
        except:
                continue

order = ARIMAX_model[AIC_arima.index(min(AIC_arima))][1]
print(order)
model = ARIMA(df['Solar'], order)
fit = model.fit()

#split into test and train
percentage = 0.75
series = df['Solar'].tolist()
size = int(len(series) * 0.75)
train, test = series[0:size], series[size:len(series)]
model = ARIMA(train , order = (7,0,2))
model_fit = model.fit()

model=ARIMA(df['Solar'],order=(7,0,2))
model=model.fit(transparams=False)
model.summary()

#ARIMA Forecast
history = train
for t in range(len(test)):
    model = ARIMA(history, order=(7,0,2))
    model_fit = model.fit(disp=0)
    output = model_fit.forecast()
    yhat = output[0]
    resid_test.append(test[t] - output[0])
    predicted1.append(yhat)
    obs = test[t]
    history.append(obs)
    print('predicted=%f, expected=%f' % (yhat, obs))
test_resid = []
for i in resid_test:
    test_resid.append(i[0])
error = mean_squared_error(test, predicted1)
print('Test MSE: %.3f' % error)
plt.plot(test)
plt.plot(predicted1)
plt.show()

#Resuidal Analysis
train, test = series[0:size], series[size:len(series)]
model = ARIMA(train, order=(7,0,2))
model_fit = model.fit(disp=0)
print(model_fit.summary())
residuals = pd.DataFrame(model_fit.resid)
residuals.plot()
plt.show()
residuals.plot(kind='kde')
plt.show()
print(residuals.describe())

#ARIMA-ANN Hybrid
window_size = 50
def make_model(window_size):
    model = Sequential()
    model.add(Dense(50, input_dim=window_size, kernel_initializer="unifor
    activation="tanh"))
    model.add(Dense(25, kernel_initializer="uniform", activation="tanh"))
    model.add(Dense(1))
    model.add(Activation("linear"))
    model.compile(loss='mean_squared_error', optimizer='adam')
    return model

model = make_model(50)
min_max_scaler = preprocessing.MinMaxScaler()
train = np.array(residuals).reshape(-1,1)
```

```
train_scaled = min_max_scaler.fit_transform(residuals)

train_X,train_Y = [],[]
for i in range(0 , len(train_scaled) - window_size):
    train_X.append(train_scaled[i:i+window_size])
    train_Y.append(train_scaled[i+window_size])

new_train_X,new_train_Y = [],[]
for i in train_X:
    new_train_X.append(i.reshape(-1))
for i in train_Y:
    new_train_Y.append(i.reshape(-1))
new_train_X = np.array(new_train_X)
new_train_Y = np.array(new_train_Y)
model.fit(new_train_X,new_train_Y, epochs=500, batch_size=512, validation

test_extended = train.tolist()[-1*window_size:] + test_resid
test_data = []
for i in test_extended:
    try:
        test_data.append(i[0])
    except:
        test_data.append(i)
test_data = np.array(test_data).reshape(-1,1)
min_max_scaler = preprocessing.MinMaxScaler()
test_scaled = min_max_scaler.fit_transform(test_data)
test_X,test_Y = [],[]
for i in range(0 , len(test_scaled) - window_size):
    test_X.append(test_scaled[i:i+window_size])
    test_Y.append(test_scaled[i+window_size])
    new_test_X,new_test_Y = [],[]
for i in test_X:
    new_test_X.append(i.reshape(-1))
for i in test_Y:
    new_test_Y.append(i.reshape(-1))
new_test_X = np.array(new_test_X)
new_test_Y = np.array(new_test_Y)
predictions = model.predict(new_test_X)
```

```
predictions_rescaled=min_max_scaler.inverse_transform(predictions)
Y = pd.DataFrame(new_test_Y)
pred = pd.DataFrame(predictions)
plt.plot(Y)
plt.plot(pred , color = 'r')
plt.show()
error = mse(test_resid ,predictions_rescaled)
print('Test␣MSE:␣%.3f' % error)

red_final = predictions_rescaled + predicted1
Y = pd.DataFrame(test)
pred = pd.DataFrame(pred_final)
plt.plot(Y)
plt.plot(pred , color = 'r')
plt.show()


#LSTM
scaler = MinMaxScaler()
scaler.fit(train)
scaled_train = scaler.transform(train)
n_input = 12
n_features = 1
generator = TimeseriesGenerator(scaled_train, scaled_train, length=n_inpu
model = Sequential()
model.add(LSTM(100, activation='relu', input_shape=(n_input, n_features))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')
model.fit(generator,epochs=50)
test_predictions = []

first_eval_batch = scaled_train[-n_input:]
current_batch = first_eval_batch.reshape((1, n_input, n_features))

for i in range(len(test)):

    current_pred = model.predict(current_batch)[0]
    test_predictions.append(current_pred)
```