Maharashtra Rainfall forecasting

ARTICLE HISTORY

Compiled June 1, 2019

ABSTRACT

One of the main components of the hydrological cycle is Precipitation. Forecasting accurate precipitation is really important in our day to day activities. It helps us to manage water resources and crop production. Farmers can make appropriate decisions based on future precipitation, and this, in turn, helps us in many ways. We cannot easily predict precipitation as it is very dynamic in nature and also it depends on different factors. Time series forecasting has been a fascinating problem for many years and different techniques have been proposed. Traditional forecasting methods like ARIMA and modern methods like Artificial Neural Networks(ANN), Recurrent Neural Networks(RNN) and Long Short Term Memory(LSTM) are the most popular methods to forecast the precipitation. Artificial Neural Networks and Deep learning methods have been widely used to remember long term dependencies. In this paper, we compared ARIMA, ANNs, RNN, and LSTM on all the districts of Maharashtra and the thus obtained results were further analysed in the perspective of the different climatic zones of the state. The rainfall in Maharashtra is highly unstable during the periods of the rainy season. We have used the monthly precipitation data of Maharashtra from 1901 to 2002, provided by the Indian Water Portal. The data of the last five years, that is, from 1997 to 2002, is used for the validation of models. A comparison of these methods on all the districts of Maharashtra, which were further categorized as different climatic zones, has been presented. For the evaluation of the

performance of these methods, Root Mean Square Error(RMSE), Mean Absolute Error(MAE) and Correlation Coefficient(R) were used. Different methods performed well in different districts of Maharashtra. The research showed that RNN and LSTM dominated ARIMA and ANN methods. Adding to these, we also proposed a few ways of using different data as input for LSTM which enhanced the performance of results in seasonal months. As a whole, this study shows that we can apply deep learning methods for forecasting precipitation and the effective usage of Long Short Term Memory(LSTM) to improve the performance of rainy seasons.

Keywords: Monthly Precipitation, Artificial Neural Networks, Long Short Term Memory, ARIMA, Forecasting.

1. Introduction

Precipitation is one of the most crucial elements in the hydrological cycle. It is the only source of fresh water in most of the regions of India. Forecasting rainfall ahead of time helps us to manage water resources, avoid floods, droughts and also used in the estimation of various useful parameters such as runoff, agriculture, groundwater storage. Farmers can take useful decisions in crop production and decrease the percentage of crop damage. Hence, it is important to understand the dynamic nature of the precipitation.

The dynamic nature of precipitation has become a major challenge to accurately forecast the precipitation. In general, the forecasting methods are divided into knowledge-driven and data-driven approaches (Ouyang et al 2016). Due to the recent advancements in the algorithms and in the computational power, data-driven approaches are dominating knowledge-driven approaches. Knowledge-driven approaches require different factors like temperature, wind,

pressure, and humidity to forecast. The data of these factors may not be available all the time. Data-driven approaches take input as previous precipitation data to forecast the rainfall. In today's world, data has been extensively generated, stored and processed efficiently so that we are able to solve most of the problems using large datasets (Deepak Kumar et al 2019).

The most popular statistical data-driven methods are Support Vector Regression (Vapnik 1995), Gene Expression Programming (GEP), Box-Jenkins methods(Box and Jenkins 1970), which include Auto-Regressive(AR), Moving Average(MA), Auto-Regressive Moving Average(ARMA) and Auto Regressive Integrated Moving Average(ARIMA) and Seasonal Auto-Regressive Integrated Moving Average(SARIMA). The performance of the linear statistical methods for non-linear processes may not be good because the methods were developed based on the assumption of linear data (Mohini P. Darji et al 2015). Artificial Neural Networks(ANN) has been used as an alternative technique for time series forecasting and it acquired huge popularity in recent years. ANNs are non -linear, data-driven and universal approximators (Guoqiang Zhang et al 1997). ANNs can find hidden patterns from the raw input data and provide generalized results based on the experience. They can capture non-linearity information from the data. A variation of ANNs was developed to overcome the drawbacks of SARIMA, that could improve the performance of seasonal time series (Coskun Hamzaçebi 2008). A hybrid methodology was proposed which combines ARIMA and ANN to take advantage of both linear and non-linear modelling. Earlier studies showed that hybrid models performed better than individual models

(Peter Zhang 2001). A recursive approach was developed to forecast long term monthly precipitation using Genetic Programming(GP), which uses only the previous monthly precipitation (Suning Liu 2018). In general, ANN is commonly preferred to GEP to forecast precipitation. But, GEP is used in other fields. In addition, two different hybrid models GEP-ARCH and ANN-ARCH were introduced to predict future precipitation. The results concluded that the hybrid methods have shown better results than GEP and ANN (Saeid Mehdizadeh et al 2017).

In recent years, deep learning has gained immense popularity in various domains. Researchers were able to cross the state-of-the-art models with the help of deep learning (Yann LeCun et al 2015). It has created benchmarks in different fields like signal processing (Li Deng et al 2013), image processing (Athanasios Voulodimos et al 2018), natural language processing and understanding (Tom Young et al 2018). Deep learning techniques have the capability to automatically detect the latent features from the raw data. As the number of layers increases, they can incrementally detect more complex features from those detected in the lower layers of the network without any human intervention. This reduces the risk of missing any important features which are very crucial in performing the desired task, contrary to the method of using conventional hand engineered features. Some popular deep learning algorithms such as the convolutional neural networks(CNNs) and recurrent neural networks(RNNs) are used in image processing/computer vision applications and in natural language/signal processing applications respectively. This has inspired the deep learning community to ap-

ply these techniques in hydrological time series applications. RNNs are generally used for time-dependent applications. One of the most popular and successful variants of RNN is Long Short Term Memory(LSTM), which can store the historical long-range information in its memory cell. It works with the advantage of having a unique gated mechanism. In recent years, Kratzert et al 2018 used long short term memory(LSTM) to predict rainfall run-off. It was observed that LSTMs were able to capture long term dependencies in the provided data. Caihong Hu et al 2018 have observed better results using LSTM when compared to ANN in run-off forecasting, and have concluded that an LSTM network is suitable to use in hydrological research. Deepak Kumar et al 2019 compared between the sequential methods RNN and LSTM by forecasting monthly precipitation using these two methods on different homogeneous rainfall regions of India. They concluded that both methods performed well in various homogeneous rainfall regions of India.

The objective of this study is to compare different time series forecasting methods like ARIMA, different variants of ANNs, RNN, and LSTM on the monthly mean rainfall data of all the districts in the Maharashtra state. In this study, we proposed different ways of using monthly mean rainfall data as input for LSTM to predict the monthly mean precipitation in the rainy seasons, which helped the model to improve the performance. The paper is organized as follows: The details of the data and methods are described in Section 2 and Section 3 respectively; Section 4 describes the model development and performance evaluation criteria; Section 5 presents the obtained results and discussion, and Section 6

concludes the paper.

2. Study Region and Data

The Indian subcontinent has many states, out of which Maharashtra is the second most populous and third largest state extending upto 307,713 km^2 in the tropical monsoon region. Maharashtra, which is bounded by latitude 15.5 - 22 N and longitude 72 - 81 E, occupies the western and central part of India and has a long coastline of 720 kilometres along the Arabian Sea.

Being one of the agriculture based states, Maharashtra stands as one of the leading producers of the agricultural products like oilseeds, rice, bajra and many others. However, there has been an increasing demand and severe necessity to predict the future precipitation levels of this state, as there has been existing dynamic fluctuations in the rainfall measures, due to which many farmers are even committing suicides. News articles also stated that

In this study, we used the monthly mean precipitation data recorded for a period of 104 years (1901 - 2003) for 29 districts (out of 34) of Maharashtra state. The districts of the state were categorized into climatic zones(Dias et al., 2015), namely the Konkan zone, the Madhya-Maharashtra zone, the Marathwada zone and the Vidarbha zone. The last three zones are of our interest for discussion in our paper. The data was provided by the India Water Portal, that shares knowledge, and builds communities around water and related issues in India. The districts that were used for the analysis are shown on the Maharashtra map

in Fig. 1.

The number of data points available for each district is 1248 (104 years * 12 months) over 104 years. The first 99 years (99 * 12 = 1188) are used for training and the next 5 years (5 * 12 = 60) are used for evaluation of the model. A common statistical report of monthly mean precipitation data of 29 districts of Maharashtra is provided in the Table. 1, which includes minimum precipitation, maximum precipitation, mean precipitation, standard deviation. Latitude and longitude of each district are also included in the Table. 1._$ $ $ $ $ $ $ $ $ $ $

insert Link in the bottom of the page ##_$ $ $ $ $

3. Methods

3.1. Auto-Regressive Integrated Moving Average (ARIMA)

ARIMA is a novel statistical class of models to capture the structure of a time series data, analyze and forecast the future outcomings. It works best on the data which exhibits a stable or consistent pattern over time with a minimum number of outliers. ARIMA can be regarded as the generalization of ARMA. ARMA means Autoregressive Moving Average. This has been designed to deal with stationary time series data. Stationarity implies that the data has a constant mean and variance over time. ARMA works by accounting for the past values of the observation and white noise. It has been extended to deal with non-stationary time series data by introducing the notion of integration in its generalized version ARIMA.

Autoregression (AR(p)) is the method of capturing the relationship between the current observation and some of its lagged values. The value of current observation z_t is obtained by regressing over past values of some certain number of lags. This is done by expressing the current observation in terms of the linear combination of its previous lags plus some random error. The number of the previous lags to be considered is a parameter represented by p. If p is 2, then the model is of "second order" with the current observation z_t being a combination of z_{t-1}, z_{t-2} and error term a_t.

Integration is a way of making the time series a stationary one by differencing a value from its previous one. The parameter representing as of how many times this operation should take place is represented by d. If the resultant series becomes stationary upon differencing once(d=1), then the data is said to be "first differenced". If it is still not stationary, one may proceed to difference the resultant series and this can be done until the outcome is found to be stationary.

Moving Average (MA(q)) is a model which tries to relate the current observation with the previous error terms and the current error term. The parameter as of how many previous error terms should be considered is represented by q. If q is 1, it means that the current observation z_t can be expressed as the linear combination of the current error term a_t and the error term of the previous timestep a_{t-1}.

ARIMA can be regarded as a mixed model with parameter set (p,d,q). The value of current observation z(t) is obtained by regressing over past values of some certain number of lags and random errors. Mathematically, ARIMA(p, d,

q) model can be expressed as:

Write equation - Screenshot (ARIMA)

where z_t is the current observation, z_{t-1} to z_{t-p} are the actual values and a_t to a_{t-q}q) are the random errors. Phi and theta are the parameters of the model. In simple terms, AR can be regarded as ARIMA(p,0,0) and MA can be regarded as ARIMA(0,0,q).

In ARIMA modelling, we check whether the series is stationary or not. The value of d is zero if the series is stationary. Otherwise, the value of d is found by differencing the time series until it becomes stationary. The values of the parameters p and q could be found by analyzing the autocorrelation function(ACF) and the PACF plot (partial correlogram).

3.2. Artificial Neural Networks(ANN)

Conventional methods like ARIMA assume that the time series could be modelled using linear processes, which is in contrary to the real world scenario. Though ARIMA is designed to deal with non-stationary time series, it is sufficiently not able to capture the non-linearities in the real-world data. Thus there is an increasing demand to solve the real world problems by incorporating the ability to account for non-linearity, in the algorithms we use. For this purpose, ANNs have been the most flexible algorithms to use because of their efficient ability to deal with non-linear, non-stationary and complex time series data.

Artificial Neural Networks(ANN) have been inspired by the biological functioning of the human brain and regarded as one of the best universal approxima-

tors for solving various kinds of real-world problems in different fields of science and technology (Widrow et al., 1994). The commonly used ANN network is Feedforward neural network (FNN). It consists of an input layer, a hidden layer and an output layer. They work by capturing the latent features from the given input vectors and use these latent features to produce the desired output. The relation between the input and output values is represented using the below mathematical equation:

$$\widehat{y}_t \;=\; \alpha_0 \;+ \sum_{j=1}^{q} \alpha_j g \left(\beta_{0j} \;+\; \sum_{i=1}^{p} \beta_{ij} y_{t-i} \right) \;+\; \varepsilon_t$$

⇒ Write about the variables that are used in the equation...

The commonly used non-linear activation function is the logistic sigmoid function g(x) = 1/1+e^-x. Some of the other popular activation functions are hyperbolic tangent(tanh) and rectified linear unit(ReLU). The selection of an appropriate number of neurons to be used in the input and hidden layers(p, q) is an important task in ANN modelling. There is no theory that can help us to choose values of p and q. The values of p and q are dependent on the data. Hence, experiments are to be conducted using different values of p and q to find the optimal values. After the selection of those values, by defining an appropriate objective function (or cost function), the error between the target value and the output value could be minimized by calculating the gradients of the cost function with respect to each of the parameters and updating those parameters for some number of epochs. Ultimately we end up with the best parameters which help the network output values in the proximity to the target values.

3.3. Time Lagged Neural Networks(TLNN)

Time Lagged Neural Network (TLNN) is a variation of FNN. Here the input fed to the network is a set of values at some certain lags. For example, in the given figure, the input fed is the set of values at lags 1, 2, 3 and 12. Intuitively, we are considering the values at the lags t-1, t-2, t-3 and t-12 th time steps in order to predict the value at t th time step, considering the seasonality period for the time series to be 12.

A constant term that is connected to all the units of the next layer, acts as a bias term. It is generally assigned a value of 1. The relation between input and output values is represented using the below mathematical equation:

$$\widehat{y}_t \;=\; \phi_0 \;+ \left\{ w_{c0} \;+ \sum_h w_{h0}\phi_h \left(w_{ch} \;+\; \sum_i w_{ih} y_{t-j_i} \right) \right\}$$

$\Rightarrow$ Write about the variables that are used in the equation

A suitable number of previous lags to be considered for the input is selected by performing experiments.

3.4. Seasonal Artificial Neural Networks(SANN)

Seasonal Artificial Neural Network (SANN) is a variation of FNN that is developed to improve the performance for seasonal time series. SANN is capable of finding seasonal patterns in the time series without explicitly removing them from the data, unlike the conventional methods like SARIMA. In this model, the seasonality of the time series determines the value of the parameter s which helps to fix the number of input and output units. The observations of i th seasonal

time period are used as input values and the observations of i+1 th seasonal time period are used as output values. The value of s is 12 if it is a monthly time series and 4 if it is quarterly. The number of hidden units is determined by performing various experiments. The relation between input and output values is represented using the below mathematical equation:

$$\widehat{y}_{t+l} \;=\; \alpha_l \;+ \sum_{j=1}^{m} w_{jl} f\left(\theta_j \;+\; \sum_{i=0}^{s-1} v_{ij} y_{t-i}\right) \;,\; \forall t;\; l = 1, 2, 3, ..., s$$

$\Rightarrow$ Write about the variables that are used in the equation

3.5. Recurrent Neural Networks(RNN)

Recurrent Neural Network(RNN) is one of the most popular deep learning algorithms. Conventional ANNs don't consider the temporal dependencies of the input elements, whereas the time series itself implies the dependencies of the values with respect to time. Thus it is an obvious necessity to account for the temporal dependencies between the observations at the sequential timesteps. RNNs have been proved to be the state-of-the-art techniques to solve the problems in Natural Language Processing and Understanding(Mikolov et al., 2010), Speech Recognition(Graves et al., 2013), Machine Translation(Sutskever et al., 2014), etc., which essentially demand to capture the temporal dependencies existing in the sequential data such as text and signals. Since time series data can also be grouped under the class of temporal data, the usage of RNNs in solving time series related problems have been an inspiration to set new heights for the researchers in this field.

RNNs are a powerful class of models which work by the principle of weight sharing at the corresponding edges of the network. Thus an RNN can be compactly represented as both folded and unfolded versions, as shown in fig.

It efficiently deals with temporal data by considering the useful information from the previous states along with the input at the current timestep. To be mathematical, at timestep t, the hidden state ht is calculated by considering the input x_t at that time step and the hidden state information at the previous state h_{t-1}. The output y_t is computed by further processing the current hidden state information.

$$h_t = \sigma (U.x + W.h_{t-1} + b_h)$$

$$\widehat{y_t} = softmax (V.h_t + b_y)$$

Here, while and by are the bias terms, U represents the parameter matrix between the input and hidden units whereas W represents that between the adjacent hidden units. V is the weight matrix for further processing h_t and producing the output $\hat{y}$.

3.6. Long Short Term Memory(LSTM)

LSTM is the advanced version of RNN and has many advantages over RNN. LSTMs are inherently grouped under the class of RNNs and were initially proposed by Hochreiter and Schmidhuber(1997). Though RNNs are able to capture temporal dependencies, as it has fallen prey to the gradient vanishing problem due to which they have not sufficiently been successful in memorizing useful long term information. As the length of the RNN increases, the gradients are to

traverse back longer through many layers. The exponential decay of the gradients while going back in the network is known as gradient decay. This results in illustrating a small impact of the shallow layers on the output, thus negatively affecting the information retention capacity over a larger number of layers. LSTM uses different gates in its memory cell to allow the information to pass through more time steps.

An LSTM provides the flexibility to use it as a different sequential model such as one-to-one, one to many, many to one and many to many. In the time series forecasting scenario, we generally consider a many to one model, which takes many previous input values to predict the current value. When compared to RNNs, LSTMs take the additional advantage of the unique gated mechanism and memory cell. This makes the LSTMs not only very good at accounting for the temporal dependencies of the input elements but also great at selectively remembering the long distance dependencies of the current time step observation on the previous pattern. The selection of what is to be remembered and what is to be forgotten, based on the hidden information from the previous time steps and the input at the current time step is decided by the gates. The gated mechanism allows to cleverly select the useful information from the input elements and from the hidden layers.

A typical LSTM network consists of input units in the input layer, one or more memory cells in the hidden layers and finally the output layer. Each memory cell has the above-mentioned gate mechanism built in it, which makes an LSTM cell different from an RNN cell. In each memory cell, there are three gates

namely a forget gate, an update gate, and an output gate. The forget gate(f_t) decides what information could be ignored from the memory in the previous cell state(c_{t-1}), by considering the h_{t-1} and x_t. The update gate(u_t) determines what new information is to be added to the memory and carry it forward for future use. Finally, the output gate(o_t) defines which information from the cell state could be used as output.

$$
\begin{aligned}
f_t &= \sigma(W_f[h_{t-1}, x_t] + b_f) \\
u_t &= \sigma(W_i[h_{t-1}, x_t] + b_i) \; + \; \tan h(\; W_c[h_{t-1}, x_t] + b_c) \\
c_t &= f_t \; * \; c_{t-1} \; + \; u_t \\
\widehat{y_t} &= \sigma(W_o[h_{t-1}, x_t] + b_o) \\
h_t &= o_t \; * \; \tan h(c_t)
\end{aligned}
$$

In the above mathematical equations, W's and b's represent the weight matrices and bias terms at the corresponding edges on the network.

4. Model development and Evaluation criteria

4.1. Model development

In the process of developing the best models, one of the often faced challenges is the selection of optimal number of values to feed to the model. In ARIMA, that problem is solved comparatively easier with the help of ACF and PACF correlograms as already mentioned, along with the assistance of AIC measure which helped search for the best parameters in the neighborhood of the parameters yielded by the correlograms. However, the choice of the number of input neurons

and hidden units in the rest of the methods which involve neural networks is based on the trail-and-error method. There is not exactly a specific scientific or proven method to determine these two parameters in the history of the neural networks. For each of the methods, we have experimented with different permutations of the number of input and hidden units between 1 and 15, and have presented the model which showed the best performance as the best one.

Our experiments using all the above-mentioned methods seem to work better for all other months of a year, except for those months of the rainy season(June-Sept). The data of these months are dynamic and non-stable in each year. These are also extreme values when compared to the general trend of the range of the data. Thus, there has been an obvious need to deal with them in a special way in order to get better results. We have presented three approaches, where we have mainly experimented with the explanatory variables to be fed to the model.

The first approach is as explained below. For predicting the monthly mean precipitation value of a month of the rainy season, say June, we have considered the mean precipitation values of its previous months as the explanatory variables to be fed to the network. The number of previous months to be considered is based on the trail and error method, as usual. But using this approach, the results are not able to fit into the general range of precipitation values which tend to occur during the rainy seasons over the years, as exhibited in the data. Thus, this method is underestimated the precipitation values.

In the second approach, to predict the mean precipitation value of the month of June, the values of the same month in the previous years have been considered

as the input to the model. The data provided in this method has been sufficient enough to capture the range of precipitation values of the months in the rainy season and thus has shown better results when compared to the first approach.

The third method is the hybrid approach of the first two methods mentioned above. Thus, there are two simultaneous inputs to this method. The network for this method consists of two parallel LSTM networks followed by a dense layer. To predict the mean precipitation of the month of June, while the data of previous months is fed to one LSTM network, the data of the same month in the previous years is fed simultaneously to the other LSTM network. The outputs of these two LSTMs are then concatenated and used to finally feed the dense layer to predict the mean precipitation value of the month of June. This method was able to capture the the ranges of extremities and the forecast the values probable in the rainy seasons, performing better than both of the above mentioned methods .

4.2. Performance Evaluation criteria

To evaluate the performance of a model, it is suggested to use at least one absolute error measure(like RMSE, MAE) and relative error measure(like R). In this study, we have used the Root Mean Square Error(RMSE), Mean Absolute Error(MAE) and correlation coefficient(R) to evaluate the performance of the model. The mathematical formulae of the three evaluation metrics are given below:

→ Write three equations

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(E_i - O_i)}{n}}$$

$$R = \left(\frac{\sum_{i=1}^{n}(E_i - \bar{E}_i)(O_i - \bar{O}_i)}{\sqrt{\sum_{i=1}^{n}(E_i - \bar{E}_i)^2 \sum_{i=1}^{n}(O_i - \bar{O}_i)^2}} \right)$$

$$MAE = \left(\frac{100}{n} \sum_{i=1}^{n} \left| \frac{E_i - O_i}{O_i} \right| \right)$$

where O_i is the actual monthly mean rainfall, E_i is the predicted monthly mean rainfall, $O_i$bar is the actual monthly mean rainfall, $E_i$bar is the predicted monthly mean rainfall of that observation under consideration and n is the total number of time steps that are forecasted. The range of the output for each metric is different. RMSE and MAE are the absolute error measures, and whose range is [0,). The lower their values, the higher the performance of the model is. The correlation coefficient(R) is a relative error measure, and whose range is [-1, 1]. It is the measure of how similar the predicted values are to actual values. If the value of R is close to 1, then the actual and predicted values are very close to each other and vice versa.

5. Results and discussion

For any model, the selection of appropriate explanatory variables greatly shows its impression in obtaining the best results. Here the explanatory variables are the previous times. For each of the methods, the explanatory variables were varied from 1 to 25 and the ones which yielded the best results were picked as

the optimal number of explanatory variables for that model.

The obtained results vividly demontrated that the rainfall trends of the distrcitcs in a climatic region are highly correlated with its best model in terms of the evaluation metrics like RMSE, R and MAE values shown below. LSTM has outperformed all the other models in the Madhya Maharashtra and Vidarbha states in accordance with the highly unstable rainfall records over yeats in those areas. On the other hand, ARIMA has predominantly outperformed all the other models in the Marathwada region, in accordance with its comparitively more stable rainfall patters.

6. Conclusion

In this study, we presented different methods of forecasting mean precipitation values of the Maharashtra districts, India. The Vidarbha zone and the Madhya Maharashtra zones received unstable rainfall in the period of 1901-2002 where as the Marathwada state has received comparatively and acceptably stable patterns of rainfall in the same period. Thus intuitively, the precipitation process of the Vidarbha and Madhya Maharashtra regions could be fit more appropriately by non-linear methods like LSTMs or RNNs, given the fact that the data is temporal. This intuition is well well explained and supported by the observations in our study, which proved that LSTMs or RNNs have better performed in forecasting the precipitation of those regions. Similarly, the precipitation of the Marathwada region, which has exhibited nearly stable behaviour, was well

Table 1. Inspection of the RMSE metric on the models

	ARIMA	FNN	TLNN	SANN	RNN	LSTM
Ahmadnagar	46.97	47.98	48.851	58.554	43.701	**43.575**
Akola	55.675	68.188	65.208	71.445	59.735	**51.938**
Amravati	54.556	68.978	65.373	72.14	59.467	**51.038**
Aurangabad	**40.869**	50.395	51.179	56.305	41.073	44.477
Bhandara	54.675	76.394	68.005	89.188	**46.643**	49.475
Bid	**39.095**	48.39	48.697	57.443	40.619	40.135
Buldana	50.733	62.979	61.928	66.494	49.209	**47.933**
Chandrapur	49.873	68.735	63.248	81.886	**44.151**	53.655
Dhule	78.134	64.978	67.248	94.958	64.259	**62.401**
Garhchiroli	48.285	63.591	62.363	77.893	42.972	**42.511**
Gondiya	55.019	70.032	68.127	94.172	**47.924**	49.002
Hingoli	**56.431**	74.321	74.737	80.835	57.859	60.061
Jalgaon	46.47	49.405	48.797	56.254	41.446	**40.782**
Jalna	**45.198**	58.563	60.345	63.985	48.199	53.112
Kolhapur	115.524	118.83	118.065	159.452	114.616	**103.72**
Latur	45.396	54.907	54.823	63.904	**44.132**	44.646
Nagpur	52.486	72.761	67.233	81.744	**46.626**	48.971
Nanded	55.923	71.782	71.743	81.184	**54.129**	54.253
Nandurbar	106.349	**89.782**	91.756	128.469	104.313	113.412
Nashik	79.613	75.327	74.764	101.203	**65.6**	65.778
Osmanabad	38.91	43.425	44.069	54.098	**35.968**	37.57
Parbhani	**47.121**	63.427	62.745	70.913	52.895	51.781
Pune	84.78	83.081	86.27	105.669	**73.924**	75.261
Sangli	66.433	61.34	60.7	82.527	**82.527**	58.985
Satara	98.92	98.958	102.996	122.347	83.291	**82.051**
Solapur	39.367	38.975	39.776	50.496	37.302	**36.5**
Wardha	53.208	67.526	67.283	78.66	50	**48.436**
Washim	59.437	76.065	73.941	80.319	**56.724**	59.482
Yavatmal	55.536	70.582	70.168	81.3	**48.672**	54.075

Table 2. R value for each of the models

	ARIMA	**FNN**	**TLNN**	**SANN**	**RNN**	**LSTM**
Ahmadnagar	0.833	0.773	0.764	0.662	0.837	0.84
Akola	0.765	0.638	0.673	0.584	0.721	0.793
Amravati	0.799	0.667	0.726	0.612	0.761	0.823
Aurangabad	0.801	0.675	0.673	0.576	0.807	0.746
Bhandara	0.907	0.831	0.889	0.69	0.926	0.916
Bid	0.829	0.738	0.75	0.578	0.801	0.81
Buldana	0.779	0.639	0.67	0.581	0.792	0.799
Chandrapur	0.904	0.88	0.899	0.703	0.928	0.885
Dhule	0.822	0.829	0.815	0.668	0.867	0.865
Garchiroli	0.9	0.866	0.885	0.701	0.922	0.923
Gondiya	0.914	0.871	0.885	0.683	0.929	0.929
Hingoli	0.812	0.687	0.729	0.587	0.798	0.805
Jalgaon	0.791	0.723	0.727	0.64	0.837	0.825
Jalna	0.797	0.64	0.646	0.55	0.757	0.708
Kolhapur	0.892	0.87	0.864	0.716	0.878	0.901
Latur	0.819	0.769	0.775	0.602	0.827	0.824
Nagpur	0.89	0.777	0.853	0.673	0.919	0.897
Nanded	0.827	0.778	0.772	0.618	0.838	0.836
Nandurbar	0.83	0.848	0.839	0.703	0.81	0.782
Nashik	0.853	0.849	0.842	0.711	0.894	0.877
Osmanabad	0.81	0.772	0.785	0.584	0.832	0.81
Parbhani	0.828	0.705	0.726	0.584	0.774	0.783
Pune	0.848	0.824	0.815	0.69	0.88	0.861
Sangli	0.878	0.878	0.87	0.739	0.88	0.881
Satara	0.883	0.859	0.846	0.746	0.89	0.895
Solapur	0.803	0.816	0.824	0.607	0.817	0.82
Wardha	0.871	0.825	0.851	0.663	0.878	0.889
Washim	0.768	0.607	0.678	0.555	0.784	0.767
Yavtmal	0.846	0.818	0.829	0.64	0.88	0.852

Table 3. MAE score for each of the models

	ARIMA**FNN**	**TLNN**	**SANN**	**RNN**	**LSTM**	
Ahmadnagar	29.399	31.928	35.201	42.369	29.608	28.029
Akola	32.849	52.179	44.702	47.951	36.736	31.373
Amravati	32.977	53.228	48.988	53.403	37.478	32.2
Aurangabad	25.375	36.601	37.002	39.128	27.734	26.058
Bhandara	34.89	54.913	54.391	71.199	30.918	32.594
Bid	26.489	35.407	35.124	40.732	26.133	28.39
Buldana	30.218	45.58	40.799	44.441	29.382	28.236
Chandrapur	33.214	51.681	45.87	62.363	31.217	33.389
Dhule	44.295	44.491	48.478	61.923	36.753	35.194
Garchiroli	31.965	47.244	47.249	57.15	27.276	26.572
Gondiya	34.784	50.237	54.734	70.169	33.15	30.809
Hingoli	32.486	54.716	52.722	58.799	34.062	38.818
Jalgaon	28.78	36.035	35.114	42.204	26.225	24.292
Jalna	27.113	39.023	42.874	42.715	29.503	32.374
Kolhapur	61.72	71.715	73.87	117.422	80.653	59.865
Latur	30.612	41.109	43.646	47.729	28.621	31.273
Nagpur	34.132	52.75	53.712	63.201	30.317	32.336
Nanded	34.099	53.276	55.139	56.671	32.488	33.127
Nandurbar	56.722	59.88	64.892	87.128	82.084	61.485
Nashik	44.811	48.831	53.675	67.15	37.526	38.535
Osmanabad	27.197	34.88	35.747	40.111	28.307	25.358
Parbhani	29.173	45.742	44.568	49.281	31.5	32.081
Pune	50.5	56.984	62.689	78.037	43.892	44.9
Sangli	36.362	43.971	42.262	60.875	36.334	39.912
Satara	54.407	61.305	71.913	84.677	48.265	52.729
Solapur	27.447	31.412	32.59	37.878	29.175	24.614
Wardha	32.913	52.94	53.013	58.796	33.578	28.771
Washim	32.833	55.56	49.455	52.289	32.575	33.366
Yavatmal	32.408	54.942	52.706	58.937	29.415	32.212

fitted by the ARIMA method, supporting the intuition.

However, the above mentioned methods were not appropriately able to capture the dynamism in the precipitation values of the rainy seasons. Thus we have experimented different types of methods by changing the explanatory variables to LSTM network. The type of method which was fed with the precipitation values of the same month in previous years, and those values of the previous months of the same month, performed better. This method was able to capture the dynamisms and forecast the values probable in the rainy seasons.

Therefore, LSTMs could be regarded as the potential valid methods for the prediction of precipitation values for the districts of Maharashtra and were able to efficiently capture the dynamisms exhibited in the precipitation patterns of the rainy seasons. LSTMs are intelligent and reliable algorithms in the machine learning and are able to learn the patterns by themselves exhibited by the abundant amounts of data, relaxing us from the hectic task of studying and researching about the underlying processes. In continuation to current work, the scope for the future work may include the research for the best methods applicable to extreme climatic scenarios like floods and droughts.

7. References

DEEPAK KUMAR, ANSHUMAN SINGH, PIJUSH SAMUI & RISHI KUMAR JHA Forecasting monthly precipitation using sequential modelling *Hydrological Sciences Journal ISSN: 0262-6667 (Print) 2150-3435*

(Online).

RATNADIP ADHIKARI, R. K. AGRAWAL An Introductory Study on Time Series Modeling and Forecasting

ASHEL DIAS, RUTUJA DHAWDE, NUZHAT SURVE, AINA WEINBERG, TANNAZ BIRDI AND NERGES MISTRY Impact of climate changes on water availability and quality in the state of Maharashtra in Western India *Asian Jr. of Microbiol. Biotech. Env. Sc. Vol. 17, No. (4) : 2015 : 1071-1081*

V.K.SOMVANSHI, O.P.PANDEY, P.K.AGRAWAL, N.V.KALANKER 1 , M.RAVI PRAKASH AND RAMESH CHAND Modelling and prediction of rainfall using artificial neural network and ARIMA techniques *J. Ind. Geophys. Union (April 2006) Vol.10, No.2, pp.141-151*

KRATZERT, F.; KLOTZ, D.; BRENNER, C.; SCHULZ, K.; HERRNEGGER, M. Rainfall-Runoff modelling using Long-short Term-Memory (LSTM) networks. *Hydrol. Earth Syst. Sci. 2018.*

GUOQIANG ZHANG, B. EDDY PATUWO, MICHAEL Y. HU Forecasting with artificial neural networks: The state of the art *International Journal of Forecasting 14 (1998) 35–62*

G. PETER ZHANG Time series forecasting using a hybrid ARIMA and neural network model *Neurocomputing 50 (2003) 159 – 175*

MOHINI P. DARJI, VIPUL K. DABHI, HARSHADKUMAR B.PRAJAPATI Rainfall Forecasting Using Neural Network: A Survey *(ICACEA)*

QI OUYANG, WENXI LU, XIN XIN, YU ZHANG, WEIGUO CHENG, TING YU Monthly Rainfall Forecasting Using EEMD-SVR Based

on Phase-Space Reconstruction *Water Resour Manage*

MOUMITA SAHA, PABITRA MITRA AND RAVI S NANJUNDIAH Deep learning for predicting the monsoon over the homogeneous regions of India *J. Earth Syst. Sci. (2017) 126:54*

CAIHONG HU, QIANG WU, HUI LI, SHENGQI JIAN, NAN LI AND ZHENGZHENG LOU Deep Learning with a Long Short-Term Memory Networks Approach for Rainfall-Runoff Simulation *Water Article*

DUONG TRAN ANH, MINH DUC BUI, P. RUTSCHMANN Long short term memory for monthly rainfall prediction in Camau, Vietnam *ResearchGate*

HAFZULLAH AKSOY, AHMAD DAHAMSHEH Artificial neural network models for forecasting monthly precipitation in Jordan *Stoch Environ Res Risk Assess (2009) 23:917–931*

****

Figure 1 Climatic categorization of the districts in the state of Maharashtra.

Figure 2 Architecture of a Feed Forward Neural Network

Figure 3 Architecture of a Time Lagged Neural Network

Figure 4 Architecture of a Seasonal Artificial Neural Network

Figure 5 Rolled and unrolled structures of a Recurrent Neural Network

Figure 6 Figure illustrating the architecture of an LSTM cell

Figure 7 Error plots showing large variation between the actual and forecasted values in the months of rainy season

Figure 8 District wise best methods

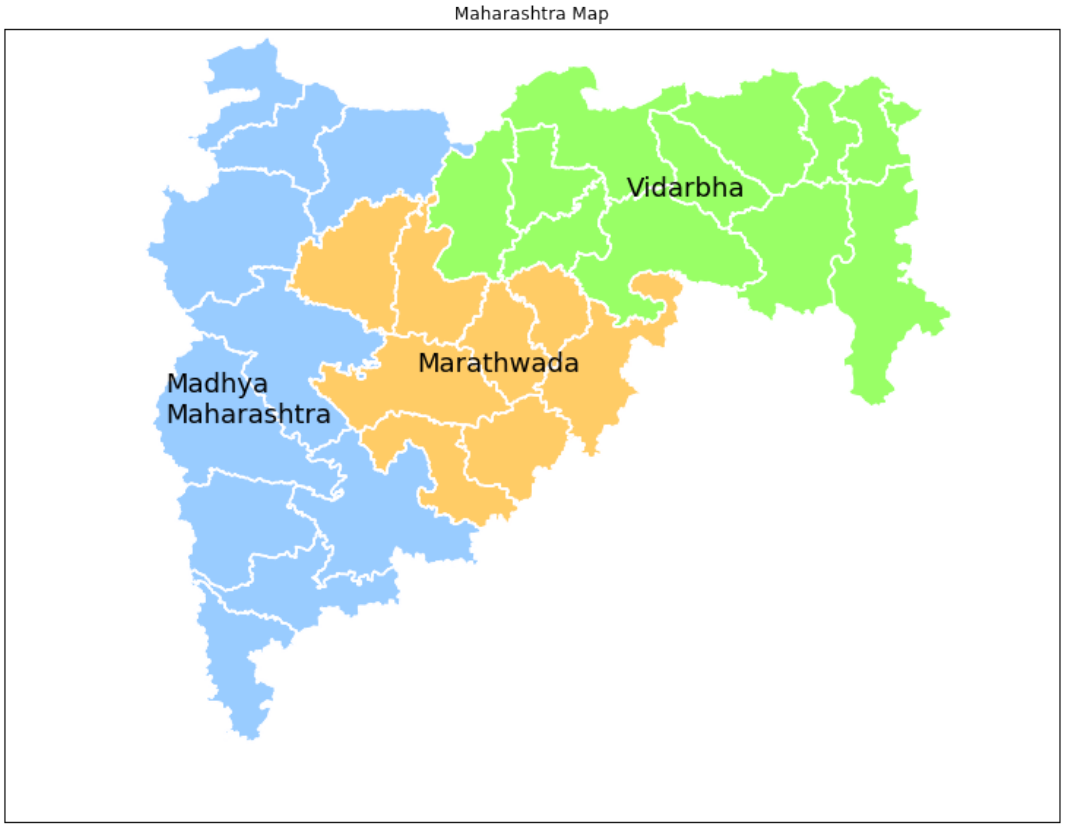

Figure 1. Climatic categorization of the districts in the state of Maharashtra.

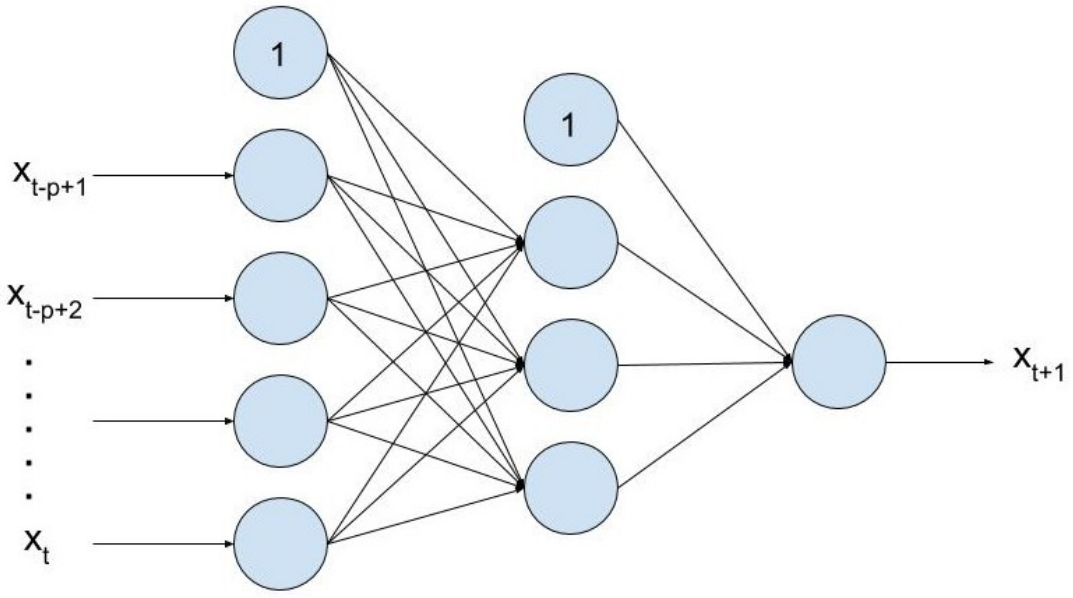

Figure 2. Architecture of a Feed Forward Neural Network

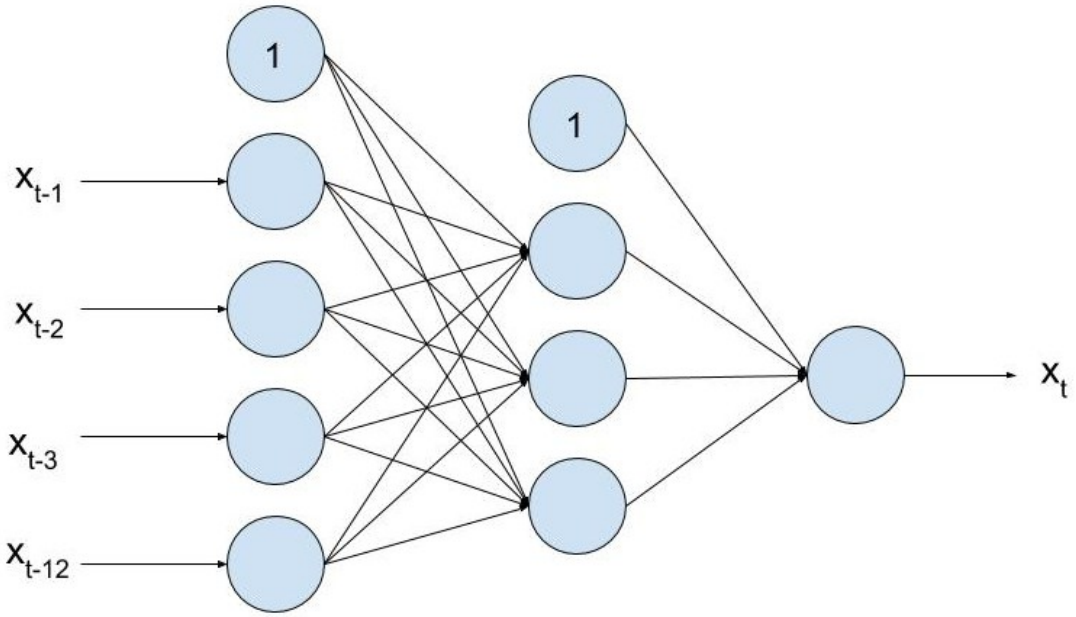

Figure 3. Architecture of a Time Lagged Neural Network

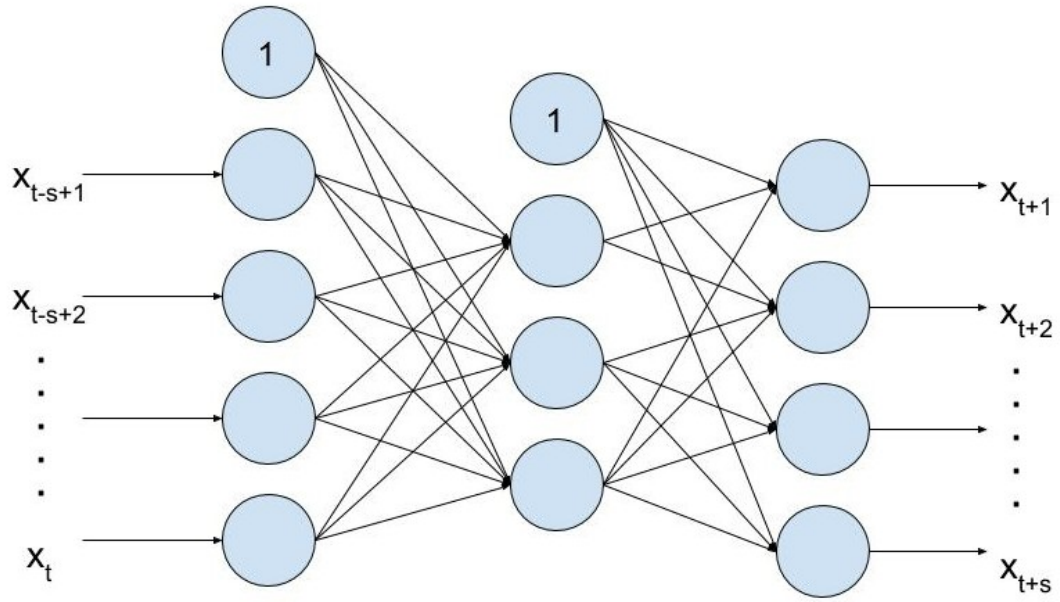

Figure 4. Architecture of a Seasonal Artificial Neural Network

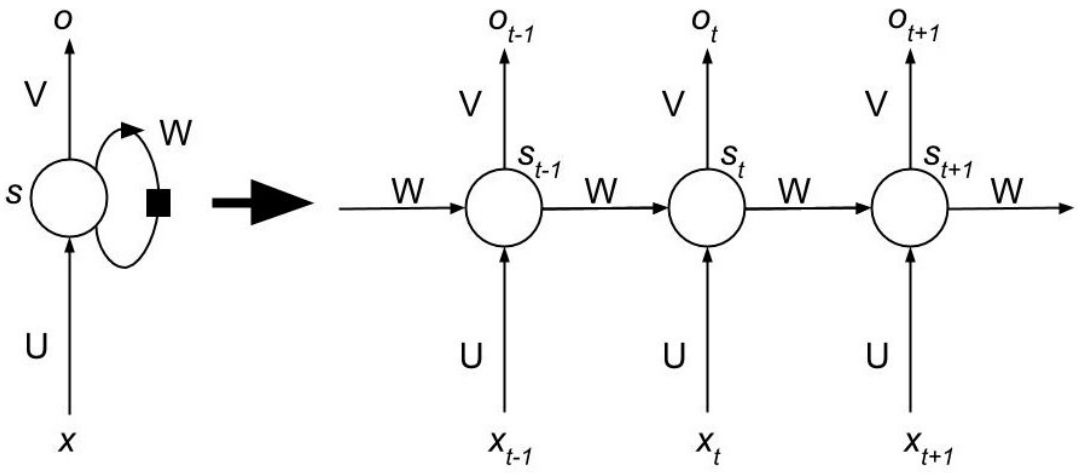

Figure 5. Rolled and unrolled structures of a Recurrent Neural Network

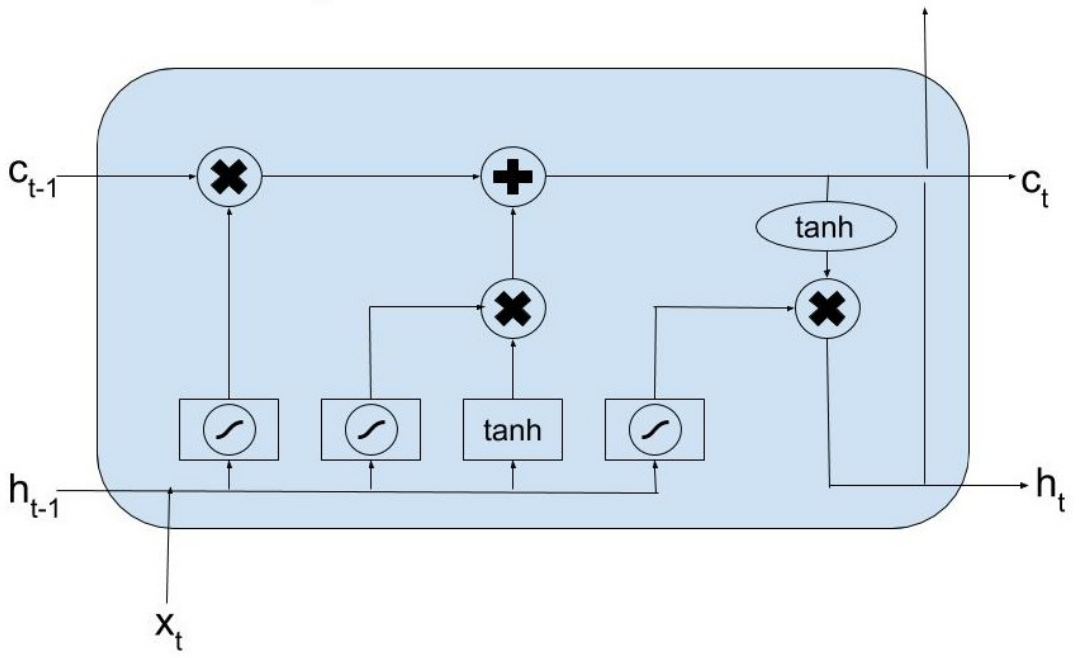

Figure 6. Figure illustrating the architecture of an LSTM cell

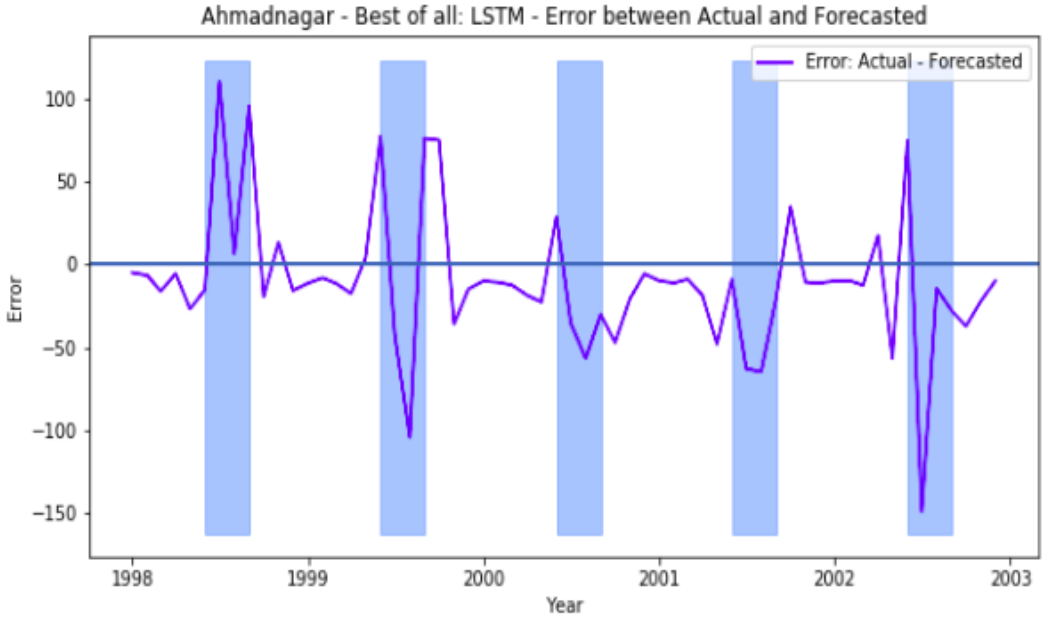

Figure 7. Error plots showing large variation between the actual and forecasted values in the months of rainy season

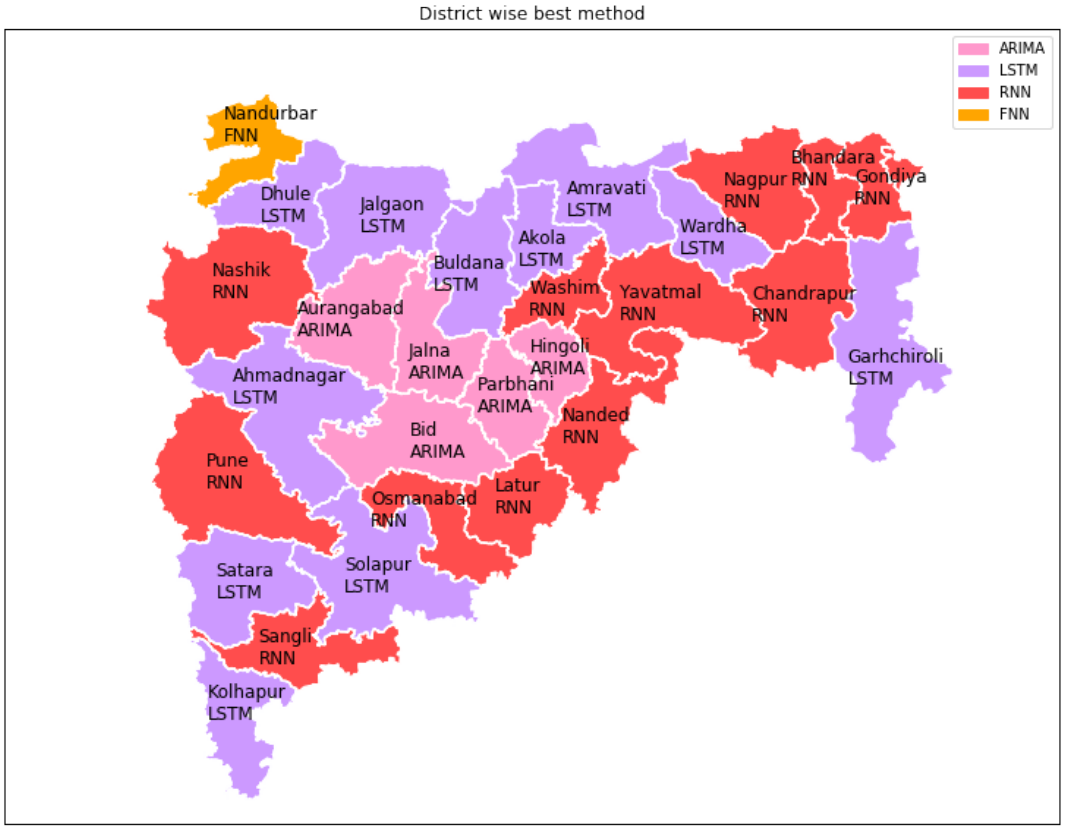

Figure 8. District wise best methods