Deep Learning based Monsoon Forecasting

A Project Report

submitted by

SAURABH KATARIYAR

in partial fulfilment of the requirements
for the award of the degree of

MASTER OF TECHNOLOGY

DEPARTMENT OF APPLIED MECHANICS
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

MAY 2019

THESIS CERTIFICATE

This is to certify that the thesis titled **Deep Learning based Monsoon Forecasting**, submitted by **Saurabh Katariyar**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. BSV Prasad Patnaik
Research Guide
Professor
Dept. of Applied Mechanics
IIT Madras, 600 036

Dr. Bipin Kumar
Research Guide
Scientist - 'E'
HPCS
IITM Pune, 411 021

Place: Pune / Chennai

Date:

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my research supervisors, Dr. Bipin Kumar and Dr. BSV Prasad Patnaik. I would like to thank you very much for your support and understanding throghout this project.

I would also like to thank Dr. Rajib Chattopadhyay (Scientist 'D', IITM,Pune) and Mr. Manmeet Singh (Scientist 'C', IITM,Pune) for their valuable inputs and involvement in every step throughout the process. Dr. Rajib Chattopadhyay helped me with most of domain insights for the work. I would also take this opportunity to thank him for the constant criticism of my work. Mr. Manmeet Singh was constant support to me throughout this work. His insights on modelling proved to be great help. He has been like a elder brother figure to me.

I would also like to show gratitude to Dr. Balaji Srinivasan. Dr. Balaji Srinivasan was my Machine Learning for Science and Engineers (ID5030) professor at IIT Madras. His teaching style and enthusiasm for the topic made a strong impression on me and I have always carried positive memories of his classes with me.

Getting through my project required more than academic support, and I have many, many people to thank for listening to and, at times, having to tolerate me over the past one year. I cannot begin to express my gratitude and appreciation for their friendship.

Most importantly, none of this could have happened without my family. This thesis stands as a testament to your unconditional love and encouragement.

ABSTRACT

Monsoon prediction for a tropical countries become very important. Agrarian society like India suffers a lot due to bad quality monsoon predictions. Lately, extreme rainfall events have been observed more frequently than ever before, causing large scale destruction of lives and property. Monsoon is highly non-linear dynamical system. Hence, Indian summer monsoon rainfall (ISMR) prediction at small regional scale is a very challenging task. Conventionally, monsoon is predicted as a derived results from physics based dynamical models. We propose a new deep learning based approach to solve the problem at much smaller spatial scale ($1^\circ \times 1^\circ$) than most existing statistical systems. We have used convLSTM based architecture which captures spatio-temporal pattern in ISMR. The algorithm is developed using single variable (i.e. rainfall) to forecast upto five days lead time in future. It is seen that most part of India gives high correlation for three day lead time.This work is very first attempt to capture such spatio-temporal pattern in monsoon using statistical techniques.

KEYWORDS: Deep Learning, convLSTM

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

IITM	Indian Institute of Tropical Meteorology, Pune
CCCma	Canadian Centre for Climate Modelling and Analysis
CGCM	Coupled Global Climate Models
SVM	Support Vector Machine
ANN	Artificial Neural Network
ISMR	Indian Summer Monsoon Rainfall
NCEP	National Center for Environmental Prediction
SST	Sea Surface Temperature
SLP	Sea Level Pressure
AT	Air Temperature
IMD	Indian Meteorological Department
LSTM	Long Short Term Memory algorithm
CNN	convolutional Neural Network
RNN	Recurrent Neural Network
convLSTM	Convolutional LSTM model
ROVER	Real-time Optical flow by Variational methods for Echoes of Radar
MSE	Mean Squared Error
MAE	Mean Absolute Error
NLP	Natural Language Processing
FC-LSTM	Fully Connected Long Short Term Memory

NOTATION

σ	sigmoid function, $\sigma(z) = \frac{1}{1+(e)^{-z}}$
$\circ$	Hadamard Product
$*$	Convolution Operator

CHAPTER 1

INTRODUCTION

Atmosphere and Earth systems, being one of the most complex system in nature, is one of the hardest subject to be studied in the world for many centuries. Nature is perhaps the most dynamic system to analyze. We have a long history of scientists and physicists trying to study the physics related to these systems. We have made tremendous achievements in the field and still there is a lot of scope for improvements. The dynamical system modeling of nature has been developing steadily over the years and constant stride is being made to improve the results. We see constant improvement in parametrization of climatic variables for better result in the dynamic models. The current state of the art dynamic models give accurate and precise results for variables like air temperature, wind speed etc. These models have shown great capacity in predicting hazardous conditions like cyclone and typhoons. Even after large scale development, the problem of weather forecasting is still a great challenge. With extreme events becoming more frequent due to the advent of global warming and pollution, the earth and it's people are possibly under serious threat more than ever before. Frequent outpours and heat waves are common observation now a days. So, precise and accurate weather forecasting system is the need of hour.

Weather forecasting has been a very complex and long studied problem in the world. Forecasting is a method of knowing the future before hand. Nature, being a highly non-linear system, is hard to forecast. However, we have been successful in forecasting several climatic variables with very good precision. Air Temperature and Wind Speed have a good accuracy of forecast from the dynamic models. Precipitation is one of the derived variable out of dynamical model. On the one hand when we are able to forecast many climatic variables with great accuracy, derived variable such as precipitation forecasting still exists as an open challenge to the community.

Among various regions of the globe, tropical region is perhaps the most dynamic system. Tropical systems depend on various fluctuating climatic parameters such as

El-Nino and La-Nina. Forecast of any climatic variable in this region is a much harder problem. In this work, we have tried to forecast monsoon over the Indian region. Precipitation forecast for the Indian region can have a very positive impact on our country and it's economy.

The Indian economy is mostly driven by agriculture. Indian summer monsoon rainfall (ISMR) is very important phenomena which directly impacts the Indian agricultural production. Agriculture being an important sector in India, ISMR drives GDP of the country, directly or indirectly. In a country like India where farmers still depend on monsoon for agriculture, a failure in monsoon prediction can have hazardous effect on the economy, such as loan deficit and unbalanced trade. According to Deshpande and Prabhu (2005), over 48 per cent of the farmers are indebted and nearly two-thirds of the farmers are frustrated with their profession. According to Mahapatra (2017), the Government of India reported over 1200 farmer suicide per year since 2013. Changing climatic conditions such as out pour of monsoon rainfall led to hazardous condition in Western ghats of India, leading to loss of life and property worth millions. The recent events of Kerala floods in 2018 is one such unfortunate event.Mishra *et al.* (2018) discuss the extreme rainfall event in Kerala, which was about 53% more than normal. The event had long return periods in the range of 100 years.

Although dynamic modeling techniques have been used to predict variables, we are seeing a gradual shift in paradigm for the last two-three decades. Several work has been done using statistics based approach to solve prevalent problems. In 1.1, we will see some of the works done till date. Most of the work solves problem of monsoon forecasting in India but spatial pattern of the monsoon is not explained. These works have forecasted averaged value over India or regions. We have shown from our work that spatial pattern in ISMR can be used to forecast rainfall at much higher resolution using single model. We have used convLSTM based deep learning approach for our work. We intend to forecast daily rainfall for up to 5 days lead time. Our study is uni-variate, rainfall being the single variable used.

1.1 Literature Review

With such extreme weather conditions becoming more frequent recently, accurate prediction of precipitation is a need of hour. Although the dynamic models such as the recent third generation Coupled Global Climate Models (CGCM), such as Canadian Center for Climate Modeling and Analysis (CCCma) have shown tremendous improvements in prediction, their forecast for extreme rainfall events is not up to the mark. Unlike dynamic models which explains the physics and dynamics of any process, statistical modeling is a data driven approach which recognizes the pattern in historical data. There has been numerous studies based on statistical modeling of climatic system which attempts to find relation between different climate variables and rainfall. Regression based models were used by Thapliyal and Kulshrestha (1992), Gowariker *et al.* (1991) to forecast ISMR. Recent advancement in computational infrastructure and abundance availability of data has led to a revolution. Now a days, data driven statistical models have proved to be good alternative approach to mathematical models for some specific case study.

Tripathi *et al.* (2006) studied the use of machine learning algorithm for rainfall prediction using down scaling property of support vector machine (SVM) algorithm. The paper shows that SVM performs better than artificial neural networks (ANN) in down scaling models.The author developed different models for each sub divisons. The suggested model predicts average rainfall for various sub divisons in India as shown in figure1.1 using grid point climate data at monthly time scale. The models are trained using monthly reanalysis data maintained by National Center for Environmental Prediction (NCEP).

Saha *et al.* (2016*a*) developed ANN based auto-encoder to generate non-linear feature vector and predicted ISMR using the generated features. Auto-encoders are unsupervised machine learning algorithm. The paper discusses a non-linear dimensionality reduction technique, sparse auto-encoder to reduce global climatic variables such as sea surface temperature (SST), air temperature (AT) and sea level pressure (SLP). The sparse auto-encoder can be seen in figure 1.2. Reduced feature space is used to predict ISMR using ensemble models viz., regression trees and bagged decision trees. The

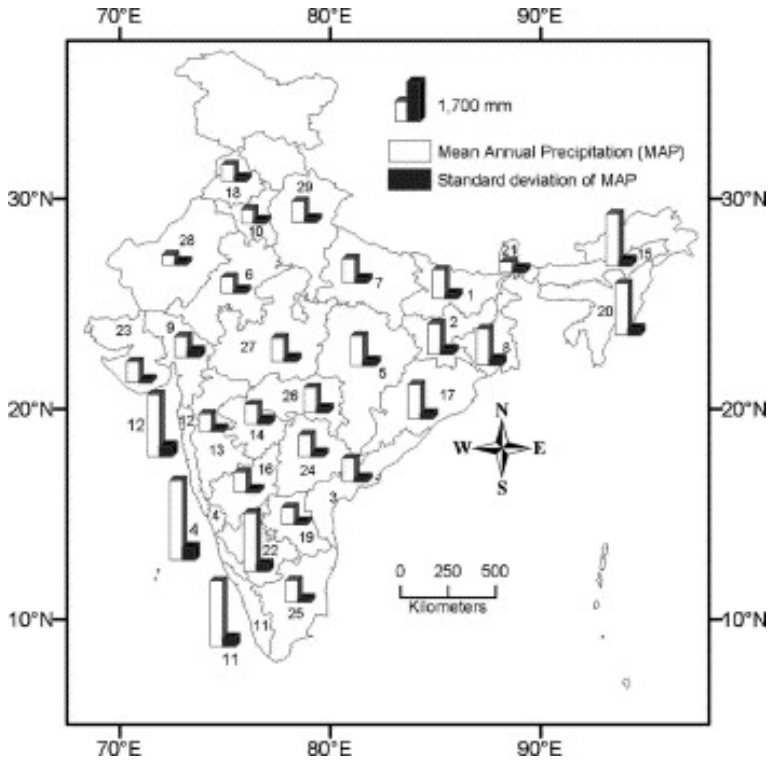


Figure 1.1: Meteorological sub divison of India - Study area of Tripathi *et al.* (2006)

complete methodology is given in the flowchart as shown in fig1.3. The auto-encoders develop some complex features which have a better correlation with ISMR. Ensemble based models trained on the generated features predicts rainfall better than state of the art dynamic models. The developed model shows mean absolute error (MAE) of 4.5% in predicting ISMR.

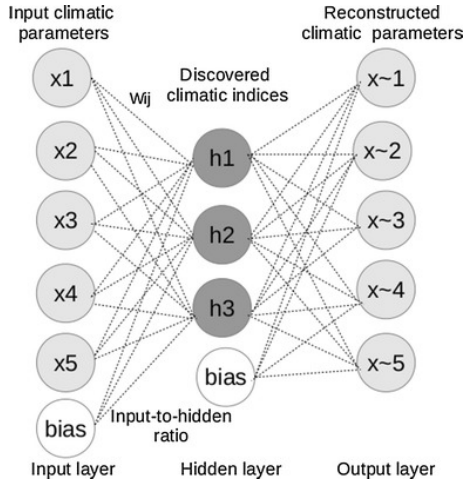


Figure 1.2: Architecture of auto-encoder. Saha *et al.* (2016*a*)

Saha *et al.* (2017) discussed stacked auto-encoder for non-linear dimensionality reduction to generate complex feature for regional ISMR prediction. The schematic rep-

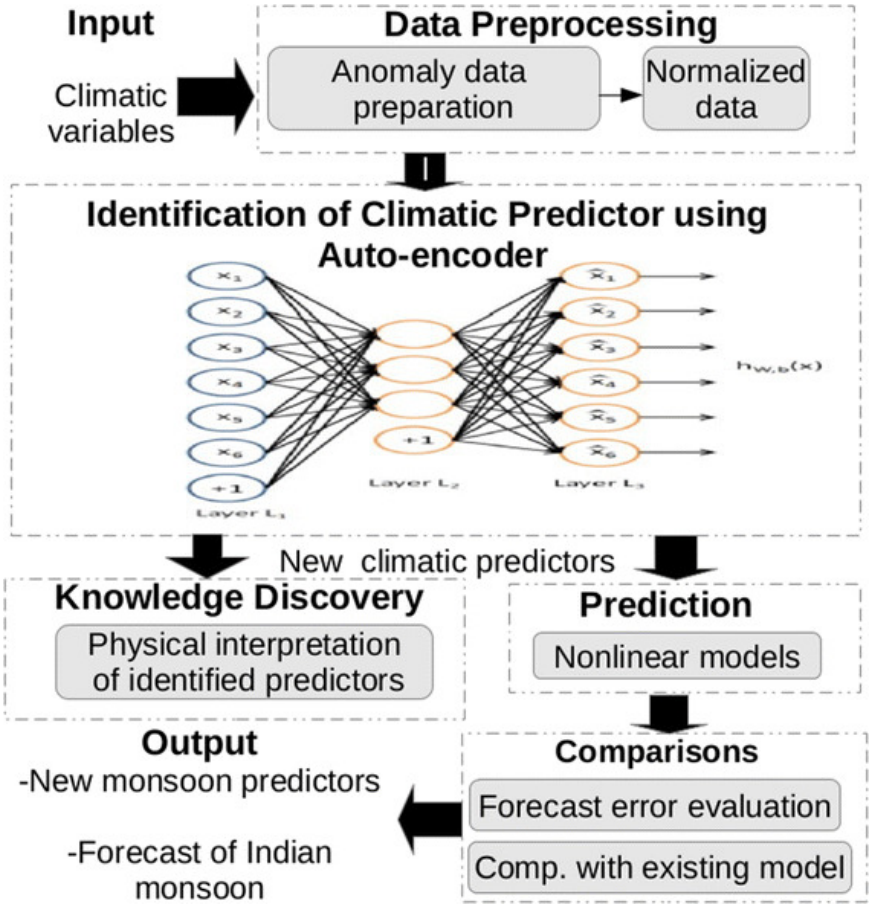


Figure 1.3: Methodology. Saha *et al.* (2016*a*)

resentation of stacked auto-encoder is shown in figure1.4. This is a development over sparse auto-encoder and has several layer deep neural network. The developed encoders were successful in better dimensional reduction with lower loss in variance. Again the generated features were used to predict monsoon in different geographies, namely central, north-east, north-west and south-peninsular India. Four different ensemble models are developed for each regions. The models were trained using different set of generated features for each regions. The models predict monsoon with errors of 4.1%, 5.1%, 5.5% and 6.4% in central, north-east, north-west and south-peninsular regions respectively. The stacked auto-encoder has two parts in it's architecture viz., encoder and decoder. The unsupervised learning for the auto-encoder is done by minimizing reconstruction loss. Reconstruction loss is calculated when inputs and outputs are same. The encoder generates complex features reducing dimension of the input space and the decoder reconstructs the output (same as input) from encoded feature space.

Early and late monsoon for Indian subcontinent is an important phenomena. Predicting early and late monsoon becomes non-trivial as the responsible climatic variables

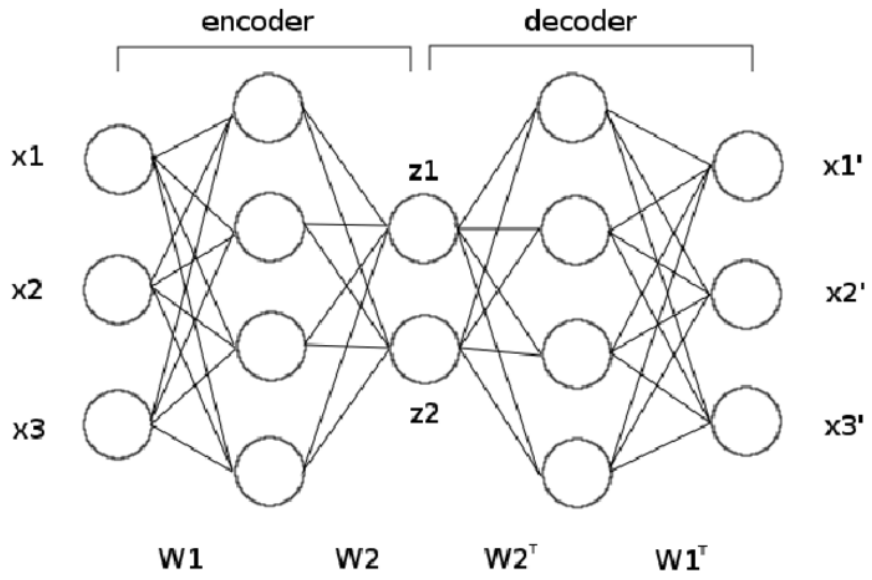


Figure 1.4: Stacked Autoencoder architecture

are very different for these events. Saha *et al.* (2016*b*) developed a model using stacked auto-encoder to predict the predictors for both early and late monsoon events for India. The early and late monsoon events are predicted by training ensemble models on the generated predictors.

All the work discussed above predicts average rainfall over large geographies. Such problems are time series problem without taking spatial pattern into account.

In recent years, deep learning started to make it's mark. Due to advent of the more complex algorithms such as CNN and RNN, scientists and academicians all over the world have started using more complex networks for solving the problems. Shi *et al.* (2015) used one such architecture for precipitation nowcasting. The paper discussed Convolutional LSTM algorithm which was able to forecast using spatio-temporal data. convLSTM incorporates the concept of two deep learning architecture, viz. CNN and RNN. The model was used to predict rainfall intensity over a range of 0-6 hours using real time radar data over Hong Kong at time scale of 6-10 minutes. The model was shown to perform better than state of the art Real-time Optical flow by Variational methods for Echoes of Radar (ROVER) algorithm. The model reports rainfall-MSE of 1.420.

The purpose of our study is to develop a model which accurately predicts precipitation for the whole Indian region. We have used convLSTM architecture for our work. For training our model, we have used daily IMD gridded dataset. We have trained our model on this data from 1964-2004. For testing our model, we have used the same data from 2005-2012.

The thesis is organized as follows. Chapter 2 discusses basic overview of proposed model. Chapter **??** briefs about the network architecture of the model. Chapter 4 describes the overall method adopted through the project. Chapter 5 includes the set of experiments performed while training. Chapter 6 discusses the result obtained. Finally we conclude in chapter 7 along with the future work.

CHAPTER 2

Overview of convLSTM

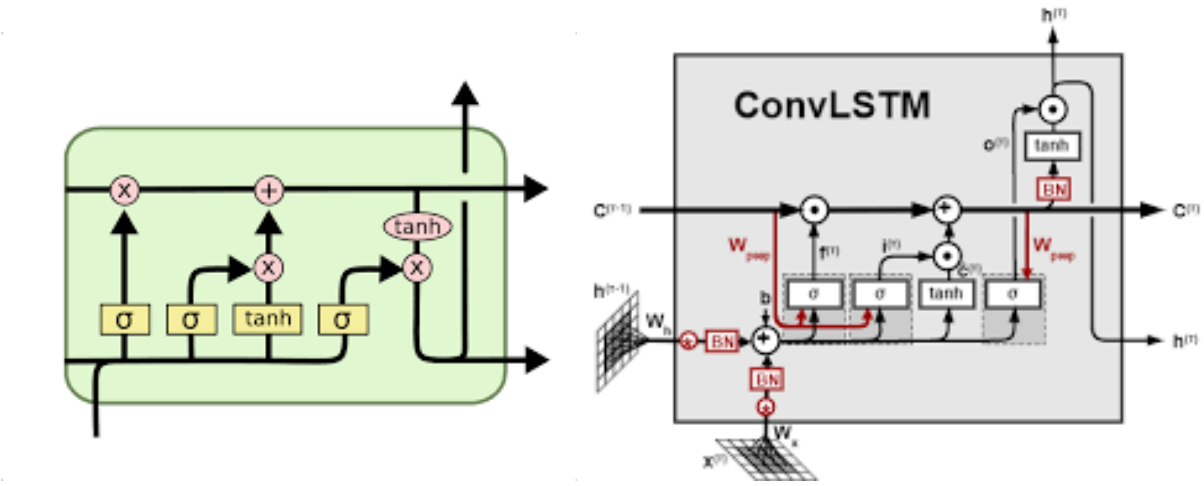
In the era of deep learning and abundant availability of data, we have started to make data driven decisions. Deep learning algorithms can capture non-linearity of complex systems to derive some meaningful results. Out of many such problems, forecasting a sequential data is of great importance. Many attempts have been made to solve such problems using traditional machine learning and time series methods with great success.

Special class of network called Recurrent Neural Networks (RNN) became popular for sequential problem solving. With time RNN module saw many developments from vanilla RNN, Gated Recurrent Unit (GRU) to the more complex long short term memory (LSTM). LSTM was developed by Hochreiter and Schmidhuber (1997) in the year 1997. But only recently, we have had appropriate computational infrastructure to derive full potential out of this algorithm. This class of algorithm are basically applied to make decision out of a sequential data. Suddenly LSTM became a go-to algorithm for any sequential data, such as natural language processing (NLP) and time series analysis. Basically, LSTM is a module, which when stacked together forms a network.

Simple RNN modules fail to capture long term dependencies due to vanishing gradient problem. LSTM are robust to vanishing gradient due to dedicated cell state C_t, which has very minor linear interaction with the module aiding in unchanged information flow Olah (2015). The module, as represented in figure 2.1a has three gates, input, output and forget gates. These gates decide information flow out and in of these repeating modules in the network.

There are many versions to this architecture. One such architecture was developed by Gers and Schmidhuber (2000). In this variant, peephole connections are added so that the gates of the LSTM module can see the cell state. The important mathematical

Fully Connected LSTM v/s Convolutional LSTM



(a) FC-LSTM Olah (2015) (b) convLSTM

Figure 2.1: Long Short Term Memory

equations are shown by equations 2.1 where $\circ$ refers to Hadamard product.

$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci} \circ c_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf} \circ c_{t-1} + b_f) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co} \circ c_t + b_o) \\
h_t &= o_t \circ \tanh(c_t)
\end{aligned}
\tag{2.1}
$$

Simple LSTM are also known as fully connected Long Short term Memory (FC-LSTM). One of the major drawbacks of the FC-LSTM is the handling of spatio-temporal data. To overcome this problem, Shi *et al.* (2015) developed Convolutional Long Short Term Memory (convLSTM) model which uses convolutional operations in the LSTM network. The convLSTM is shown in figure 2.1b. The difference between FC-LSTM and convLSTM can be easily seen in figure2.1. In convLSTM inputs, cell outputs, hidden states and gates i_t, o_t, f_t are 3D tensors.The convolution operation is performed on these tensors. The important equations in convLSTM are shown in equation 2.2, where $*$ stands for convolution operation and $\circ$ stands for Hadamard product.

$$
\begin{aligned}
i_t &= \sigma(W_{xi} * \mathcal{X}_t + W_{hi} * \mathcal{H}_{t-1} + W_{ci} \circ \mathcal{C}_{t-1} + b_i) \\
f_t &= \sigma(W_{xf} * \mathcal{X}_t + W_{hf} * \mathcal{H}_{t-1} + W_{cf} \circ \mathcal{C}_{t-1} + b_f) \\
\mathcal{C}_t &= f_t \circ \mathcal{C}_{t-1} + i_t \circ \tanh(W_{xc} * \mathcal{X}_t + W_{hc} * \mathcal{H}_{t-1} + b_c) \\
o_t &= \sigma(W_{xo} * \mathcal{X}_t + W_{ho} * \mathcal{H}_{t-1} + W_{co} \circ c_t + b_o) \\
\mathcal{H}_t &= o_t \circ \tanh(\mathcal{C}_t)
\end{aligned}
\tag{2.2}
$$

In the above equation, $\mathcal{X}_1, ..., \mathcal{X}_t$ denotes cell input at any time step. Similarly, $\mathcal{C}_1, ..., \mathcal{C}_t$ denotes cell outputs and $\mathcal{H}_1, ..., \mathcal{H}_t$ denotes the hidden states. All the cell value are 3D tensors.

CHAPTER 3

Model Architecture

In the chapter 2, we discussed about convLSTM module. When such modules are stacked together, a complex architecture is formed. One such architecture is shown in figure 3.1. The figure shows the final architecture we have used for our study. We performed pre-defined sets of experiment as will be discussed in chapter 5 to come up with this architecture.

The overall architecture is a 6 layer deep convLSTM network, with each layer being $n + 5$ cells wide, n being number of input days (or cells). Since we intend to forecast daily rainfall for 5 days lead time, we have $n + 5$ days wide network. $n + 5$ convLSTM modules are stacked vertically to form one layer. Such layers are horizontally stacked to get a 6 layer deep network. Each layer contains 80 filters each. This architecture works by encoding spatio-temporal pattern and then forecasting future patterns. The information from past n days are encoded and this information is used to forecast the coming 5 days. The discussion in chapter 5 includes detail procedures and experiments for this architecture.

The table 3.1 shows shape and bulkiness of the model. It can be seen that the model is trying to learn about 0.6 million parameters in the final architecture.

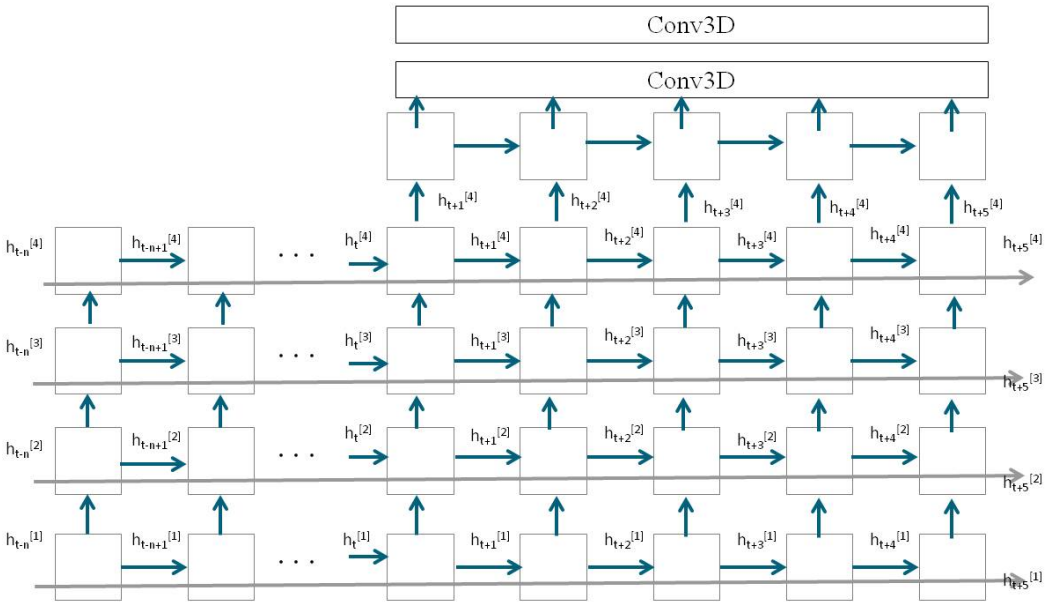


Figure 3.1: Model Architecture

Table 3.1: Model Summary

Layers	Output Shape	Number of Parameters
convLSTM	(None, None, 357, 1, 80)	78080
Batch Normalization	(None, None, 357, 1, 80)	320
convLSTM	(None, None, 357, 1, 80)	153920
Batch Normalization	(None, None, 357, 1, 80)	320
convLSTM	(None, None, 357, 1, 80)	153920
Batch Normalization	(None, None, 357, 1, 80)	320
convLSTM	(None, None, 357, 1, 80)	153920
Batch Normalization	(None, None, 357, 1, 80)	320
lambda	(None, None, 357, 1, 80)	0
conv3D	(None, None, 357, 1, 80)	57680
conv3D	(None, None, 357, 1, 1)	9681
Total Parameters : 608,481		
Trainable Parameters : 607,841		
Non-trainable Parameters : 640		

CHAPTER 4

METHODOLOGY

In statistics, weather prediction including precipitation is basically a time series problem. Chapter 1 discussed the work attempted till now. Most statistical model developed for ISMR are based on machine learning algorithm. These models are developed for average rainfall over a geography Saha *et al.* (2016*b*) Saha *et al.* (2016*a*) Saha *et al.* (2017). We propose a model to predict precipitation for entire Indian sub-continent. Our model captures the spatio-temporal pattern in historical data to forecast future events. Our study region as shown in figure 4.1 extends from 6.5° to 38.5° latitudes and 66.5° to 100.5° longitudes.

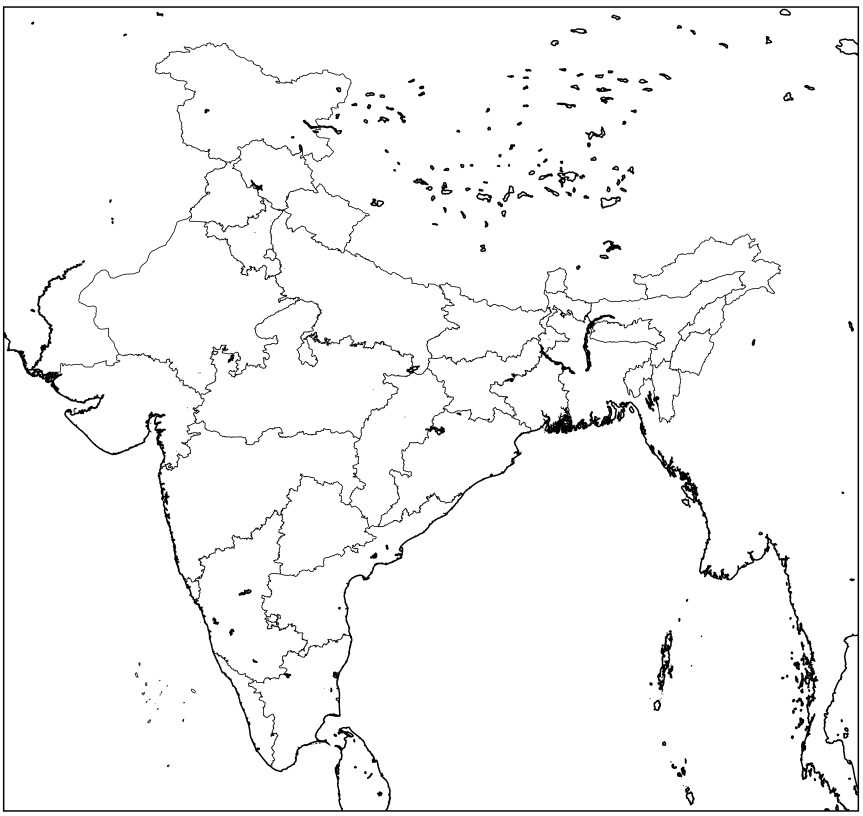


Figure 4.1: Region under study

Unlike most of the time series problem, we are solving spatio-temporal problem with one model. Our model uses convolutional LSTM network architecture (discussed

in chapter2) to predict precipitation. The convLSTM uses the concept of CNN and RNN architecture of deep learning. Our problem can be defined as the equation 4.1.

$$\widetilde{\mathcal{X}}_{t+1}, .., \widetilde{\mathcal{X}}_{t+k} = \underset{\mathcal{X}_{t+1}, ..\mathcal{X}_{t+k}}{arg\ max}\ p\left(\mathcal{X}_{t+1}, ..\mathcal{X}_{t+k} \mid \widetilde{\mathcal{X}}_{t-J+1}, \widetilde{\mathcal{X}}_{t-J+2}, .., \widetilde{\mathcal{X}}_{t}\right) \tag{4.1}$$

The figure 4.2 shows flowchart of the study. The properties of raw data is briefly explained in 4.1.1. Detail discussion of the data handling and preparation is explained in 4.1.2. Finally, the preparation of model-ready data is explained in 4.1.3

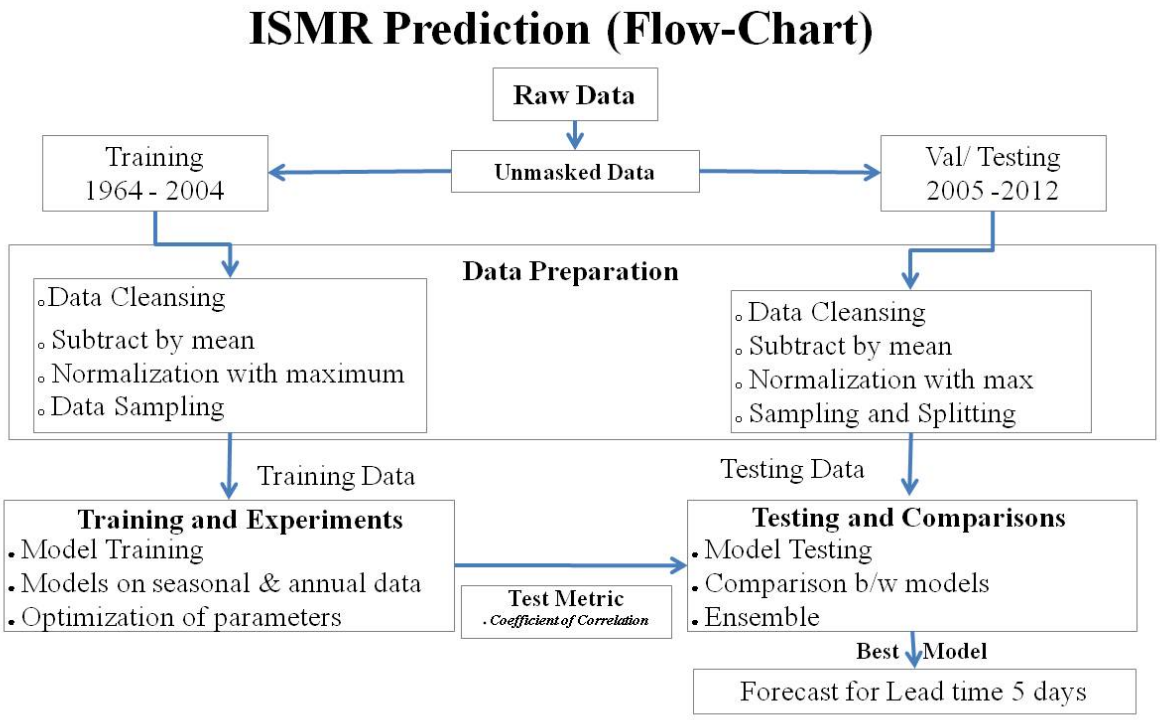


Figure 4.2: Overall flowchart

4.1 Data Preparation

The study involves training a deep learning model to forecast ISMR. convLSTM used for this study comes under supervised class of algorithm. We need to prepare our data before feeding it to the network algorithm for training. The preparation involves cleansing of data, sampling and conversion to supervised form. The following sections briefly

explains the steps undertaken.

4.1.1 Raw IMD Data

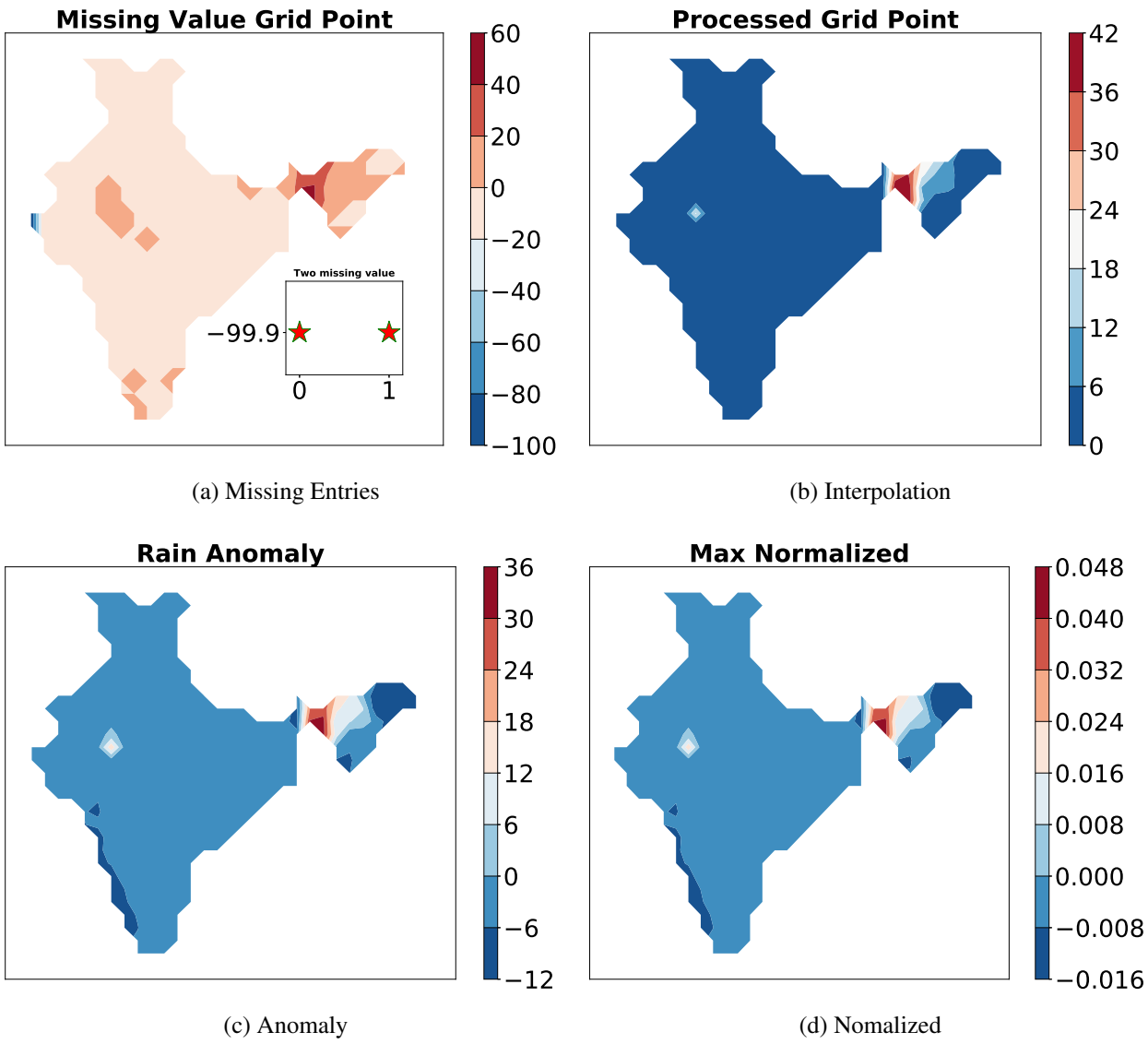
We have used gridded IMD rainfall data from 1964-2012 for this study.Rajeevan *et al.* (2006) The data was taken on a daily basis. Since the data was actual observation, many points in the study area had no observed value. The area corresponding to sea and landmass outside India had no value. The used data was masked with the fill value of -9.99×10^{-8} at every point where the observation was lacking. The data was three dimensional array, with dimensions corresponding to time, latitude and longitude respectively. The format of data was netCDF4.

4.1.2 Data Pre-processing

Data pre-processing is done to ensure that the data is properly clean before sampling. As discussed in section 4.1.1, the data is masked in the region where there is no observation. We observe that at multiple time steps there are values of absolute rainfall less than zero, which is not possible. Analyzing further, we see that at all erroneous time steps (272 time steps) there are two such values, making total 544 missing value. Also we see that all value are -99.9 as can be seen in figure 4.3a. From the figure 4.3a it can be seen that there are two values in Gujarat which are missing. The source of such error can be failure of observation mechanism.

To treat such missing entries, we impute these points with mean of nearest four neighbor. We have done spatial interpolation to impute the missing value. Figure 4.3b shows the treated values. It can be seen that the missing entries have successfully been imputed.

Now that the data is cleaned, we subtract temporal mean of every grid point to convert absolute rainfall into anomalies. Figure 4.3c demonstrates the anomalies of rainfall. We ensure that the data is properly scaled by applying max normalization to the anomalous data. Final processed data can be seen in figure 4.3d.



(a) Missing Entries

(b) Interpolation

(c) Anomaly

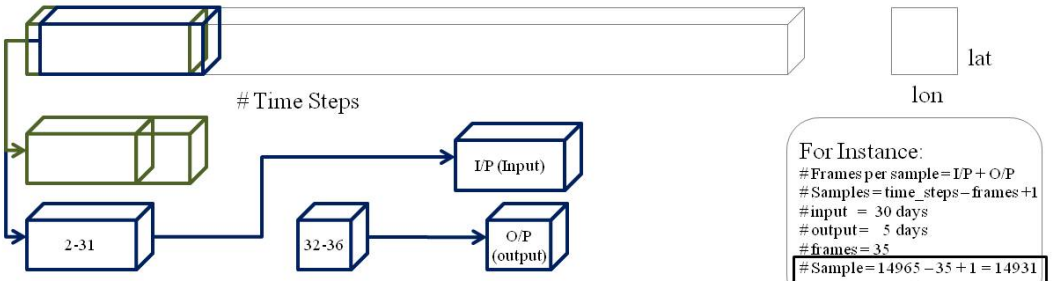
(d) Nomalized

Figure 4.3: Data Pre-processing

4.1.3 Data Sampling

The processed data has to be converted to supervised learning problem before feeding to the network. The network expects data in 5-dimensions (number of samples, number of time steps, latitude, longitude, number of variables). Further we have to sample in such a way that there is an input and a output data set for supervised learning. Fifth dimension is 1 as we are using only one variable i.e. rainfall.

Sampling is done with repetition. We sample by sliding a window of size (n+5,latitude, longitude). This window is then split into two sets. Number of sample being the win-



Model	Purpose of Sampling	Data Used		Number of Samples
		Years	Months	
Seasonal Model	Training	1964-2004	JJAS	3608
	Testing	2005-2012	JJAS	704
Annual Model	Training	1964-2004	All months	14931
	Testing	2005-2012	JJAS	704

Figure 4.4: Sampling Methodology

dow operation from which the data is generated.

- Input data set : (sample, n, latitude, longitude, 1)
- Output data set : (sample, 5, latitude, longitude, 1)

The number of samples generated can be calculated as

$$S = N - w + 1$$

where S is number of samples, N is total time frames and w is the window time frames.

One such detail calculation is demonstrated by figure 4.4Ẇe train two models using different data sets.

- Seasonal Model : Model developed using monsoonal months (JJAS) [1] only
- Annual Model : Model developed using all months

For testing purpose we use same data set for comparison.

[1] Months of June, July, August and September

As discussed in section 4.1.1, almost 70% of the data per time step is masked. Convolution over mask fill value will not give us expected result as it does not represent reality. We approach training by extracting meaningful data from each time step and converting it to one dimensional vector. Initially the data shape per time was $R^{lat \times lon}$. We convert this shape into

$$R^{33 \times 35} \Longrightarrow R^{357 \times 1}$$

The final data shape of our model is

$$R^{samples \times timeframes \times 33 \times 35 \times 1} \Longrightarrow R^{samples \times timeframes \times 357 \times 1 \times 1}$$

.

4.2 Convolution Operation

As discussed in chapter 3, we have used 2 dimensional convLSTM modules for development of algorithm. This module uses 2 dimensional filters for convolution operation. It is discussed above in section 4.1.3 that we have converted our data into single vector of shape $R^{357 \times 1}$ at each time step. We cannot use normal two dimension filter in our case. We have used a one dimensional filter over our reshaped data. Similarly, the fifth and sixth layer are three dimensional convolution layer. It expects 3-D filter but we have forced the second dimension to be of size 1.

CHAPTER 5

Experiments and Training

The model architecture was discussed in chapter 3. Every deep learning model has to be tuned for optimal performance. The process is called hyper-parameter optimization, in which we perform different sets of experiments to get minimum loss. There are numerous parameters that can be tuned. Number of layers, number of cell in each layers, activation function, optimization algorithm, learning rate, momentum rate, number of filters, number of epochs, batch size etc. are some of the many parameters that can be tuned to get best performance of any model. We have used grid search cross validation technique to design experiment. We form grids of potential values of parameters and search for the best set. Cross validation is performed to measure the performance. We have used gridsearchCV function of scikit-learn package to perform experimentation.Pedregosa *et al.* (2011)

Our model has a lot of parameters that can be trained but we have conducted only three experiments to boost our model performance.

- **Number of Input Days:** In this experiment, we search the best value for number of input days for forecasting 5 days lead monsoon. The grid is one dimensional. The metric used to measure performance is average coefficient of correlation and root mean square error (RMSE). The metrics have to be single scalar, so we average both in spatial and temporal domain.
- **Optimization Algorithm:** In this experiment, we search the best class of optimization algorithm for optimal performance. Optimization algorithms reduces the loss function. We select 8 different gradient based algorithms and perform one dimensional search to find the best algorithm.
- **Learning Rate and Momentum Rate:** Learning rate and Momentum rate are hyper-parameters of the optimization algorithm, which decide how fast the model learns and ensures the proper minimization of objective or loss function. We perform 2-dimensional grid search to find out the best combination.

For training, we have used keras package in python with tensorflow running in the backend. Abadi *et al.* (2015) Chollet *et al.* (2015) After searching for the optimal values of the parameters, we train the model using early stopping and model checkpoint functions in keras. These functions help us to save best model.

We have developed three different models as discussed below.

5.1 Seasonal Model

As discussed in section 4.1.3, seasonal model is developed by using monsoonal month data (JJAS). To train this model we perform the set of experiments as mentioned above to seasonal data and further train on the optimal parameters with early stopping and model checkpoint.

Table 5.1 shows the one dimensional grid for number of input days. It can be clearly seen that 30 days performed best on both metric.

Table 5.1: Optimization of number of Input Days for Seasonal Model

Number of Days							
Metric	5	10	15	20	25	**30**	35
Coefficient of Correlation	0.207	0.241	0.236	0.265	0.269	**0.27**	0.266
RMSE (mm)	17.4	17.2	17.4	17.1	17.1	**17.1**	17.1

Table 5.2 represents the search result for optimization algorithm. We see that Adam algorithm outperforms other.

Table 5.2: Selecting best Optimizing Algorithm for Seasonal Model

Optimization Algorithm							
Metric	**Adam**	SGD	RMSprop	Adagrad	Adadelta	Adamax	Nadam
MSE ($\times 10^{-4}$)	**4.99**	6.0	14.2	5.36	5.44	5.39	5.05

Adam algorithm has two parameters, viz. learning rate and momentum rate. Table 5.3 shows the 2-dimensional grid search for the best set. We select the set of those values which give least error.

Table 5.3: Optimization of learning rate and momentum rate for Seasonal Model using Adam optimization algorithm

		Learning Rate				
Momentum Rate		10^{-5}	$\mathbf{10^{-4}}$	10^{-3}	10^{-2}	10^{-1}
	0.8	15.35	8.01	8.32	13.68	5.9×10^4
	0.85	12	7.4	8.05	130.8	2.7×10^3
	0.9	10.79	7.4	9.37	18.83	337.6
	0.95	12.35	7.57	6.03	16.03	3.7×10^3
	0.98	13.10	6.13	7.66	15.49	124.13
	0.99	10.91	**5.80**	6.80	3.1×104	2.1×10^4

5.2 Annual Model

Similar to seasonal model, as discussed in section 5.1, annual model is developed by using all the months. Same set of experiments are performed to perform search for best parameters. Since, this model is trained on annual data, we see different set of optimal parameters.

Table 5.4 shows the grid for number of input days. We see that 30 days perform best for RMSE but 25 days perform slightly better in terms of coefficient of correlation. But we select 30 days, as we want to build a ensemble model using both seasonal and annual model.

Table 5.4: Optimization of number of Input Days for Annual Model

Number of Days							
Metric	5	10	15	20	25	**30**	35
Coefficient of Correlation	0.227	0.267	0.271	0.272	0.279	**0.277**	0.275
RMSE (mm)	17.2	17.1	17.1	17.1	17.4	**17.0**	17.0

We see from table 5.5 that Adamax performs best.

Table 5.5: Selecting best Optimizing Algorithm for Annual Model

Optimization Algorithm							
Metric	Adam	SGD	RMSprop	Adagrad	Adadelta	**Adamax**	Nadam
MSE ($\times 10^{-4}$)	2.11	2.31	3.68	2.37	2.25	**2.09**	5.49

We tune corresponding learning rate and momentum rate for Adamax algorithm. The search results for 2-dimensional grid can be seen in table 5.6.

Table 5.6: Optimization of learning rate and momentum rate for Annual Model using Adamax optimization algorithm

	Learning Rate				
Momentum Rate		10^{-5}	10^{-4}	$\mathbf{10^{-3}}$	10^{-2}
	0.8	3.98	2.33	3.10	16.40
	0.85	4.70	2.28	3.21	6.57
	0.9	3.73	2.32	2.24	3.41
	0.95	5.24	2.24	**2.14**	2.92
	0.98	4.29	2.24	2.44	3.38
	0.99	4.74	2.21	2.22	4.04

5.3 Ensemble Model

We develop ensemble of seasonal and annual model. Seasonal model captures the monsoonal pattern and annual model captures the long term pattern. The ensemble of these two gives better result than any of the two individually, as will be discussed in chapter 6. Ensemble is simply the average of the two predictions.

CHAPTER 6

Results

We finally predict rainfall for 5 days lead time for JJAS months for all the three models. We have used data from 2005-2012 for testing, which generates a sample size of 704. We have used coefficient of correlation at every grid point as our performance metric. The coefficient of correlation is calculated between the predicted time series and observed time series at every point. For such large sample size, we observe our results to be statistically significant. To give a clear picture of the result, we have plotted root mean square error (RMSE).

6.1 Results from Seasonal Model

The coefficient of correlation for all the five days is shown in figure 6.1. It can be seen that we are able to predict rainfall with good correlation for the first three days. However, the correlation falls down as we progress towards 5 day lead time. The western ghats, central India, north-east and Ladakh region gives good correlation compared to other parts of India.

The RMSE plot in figure 6.2 shows the error between observed and predicted time series. From the plots it can be seen that the prediction for all the days have almost same error range.

6.2 Results from Annual model

The annual is also tested on the same data-set as in seasonal model. Compared to seasonal model annual model has larger data to train. The results can be seen to improve in some cases and deteriorate in other cases, as compared to seasonal model. We have used same metric to evaluate, coefficient of correlation and RMSE. The correlation plot

for all the 5 days lead time is shown in figure 6.3. We observe that the maximum correlation increases from 0.56 to 0.64 in 2 day lead time case, which is an improvement. However the correlation drops for the 3 and 4 day lead time. The annual model performs equally good for the first three days. The parts of India which performed well in annual model can be seen as similar to seasonal model. The western ghats, north-east India, central parts and Ladakh region prediction have good correlation.

RMSE for all five lead days can be observed from the plots in figure 6.4Ṫhere is no significant change in the RMSE as can be seen by comparing the corresponding plot for seasonal model.

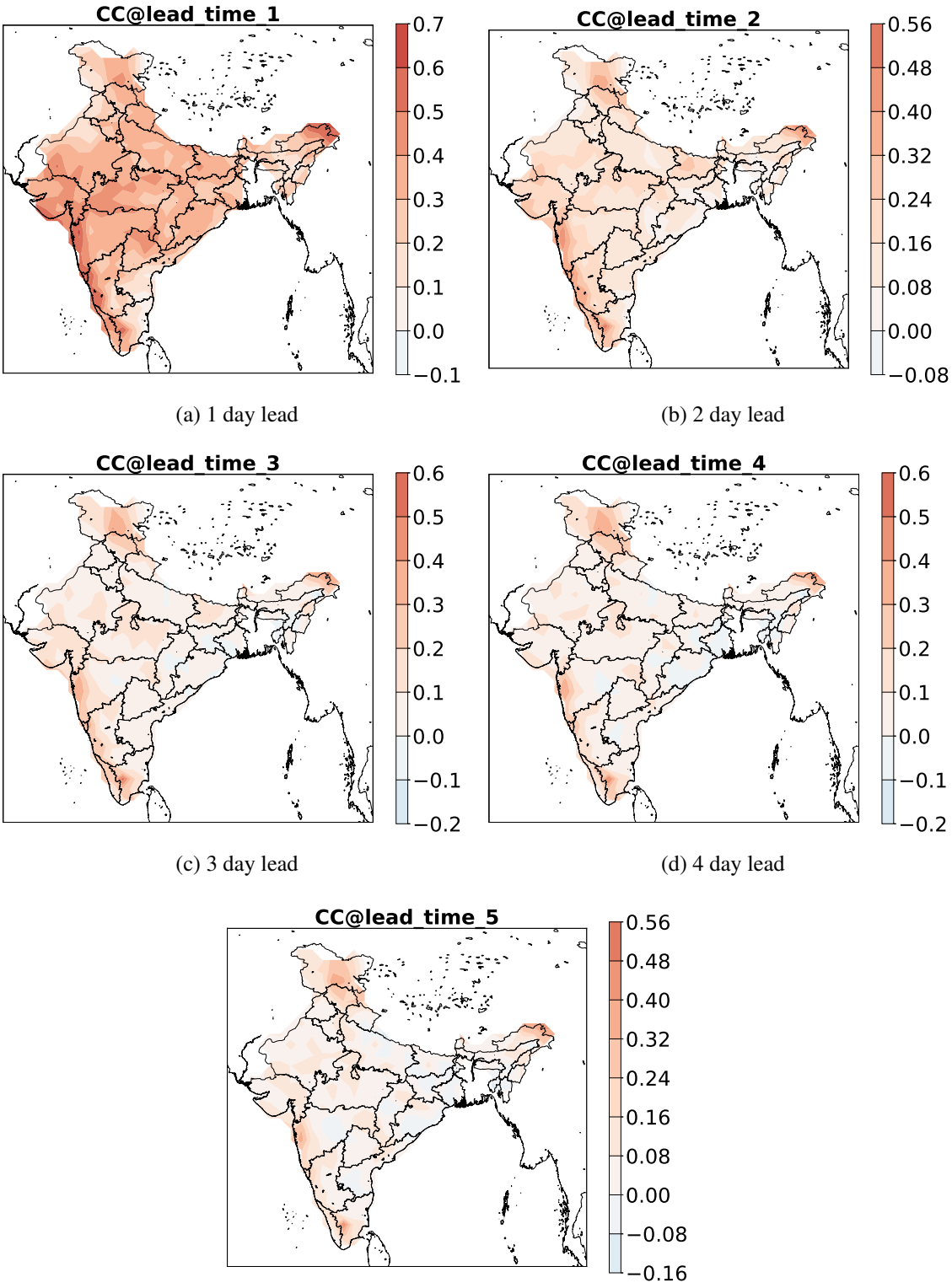
6.3 Results from Ensemble Model

Ensemble results are calculated by averaging the predictions from seasonal and annual model. We observe that the ensemble model outperforms both seasonal and annual model. Testing the predictions from ensemble on the same 704 samples, we find that the model improves the coefficient of correlation for 2, 3 and 5 day lead time as compared to the seasonal model. But, the 4^{th} day results deteriorate. The plots in figure 6.5 shows the improvements as compared to the corresponding plots in the seasonal model. We observe that the ensemble model captures the northern India, central India, north-east, western ghats and Ladakh regions with good correlation.

The RMSE plots can be observed from figure 6.6. The figures resembles similar to that of the seasonal and annual models. There is no improvements in RMSE but the trends in ISMR is better captured in ensemble model.

We have shown the year-wise analysis of the predictions in Appendix A.

Results from Seasonal Model - Coefficient of Correlation (CC)



(a) 1 day lead

(b) 2 day lead

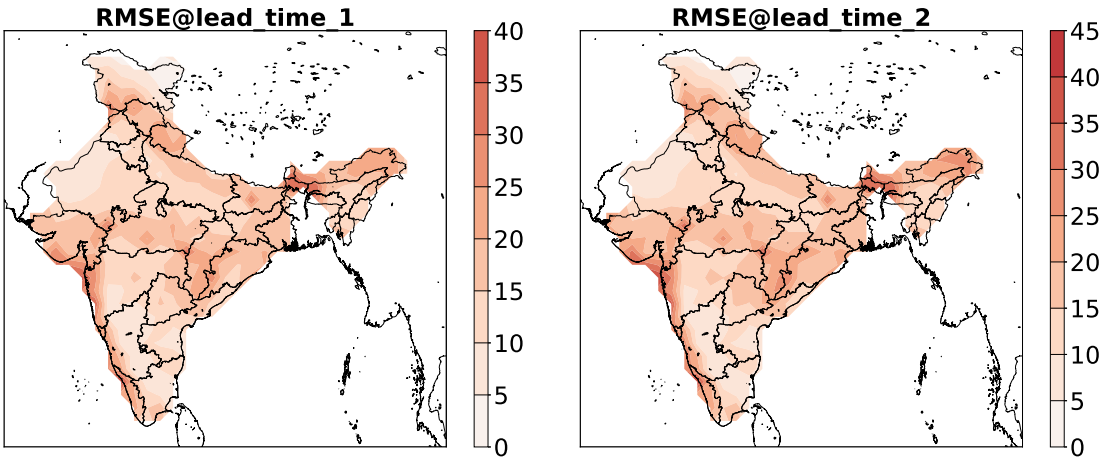
(c) 3 day lead

(d) 4 day lead

(e) 5 day lead

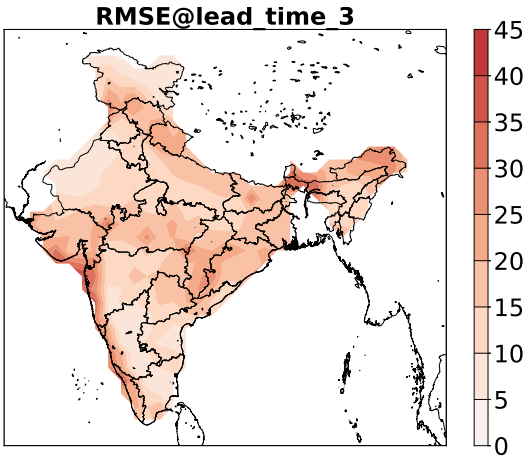
Figure 6.1: Coefficient of Correlation [Seasonal]

Results from Seasonal Model - Root mean square error (RMSE [in mm])

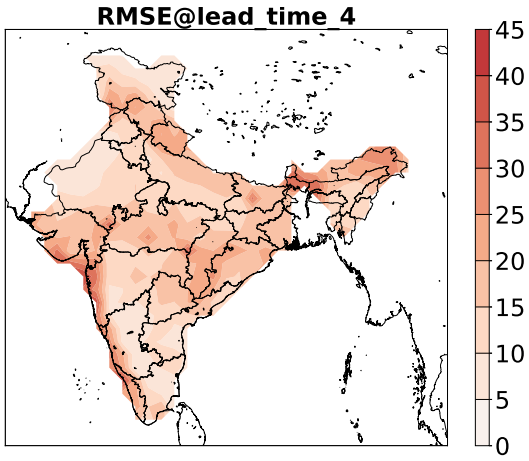


(a) 1 day lead

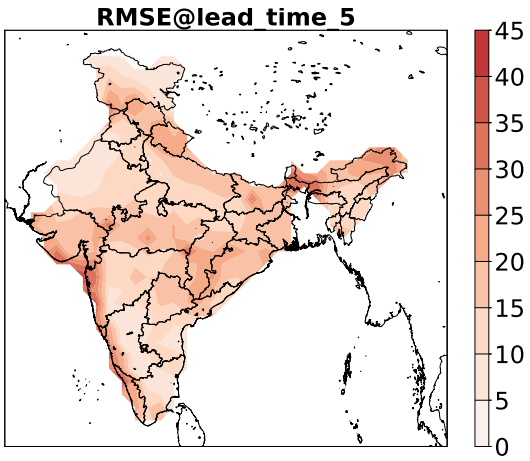
(b) 2 day lead



(c) 3 day lead



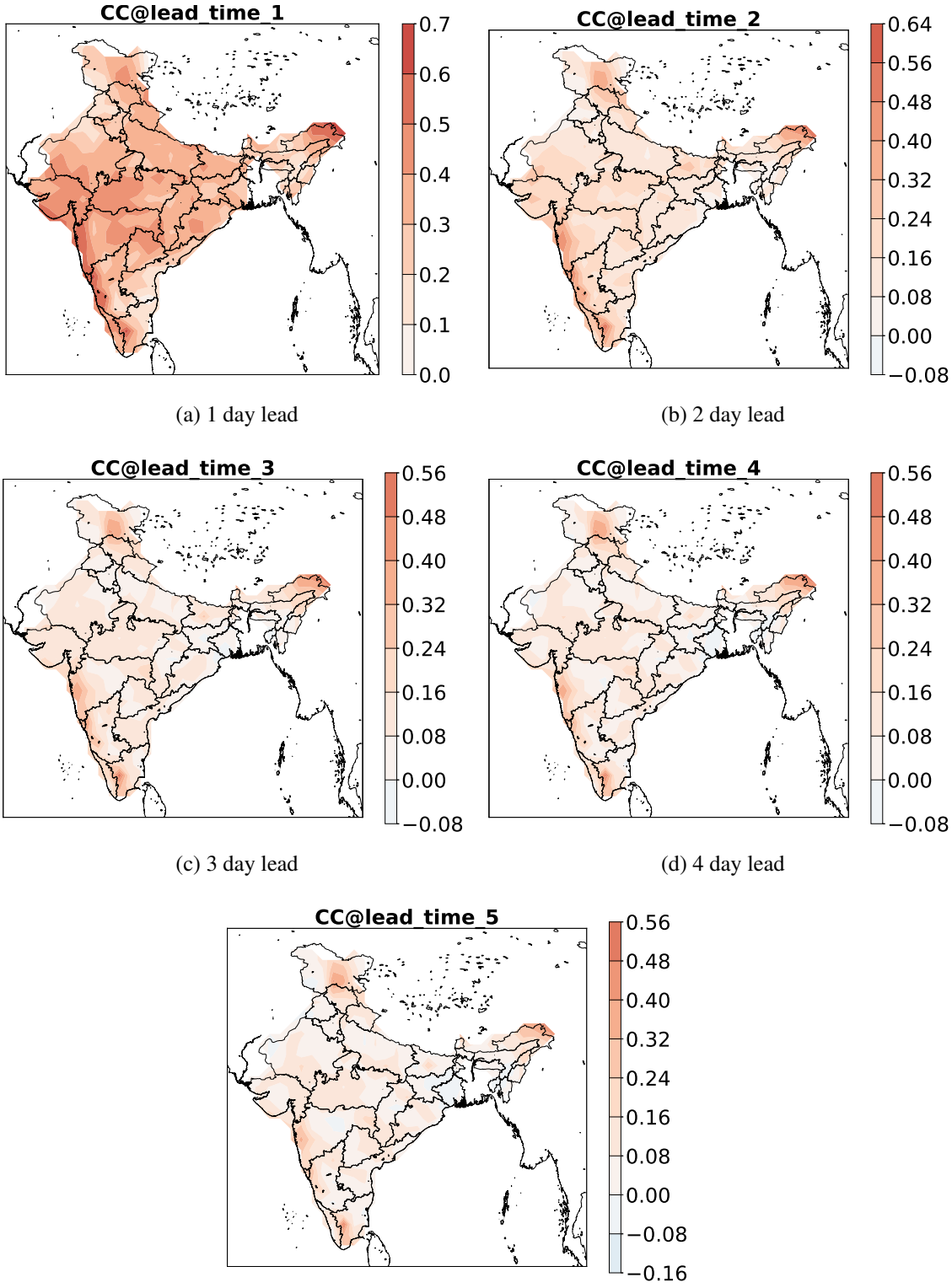
(d) 4 day lead



(e) 5 day lead

Figure 6.2: Root Mean Square Error [Seasonal]

Results from Annual Model - Coefficient of Correlation (CC)



(a) 1 day lead

(b) 2 day lead

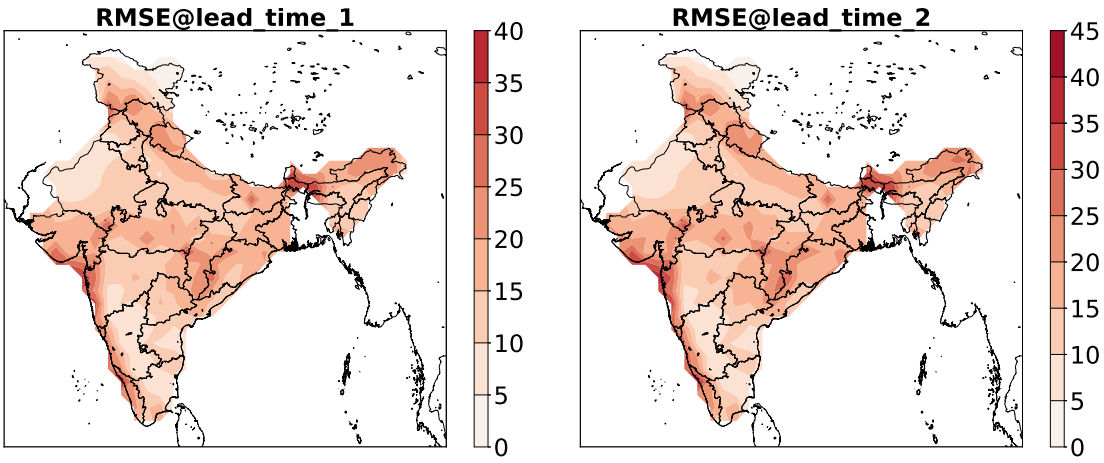
(c) 3 day lead

(d) 4 day lead

(e) 5 day lead

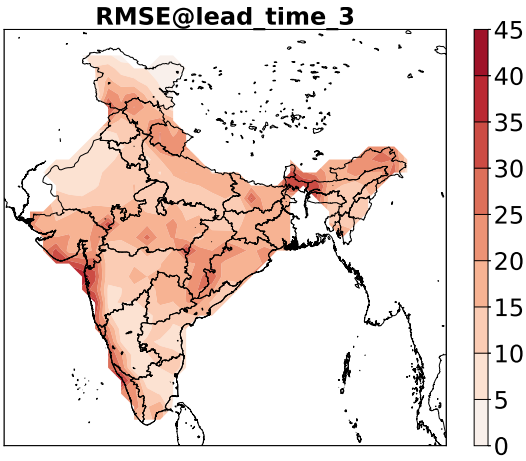
Figure 6.3: Coefficient of Correlation [Annual]

Results from Annual Model - Root mean square error (RMSE [in mm])

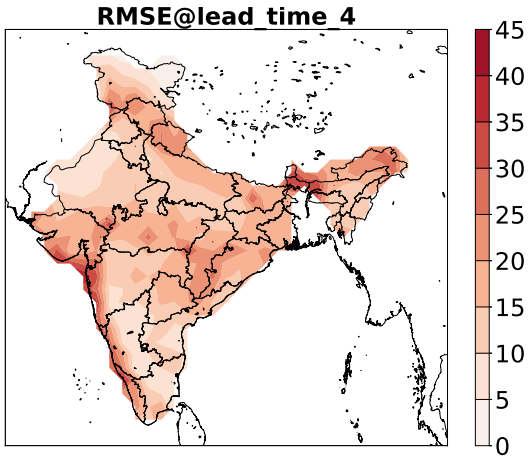


(a) 1 day lead

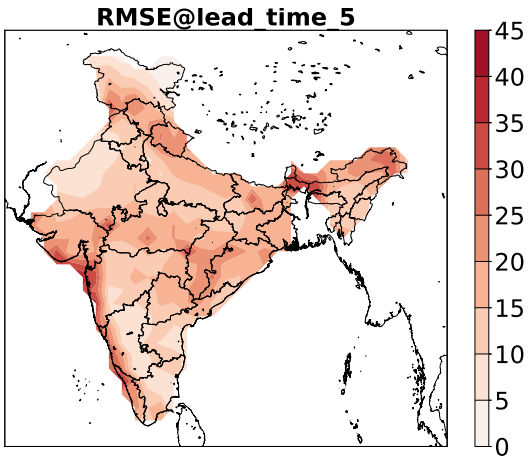
(b) 2 day lead



(c) 3 day lead



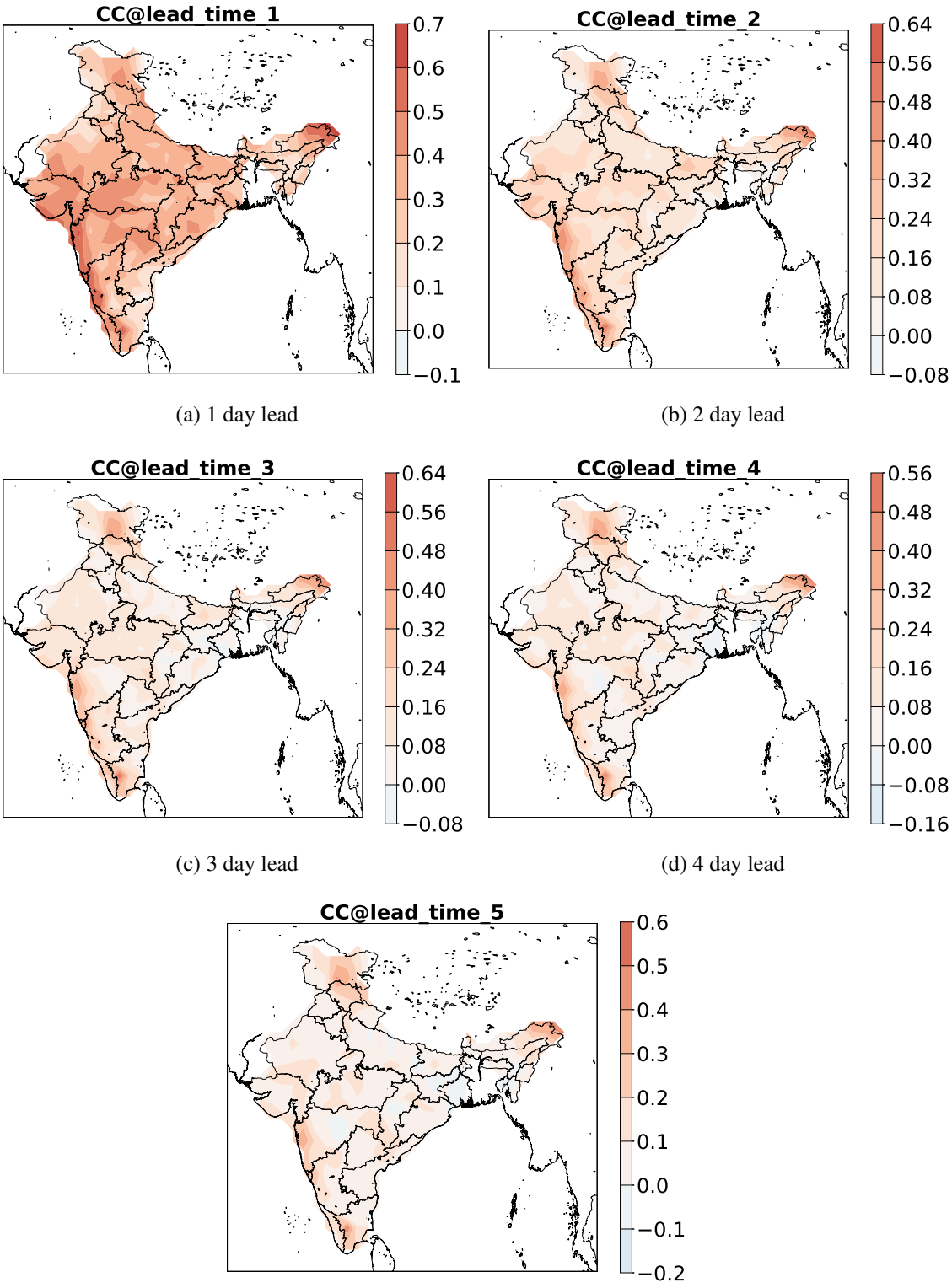
(d) 4 day lead



(e) 5 day lead

Figure 6.4: Root Mean Square Error [Annual]

Results from Ensemble Model - Coefficient of Correlation (CC)



(a) 1 day lead

(b) 2 day lead

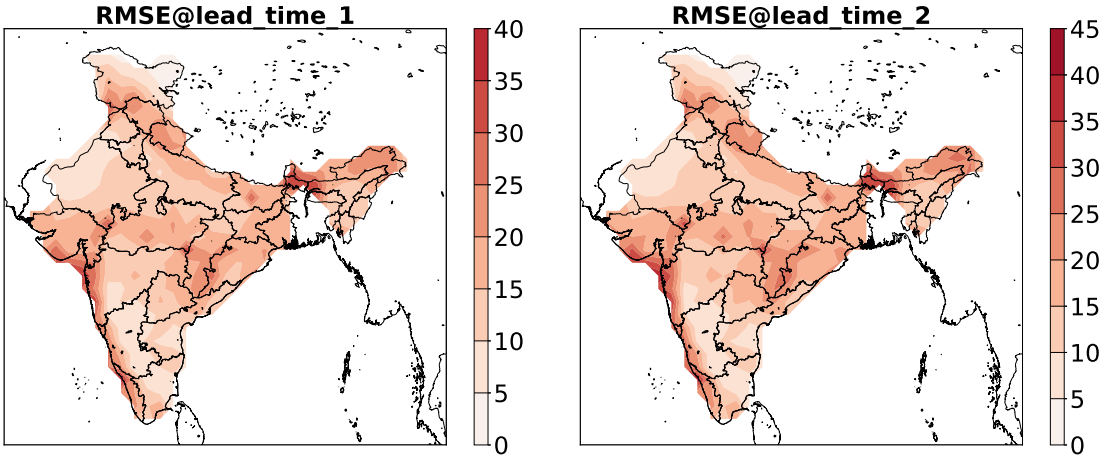
(c) 3 day lead

(d) 4 day lead

(e) 5 day lead

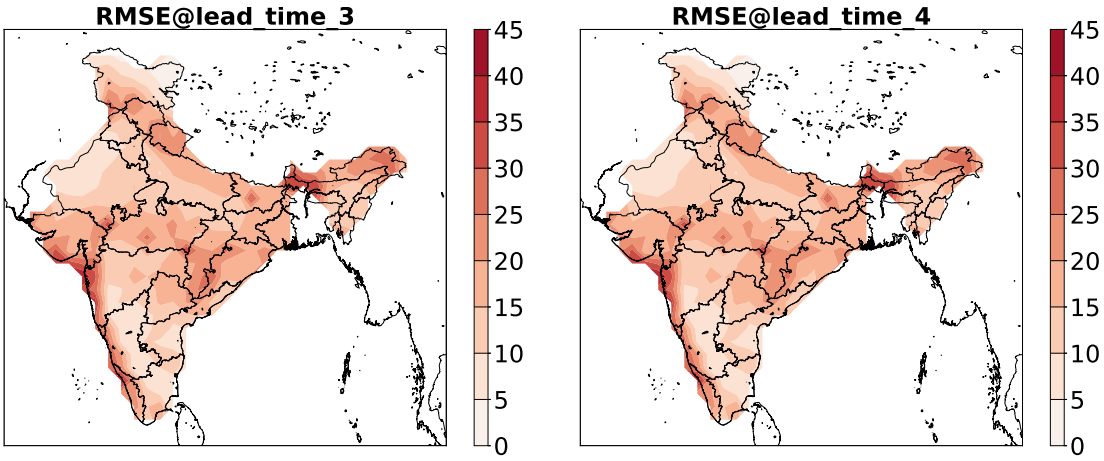
Figure 6.5: Coefficient of Correlation [Ensemble]

Results from Ensemble Model - Root mean square error (RMSE [in mm])



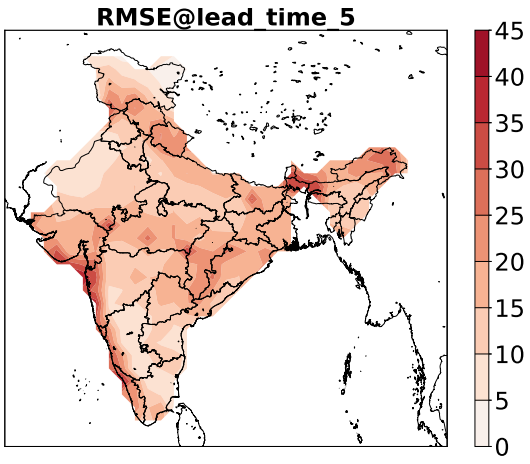
(a) 1 day lead

(b) 2 day lead



(c) 3 day lead

(d) 4 day lead



(e) 5 day lead

Figure 6.6: Root Mean Square Error [Ensemble]

CHAPTER 7

Conclusion and Future Work

In this thesis, a new statistical approach to capture spatio-temporal pattern is discussed. The deep learning model based on conv-LSTM algorithm is shown to successfully predict future rainfall events using past spatio-temporal pattern. The model is computationally efficient, taking considerably less time than dynamic models to predict future rainfall which can be very helpful. The model shows good results in most parts of India, for up to 3 days lead. It is shown that the model performs better in western ghats, northern India, north-east, parts of central India and Ladakh regions. The model is shown to give good spatial pattern for these parts of India.

The work also discussed the modeling aspects of the prediction problem using deep learning. It is shown that artificial intelligence and deep learning can be very helpful in atmospheric and environmental sciences domain. This work presents a uni-variate study using rainfall as the only variable.

The future direction of this work include multi-variate analysis using several other climatic variables such as sea surface temperature (SST), sea level pressure (SLP), air temperature (AT), humidity etc. Other data sets can be used to include climate on oceans. Other models such as temporal convolutional neural networks (T-CNN), generative adversarial networks (GAN) can be used. The satellite data can be used along with observation data to model. The dynamic simulation results can be also used to develop deep learning model. The idea of deep learning can be extended to solve more problem in atmospheric science.

APPENDIX A

Detail Interpretation of Results

In this chapter, we discuss the detail analysis of the performance. We have plotted the results for each year separately to observe the model performance.

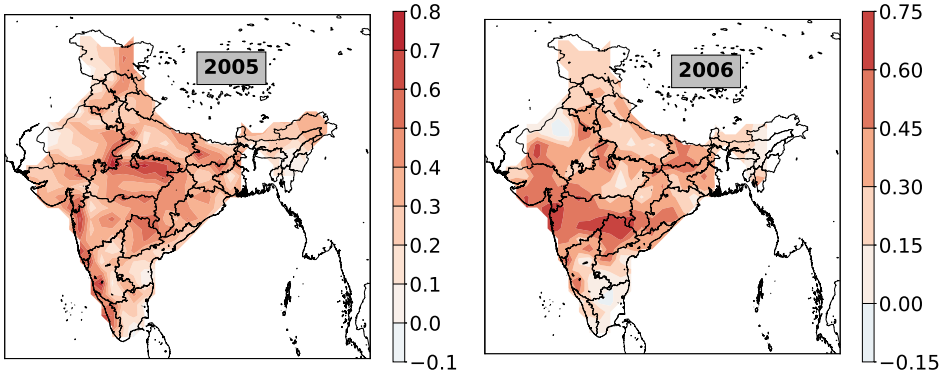
A.1 Yearly Coefficient of Correlation

Figure A.1 shows the point wise coefficient of correlation for 1 day lead time for 2005-2012. We see that model performs equally good in all the years. We observe some negative correlated points but the regions of western ghats, central India, north-east India, Ladakh and northern India performs equally good. Regions such as Tamil Nadu and parts of Punjab does not performs good.

Figure A.2

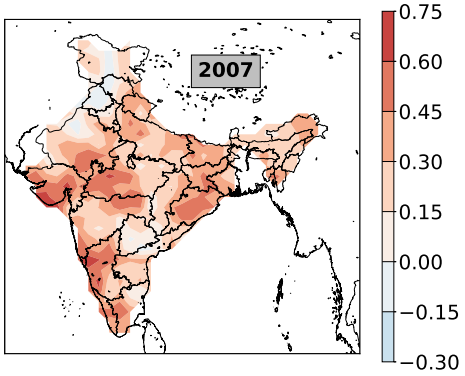
A.2 Yearly Root Mean Square Error

Yearwise Coefficient of Correlation (CC) - 1 day lead time

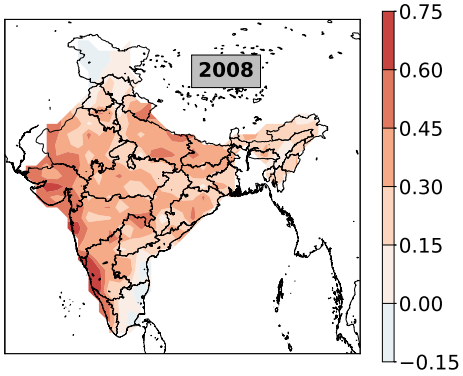


(a) 2005

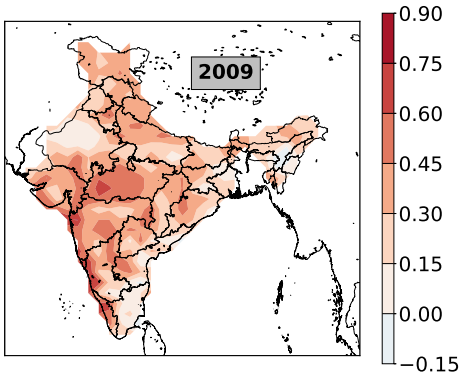
(b) 2006



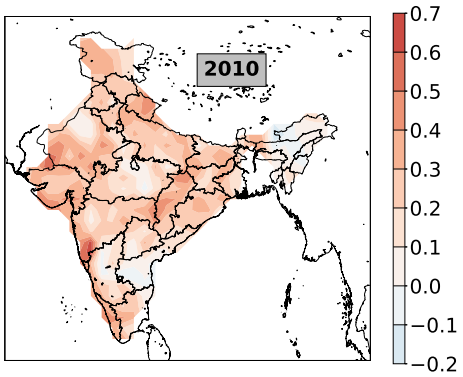
(c) 2007



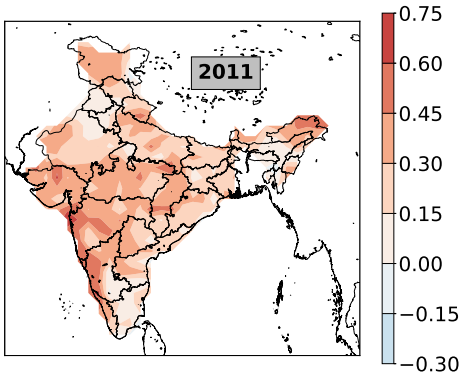
(d) 2008



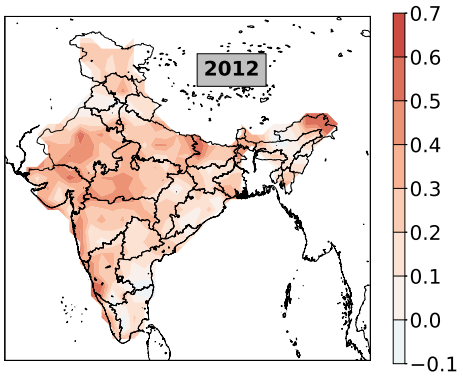
(e) 2009



(f) 2010



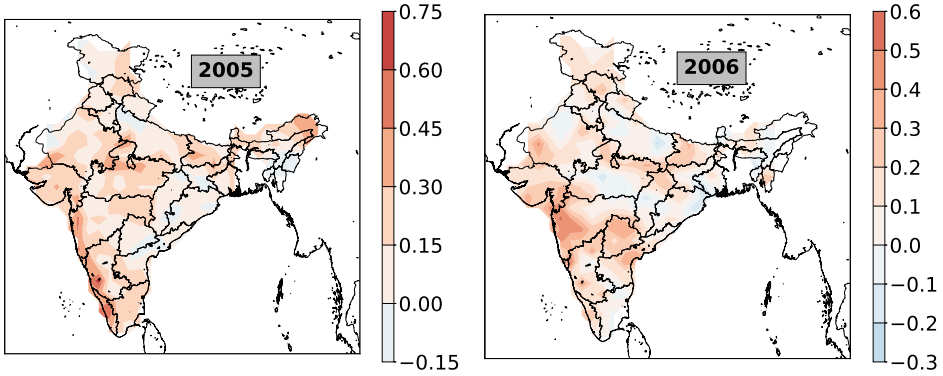
(g) 2011



(h) 2012

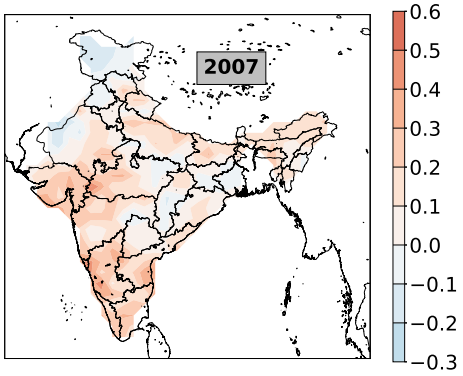
Figure A.1: Yearwise Coefficient of Correlation for lead time 1 day

Yearwise Coefficient of Correlation (CC) - 2 day lead time

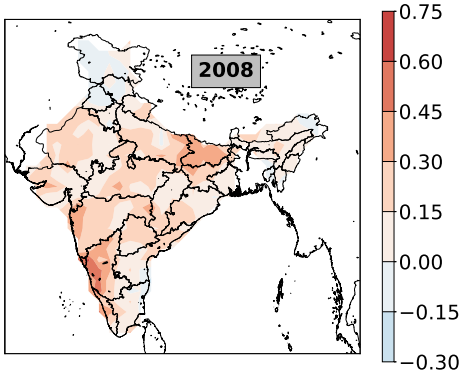


(a) 2005

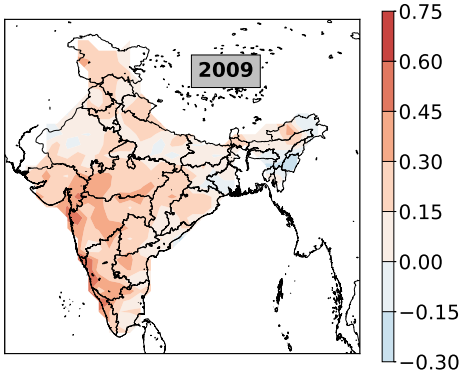
(b) 2006



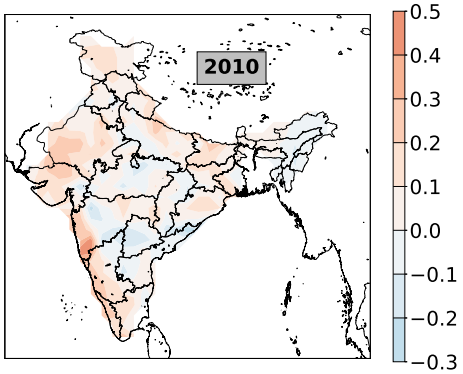
(c) 2007



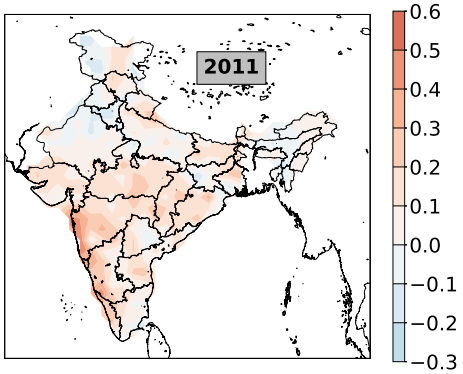
(d) 2008



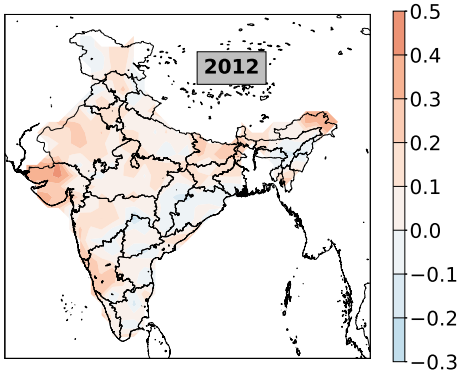
(e) 2009



(f) 2010



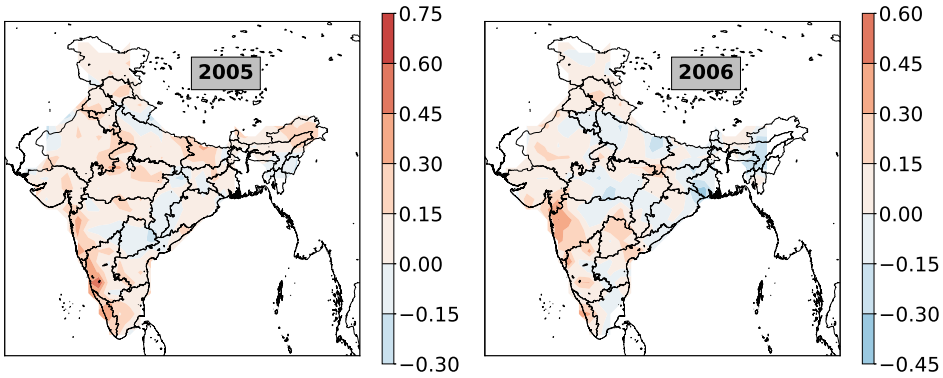
(g) 2011



(h) 2012

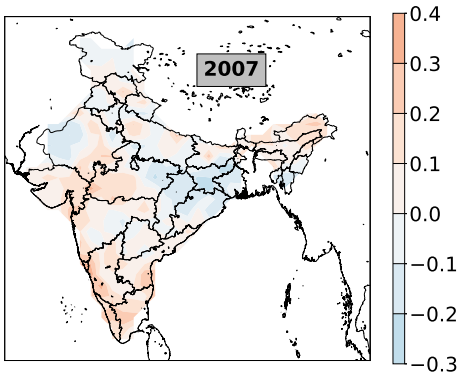
Figure A.2: Yearwise Coefficient of Correlation for lead time 2 day

Yearwise Coefficient of Correlation (CC) - 3 day lead time

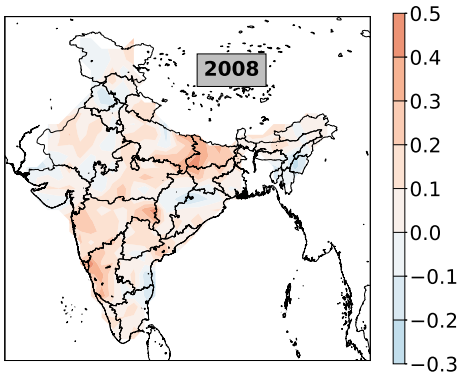


(a) 2005

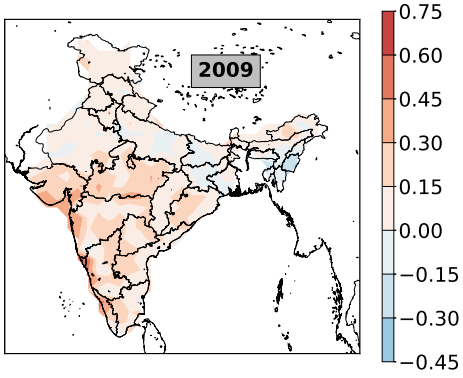
(b) 2006



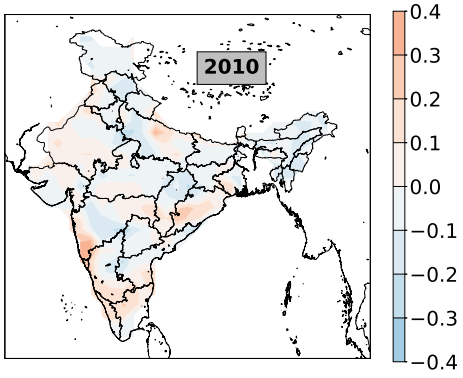
(c) 2007



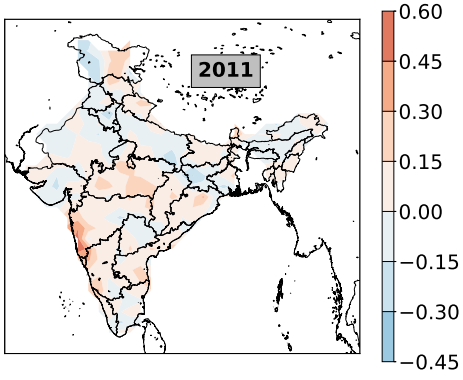
(d) 2008



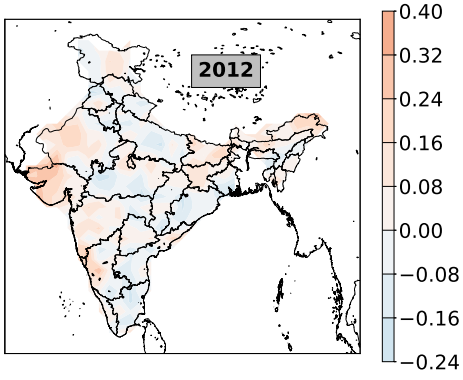
(e) 2009



(f) 2010



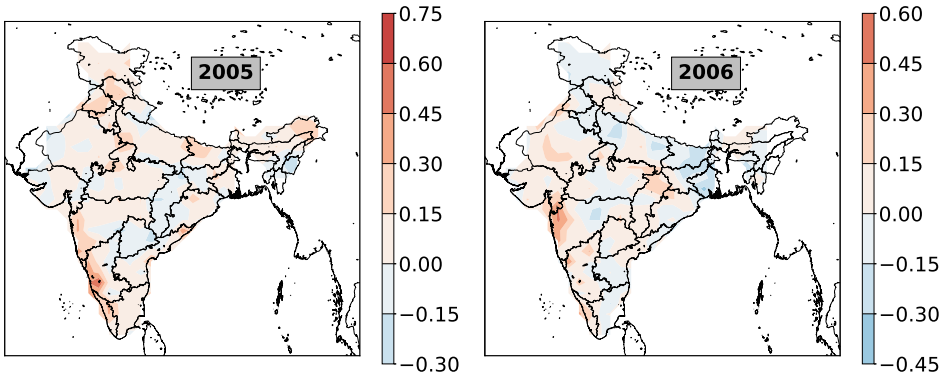
(g) 2011



(h) 2012

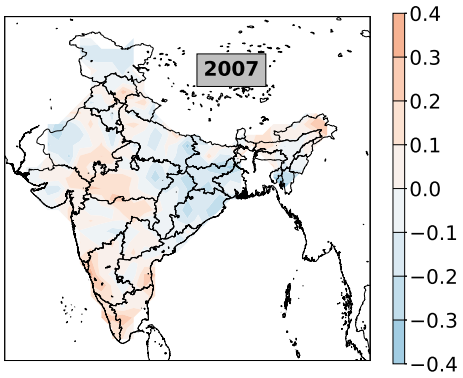
Figure A.3: Yearwise Coefficient of Correlation for lead time 3 day

Yearwise Coefficient of Correlation (CC) - 4 day lead time

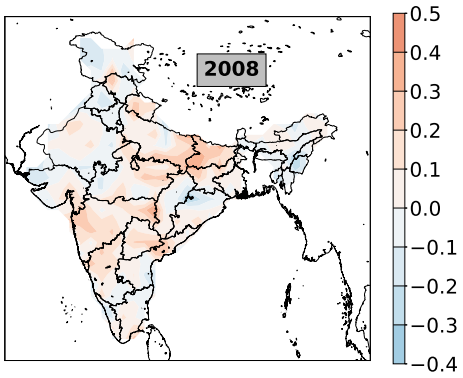


(a) 2005

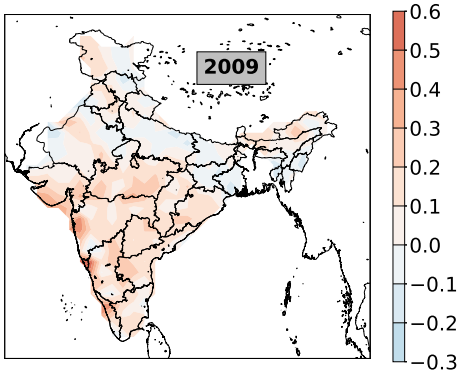
(b) 2006



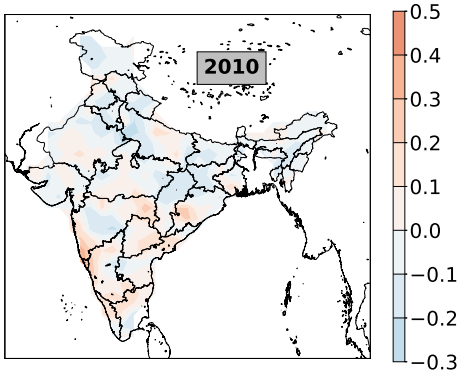
(c) 2007



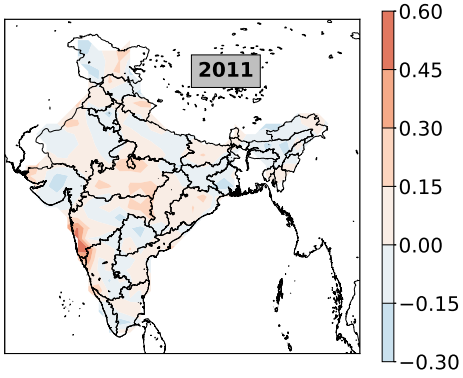
(d) 2008



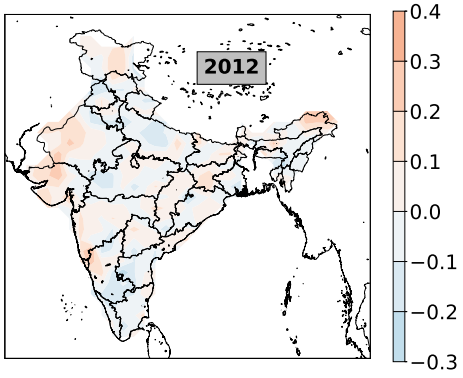
(e) 2009



(f) 2010



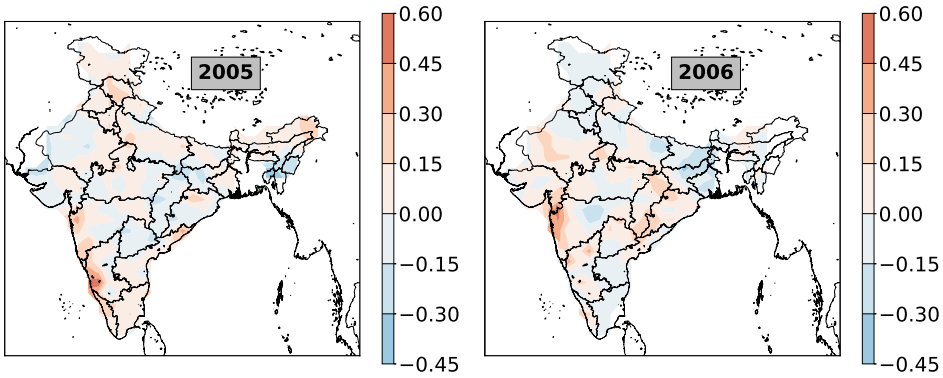
(g) 2011



(h) 2012

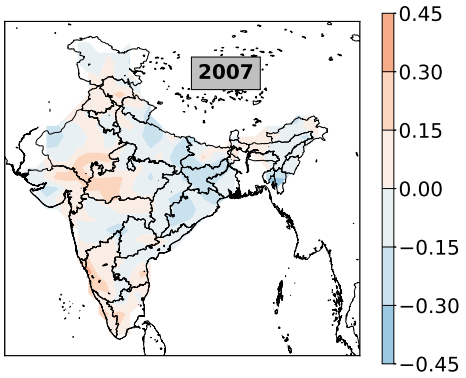
Figure A.4: Yearwise Coefficient of Correlation for lead time 4 day

Yearwise Coefficient of Correlation (CC) - 4 day lead time

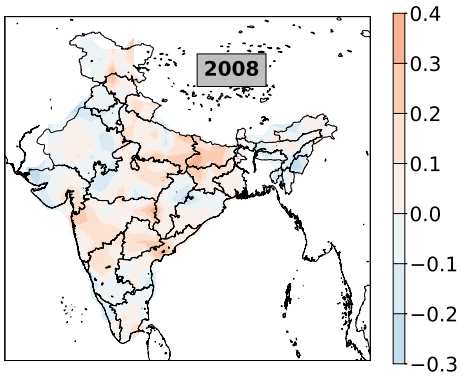


(a) 2005

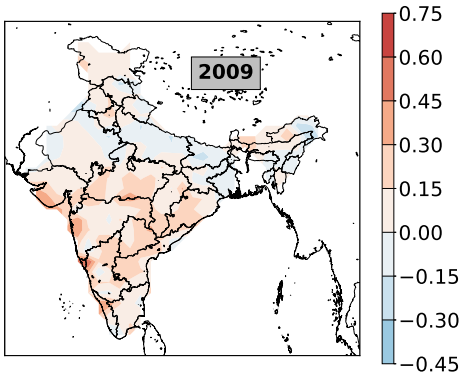
(b) 2006



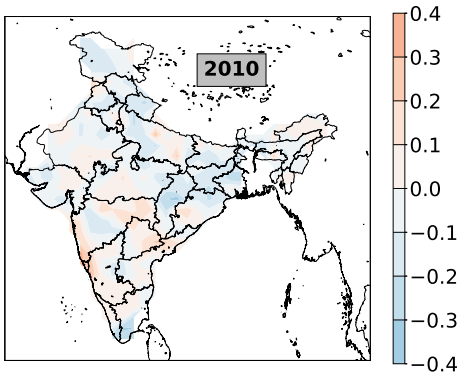
(c) 2007



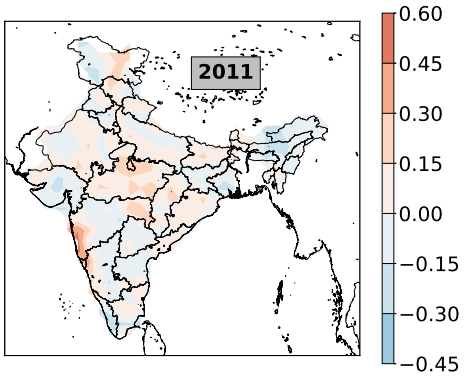
(d) 2008



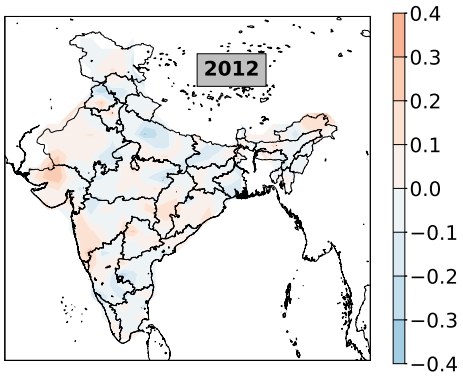
(e) 2009



(f) 2010



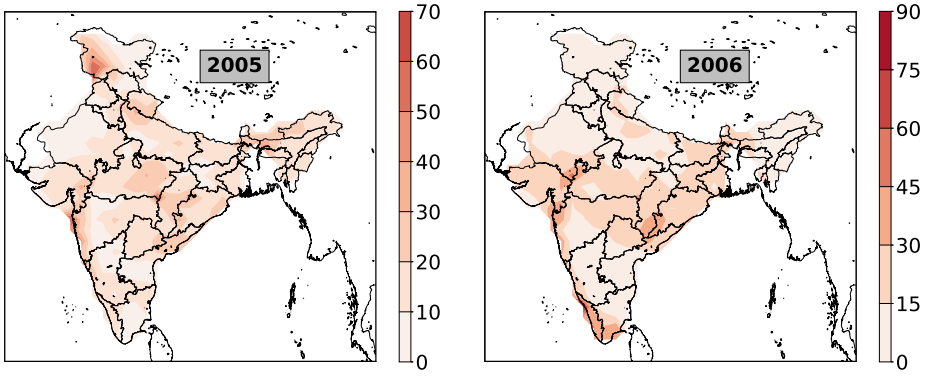
(g) 2011



(h) 2012

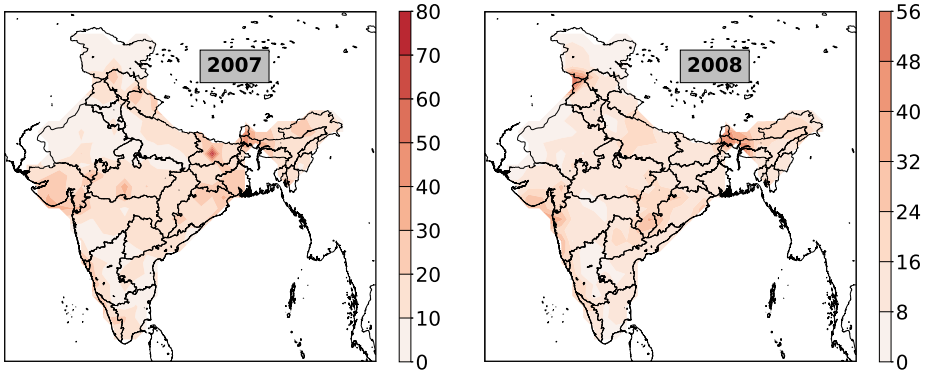
Figure A.5: Yearwise Coefficient of Correlation for lead time 5 day

Yearwise Root Mean Square Error (RMSE [in mm]) - 1 day lead time



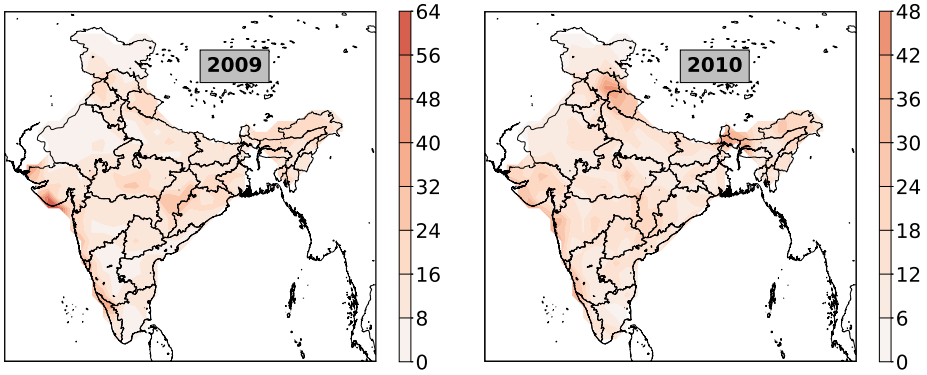
(a) 2005

(b) 2006



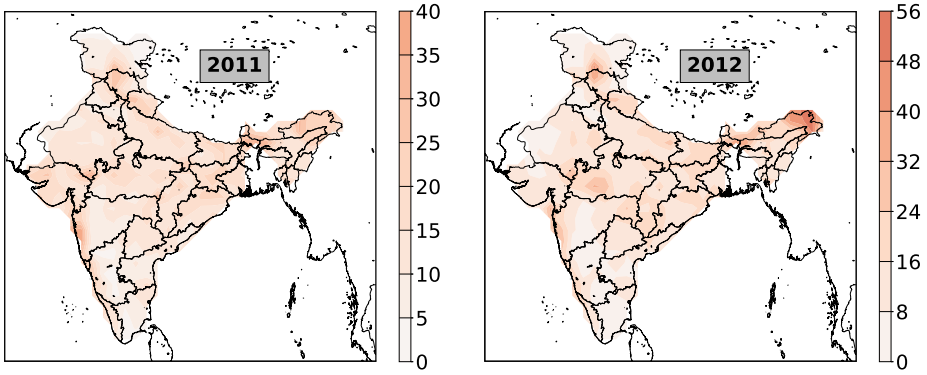
(c) 2007

(d) 2008



(e) 2009

(f) 2010

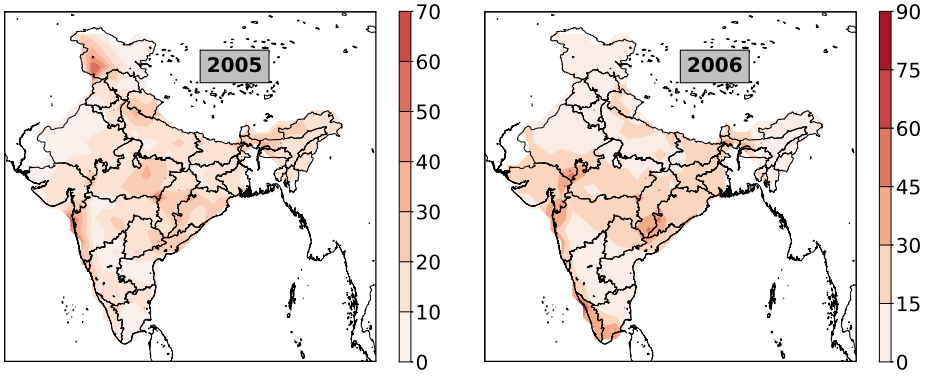


(g) 2011

(h) 2012

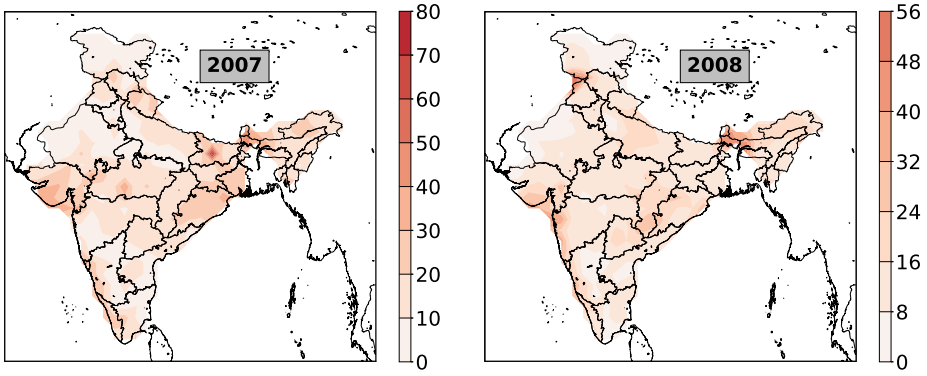
Figure A.6: Yearwise RMSE for lead time 1 day

Yearwise Root Mean Square Error (RMSE [in mm]) - 2 day lead time



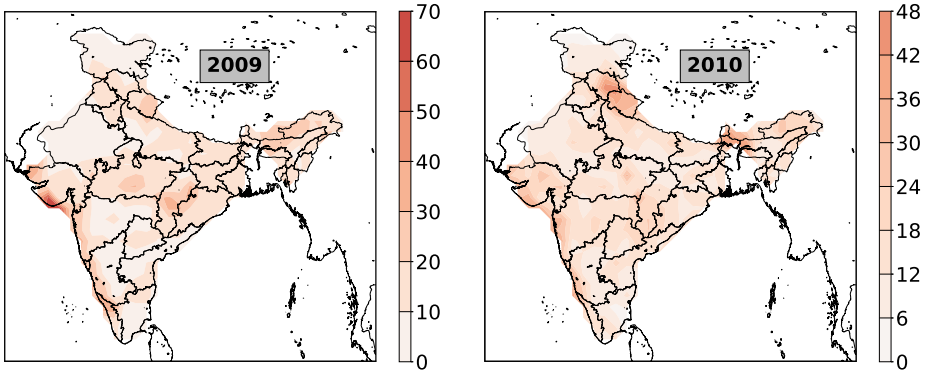
(a) 2005

(b) 2006



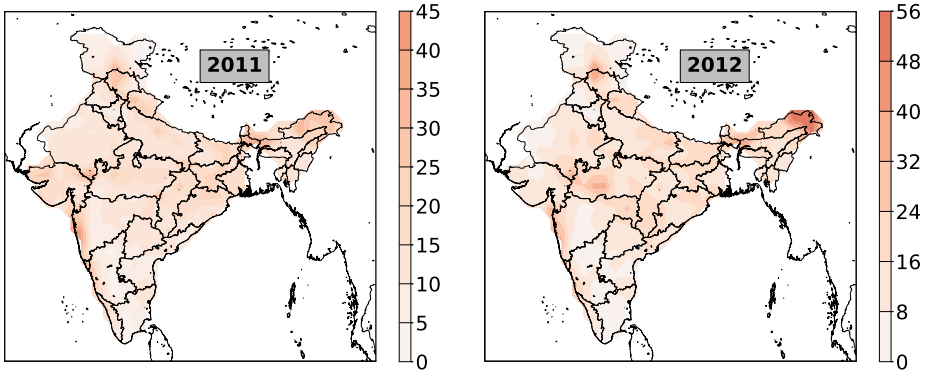
(c) 2007

(d) 2008



(e) 2009

(f) 2010

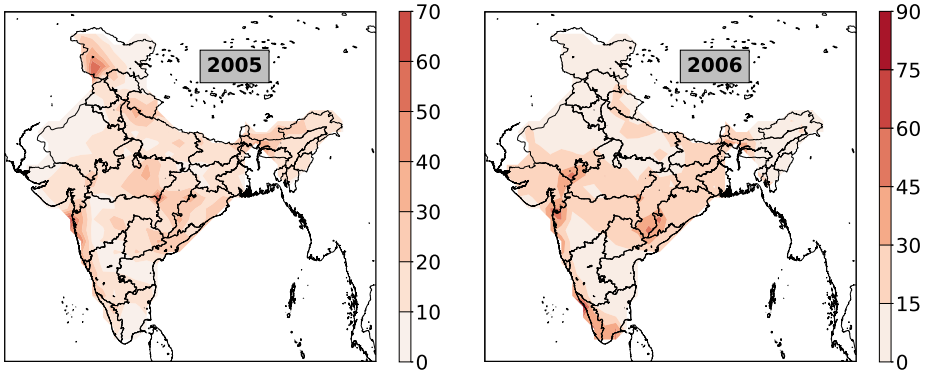


(g) 2011

(h) 2012

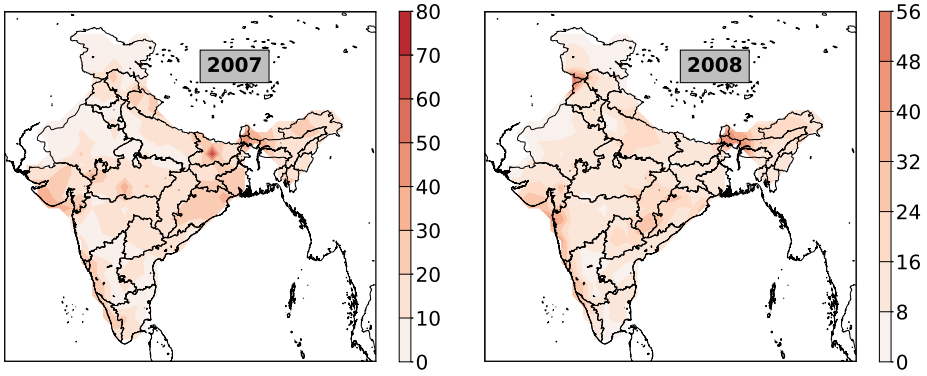
Figure A.7: Yearwise RMSE for lead time 2 day

Yearwise Root Mean Square Error (RMSE [in mm]) - 3 day lead time



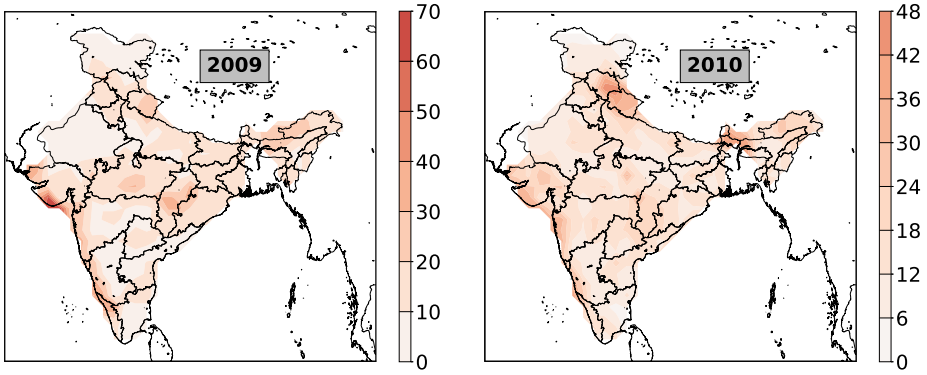
(a) 2005

(b) 2006



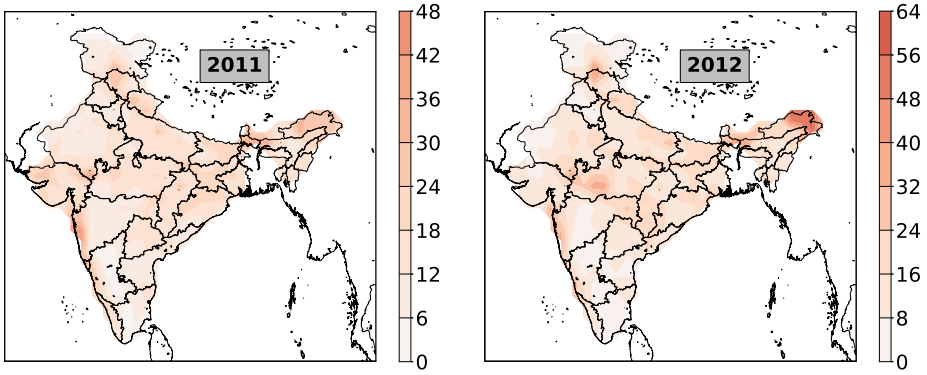
(c) 2007

(d) 2008



(e) 2009

(f) 2010

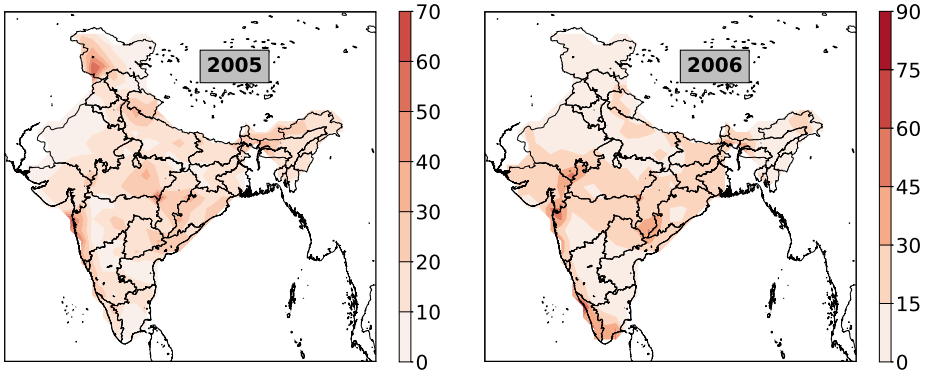


(g) 2011

(h) 2012

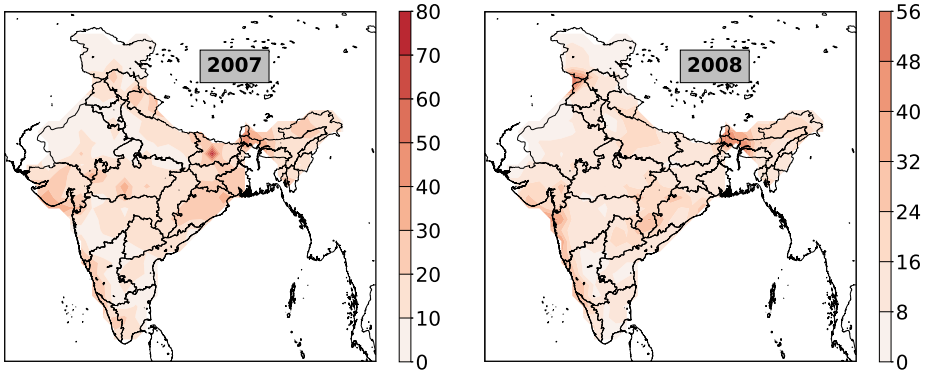
Figure A.8: Yearwise RMSE for lead time 3 day

Yearwise Root Mean Square Error (RMSE [in mm]) - 4 day lead time



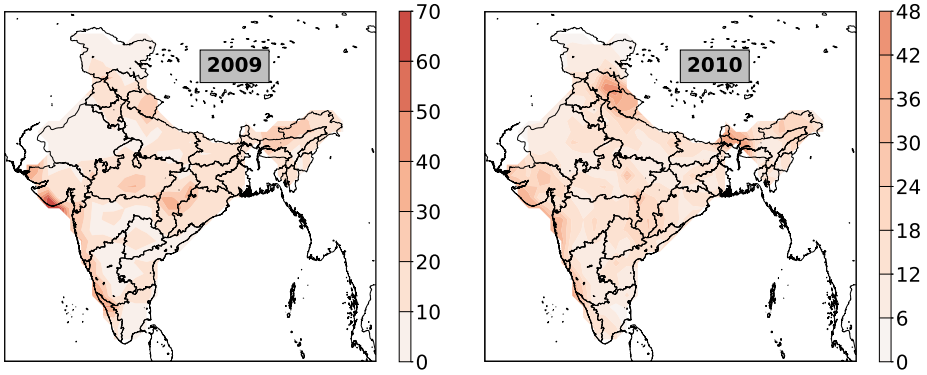
(a) 2005

(b) 2006



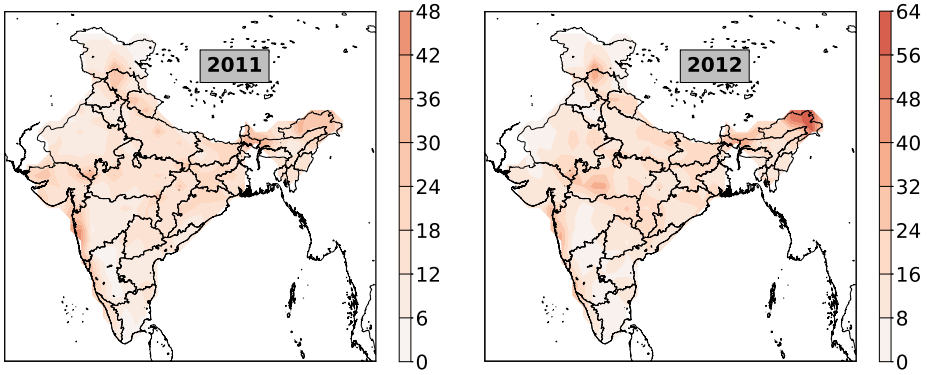
(c) 2007

(d) 2008



(e) 2009

(f) 2010

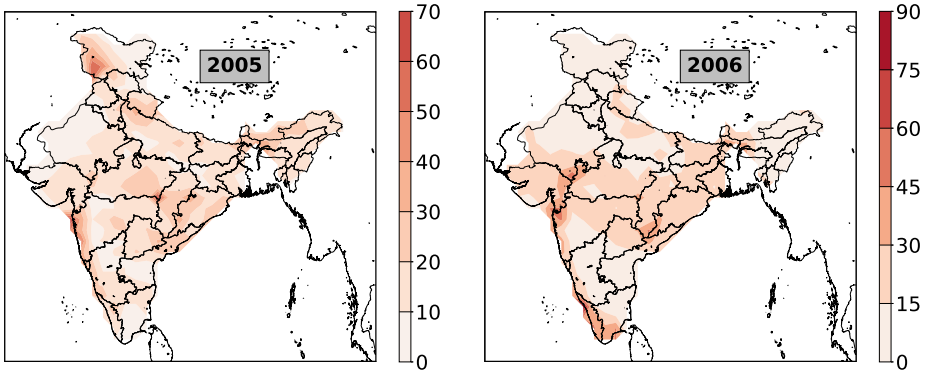


(g) 2011

(h) 2012

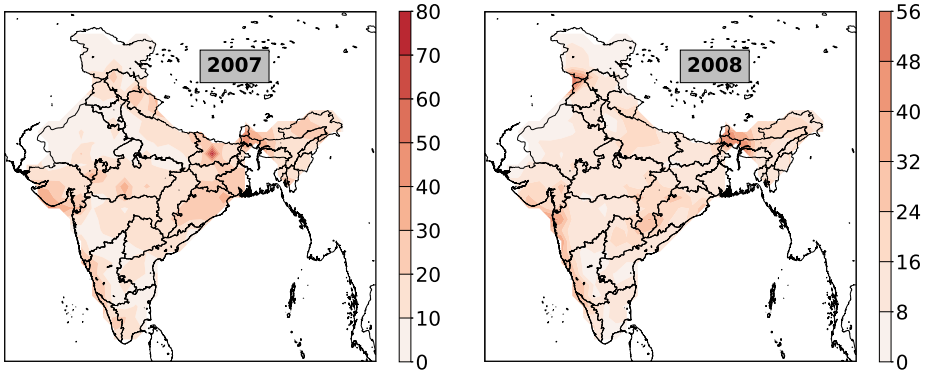
Figure A.9: Yearwise RMSE for lead time 4 day

Yearwise Root Mean Square Error (RMSE [in mm]) - 5 day lead time



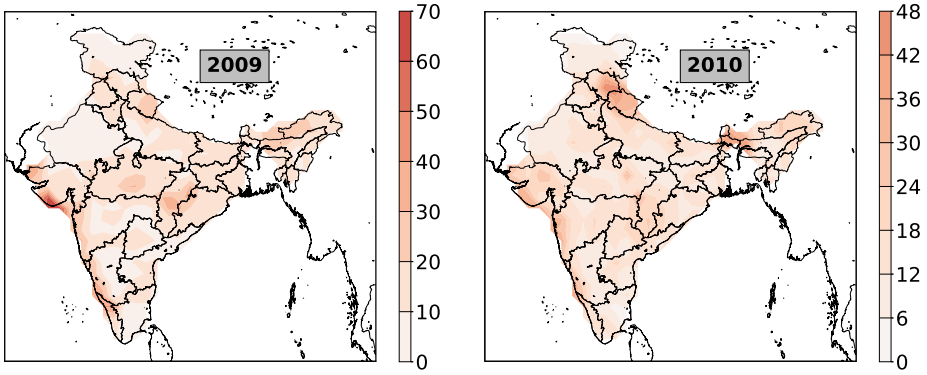
(a) 2005

(b) 2006



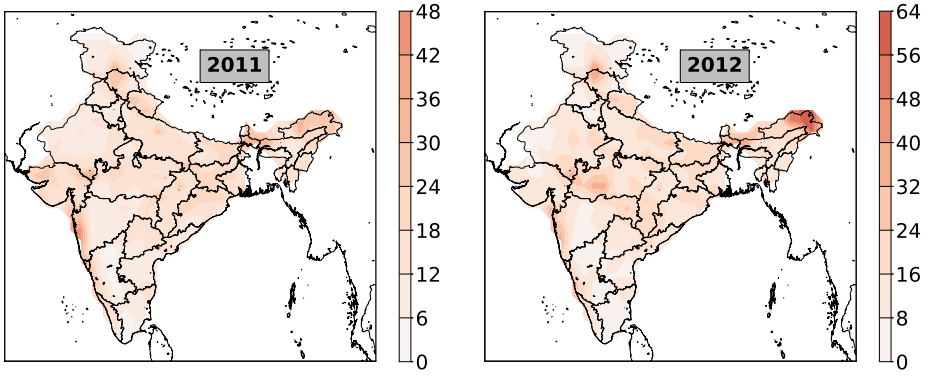
(c) 2007

(d) 2008



(e) 2009

(f) 2010



(g) 2011

(h) 2012

Figure A.10: Yearwise RMSE for lead time 5 day

REFERENCES

1. **Abadi, M.**, **A. Agarwal**, **P. Barham**, **E. Brevdo**, **Z. Chen**, **C. Citro**, **G. S. Corrado**, **A. Davis**, **J. Dean**, **M. Devin**, **S. Ghemawat**, **I. Goodfellow**, **A. Harp**, **G. Irving**, **M. Isard**, **Y. Jia**, **R. Jozefowicz**, **L. Kaiser**, **M. Kudlur**, **J. Levenberg**, **D. Mané**, **R. Monga**, **S. Moore**, **D. Murray**, **C. Olah**, **M. Schuster**, **J. Shlens**, **B. Steiner**, **I. Sutskever**, **K. Talwar**, **P. Tucker**, **V. Vanhoucke**, **V. Vasudevan**, **F. Viégas**, **O. Vinyals**, **P. Warden**, **M. Wattenberg**, **M. Wicke**, **Y. Yu**, and **X. Zheng** (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

2. **Chollet, F.** *et al.* (2015). Keras. `https://keras.io`.

3. **Deshpande, R.** and **N. Prabhu** (2005). Farmers' distress: Proof beyond question. *Economic and Political Weekly*, **40**, 4663–4665.

4. **Gers, F. A.** and **J. Schmidhuber** (2000). Recurrent nets that time and count. Technical report.

5. **Gowariker, V.**, **V. Thapliyal**, **S. Kulshrestha**, **G. Mandal**, **N. Sen Roy**, and **D. Sikka** (1991). A power regression model for long range forecast of southwest monsoon rainfall over india. *Mausam*, **42**, 125–130.

6. **Hochreiter, S.** and **J. Schmidhuber** (1997). Long short-term memory. *Neural Comput.*, **9**(8), 1735–1780. ISSN 0899-7667. URL `http://dx.doi.org/10.1162/neco.1997.9.8.1735`.

7. **Mahapatra, D.** (2017). Over 12,000 farmer suicides per year, centre tells supreme court. *The Times of India*. URL `https://timesofindia.indiatimes.com/india/over-12000-farmer-suicides-per-year-centre-tells-supreme-court/articleshow/58486441.cms`.

8. **Mishra, V.**, **S. Aadhar**, **H. Shah**, **R. Kumar**, **D. Ranjan Pattanaik**, and **A. Tiwari** (2018). The kerala flood of 2018: combined impact of extreme rainfall and reservoir storage. *Hydrology and Earth System Sciences Discussions*, 1–13.

9. **Olah, C.** (2015). Understanding lstm networks. URL `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`.

10. **Pedregosa, F.**, **G. Varoquaux**, **A. Gramfort**, **V. Michel**, **B. Thirion**, **O. Grisel**, **M. Blondel**, **P. Prettenhofer**, **R. Weiss**, **V. Dubourg**, **J. Vanderplas**, **A. Passos**, **D. Cournapeau**, **M. Brucher**, **M. Perrot**, and **E. Duchesnay** (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, **12**, 2825–2830.

11. **Rajeevan, M.**, **J. Bhate**, **J. D. Kale**, and **B. Lal** (2006). A high resolution daily gridded rainfall data for the indian region: Analysis of break and active monsoon spells. *Curr Sci India*, **91**.

12. **Saha, M.**, **P. Mitra**, and **R. S. Nanjundiah** (2016*a*). Autoencoder-based identification of predictors of indian monsoon. *Meteorology and Atmospheric Physics*, **128**(5), 613–628. ISSN 1436-5065. URL https://doi.org/10.1007/s00703-016-0431-7.

13. **Saha, M.**, **P. Mitra**, and **R. S. Nanjundiah** (2016*b*). Predictor discovery for early-late indian summer monsoon using stacked autoencoder. *Procedia Computer Science*, **80**, 565 – 576. ISSN 1877-0509. URL http://www.sciencedirect.com/science/article/pii/S1877050916307517. International Conference on Computational Science 2016, ICCS 2016, 6-8 June 2016, San Diego, California, USA.

14. **Saha, M.**, **P. Mitra**, and **R. S. Nanjundiah** (2017). Deep learning for predicting the monsoon over the homogeneous regions of india. *Journal of Earth System Science*, **126**(4), 54. ISSN 0973-774X. URL https://doi.org/10.1007/s12040-017-0838-7.

15. **Shi, X.**, **Z. Chen**, **H. Wang**, **D.-Y. Yeung**, **W.-k. Wong**, and **W.-c. Woo**, Convolutional lstm network: A machine learning approach for precipitation nowcasting. *In Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15. MIT Press, Cambridge, MA, USA, 2015. URL http://dl.acm.org/citation.cfm?id=2969239.2969329.

16. **Thapliyal, V.** and **S. Kulshrestha** (1992). Recent models for long-range forecasting of southwest monsoon rainfall over india. *Mausam*, **43**, 239–248.

17. **Tripathi, S.**, **V. Srinivas**, and **R. S. Nanjundiah** (2006). Downscaling of precipitation for climate change scenarios: A support vector machine approach. *Journal of Hydrology*, **330**(3), 621 – 640. ISSN 0022-1694. URL http://www.sciencedirect.com/science/article/pii/S0022169406002368.