Indian Statistical Institute

Daily Rainfall Forecasting over West Bengal with Spatio-Temporal Graph Networks

From Deterministic Forecasting to Calibrated Extreme-Risk

A dissertation submitted in partial fulfilment of the requirements for the degree of

Master of Technology in Computer Science

by

Risheek Ghosh

(Roll No. *CS2424*)

under the supervision of

Dr. Sarbani Palit

Computer Vision & Pattern Recognition Unit

Indian Statistical Institute, Kolkata

Indian Statistical Institute

Daily Rainfall Forecasting over West Bengal with Spatio-Temporal Graph Networks

From Deterministic Forecasting to Calibrated Extreme-Risk

A dissertation submitted in partial fulfilment of the requirements for the degree of

Master of Technology in Computer Science

by

Risheek Ghosh

(Roll No. *CS2424*)

under the supervision of

Dr. Sarbani Palit

Computer Vision & Pattern Recognition Unit

Indian Statistical Institute, Kolkata

Abstract

Daily rainfall is hard to forecast where most days are dry, a few days are very wet, and almost all of the rain arrives in one season. This dissertation studies day-ahead rainfall forecasting over West Bengal, India, using models that learn from both the temporal and the spatial layout of measuring stations. We first build up a forecasting model in stages, from simple models that look at one station's past to a graph-based model that links nearby stations, and we identify the strongest deterministic forecaster among them. We then ask whether the choice of input data changes the story, by comparing rain-gauge observations against a reanalysis product on the same stations; the reanalysis turns out to be a smoothed stand-in that misses the sharpest days. Looking closely at the best model, we find it behaves like an estimator of the average rainfall for each day, which is why it cannot place the rare heavy events: almost all of the error comes from a small number of very wet days, and a squared-error objective drives the model to predict too little on exactly those days. We show that self-supervised pretraining and related representation tricks do not move this ceiling. Finally, we change what the model predicts: instead of a single number we predict a full range of possible rainfall values. This probabilistic model is well calibrated and produces useful, reliable warnings for heavy-rain thresholds that the single-number model could never flag, at the modest cost of a slightly worse typical-day error. The contribution is a clear, honest account of where standard spatio-temporal deep learning succeeds and fails for daily rainfall, and a simple change of formulation that recovers useful accuracy on the events that matter most.

Certificate

Thesis title: Daily Rainfall Forecasting over West Bengal with Spatio-Temporal Graph Networks

Name of the student: Risheek Ghosh **Roll No.:** CS2424

Degree for which submitted: Master of Technology in Computer Science

Thesis supervisor: Dr. Sarbani Palit

Month and year of thesis submission: June 2026

It is hereby certified that the work contained in this thesis titled **Daily Rainfall Forecasting over West Bengal with Spatio-Temporal Graph Networks** by Risheek Ghosh has been carried out under my supervision and that it has not been submitted elsewhere for the award of any academic degree, to the best of my knowledge.

Dr. Sarbani Palit
Computer Vision & Pattern Recognition Unit
Indian Statistical Institute, Kolkata

Declaration

I hereby declare that the thesis titled **Daily Rainfall Forecasting over West Bengal with Spatio-Temporal Graph Networks** has been authored by me (Risheek Ghosh). It presents the dissertation work carried out by me under the supervision of Dr. Sarbani Palit.

To the best of my knowledge, this is an original work in both research content and presentation. It has not been submitted elsewhere, in whole or in part, for any degree. All relevant prior work and collaborations have been duly acknowledged and cited in accordance with academic standards.

Risheek Ghosh
Roll No. CS2424
Master of Technology in Computer Science
Indian Statistical Institute, Kolkata

Acknowledgements

I am deeply grateful to my supervisor, **Dr. Sarbani Palit**, for the guidance and support she extended throughout this project. Her patient feedback, technical insight, and steady encouragement shaped this work at every stage, and I consider myself fortunate to have learned under her supervision. I am also thankful to **Dr. Kaushik Jana**, who guided me through many aspects of this project; his suggestions and discussions were a valuable part of this journey.

I am especially thankful to **Prof. Mandar Mitra**, who has helped and guided me throughout my journey at the Indian Statistical Institute. His advice—in both academic matters and general counsel—has been a constant source of reassurance and perspective, and I have grown a great deal under his mentorship.

I would also like to thank my mentor, **Geetakrishnasai Gunapati**, Applied Scientist at Amazon, who guided me during my internship. He was a constant source of support, and I learned a great deal working under his guidance.

Finally, I thank my parents and my friends for their unconditional support, patience, and encouragement, without which this work would not have been possible.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Motivation

A forecast for the next day at the local level can be very helpful in practical scenarios because rainfall plays an important role in farming activities, water resources, and flood hazards in West Bengal. Rainfall on a daily basis poses some difficulties for several traditional models, which makes it a very interesting problem for recent developments in machine learning. There will be days when there will be no rainfall, but on rare occasions, rainfall can take place with significant variation in quantity. Therefore, a model that performs well on most days could still fail on days which matters the most.

1.2 What makes the data hard

These three characteristics of the data will serve as the foundation for all further analysis. Firstly, the model may easily obtain a high average performance with very few forecasts because of the large number of dry days present in the dataset. Secondly, only a small number of days contribute significantly to the rainfall totals and those are the hardest to predict. Thirdly, rainfall is largely seasonal, with the majority falling between June and September.

1.3 Problem statement and research questions

We study one task throughout: given the recent past at every station, predict tomorrow's rainfall at every station. Around this task we ask three questions. Will spatio temporal model predict the daily rainfall more accurately than other models? Does what can be achieved differ depending on the source of the input data—direct gauge measurements (by IMD) versus a reanalysis product (ERA5 Land data)? And

is it possible to forecast heavy-rain events—which the conventional method struggles to handle—in a more reasonable manner?

1.4 Contributions

This dissertation makes the following contributions.

- A like-for-like comparison of forecasting models for daily West Bengal rainfall, from temporal-only models to a graph-based spatio-temporal model, under one evaluation framework.
- A study of how much the input data source matters, comparing gauge observations with a reanalysis product on a shared set of stations, and a characterisation of how the reanalysis differs.
- A diagnosis of why the best deterministic model plateaus: it estimates the average rainfall for each day, and a squared-error objective makes it under-predict the rare heavy days, where almost all of the error lives.
- An honest assessment of self-supervised pretraining and related methods, which we find do not raise this ceiling.
- A change of formulation, from predicting a single value to predicting a full distribution, that gives calibrated and skilful warnings for heavy-rain thresholds at a small cost in typical-day accuracy.

1.5 Organisation of the dissertation

Chapter 2 reviews the relevant literature. Chapter 3 defines the study area, datasets used, and dataset processing steps. Chapter 4 formulates and compares forecast models. Chapter 5 performs evaluation between gauged measurements and reanalysis. Chapter 6 evaluates the generality of the approach for another area. Chapter 7 examines the weaknesses of the optimal deterministic model. Chapter 8 explores whether the weakness of Chapter 7 can be overcome by representation learning. Chapter 9 discusses the probabilistic model formulation. Chapter 10 offers concluding remarks.

CHAPTER 2

Background and Related Work

This chapter reviews the ideas the dissertation builds on, in roughly the order the later chapters use them: the two broad traditions of rainfall forecasting, the sequence and spatial models that make up the deterministic forecaster, learning from unlabelled data, the difference between gauge and reanalysis inputs, and the tools of probabilistic forecasting and its verification. It closes by stating the gap the thesis addresses.

2.1 Forecasting rainfall: physical and data-driven approaches

There are two broad ways to forecast rainfall. First, the numerical weather prediction, which solves the physical equations of the atmosphere forward in time on a grid and is the backbone of operational forecasting , it is strong at short range, but it is computationally very heavy and also struggles with local daily rainfall, especially the convective downpours of the monsoon, which happen at scales finer than the model grid. Second, the data-driven or statistical–machine-learning approaches, which learns patterns directly from historical records without simulating the physics. It is cheap to run and can be skilful at the scale of individual locations, but it is bounded by the quantity and quality of the data it learns from. This dissertation works entirely in the second tradition, at daily resolution and at the scale of individual stations.

Rainfall prediction over India has a long statistical history, much of it aimed at seasonal monsoon totals, and machine learning has been applied to it increasingly over the past decade. Recent examples forecast the Indian summer monsoon with deep networks [1], and predict daily rainfall a few days ahead at Indian locations with hybrid convolutional and fully-connected networks that outperform earlier neural and kernel methods [2]. Closest in setting to the present work is a recent study of West Bengal rainfall [3]. It draws on the same century-long gauge record used here, but

aggregates it in the opposite direction — summing the station readings to district level and then to monthly totals for nineteen districts — and forecasts those monthly series nine years ahead with a hierarchical scheme: yearly summary features (annual totals, seasonal shares, variability, skewness, extremes) are forecast first and then fed, together with a district's own and its neighbours' monthly lags, into multi-layer perceptrons, tuned with expanding-window time-series cross-validation. It shares this thesis's region and source data and informed the district-level framing of the exploratory analysis here, but it solves a different problem — long-range monthly totals by district rather than next-day rainfall at individual gauges — and its spatial component is neighbour-lag features inside a perceptron rather than the graph network used below. By aggregating to monthly district totals it also smooths away the daily zero-inflation and heavy-tailed extremes that are the central difficulty of this thesis.

2.2 Sequence models for time series

A rainfall record at one station is a time series, and the natural tools for time series are recurrent neural networks, which read a sequence step by step while carrying a hidden state that summarises the past. Plain recurrent networks are hard to train over long sequences because the gradients used to learn either vanish or explode. The long short-term memory network [4] solved this with a gated memory cell that can hold information across many steps, and the gated recurrent unit [5] achieved much the same with a simpler, lighter gating scheme. Both are standard choices for hydrological and rainfall time series and appear throughout this thesis. Their limitation, for a network of rain gauges, is that on their own they treat each station as an independent series and ignore the fact that nearby stations rain together.

2.3 Spatial and spatio-temporal models

Capturing space as well as time calls for spatio-temporal models, and there are two main ways to do it. The grid-based approach treats the rainfall field as a sequence of images and learns over it with convolutions; the convolutional LSTM [6] put convolutions inside the recurrent cell and set the template for precipitation nowcasting from radar and gridded data. It is a natural fit for dense grids but less so for rain gauges scattered irregularly across a region.

The graph-based approach tackles the irregular geometry directly: each station

is modelled as a node, and edges connect stations that are close to each other. Graph neural networks then learn, by passing information along these edges. Graph convolutional networks [7] aggregate each node's neighbours with fixed weights, usually uniform, while graph attention networks [8] let the network learn how much weight to give to each neighbour. Pairing such a spatial module with a temporal one — a recurrent or convolutional sequence model — yields a spatio-temporal graph network. This design is used heavily in traffic forecasting, where sensors on a road network pose the same irregular-graph problem as rain stations: influential examples include spatio-temporal graph convolutional networks [9], the diffusion convolutional recurrent network [10], Graph WaveNet [11], and models that learns the graph itself rather than fixing it from geography [12]. The graph-attention-plus-recurrent forecaster used in this thesis belongs to this family.

2.4 Learning from unlabelled data

Self-supervised learning tries to make use of unlabelled data by first training the network on an invented "pretext" task — most commonly, hiding part of the input and asking the model to reconstruct it — and then transferring the learned representation to the real task. This masked-reconstruction idea has been carried to spatio-temporal graphs in a short but active line of work: a pretraining stage with a masked time-series transformer feeding a downstream forecaster [13], a version that masks space and time with two decoupled autoencoders [14], and a generative pretraining scheme for the same setting [15]. In weather, masked autoencoders have been pretrained on reanalysis fields [16] and on radar for precipitation nowcasting [17], though these operate on dense grids rather than gauge networks. One contrasting result cuts the other way: for contrastive (rather than masked) pretext signals, training the pretext task jointly with forecasting has been found to help more than pretraining first and fine-tuning afterwards [18]. Whether any of this helps a daily-rainfall gauge model is an open question, and one this thesis tests directly.

2.5 Gauge observations versus reanalysis

A data-driven model is only as good as its inputs, and there are two very different kinds. Rain gauges measure rainfall directly at the rainfall stations of IMD; they are the ground truth, but the network is sparse and the records have gaps. The

India Meteorological Department's high-resolution gridded daily product interpolates thousands of such gauges onto a regular grid [19]. Reanalysis takes the opposite approach: it blends a physical model with all available observations to produce a complete, gap-free record for many weather variables. ERA5 [20] and its land-surface companion ERA5-Land [21] are the widely used examples and is the source of the reanalysis inputs for our work. The convenience comes with a known catch: in reanalysis, rainfall is a model output rather than a measurement, and it tends to rain too lightly on too many days and to flatten the extreme events — a pattern documented in evaluations of ERA5 precipitation [22]. How much this difference matters for forecasting is the subject of Chapters 5 and 6.

2.6 Probabilistic forecasting and verification

While a point forecast is a one-off estimate, a probabilistic forecast gives us a distribution, which, obviously, is a much truer depiction in cases where there are uncertainties and fat tails involved. An easy way to produce a distribution is through quantile regression using pinball loss function [23]; it will be used in Chapter 9. Naturally, distribution forecasts require a different measure of assessment as well. The proper scoring rules form the basic structure: the continuous ranked probability score measures the entire forecast distribution against the observation [24, 25]; while the Brier score makes a comparison, for a yes/no event [26], which is normally expressed as a skill score based on a reference forecast. Similarly, calibration checks the integrity of the assigned probabilities by the probability integral transform [27, 28] and reliability plots. For rare events, categorical scores such as the probability of detection, false-alarm rate, and critical success index judge the quality of yes/no warnings issued at a particular threshold [29].

2.7 Summary and the gap addressed

Three points can be clearly seen from this survey. First, spatio-temporal deep learning techniques have reached maturity in terms of grid-based nowcasting and traffic-sensor graphs but still have not reached maturity for daily station-level rainfall prediction and for the comparison of different options on a single regional data set. Second, although self-supervised pre-training holds great potential for spatio-temporal graphs, few studies are available regarding gauge-scale daily weather with unclear advantages

of such pre-training in that case. Finally, although reanalysis precipitation is globally limited, its effect for daily station-scale rainfall prediction and how that is affected by the region's terrain have not been adequately analyzed before.

This dissertation thus aims to:

(i) compare spatio-temporal models for daily rainfall prediction in West Bengal with a single framework

(ii) extend the comparison between gauge-scale data and reanalysis data for different regions with varying terrains

(iii) assess self-supervised pretraining and joint training within this framework

(iv) identify what makes the best deterministic model plateau on days with heavy rainfalls; and

(v) reformulate the heavy-rainfall problem into risk-based probability estimation.

CHAPTER 3

Data and Study Region

3.1 Study region: West Bengal

West Bengal lies in eastern India and spans a range of settings, from the Himalayan foothills in the north, through the Gangetic plains, to the coast in the south. The rainfall in this region occurs due to the southwest monsoon from June through September, when most of the annual rainfall occurs. Because of the strong seasonal cycle and the diverse topography in the area, the state becomes an ideal testing ground for models that are capable of handling both the calm dry season and the wet monsoon season.

3.2 Gauge observations

In this study, the data source used is the daily rainfall data collected by the India Meteorological Department (IMD). Rain gauges are the ground instruments used to measure the amount of rainfall received within a specified area; hence, rain gauges provide the most reliable form of data when it comes to measuring rainfall. In this case, 293 gauge locations in West Bengal have been considered from 1901 to 2021.

The data is significantly sparse, and the level of sparseness dictates the approach taken during the following analysis. In comparison with a full dataset of all stations in all days, the missing data accounts for 69% of the total observations. In case the period after 1970 is considered, the number of missing observations becomes 74%. The number of stations reporting rises and falls across the century and drops sharply at the very end of the record (Figure 3.1), and a station-by-year view shows that most stations report only for limited stretches, with whole years blank rather than odd days here and there (Figure 3.2). Taken station by station, the majority are missing well over half of the full period (Figure 3.3). This block-structured sparsity is the reason the gauge network cannot, on its own, supply a complete daily map for the

main modelling work, so that work instead uses the gap-free reanalysis introduced next (Section 3.3). The gauge record's other role comes in Chapter 5, where it is the yardstick for testing how faithful that reanalysis really is.

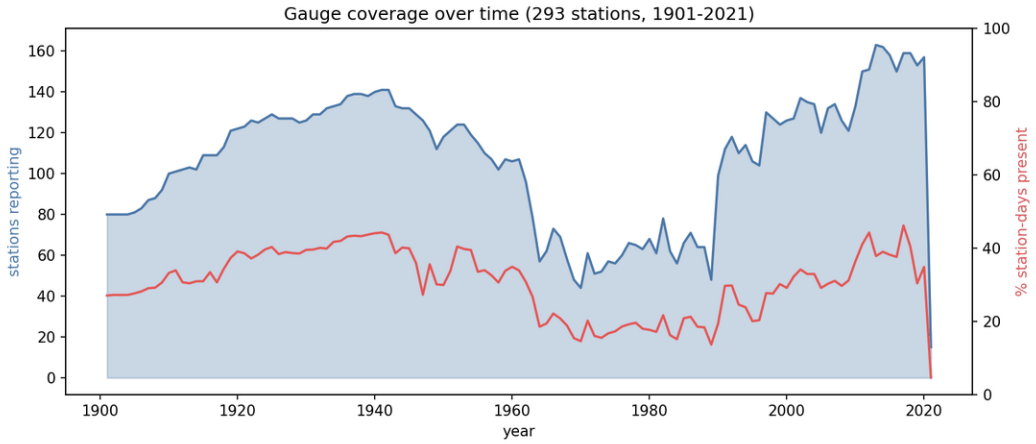


Figure 3.1. Gauge coverage over time.

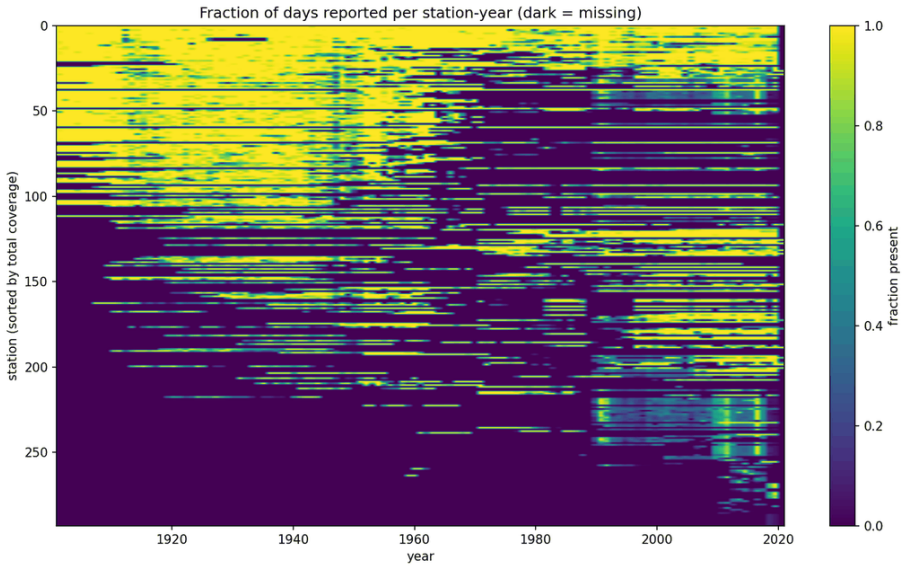


Figure 3.2. Fraction of days each station reports in each year.

3.3 Reanalysis data

The second source is the ERA5-Land reanalysis, a gridded weather record produced by combining a physical model with observations [21]. We sample it at the station locations, which gives a complete daily series at 291 of the 293 sites (two coastal points fall outside the land grid). For each station and day we take six variables: total

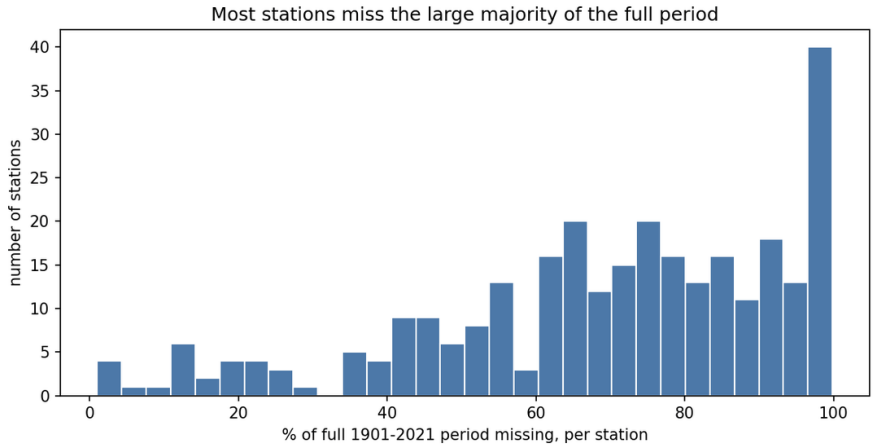


Figure 3.3. Distribution of per-station completeness.

rainfall, air temperature and dew point near the surface, surface pressure, and the two horizontal components of the wind. The series runs from 1970 to 2023. Because it is complete and covers a long span, this is the data used to develop the spatio-temporal models in Chapter 4.

3.4 Preprocessing and quality control

The raw downloads are first put into consistent units: rainfall in millimetres, temperatures in degrees Celsius, pressure in hectopascals. We then lay out each weather variable as its own table indexed by date and station. Reading across a single row gives every station's value on one day; reading down a single column gives the full day-by-day series for one station.

A few entries are still missing after this step, and we fill them in a fixed order so that the model never meets a hole in the table. For a missing value we first copy the most recent earlier reading from that same station. If there is no earlier reading to copy, we use the next available later one. If a station has no usable reading in that stretch at all, we fall back to that variable's average. The result is a complete table for every variable. Importantly, the averages and the scaling figures used in this process are computed from the training period only, so nothing about the validation or test years leaks into the inputs (Section 3.6).

3.5 Building the station graph

To let the model share information between nearby places, the stations are treated as a graph: each station is a node, and two stations are joined by an edge when they lie within 100 km of each other, measured as great-circle distance from their coordinates. Because the gauges are closely spaced and a 100 km circle covers a fair amount of ground, the resulting network is dense — on average each station is joined to about 62 others, and no station is left without neighbours. This lets a node draw on a wide surrounding area rather than just its single closest gauge. The same graph is reused by every graph-based model, so differences in results reflect the model and not the wiring.

3.6 Splitting and scaling

There is no randomness involved in the splitting of data but is done in temporal fashion. The training, validation, and testing split of data are performed in the ratio of 70:15:15 for each station. This helps reduce any problems related to data leakage. After performing data partitioning, normalization is conducted using the mean and standard deviation obtained from the training dataset. The model then outputs a scaled rain value which we rescale to millimeters per day to make the downstream analysis easier.

3.7 Exploratory data analysis

Before modelling, we summarise the reanalysis rainfall at the station locations, grouped by district, with a small set of plain plots. The district-level framing of this summary, and several of the views below, follow the exploratory approach of recent work on West Bengal rainfall [3].

Coverage is uneven across districts, which is worth remembering when reading any spatial result (Figure 3.4). The seasonal signal is strong and concentrated: average rainfall climbs steeply into the June-to-September monsoon and drops away outside it, so the monsoon months dominate both the totals and the difficulty (Figure 3.5). A district-by-month view of the same pattern makes the split between the wet and dry halves of the year easy to read at a glance (Figure 3.6).

Year-to-year totals move around a fairly stable long-term level. Annual totals are

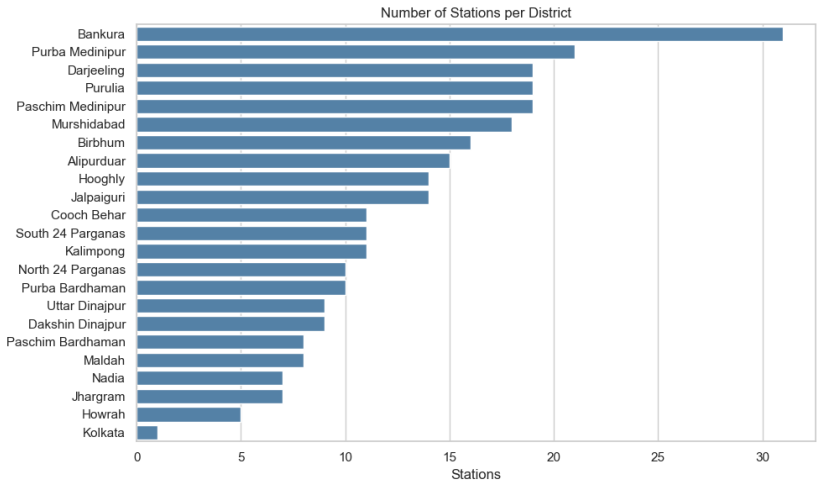


Figure 3.4. Number of gauge stations per district.

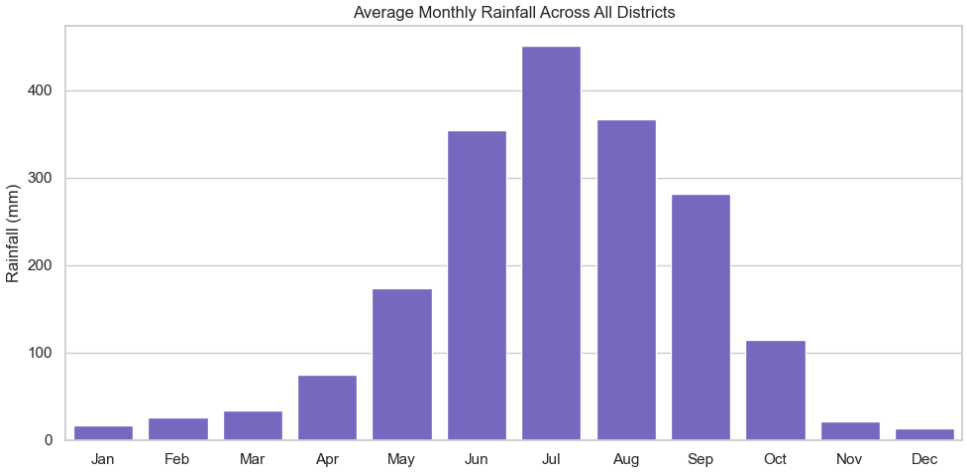


Figure 3.5. Monthly rainfall climatology, showing the June–September peak.

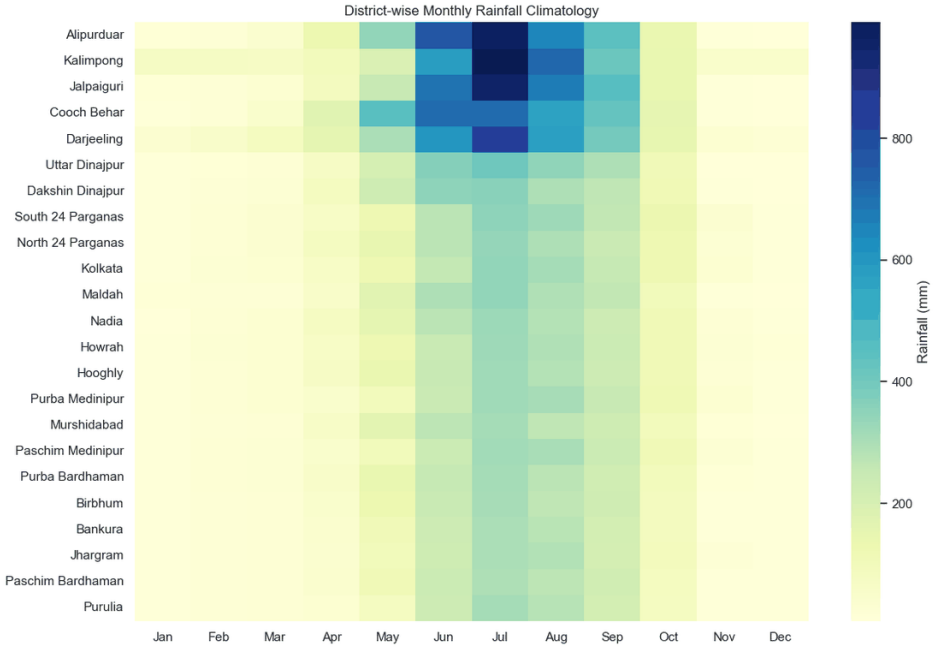
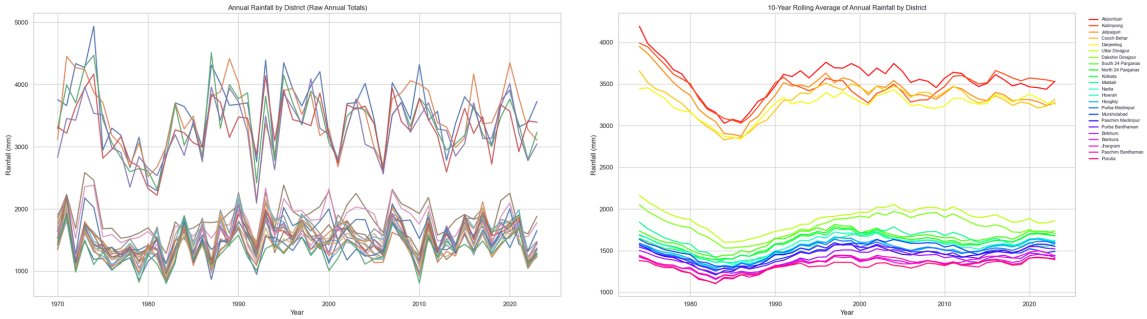


Figure 3.6. District-by-month rainfall heatmap.

noisy from one year to the next (Figure 3.7(a)), but a ten-year moving average shows no strong long-run trend across 1970–2023 (Figure 3.7(b)).



(a) **(b)**

Figure 3.7. **(a)** Annual rainfall totals by district. **(b)** Ten-year rolling mean of annual rainfall, showing no strong long-run trend across 1970–2023.

Districts do not behave independently. Monthly rainfall is strongly correlated between districts, especially neighbouring ones, which is exactly the spatial structure the graph-based models are meant to use (Figure 3.8). Mapping the long-term average rainfall across the state shows where the wetter and drier districts sit (Figure 3.9). Breaking the record into decades shows how that pattern has shifted over time (Figure 3.10). Mapping the share of rain that falls in the monsoon underlines how seasonally concentrated the rainfall is right across the state (Figure 3.11).

Finally, counting the unusually wet and unusually dry years at each district gives

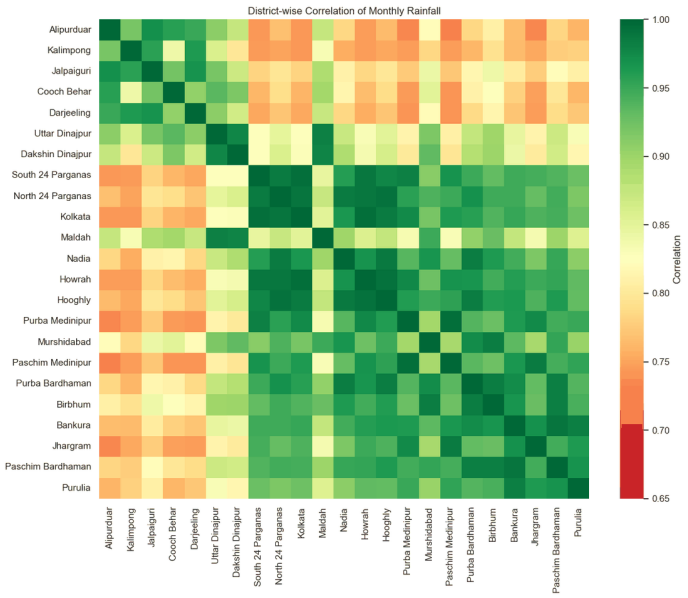


Figure 3.8. Between-district correlation of monthly rainfall.

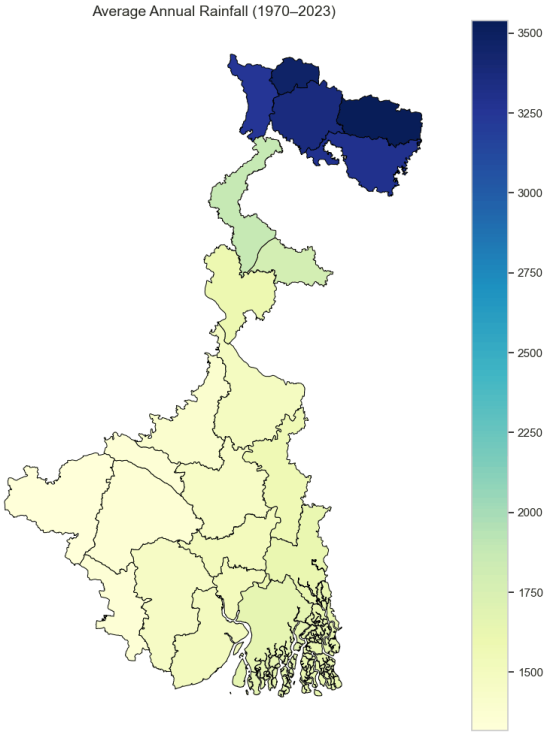


Figure 3.9. Average annual rainfall by district.

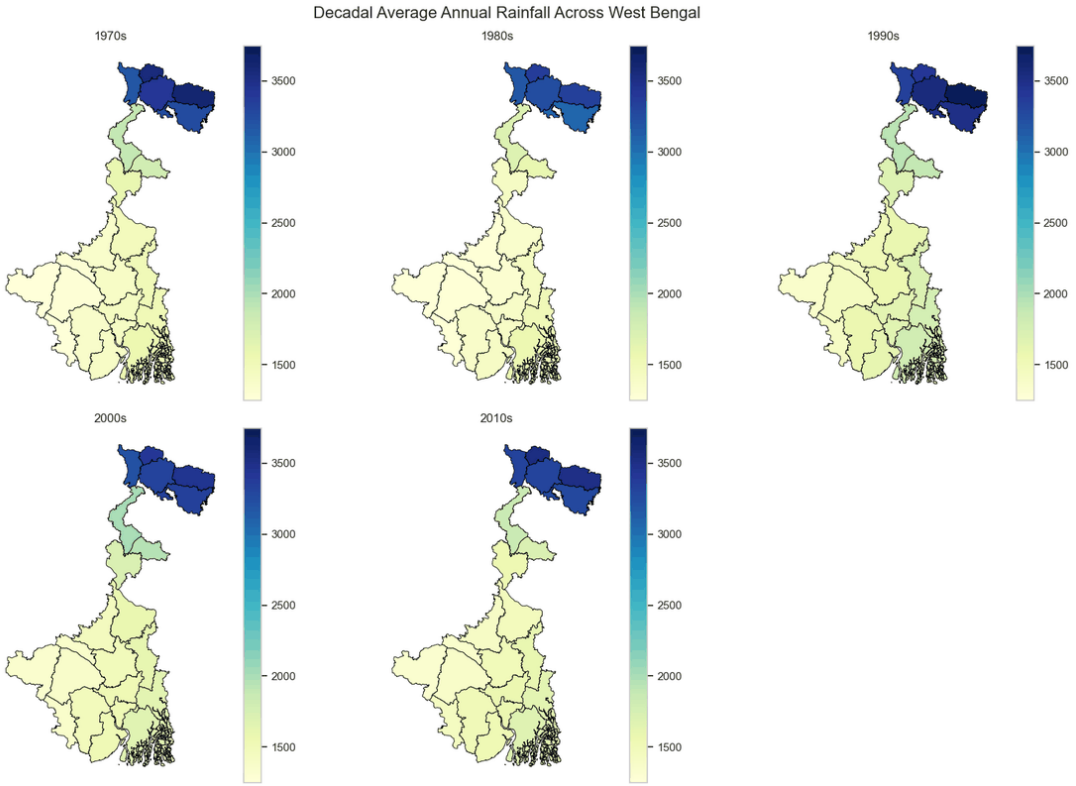


Figure 3.10. Decade-by-decade rainfall maps, 1970s to 2010s.

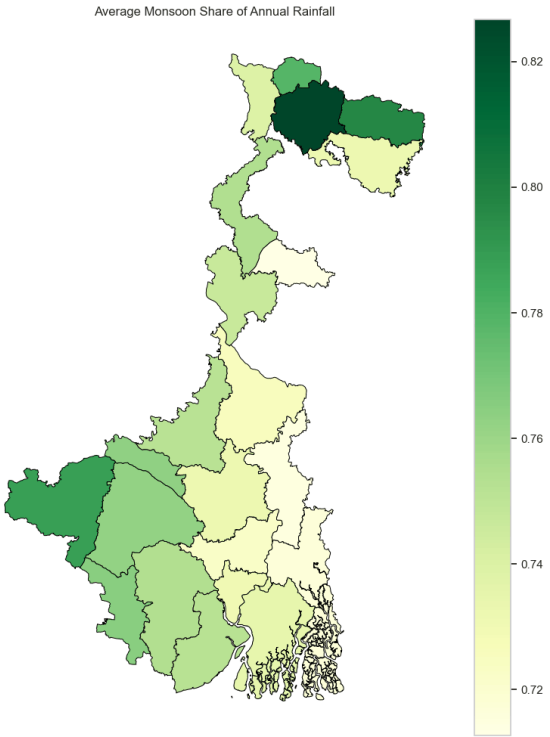
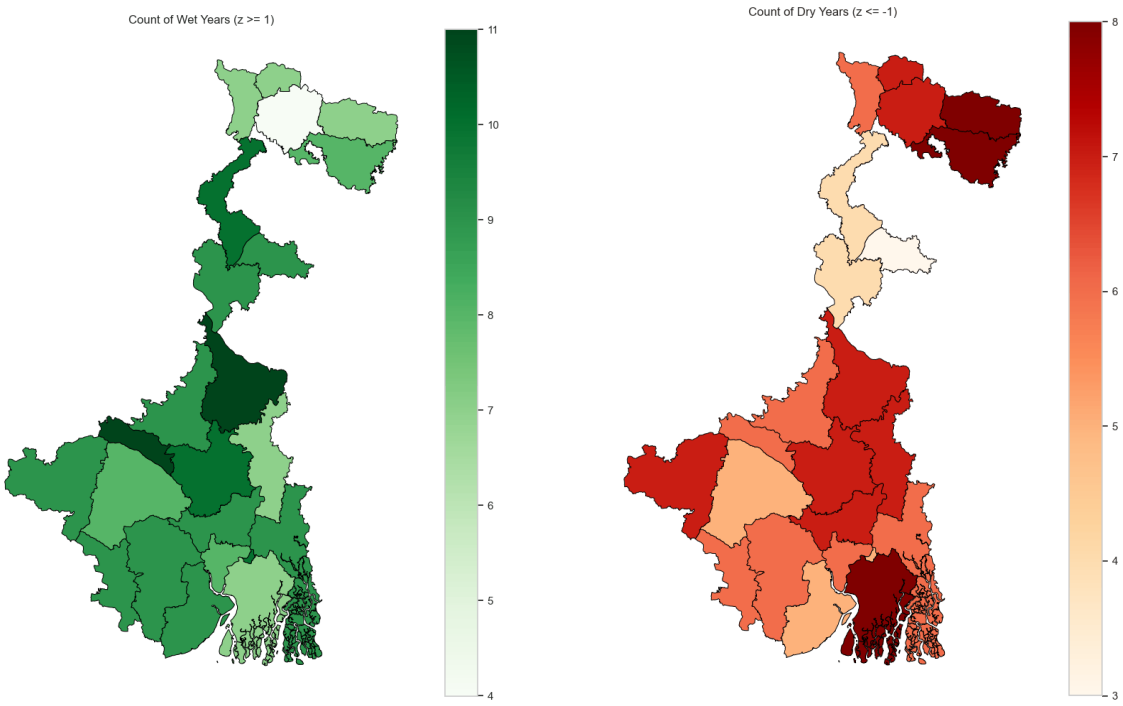


Figure 3.11. Monsoon share of annual rainfall by district.

a simple picture of how variable the climate is from one year to the next (Figure 3.12).



(a) **(b)**

Figure 3.12. Counts of unusually wet **(a)** and unusually dry **(b)** years by district.

Two points from this analysis carry into the rest of the thesis: the rain is concentrated in the monsoon, and nearby places move together. The first says where the hard forecasting cases are; the second says why a spatial model should help.

CHAPTER 4

Comparative Spatio-Temporal Modelling

This chapter builds the forecasting model in stages and picks out the strongest deterministic forecaster among them. We start with models that look only at a single station's past, then add spatial structure, and finish with models that treat the whole network of stations as a connected graph. Two things come out of the comparison: how far each kind of model can go, and where even the best of them stops improving. That stopping point is the thread the later chapters pull on.

One caveat belongs up front. The earlier models in this chapter were developed on the rain-gauge record, while the graph-based models use the gap-free reanalysis from Chapter 3. The reanalysis is smoother and easier to predict, so part of the drop in error from the early models to the later ones comes from this change of input, not from the model. The fair model-versus-model comparison is therefore between models that share the same data, and the separate question of how much the data source itself matters is the subject of Chapter 5. We point this out again when reading the results.

4.1 Forecast set-up and evaluation metrics

The task is the same for every model: read the recent past at every station and predict the next day's rainfall at every station. Each model is given a seven-day window of inputs and asked for a one-day-ahead forecast. The data preparation, the time-based split into training, validation, and test parts, and the station graph were all described in Chapter 3; here we fix only the prediction task and the way results are scored.

Every model is scored on the test period, in millimetres per day, after its predictions are converted from the standardised scale back to real units. Four numbers are used throughout [29]. The root mean squared error is defined as the square root of the

average of the squares of the differences between the observed and predicted data. This particular measure places a high penalty on large differences, making it ideal for situations where a large error is very costly. On the other hand, the mean absolute error measure takes into account the absolute differences. The bias is the average of prediction minus observation, so a positive value means the model rains too much overall and a negative value too little. The normalised root mean squared error divides the root mean squared error by the average observed rainfall, giving a unitless figure for comparing places or seasons with different rainfall amounts. We also report the root mean squared error for the monsoon months (June to September) on their own, because that is when most rain falls and when the hardest cases occur. Writing p for a prediction, o for the matching observation, and $\bar{o}$ for the mean observation:

$$\mathrm{RMSE} = \sqrt{\mathrm{mean}((p-o)^2)}, \qquad \mathrm{MAE} = \mathrm{mean}(|p-o|),$$

$$\mathrm{Bias} = \mathrm{mean}(p-o), \qquad \mathrm{NRMSE} = \frac{\mathrm{RMSE}}{\bar{o}}.$$

4.2 Temporal models

The first model is a long short-term memory network, a recurrent network that reads the input sequence one step at a time and maintains a hidden summary of what it has seen so far. We treat forecasting as sequence prediction i.e from the recent window of inputs we predict the next day's rainfall. At each step the network updates its hidden state and reads out a value,

$$h(t) = \mathrm{LSTM}(x(t),\, h(t-1)), \qquad \hat{y}(t+1) = W \cdot h(t) + b,$$

and it is trained to minimise the mean squared error between the prediction and the observed rainfall. The model runs on each station on its own, using that station's recent rainfall together with the calendar signals (Figure 4.1).

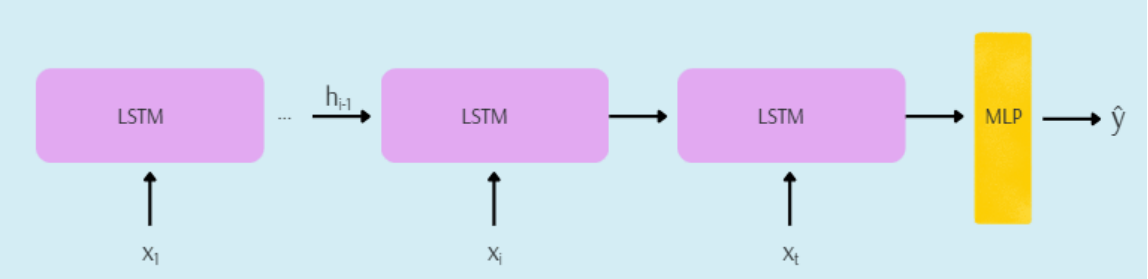


Figure 4.1. The baseline long short-term memory model, unrolled across the input window.

This temporal-only model reaches a test error of 15.82 mm/day. Its overall bias is almost zero, but its error inside the monsoon (23.03 mm/day) is far larger than outside it (8.72 mm/day) — the first sign of the central problem: the model is comfortable in the dry season and struggles when it actually rains.

We then asked whether increasing the context window length would help improving the predictions. A seven-day window gives 15.90 mm/day, and stretching the single window to fourteen days changes the validation error by less than a tenth of a millimetre, so we keep a compact seven-day window (the point is revisited with the multi-scale model below).

Because the model does well on light rainy days and poorly on heavy ones, we next updated the training objective to penalise large errors more heavily. The focal-weighted squared error scales each squared error by a factor that grows with the error size,

$$L_{\text{focal}} = \left(1 - e^{-|e|}\right)^{\gamma} \cdot e^2, \qquad \text{with } e = \hat{y} - y \text{ and } \gamma = 2,$$

so small, confident misses are discounted and large misses are emphasised. On its own this barely moves the result, to 15.81 mm/day. A seasonal version multiplies the same term by a weight that is larger in the monsoon,

$$L_{\text{seasonal}} = w(\text{month}) \cdot \left(1 - e^{-|e|}\right)^{\gamma} \cdot e^2, \qquad \text{with } w = 3 \text{ for June–September and 1 otherwise,}$$

giving 15.84 mm/day. We used the log of rainfall to handle the long tail, which in turn raised the error to 16.98 mm/day, though it lowered the average-sized error and tipped the model toward under-prediction (bias −3.63 mm/day). No change of the loss moved the headline numbers; the temporal model stayed near 15.8 mm/day.

Finally, since storms have multiple phases : a quick build-up, a sustained phase, and a slow decay, we gave the model multiple context windows of varying sizes. The multi-scale model runs a separate recurrent network over four windows — three, seven, fourteen, and thirty days ; this reached 15.83 mm/day, no better than the baseline models. We also added attention on top. Within each scale, a temporal attention layer weights the days for that scale,

$$\alpha(t) = \text{softmax}(w_s \cdot H(t)), \qquad z = \sum_t \alpha(t) \cdot H(t),$$

and a cross-scale attention layer then weights the four scale summaries,

$$\beta(s) = \mathrm{softmax}(w \cdot z(s)), \qquad z_{\text{final}} = \sum_{s} \beta(s) \cdot z(s), \qquad \hat{y} = W \cdot z_{\text{final}} + b,$$

but this did not lower the error either. The lesson is consistent across all the experiments here, giving the temporal model more of the past, or a cleverer way to read it, does not help. Chapter 7 explains why — the error is concentrated in rare heavy days that no length of history can resolve.

4.3 A grid-based model

Temporal models handle each weather station alone. The next model brings in spatial info by turning daily observations into picture form. They take daily rainfall data from stations and use linear interpolation to fill a 0.25° by 0.25° grid. At the edges, they fill gaps with the nearest neighbor method. This makes a bunch of daily rainfall image stacks.

The grid values under 2.5 mm get set to zero according to the India Meteorological Department's rules for a rainy day. By doing this, the model focuses on more significant rain amounts instead (Figure 4.2).

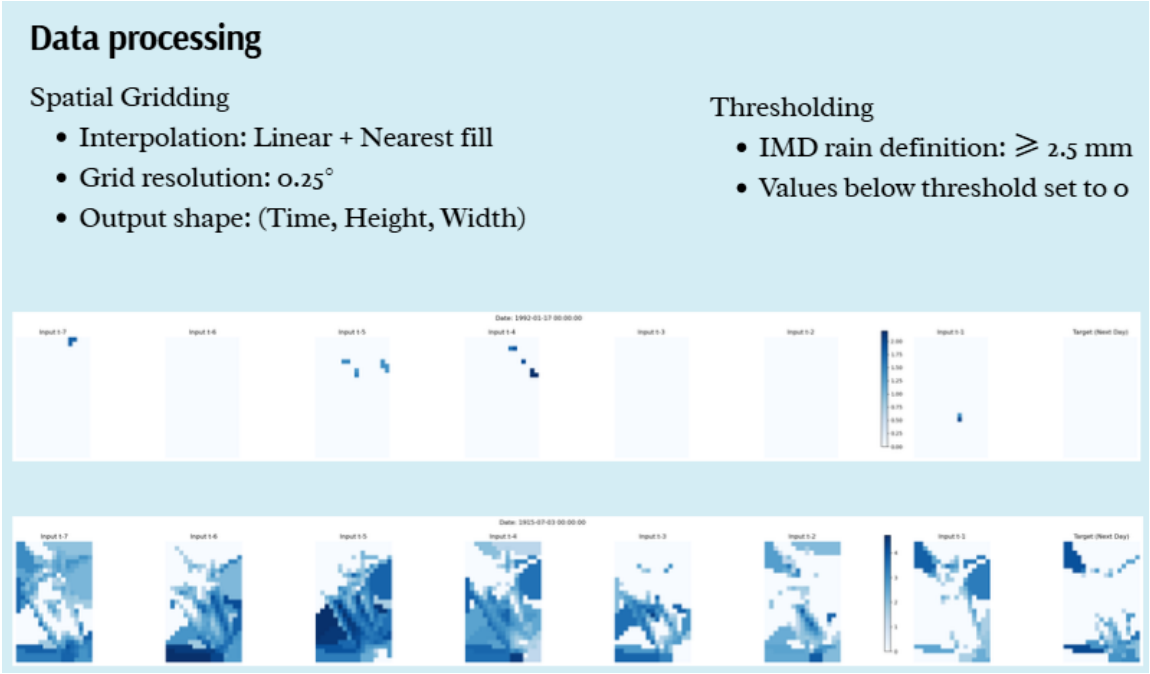


Figure 4.2. Interpolating the station readings onto a 0.25° grid and applying the 2.5 mm rainy-day threshold.

The model is a convolutional recurrent network. A convolutional backbone scans each day's grid the way an image network does, so the forecast for a cell can draw on the rainfall in the cells around it; a recurrent layer then carries this spatial summary across the seven days of the window; and a final layer reads out the next day's rainfall (Figure 4.3). Adding space this way clearly helps: with a loss that up-weights rainier cells, the model reaches 13.37 mm/day, a real step below the per-station temporal models at about 15.8 mm/day.

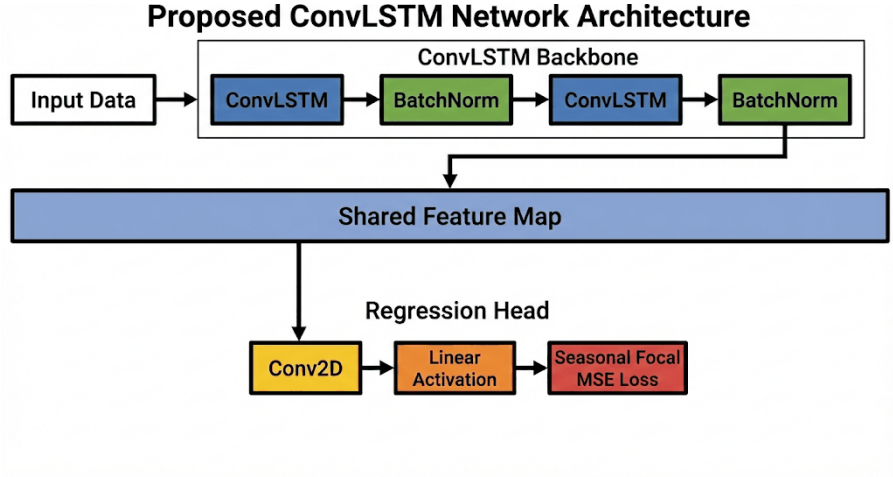


Figure 4.3. The grid-based convolutional–recurrent architecture.

The gain comes with two caveats, both of which matter later. First of all, there is a strong negative bias of −3.22 mm/day in the forecast model, resulting in low precipitation prediction. The rest of the thesis tries to address the same. Second, interpolating scattered gauges onto a grid smooths away sharp local peaks, so a grid model is structurally limited in how well it can ever place the heaviest, most localised downpours.

4.4 A classification-style model

In the last model discussed in this chapter, forecasting objective is redefined i.e. we are doing regression via classification. Instead of predicting a specific amount of rainfall, it forecasts which intensity band the next day's rain will fall into.

We select the bands from the rainfall distribution. The widths aren't uniform; they get much narrower when dealing with heavier rains. This is key because, although heavy rain is rare, the ability to flag it is super important.

For bands, we use one for zero-rainfall and then on every 5 percentiles up to the 75th percentile. Then the increments become more detailed – every 1 percentile

between the 76th and 94th, then at every 0.5 percentile from the 94.5th to the 99.5th. Lastly, one overflow band for values above 99.5 percentile.

This sums up to a total of 48 bands, mostly zoomed in on the upper end. The reasoning? To make sure a day with 60mm of rain isn't lumped together with a day exceeding 200mm – simply labeled as "heavy rain." These finer bands help distinguish among different levels of high rainfall amounts.

A causal transformer — a sequence model in which attention lets each day look back at the days before it — reads the sequence of past intensity bands and predicts the band for the next day, trained with the usual cross-entropy classification loss (Figure 4.4). To compare it with the other models, the predicted band middle value is used to map back to a representative rainfall amount.

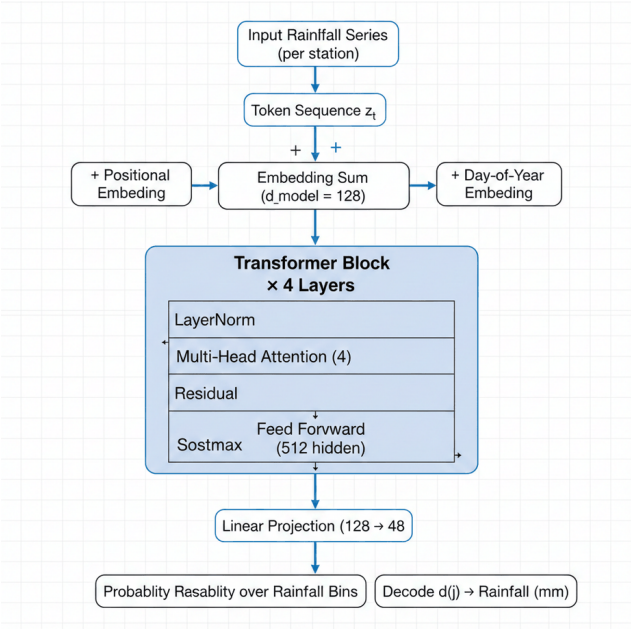


Figure 4.4. The transformer that predicts the next rainfall band.

This performs worst of all, at 18.53 mm/day, with a large dry bias (−4.94 mm/day) and the highest monsoon error (28.15 mm/day). Two things work against it. Turning a smooth quantity into bands discards information about where inside a band the true amount lies, so even a perfectly predicted band still carries error. And despite the fine upper bands, the heavy-rain classes hold too few examples to learn well, so the model leans on the common low bands and rarely commits to a heavy one. We did not take this direction further.

4.5 Graph-based models

The strongest models leave the stations as they are and connect them directly, using the 100 km graph from Chapter 3. From here on we use the gap-free reanalysis at the 291 station locations, which removes the missing-data problem that made graph training awkward on the raw gauge record.

Two graph models are compared. The first combines graph convolution — which blends each station's signal with its neighbours' — with a recurrent layer that carries the result across days, and reaches 8.99 mm/day. The second swaps plain graph convolution for graph attention, which lets a station decide how much weight to give each neighbour rather than treating them all alike. This is the best deterministic model in the study, at 8.79 mm/day (mean absolute error 4.00, bias +0.25). Running it again through a corrected version of the pipeline gives essentially the same result, 8.78 mm/day, which we use as the reference point for the later chapters. Because both graph models use the same data, the gap between them is a clean architecture comparison: letting the model weigh its neighbours helps, but only a little.

Plotting a graph model's predictions against observations shows the pattern they all share (Figure 4.5(a)). The points are clustered in the lower left corner and the model predicts upto 130 mm only, which shows that the model performs well in ordinary days but falls short on heavy ones. The error is also spatially uneven, sitting highest over the wetter northern districts (Figure 4.5(b)).

4.6 Results and comparison

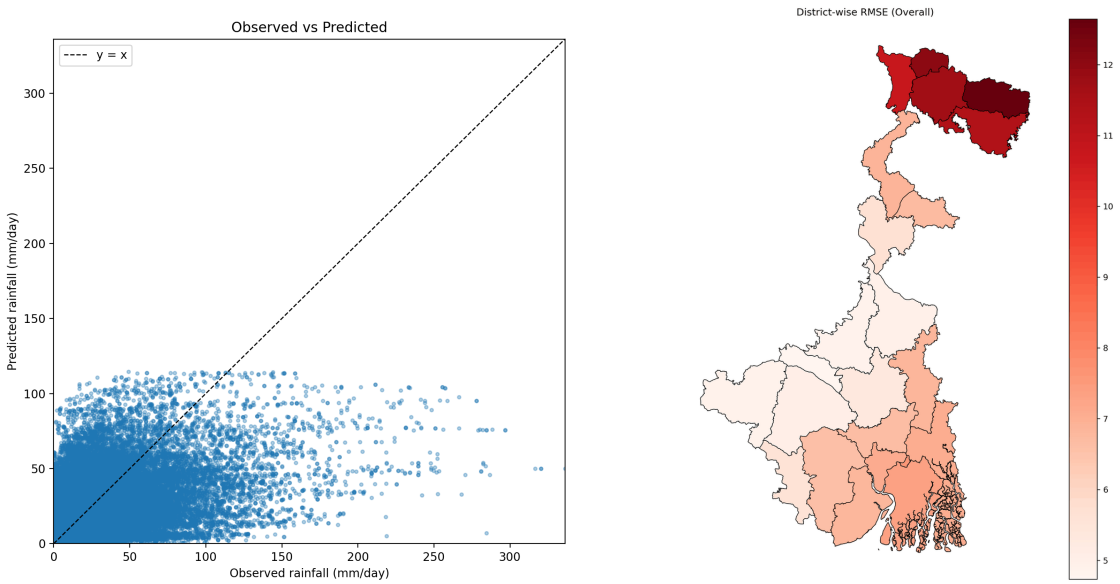
Table 4.1 collects the test-set results, and Figure 4.6 shows them at a glance.

4.7 Discussion

The graph attention model is the best deterministic forecaster here, at about 8.8 mm/day (8.79 in the table). The diagnostic and probabilistic chapters re-run the exact same model within the standard evaluation process used starting in Chapter 6,

Two main things from the table stand out:

First, the better performance seen as time goes on in the earlier models wasn't just about having a fancier structure. Transitioning from gauge data to reanalysis data made a big difference too, something Chapter 5 talks more about.



(a) **(b)**

Figure 4.5. **(a)** Observed versus predicted daily rainfall, graph-convolution model. **(b)** District map of test error, graph-convolution model.

Table 4.1. Test-set results, in millimetres per day except the unitless normalised error. The two graph models use reanalysis input; the others use gauge data, so the large step down to the graph models reflects both a better model and an easier, smoother target (Chapter 5). Within the same data, graph attention edges out plain graph convolution (8.79 against 8.99). The grid model's mean absolute and normalised errors were not re-extracted for this table.

Model	Input data	RMSE	MAE	Bias	NRMSE
Transformer (classification)	gauge	18.53	6.00	−4.94	3.17
LSTM + log-transform loss	gauge	16.98	5.66	−3.63	3.03
LSTM + seasonal-focal loss	gauge	15.84	6.98	+0.25	2.83
Multi-scale LSTM	gauge	15.83	6.90	+0.05	2.83
LSTM (baseline)	gauge	15.82	6.87	+0.16	2.83
LSTM + focal loss	gauge	15.81	6.63	−0.27	2.83
ConvLSTM (grid)	gauge	13.37	—	−3.22	—
GCN-GRU (graph)	reanalysis	8.99	4.08	+0.17	1.67
GAT-GRU (graph)	**reanalysis**	**8.79**	**4.00**	**+0.25**	**1.63**

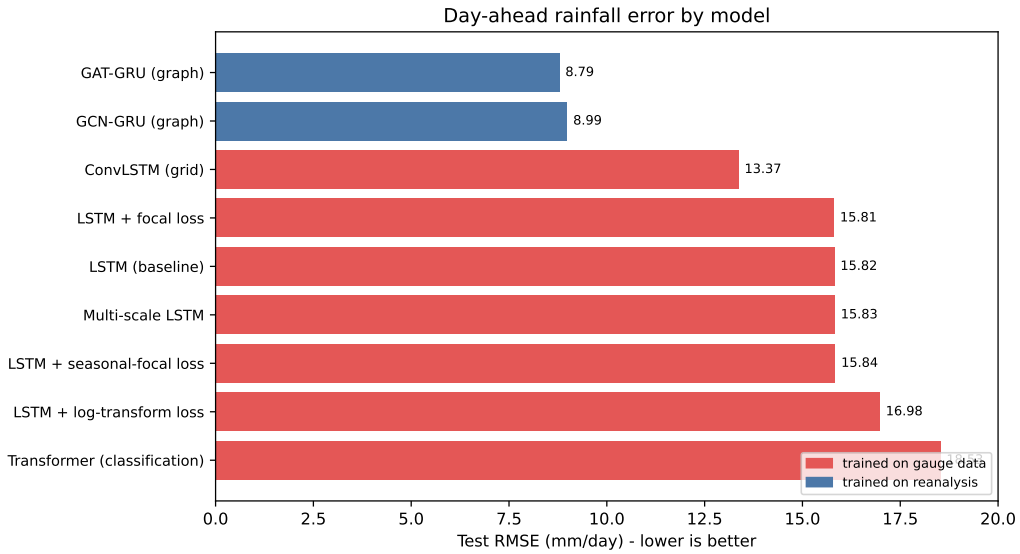


Figure 4.6. Day-ahead rainfall error by model, coloured by data source.

Secondly, and most importantly, all the models in the table – whether they're simple or complex, gauge-based or reanalysis-based – have this common problem. While they handle regular days well, they fall flat on what the researchers call "heavy-load days." Furthermore, some models mitigate this problem by making small predictions that help them avoid the tough cases, hence creating a negative bias.

So, the rest of the thesis dives into figuring out why this happens and how to fix it. In Chapter 7, we tear apart the top-performing model to try to discover the root cause. Then, in Chapters 8 and 9, they take two separate shots at solving the problem.

CHAPTER 5

Gauge versus Reanalysis

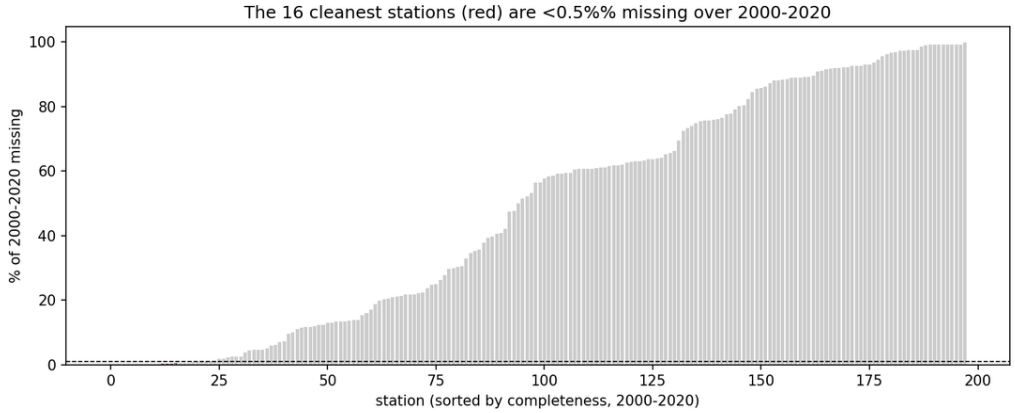
5.1 Does the data source change the story?

Chapter 4 left an open question to be answered. The best-performing graph models were trained on reanalysis data rather than the rain gauges, and we cautioned that an advantage could result from easier data rather than stronger models. This chapter settles that question. Plus, figuring this out is super important because a forecasting system's usefulness hinges on the rainfall record it's built from. We especially want to know if reanalysis data is better, despite being less accurate to actual gauge readings. To find out, we take two routes: one involves a forecast test using the same set of weather stations on both datasets, while the other directly compares the records station by station.

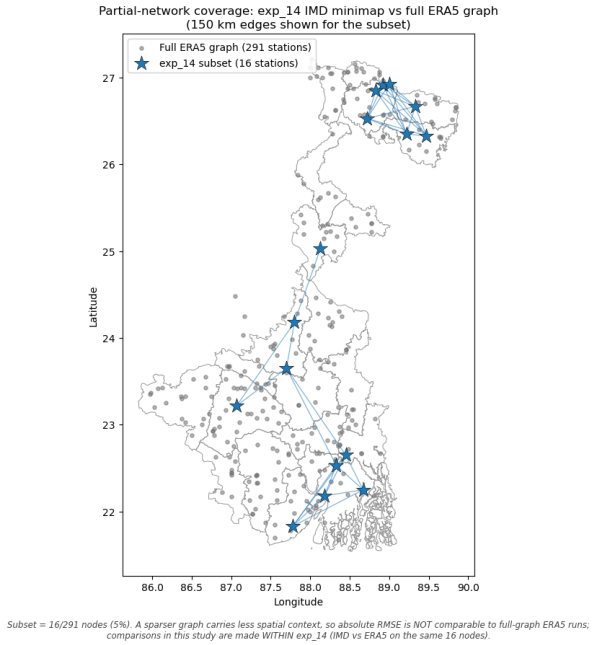
5.2 A matched experiment on a shared station subset

To fairly compare data from different sources, all other factors need to stay the same. However, the sparsity of gauge records is a big issue here. For a thorough comparison, we need the stations that are almost entirely observed during the same time period on both gauge and reanalysis data. The most thoroughly observed period goes from 2000 to 2020. Even then, only about two dozen stations report data on more than 99% of days within that time frame. We take the sixteen most complete of these — each missing under half a percent of the window, together spanning ten districts — which is enough nodes to build a meaningful graph while keeping the data essentially gap-free (Figure 5.1(a)). On this subset we train the same graph-attention model, with the same settings, three times: once on the gauge rainfall, once on the reanalysis rainfall at those same sixteen stations, and once on the reanalysis with its full set of

weather variables. Only the input data changes, so any difference in accuracy is a property of the data, not the model (Figure 5.1(b)).



(a)



(b)

Figure 5.1. **(a)** Selecting the matched subset: per-station completeness over 2000–2020, with the sixteen cleanest stations highlighted. **(b)** The network of stations used for comparison.

The result is stark. The model shows a test error of 17.60 mm/day on gauge data. But when using matched reanalysis, the error drops to 10.42 mm/day. That's about a 40% reduction. It's for the same stations, window, and model. Adding the reanalysis' extra weather variables barely helps more, reducing the error to 10.03 mm/day. This suggests that at a one-day forecast, recent rain data is way more important than other physics info.

Both models beat simple guess methods too. The persistence model, giving yesterday's value (error: 22.53 mm/day) and predicting based on the day of the year i.e. Climatology model (error: 18.50 mm/day). So, there's real value in using these models.

But why's the gauge data harder? Well, actual rainfall hits upto 350 mm in a day. Yet, the model heavily underpredicts, i.e. 70 mm max, missing those big events. It smooths out the extremes and predicts the average a lot. With reanalysis data, the targets have softer peaks, which the model nails better. This isn't about the model failing; it's that the actual rain records are naturally tougher to predict. This under-prediction of heavy days is the central limitation of the whole approach, and Chapter 7 examines it in detail for the best model.

Why is the gauge record harder? Because the reanalysis is smoother in time. Day to day, reanalysis rainfall is about twice as self-correlated as gauge rainfall (lag-one autocorrelation 0.56 against 0.31), its spread is narrower (standard deviation 13.7 against 19.5 mm), and it calls many more days wet (52% dry against 71% for the gauges) — even though the two share almost the same long-run mean of about 6.4 mm/day (Figure 5.2).

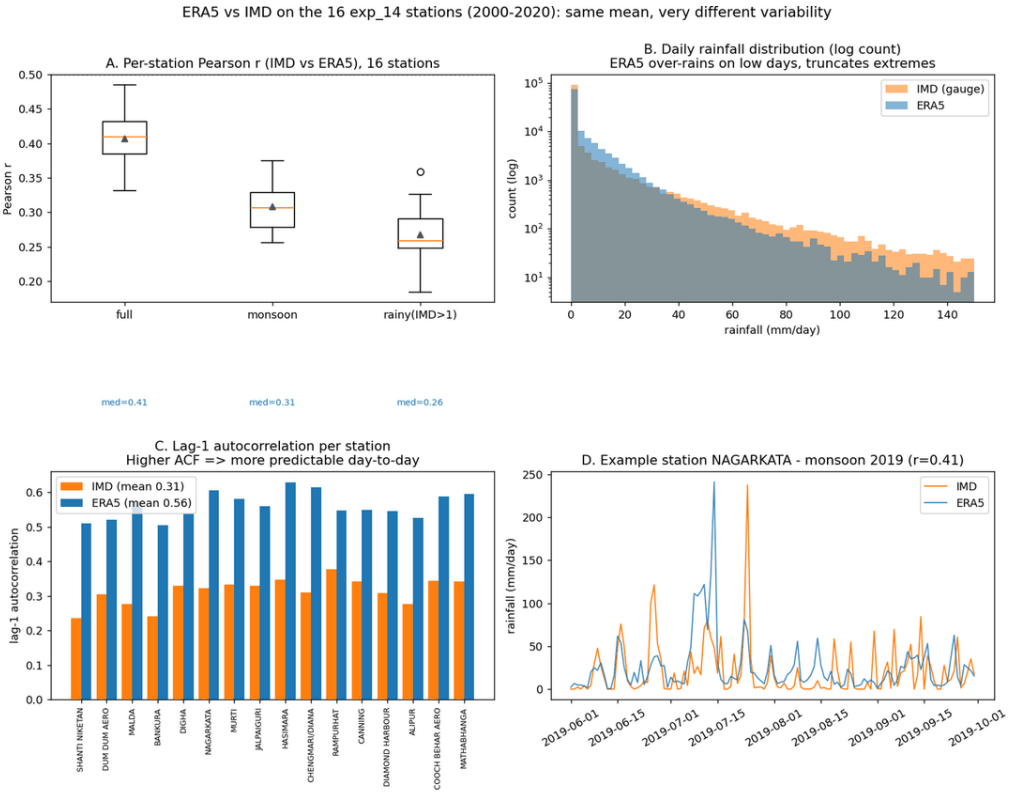


Figure 5.2. How the reanalysis differs from the gauges: correlation by season, day-to-day persistence, and wet-day frequency.

5.3 Station-by-station agreement

The matched experiment compares the data sources through a model. We directly compare the two as well. For each station — totaling 291, with 199 having at least a year of daily data overlap — we align the gauge and reanalysis series on overlapped days, without ever filling in gaps. At each station, we figure out how well they match, their average difference, and which one more often flags a day as wet. We run these checks overall, just during monsoons, and solely on rainy days.

The agreement is moderate at the daily level but gets weaker when we remove the simpler dry days. The median station correlation is 0.37 over the full series, falling to 0.29 during monsoon and 0.24 on rainy days (Figure 5.3). Once we restrict attention rainy days only, the two records agree weakly. Yet their averages line up almost exactly: the median bias between them is about −0.04 mm/day, essentially zero.

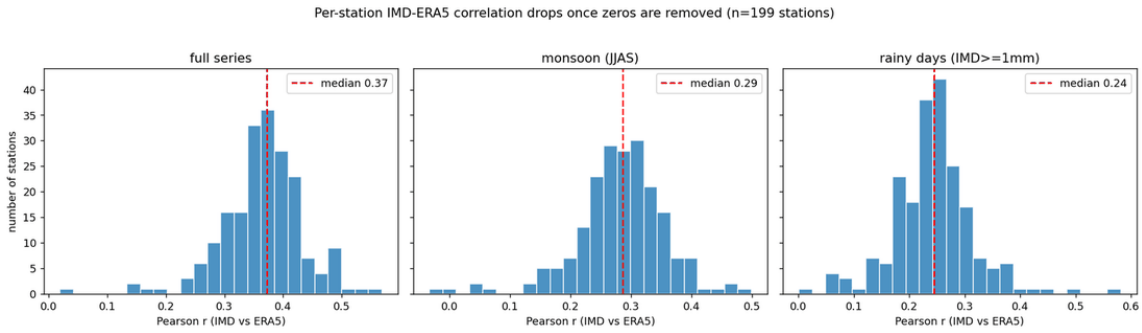


Figure 5.3. Distribution of station-by-station correlation for the full series, the monsoon, and rainy days only.

5.4 Wet-day frequency and extreme behaviour

Specifically, two differences are responsible for the moderate daily correlation noted above. One, rain. Rain is detected much more often by the reanalysis than by the gauge stations. The gauge stations record rain on an average 30% of the days compared to 51% in the reanalysis (Figure 5.4). The reanalysis detects small amounts of rain even though there may be no rain as recorded by the gauge stations. Two, the reanalysis dilutes the rainfall in times of high intensity i.e. it makes it softer and wider.An example from a single station during one monsoon shows both happening together. The reanalysis ends up showing rain more days and reduces the highest rainfall spikes (Figure 5.5). Also, the daily differences are quite big compared to

the average daily amount — around 7.6 mm/day difference, even though the overall average stays the same at about 6.4 mm/day.

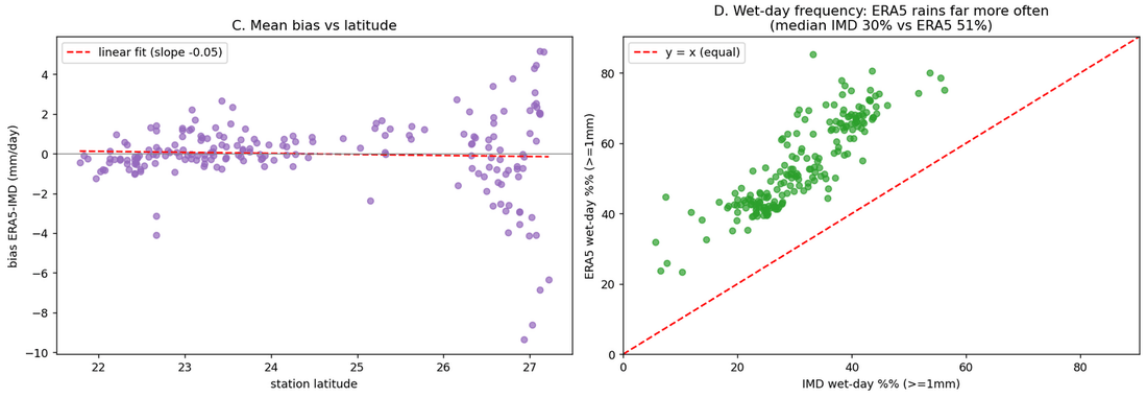


Figure 5.4. Wet-day frequency, gauge versus reanalysis, with bias against latitude.

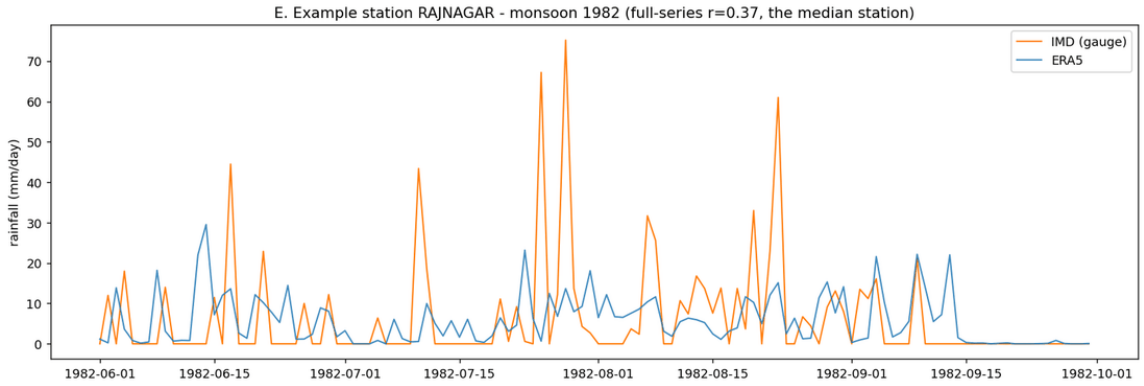


Figure 5.5. Gauge and reanalysis daily rainfall at an example station over one monsoon.

5.5 Reanalysis as a smoothed proxy, and the terrain signal

Putting the two analyses together gives one consistent picture. The reanalysis agrees with the gauges on climatology — the long-run averages match almost everywhere — which is why it is a sound foundation for the main modelling work, where the gauges are too sparse to build a dense graph. But at the daily scale it is only a moderate match: it over-reports light rain and smooths away extremes, making it an easier, gentler target than reality. This is precisely why the graph models in Chapter 4 looked so much better on reanalysis, and why their headline error cannot be read as pure model performance.

One feature of the comparison points ahead. The disagreement is not uniform across the state. Bias is near zero almost everywhere, but the largest departures, and the lowest correlations, sit over the hilly north (Figure 5.6). The reanalysis struggles most where the terrain is most complex, which is consistent with independent assessments of the reanalysis over India and elsewhere [22]. It also becomes a testable prediction: if complex terrain is where the reanalysis is weakest, a model trained on it should forecast least well over mountainous regions. Chapter 6 puts that prediction to the test by moving to two contrasting regions, one flat and one Himalayan.

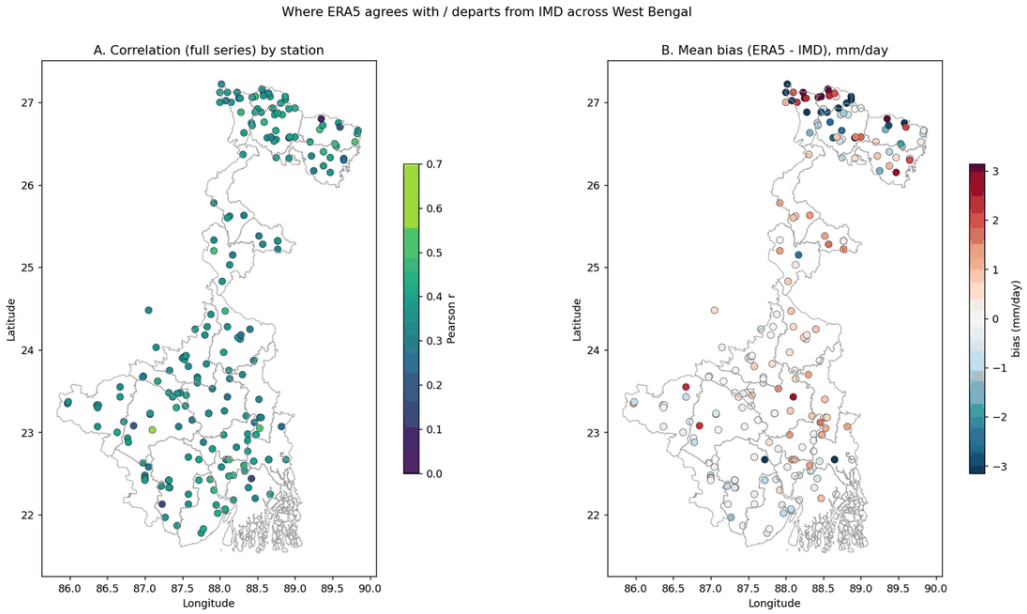


Figure 5.6. Maps of gauge–reanalysis correlation and bias across West Bengal.

5.6 Discussion

There are two practical takeaways here. First, there is a real tradeoff when choosing between the gauge and reanalysis data. Gauge readings are accurate but setting up a reliable network is almost impossible because they're sparse. On the flip side, reanalysis data is complete and agrees with Gauge wrt climatology but is less precise about extreme events. It's reasonable to use reanalysis for the main modeling tasks since we can't create a densely connected gauge network. Still, this approach comes with a limit i.e. models can't fully resolve real extremes due to reanalysis smoothing.

Secondly, the challenges with gauge records aren't a result of faulty modeling. Give the same model the easier reanalysis data, and it performs way better. Moving forward, subsequent chapters dive deep into these issues. Chapter 7 looks at why

even top models struggle with predicting heavy days, and the rest wraps up with efforts to improve that.

CHAPTER 6

Cross-Region Generalisation

Chapter 5 ended with a prediction: if the reanalysis is at its weakest over complex terrain, a model trained on it should forecast least well in mountainous regions. This chapter tests that by repeating the modelling in two regions chosen to bracket West Bengal on terrain — the flat northern Plains and the mountainous Himalaya. The outcome is two-sided, and worth stating up front. One part of the central story generalises cleanly; the specific prediction — that accuracy would worsen steadily as terrain gets rougher — does not, and the reason it does not is the most interesting result in the chapter.

6.1 Motivation: ordering the regions by terrain

West Bengal sits in the middle of a terrain range. We can arrange the Plains, West Bengal, and Himalaya in order of flattest to roughest topography by adding a flatter and a more mountainous area. The model's forecast accuracy, including its error and hit-rate on heavy days, should gradually deteriorate along this line, reaching its lowest point in the Himalaya, if complex terrain is indeed where reanalysis rainfall is most difficult to predict.

6.2 Regions and data

The two new regions are defined by state boundaries so that they are clean and reproducible: the Plains are Uttar Pradesh and Bihar, the Himalaya is Himachal Pradesh and Uttarakhand, and both are disjoint from West Bengal. As elsewhere in the thesis the nodes are real rain-gauge locations — 604 stations in the Plains and 197 in the Himalaya — but the model is fed the matched reanalysis at those points, with the same six weather variables and the same period, so the set-up mirrors the West Bengal work exactly. Per-station elevation is taken from a digital elevation model for

the terrain analysis later in the chapter.

The three regions have genuinely different rainfall, which matters for reading the results. The Plains are the driest (about 3.2 mm/day on average) and the most monsoon-concentrated (84% of rain in June–September); West Bengal is the wettest (5.3 mm/day); the Himalaya sits in between (4.4 mm/day) and is the least monsoon-dominated (67%) (Figure 6.1). The spatial pattern of rainfall also differs from region to region (Figure 6.2). These climatological differences, it turns out, dominate the raw error numbers.

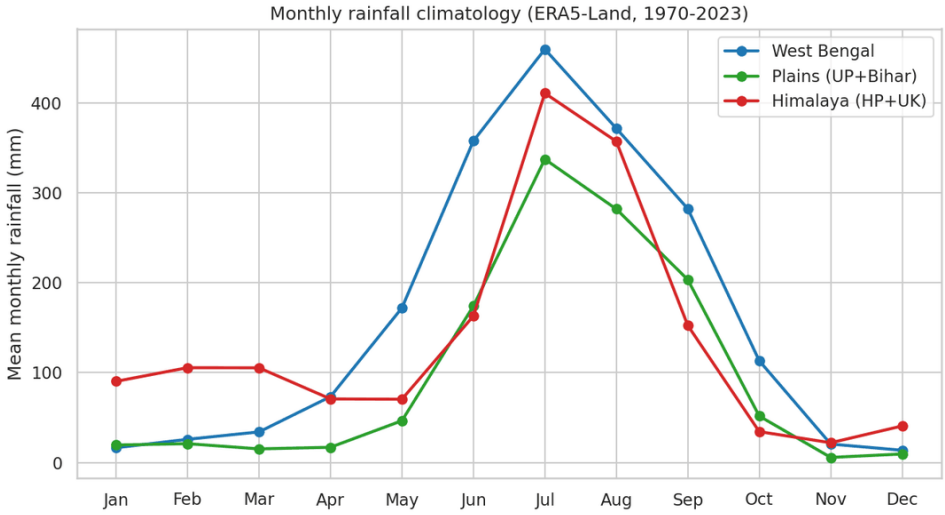


Figure 6.1. Monthly rainfall climatology for the three regions.

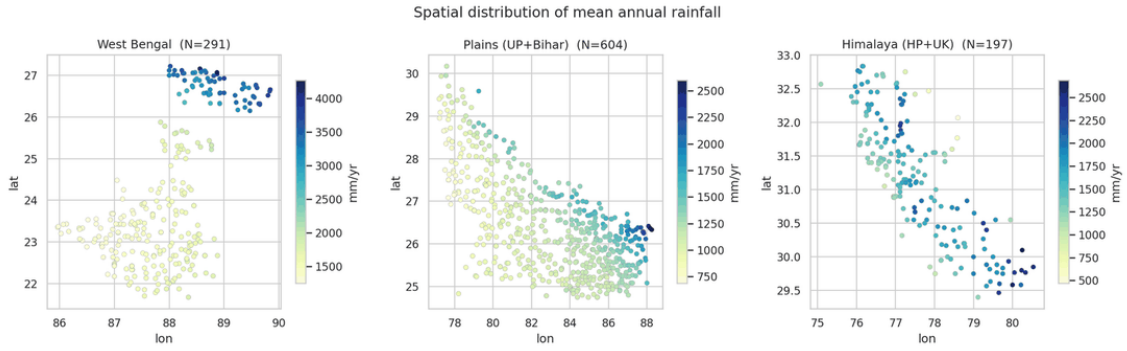


Figure 6.2. Mean annual rainfall across the three regions.

6.3 Experimental set-up

The model and its settings are exactly the same as that of the West Bengal baseline: the same graph-attention architecture, seven-day window, similar training methodologies,

and the same input variables. The regions have different numbers of stations, so the station graphs differ in size; we keep the same 100 km linking rule and report per-station errors so the comparison is not distorted by network size. One caveat applies throughout: the model is fed reanalysis only, so this is a test of how well a reanalysis-trained model generalises, not a comparison against ground truth in the new regions.

6.4 Results by region

The stats don't show the predicted order (Table 6.1, Figure 6.3). Based on root mean squared error, the Plains have the easiest challenge (6.41 mm/day), followed by the Himalaya (7.99), and West Bengal is the toughest (8.78). This is the opposite of what was predicted; the wettest area ends up with the highest error score. But when the error is adjusted for each region's average rainfall, the rankings change yet again. Now, West Bengal comes out on top (1.63), and the Plains have the highest score (1.94). In all this, "Himalaya hardest" doesn't match either of the rankings. Both are dominated by how much it rains in a region, not by how complex its terrain is.

Table 6.1. The same model in three regions (reanalysis input). Raw error and normalised error rank the regions in opposite orders, both driven by rainfall amount rather than terrain.

Region	Stations	RMSE	MAE	Bias	NRMSE	Monsoon RMSE
Plains (UP + Bihar)	604	6.41	2.78	+0.40	1.94	10.02
Himalaya (HP + UK)	197	7.99	3.74	+0.21	1.84	11.90
West Bengal	291	8.78	3.92	−0.10	1.63	12.87

Where the regions agree is on the heavy days. Heavy-rain detection is weak everywhere and collapses completely at the highest intensity: at the very-heavy threshold (≥124.5 mm/day) the model issues essentially no correct heavy forecasts in any region, so the critical success index — the share of heavy days that are both forecast and observed — is zero for all three (Figure 6.3). At the lower heavy thresholds detection is modest and tracks how many heavy days a region actually has rather than how rough its terrain is (Figure 6.4). The same under-prediction of extremes that limits the West Bengal model is present, unchanged, in both new regions.

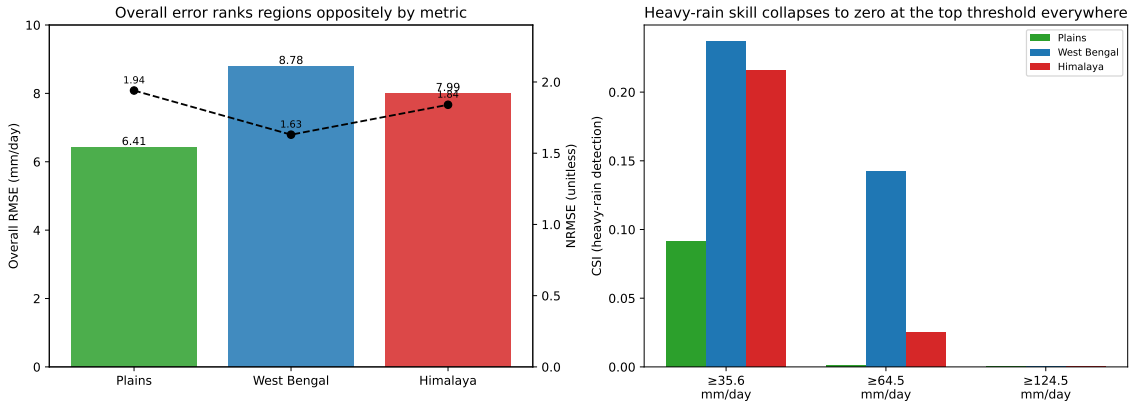


Figure 6.3. Overall error and heavy-rain detection across the three regions.

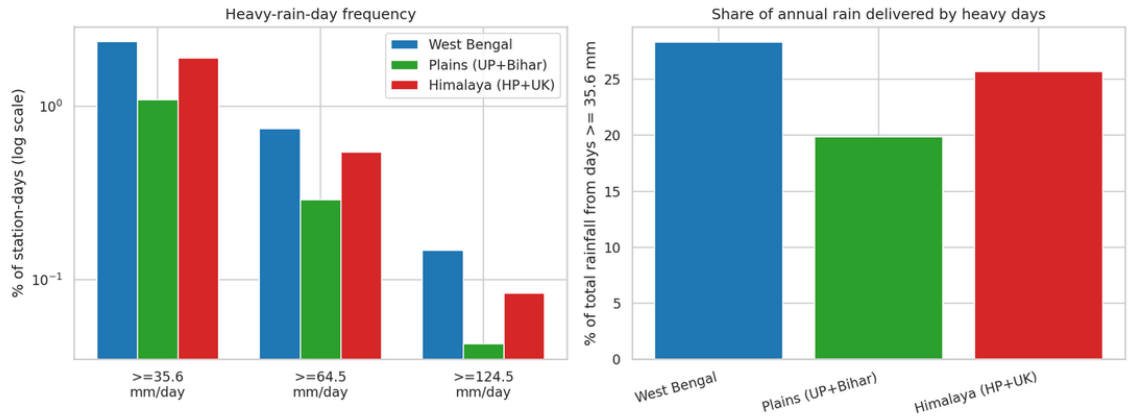


Figure 6.4. Heavy-rain-day frequency and the share of rain they deliver.

6.5 Testing the terrain hypothesis

The results cut both ways. The broad conclusion still holds: the model's trouble with pinning down which days will bring heavy rain is not something specific to West Bengal, since the same problem turns up over the flat Plains and the mountainous Himalaya alike. What fails is the narrower expectation, that the forecast accuracy ought to deteriorate steadily as you move from the flat Plains toward the mountainous Himalaya.

And the reason it fails is really the chapter's main finding, one that supports Chapter 5 : the model is never given any information about the terrain. In real mountains, how much rain falls depends heavily on elevation. The windward slopes get drenched, while the valleys sheltered behind them stays relatively dry. To check whether the reanalysis carries that signal, we plot, for each region, every station's average rainfall against its elevation and fit a straight line through the points (Figure 6.5). The strength of the link is summarised by a correlation: near zero means rainfall and elevation are essentially unrelated, near one that rain rises steadily with height. Across the Himalayan stations, whose elevations span roughly 180 to 5200 m, that correlation is just 0.09 — the reanalysis rainfall is almost flat with height. In West Bengal it is a clear 0.48. The reanalysis has ironed out the orographic structure that, on the ground, makes Himalayan rainfall so sharp and variable.

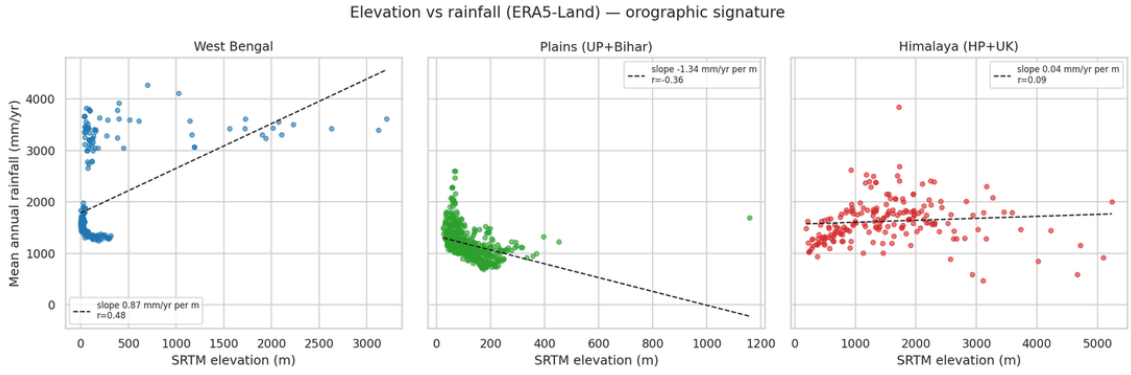


Figure 6.5. Mean annual rainfall versus station elevation, one point per station with a fitted line, for each region.

The smoothing shows up a second way too, in how alike neighbouring stations are. For each region we take every pair of stations, line up their daily rainfall over the full record, and measure how strongly the two rise and fall together — again a correlation, where one means they move in perfect step and zero that they are unrelated. There are thousands of such pairs per region, so Figure 6.6 summarises

them as one box per region — the bold line through each box is the typical (median) pair, and the box spans the middle half of all pairs. Real mountain rainfall is patchy — a storm can drench one valley and miss the next — so we would expect Himalayan stations to agree less than flat-plain ones. The reanalysis shows the opposite: the typical Himalayan pair correlates at 0.55, above West Bengal's 0.52 and the larger Plains region's 0.42. In the reanalysis, in other words, the roughest terrain looks the smoothest. (The Plains value is also dragged down by how large that region is: stations spread far apart will naturally agree with each other less. But even once you account for that, it's the Himalaya — sitting just above West Bengal — that makes for the revealing comparison.)

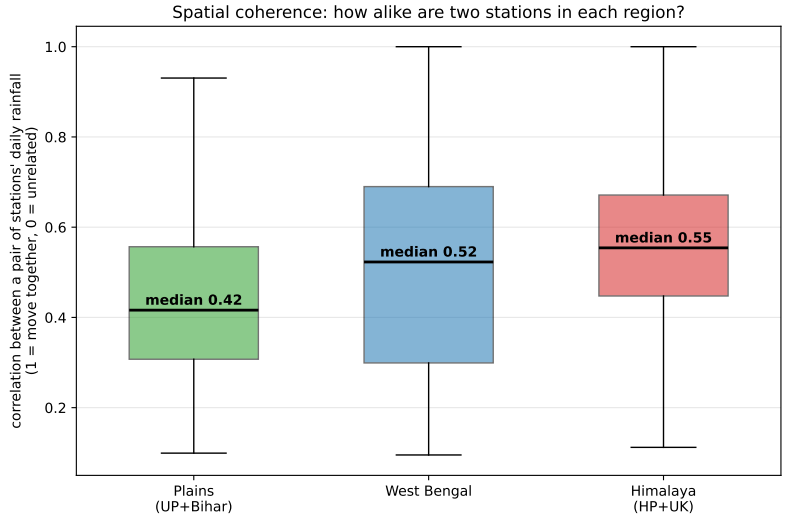


Figure 6.6. The correlation between pairs of stations' daily rainfall, summarised as one box per region: correlation is on the vertical axis (1 = the two stations move together, 0 = unrelated), the bold line is the typical (median) pair, and a higher box means stations across that region behave more alike.

This accounts for the missing trend straightforwardly enough. A model has no way of forecasting worse over complex terrain once its inputs no longer carry that terrain's distinctive, hard-to-predict variability, because the reanalysis has already smoothed it away. So the terrain effect that Chapter 5 predicts is genuine — but it lives in the data, showing up as smoothing in the reanalysis rather than as a fall in forecast accuracy. We get a small confirmation of this by feeding some of the lost information back in: when the stations' static features, true elevation among them, are added as extra inputs, the Himalayan error drops from 7.99 to 7.71 mm/day. It's a modest gain, but a consistent one, and it shows up exactly where you'd expect — in the single region where the reanalysis had flattened the elevation signal the most.

6.6 Discussion and caveats

The chapter basically half-confirms its hypothesis. It does generalize convincingly that three regions with different climates and terrains all experience similar collapses on heavy-rain days. Yet, performance worsening with terrain that was anticipated didn't appear in the error figures. Still, the data showed the mechanism is clearly there — the reanalysis smooths out mountain rainfall just like Chapter 5 explained.

These conclusions come with a few caveats, though. The model only gets reanalysis data, so the regional runs can't be checked against real-world facts like the West Bengal data were. The regions aren't identical either; they have different numbers of stations.

Even with these limitations, the cross-region test works as intended. It demonstrates that the main issue is widespread — suggesting that the reanalysis itself is where the terrain signal is lost.

CHAPTER 7

Diagnosing the Deterministic Ceiling

In the previous chapters, a recurring problem arises through various methods. Concerning West Bengal, the most accurate forecast according to the standard-model amounts to approximately 8.8 mm/day. Improving the loss function leads nowhere, deeper neural networks have little to teach, longer training times produce similar results, and different sources of data prove useless. In three areas combined, all the models show a common problem – they do not accurately capture rainy days when there is an unusually high amount of precipitation.

This chapter tries to identify the reasons for this issue. Instead of offering a summary, the chapter focuses on the functioning of the best model in terms of forecasting the weather, i.e., the corrected graph-attention baseline model, based on nearly 860,000 station-days of data. It turns out the reason is pretty clear, although obvious now in hindsight. The model predicts the typical daily rainfall, which is its best guess when trying to minimize the squared error. But averages don't reflect the extreme rainfall events, which make up the bulk of the errors.

This reveals why boosting the input reading skills won't fix anything; the output format itself needs changing. This leads us onto the next steps discussed in Chapters 8 and 9.

7.1 The model behaves like a conditional-average predictor

Take the test days and sort them by the value the model predicted, then split them into twelve equally-sized groups — call these the prediction bins. For each bin we compare two numbers: the average rainfall the model predicted, and the average

rainfall actually observed. Plotting them — both in millimetres per day — the points fall almost exactly on the line where the two are equal (Figure 7.1).

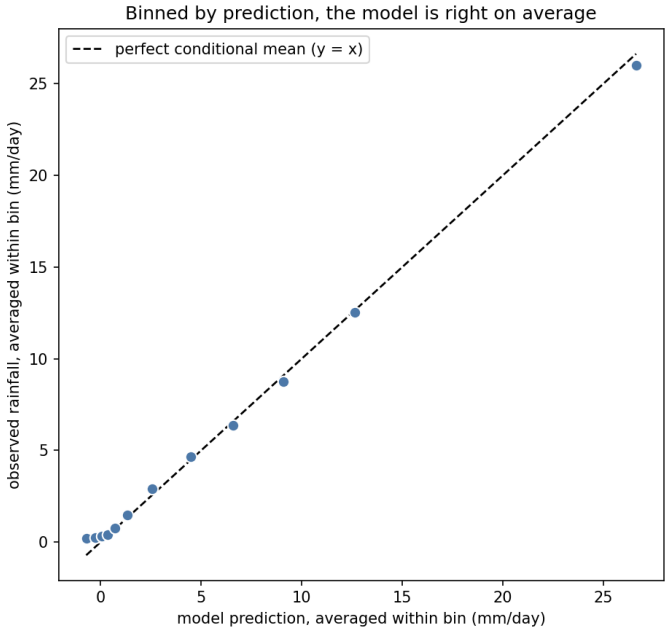


Figure 7.1. For each prediction bin, the average observed rainfall (vertical axis) against the average predicted rainfall (horizontal axis), both in mm/day; a point on the dashed line means the prediction equals the average outcome for that group of days.

Figure 7.1 shows that the model has learned the conditional mean, the average outcome given the inputs, which is exactly what a squared-error objective is designed to learn, because the mean minimizes squared error. The model is not broken or mistuned; it is doing that job almost optimally. This is also why none of the encoder changes in earlier chapters helped: we cannot improve a near-optimal estimate of the mean by reshaping the network that produces it.

It helps to split the task into two questions: will it rain? and how much? The first is largely solved — the model separates rainy days from dry ones well (a wet-day hit rate, measured by the critical success index introduced in Chapter 6, of about 0.70). The difficulty lies entirely in the second question, the amount.

7.2 Under-dispersion: the predictions cannot reach the heavy days

Predicting the conditional mean has a hard consequence for a quantity as skewed as rainfall. Most days are dry or light and a few are torrential, so the average for

any given set of conditions sits far below the heaviest outcomes those conditions can produce. The model therefore hedges toward the middle: its predictions are far less spread out than reality (a standard deviation of 8.5 mm against 12.0 for the observations), and they have a ceiling. Over the entire test set the model never once predicts more than about 107 mm in a day, while observations run up to 336 mm (Figure 7.2). The heaviest third of the observed range is simply unreachable. Forecasters call this under-dispersion: the forecast spread is too narrow and misses the tails.

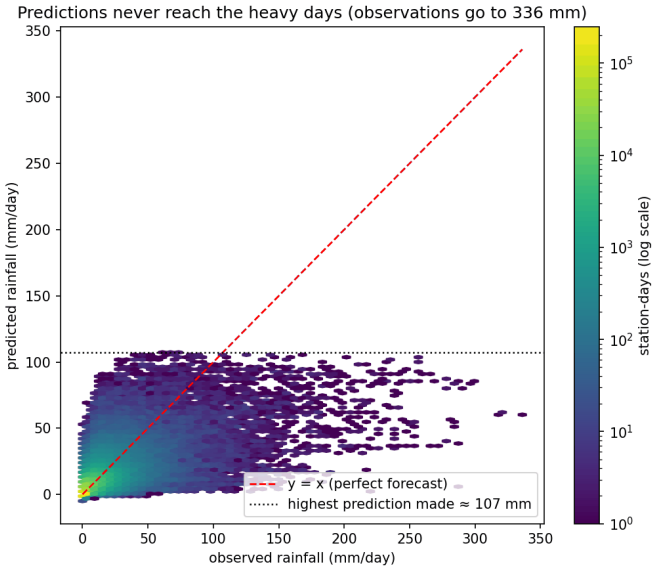


Figure 7.2. Observed versus predicted rainfall for every test station-day; the cloud bends below the one-to-one line and predictions stop at about 107 mm.

7.3 Where the error concentrates

Because the costly errors are the big misses, this ceiling dominates the overall score even though it touches only a tiny minority of days. Ranking the test days from heaviest to lightest and accumulating the squared error makes the imbalance stark: the heaviest 2.4% of days — those above the rather-heavy threshold of 35.6 mm — account for 59% of all the squared error, and the very heaviest 0.1% of days (above 124.5 mm) still account for a fifth of it (Figure 7.3). The model is accurate on the ninety-seven-plus percent of ordinary days and pays almost the whole bill on the rare heavy ones. Lowering the headline error meaningfully therefore means getting the heavy days right — precisely the days the conditional mean cannot represent.

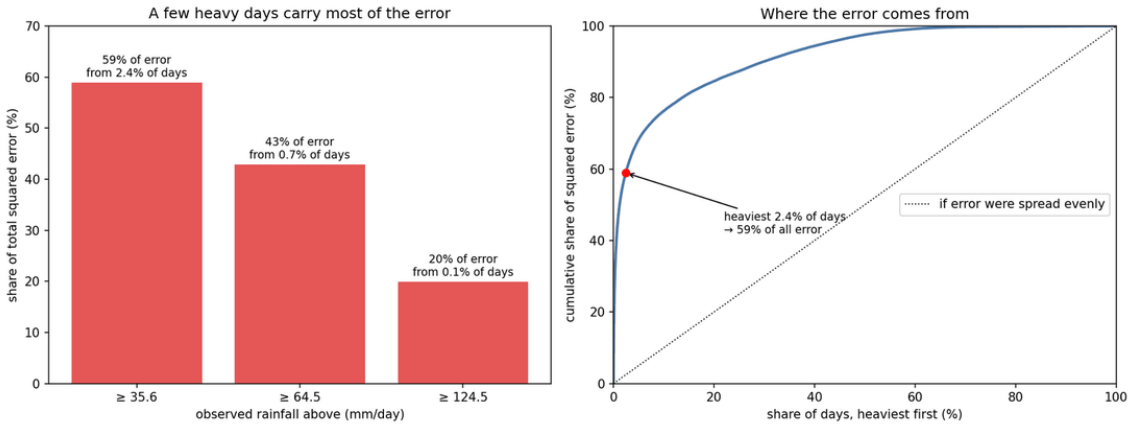


Figure 7.3. Left: the share of total squared error contributed by days above each heavy threshold; right: cumulative squared error as days are added heaviest-first.

7.4 The limit is the output and objective, not the encoder

The three findings point to one diagnosis. The model is about as good an estimator of the conditional mean as one could ask for (7.1). The trouble is that rainfall is heavy-tailed and mean sits too low and too narrow to reach the extremes (7.2) — and it's the extremes that hold nearly all the error (7.3).

This isn't the spatio-temporal encoder's fault. The graph and recurrent network have done their job; sharpening them won't help since the mean is already optimized. The issue is that the answer is just one number, trained using squared-error loss.

This changes how the rest of the thesis works. In Chapter 8, the negative side is proved by showing that even richer representations, gained from self-supervised pretraining, still don't break through the limit. Then, Chapter 9 tackles the positive aspect, switching from that single number to a whole predicted distribution. This way, a heavy day can be expressed as a probability rather than being represented by a single value that might be missed.

CHAPTER 8

Can Representation Learning Break the Ceiling?

Chapter 7 placed the blame for the ceiling on the form of the answer — a single number trained to predict the average — and explicitly cleared the encoder (Section 7.4). This chapter tests that directly. The argument has two sides. If the encoder is already capturing essentially all of the *predictable* signal — call it near-optimal — then handing it a better internal representation cannot lower the error, because what limits the error is the single-number output, not the representation. If, on the other hand, the encoder were still leaving usable information untapped, then a richer representation should improve the forecast. The experiment is therefore decisive either way: we try hard to enrich the representation, and if the forecast does not improve, the encoder was not the bottleneck. We test this with self-supervised learning, or SSL — training the encoder on a label-free task invented from the data before the forecasting task — and several related ideas. The result is the negative one the diagnosis predicts: none of them meaningfully moves the ceiling.

8.1 The idea of pretraining the encoder

A forecasting model is normally trained end to end on the one task we care about — predict tomorrow's rain. Self-supervised pretraining adds a preliminary stage: before the forecasting task, the encoder is trained on a task that needs no labels, invented from the data itself, so that it reaches the forecasting stage already knowing something about how the weather is structured. The hope is that this warm-started encoder generalises better than one trained from scratch. The standard recipe for grid- and graph-shaped data is *masked reconstruction*: hide part of the input and train the model to fill it back in, which forces it to learn how the variables and locations relate [13, 14]. We borrow that recipe and ask whether it raises rainfall-forecast accuracy.

8.2 Methods compared

Against the corrected baseline we line up six variants. The baseline is the graph-attention model of Chapter 4: twelve inputs per station-day (six reanalysis weather variables and six calendar signals), trained directly to predict next-day rain under a squared-error loss. Each variant changes exactly one thing.

Static features. We add three fixed attributes of each station — its latitude, longitude, and elevation — to the twelve inputs, so the model is told where each station sits and how high it is. This asks whether geographic context the model cannot infer from the weather alone is useful.

Deeper graph. We stack a second graph-attention layer, so that at each time step a station blends information not only with its direct neighbours but with its neighbours' neighbours (a two-hop neighbourhood). This asks whether more spatial mixing helps.

Self-supervised pretrain, then fine-tune. This is the masked-reconstruction recipe run as two stages. In the first stage there are no rainfall labels: on each training window we randomly hide about 15% of the weather readings — only the six physical variables; the calendar and location inputs stay visible and are never hidden — and we train the encoder, together with a small temporary decoder, to fill the hidden values back in from the ones it can still see. To do this well the encoder has to learn how the variables and the neighbouring stations relate. In the second stage we discard the decoder, attach a fresh rainfall output, and fine-tune the whole model on next-day rain exactly as the baseline was trained. The encoder thus begins the forecasting task already warmed up.

Self-supervised, joint. The same masked-reconstruction task, but carried out *at the same time* as the forecast rather than before it: the model has both a rainfall output and a reconstruction output, and the training loss is the sum of the forecast error and the reconstruction error. Masking is applied only during training; at test time the model sees clean inputs and uses only its rainfall output. This is the natural alternative to the two-stage version, and the pair lets us compare the two ways of using the same self-supervised signal.

Multi-task auxiliary. Here the side task is supervised rather than reconstruction: alongside predicting tomorrow's rain, the model is also asked, from a second output, to predict tomorrow's other five weather variables, with that side loss given a small weight that decreases over training. The extra predictions are never used; the intent is that forcing the shared encoder to also explain the physical drivers will regularise the rainfall output.

Three-stage. This chains the warm-ups into one pipeline: first the masked-reconstruction pretraining, then the multi-task auxiliary stage, then a final fine-tune on rainfall alone. It tests whether stacking the (individually small) effects compounds them.

Every variant uses the same encoder, data, seven-day window, graph, and training budget as the baseline, so each isolates exactly one change.

8.3 Experimental set-up

All seven models (baseline plus six variants) are trained and tested on the West Bengal reanalysis with similar methodologies of Chapter 4. They are compared on the same measures: the test error, the heavy-rain detection score, and the gap between training and validation error (which would reveal any over-fitting that a regularizer could fix). Where pretraining is used, the encoder weights are carried over exactly into the forecasting model, so the only change is the warm start.

8.4 Results: the ablation matrix

The differences are small, and most variants do not improve on the baseline (Figure 8.1, Table 8.1). Only the masked-reconstruction pretraining followed by fine-tuning , comes in below the baseline, at 8.70 mm/day against 8.78 mm/day, an improvement of under 0.1 mm that is too small to read as a real gain. The rest sit at or above the baseline error. The static features change nothing (8.78). The deeper graph is poorer (8.88), consistent with the over-smoothing that stacking graph layers is known to cause. The joint form of pretraining is also poorer (8.81). The auxiliary-task and three-stage variants land just below the baseline on average error (8.76), but only by becoming more conservative: they predict lower across the board, which lowers the average-sized error while leaving a dry bias and weaker heavy-rain detection, and they do not shrink the already-near-zero gap between training and validation error —

so the hoped-for regularisation does not appear. Heavy-rain detection tells the same story: only the pretrain-then-fine-tune model improves on the baseline, and again only slightly.

Table 8.1. All seven models on the West Bengal test set. RMSE and MAE in mm/day; CSI is the heavy-rain detection score at the ≥64.5 mm threshold. Only pretrain-then-fine-tune comes in below the baseline error, marginally, while also leading on heavy-rain detection.

Variant	RMSE	MAE	Bias	CSI ≥64.5 mm
baseline	8.781	3.918	−0.103	0.142
+ static features	8.777	4.005	+0.208	0.143
deeper graph	8.880	3.955	+0.127	0.107
SSL pretrain → fine-tune	**8.697**	3.919	+0.039	**0.167**
SSL joint	8.809	4.052	+0.092	0.103
multi-task auxiliary	8.763	3.825	−0.399	0.107
three-stage	8.758	3.864	−0.417	0.156

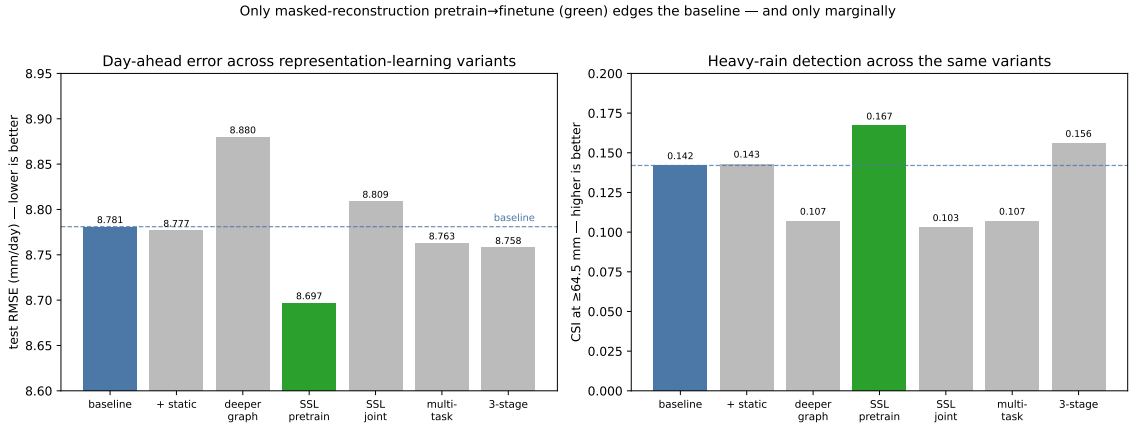


Figure 8.1. The ablation matrix: test error and heavy-rain detection across the seven models.

The pretraining worked well, the warm-started encoder showed richer internal representation with more diverse node embeddings (effective rank of about 5.9 compared to 3.1). Still, this upgrade didn't boost forecasts significantly. Here, a better encoder didn't lead to improved rainfall predictions.

8.5 Pretraining versus joint training

The one comparison that does say something clean is between the two ways of using the same self-supervised task. Used as a separate warm-up stage, masked reconstruction

is the best variant in the study; used jointly, mixed into the forecasting loss, it is among the poorest (8.70 against 8.81). The reason is visible during training, the joint version pulls the shared encoder toward features that are good for reconstruction but not for forecasting, and its forecasting loss begins to over-fit partway through, so the auxiliary task gives no benefit. This is the opposite of what has been reported for a different self-supervised family — contrastive learning, where the joint form usually wins [18]. For masked reconstruction on weather data we are not aware of a prior head-to-head, so this small "pretrain beats joint" result is a modest contribution in its own right, and it would have been worth reporting whichever side had won.

8.6 The encoder is near its ceiling

The verdict is a clean negative, and it is the one Chapter 7 led us to expect. Section 7.4 argued that the limit on the deterministic forecast is the form of the output — a single number under a squared-error loss — and not the encoder, because the encoder already estimates the conditional mean almost optimally; the direct consequence is that improving the encoder should not improve the forecast. That is exactly what this chapter finds. None of the experiments here, lifted the deterministic error by more than a negligible margin, and the variant that most enriched the encoder's representation did not turn that richness into accuracy. The encoder is not where the headroom is. The lever has to be the output itself — which is what Chapter 9 changes.

CHAPTER 9

Probabilistic Multi-Quantile Forecasting

In Chapters 7 and 8, we narrow down to one final approach. We found that the deterministic model keeps giving just the average daily rainfall i.e. predicting what happens on a typical day but missing the heavier rain events (Chapter 7). Tweaking the encoder didn't solve this issue either (Chapter 8).

So, the only option left is to adjust the output. Instead of forecasting just one amount, the new model predicts a range of possible rainfall values. With this, the likelihood of when heavy rainfall will occur becomes clearer. They made these changes while keeping things simple: the same encoder was used, but they altered how the output is shown and the way errors are calculated.

This tweak worked really well. Even under those extreme conditions that the previous model failed to predict, the forecasts were pretty accurate and well calibrated. Naturally, there is a small drawback: their predictions of typical days are a little off. When we truly need to know about severe rain, it's a good improvement overall.

9.1 From one number to a distribution

A point forecast simply gives a single number whereas a probabilistic forecast provides multiple predictions at different percentile levels, i.e. a distribution. The second option gives the likelihood of the extreme events. A distribution is not pinned to the mean, so its upper tail can express the possibility of a heavy day even when the most likely outcome is light. We therefore keep everything about the model except what it outputs.

9.2 Method: a multi-quantile model

Describing the distribution by its quantiles. Let R be the rainfall on a given day at a given station, with cumulative distribution function $F(x) = \Pr(R \le x)$ — the probability that rainfall is at most x. The *quantile* at level $\tau \in (0, 1)$ is the inverse of this,

$$Q(\tau) := \inf\{\, x : F(x) \ge \tau \,\},$$

the smallest amount the day stays at or below with probability τ. For example $Q(0.5)$ is the median and $Q(0.9)$ is the amount exceeded on only one day in ten. Giving $Q(\tau)$ for a range of levels describes the whole distribution — which is exactly what frees the forecast from the single conditional mean that Chapter 7 identified as the bottleneck.

The model outputs eleven quantiles per station per day, at the fixed levels

$$\tau \in \{0.05, 0.10, 0.25, 0.50, 0.75, 0.90, 0.95, 0.975, 0.99, 0.995, 0.999\},$$

deliberately dense in the upper tail where the heavy-rain thresholds sit. It is worth being explicit about what these eleven numbers are, because it is easy to misread: each output is a *rainfall amount in millimetres* — the value at that probability level — and not a probability. They do not sum to one, and there is no softmax; read in order from the lowest level to the highest, they trace out the predicted distribution directly.

Keeping the quantiles ordered. A valid distribution must satisfy $Q(\tau_1) \le Q(\tau_2)$ whenever $\tau_1 < \tau_2$ — a higher probability level cannot correspond to less rainfall. The output layer enforces this by construction: it predicts the lowest quantile and then a sequence of non-negative increments that are added on to reach each higher level, so the eleven values can never cross.

The training objective: the pinball loss. Each level is fitted with the *pinball loss* (also called the quantile loss), the standard objective for quantile regression [23]. For a target level τ, a predicted quantile $\hat{q}$ and an observation y, writing the forecast

error as $u = y - \hat{q}$,

$$\rho_\tau(y, \hat{q}) = \begin{cases} \tau\, u, & u \geq 0 \quad \text{(under-prediction: } y \geq \hat{q}\text{)} \\ (\tau - 1)\, u, & u < 0 \quad \text{(over-prediction: } y < \hat{q}\text{)}, \end{cases}$$

which can be written compactly as $\rho_\tau(u) = u\,(\tau - \mathbb{1}\{u < 0\})$. The two arms are asymmetric: an under-prediction is charged at rate τ and an over-prediction at rate $1 - \tau$. For a high level such as $\tau = 0.9$, under-prediction costs nine times as much as over-prediction, so the fitted value is pulled upward until only about one observation in ten lies above it — that is, to the 90th percentile (Figure 9.1, left).

The total loss averages this over all eleven levels and over every station and day:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{11} \sum_{k=1}^{11} \rho_{\tau_k}\big(y_i, \hat{q}_{i,k}\big).$$

The loss is therefore computed per level and then summed — here averaged — across the eleven levels, so each output is supervised by its own pinball term.

Why this lands on the right quantile (gradient flow). Because the loss is piecewise linear, its slope with respect to the prediction is constant on each side of the observation (Figure 9.1, right): a fixed pull of size τ for an under-prediction and $1 - \tau$ for an over-prediction, in opposite directions. The fit settles where these opposing pulls balance in expectation, i.e. where $\Pr(y \leq \hat{q}) = \tau$ — which is precisely the definition of the τ-quantile [23]. This is why minimising the pinball loss yields a calibrated quantile, rather than the mean that a squared-error loss would give.

Only the output and loss change. The encoder, the inputs, the station graph, the seven-day window and the training budget are all identical to the deterministic graph-attention model of Chapter 4; only the final layer (eleven ordered outputs instead of one) and the loss (pinball instead of squared error) differ. Any gain on the heavy days is therefore attributable to the change of formulation, not to a larger or different network.

9.3 Evaluating calibration and probabilistic skill

A distribution forecast needs different measures from a point forecast. We use four, set out one at a time with the equation first and the plain meaning after.

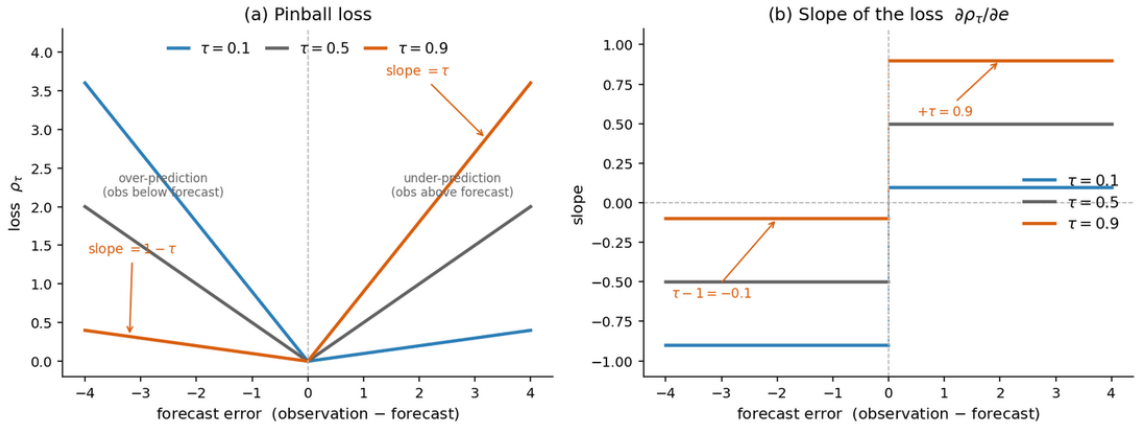


Figure 9.1. Left: the pinball loss against forecast error (observation minus forecast) for a low, median and high level ($\tau = 0.1, 0.5, 0.9$); the V is lopsided, the steep arm on the side the level cares about. Right: the slope of that loss — constant on each side, with a jump at zero — showing the asymmetric pull (size τ versus $1 - \tau$) that drives each output to its quantile.

Calibration — the probability integral transform (PIT). For each forecast, the PIT is the predicted cumulative probability at the value that actually occurred,

$$\mathrm{PIT}_i = \hat{F}_i(y_i),$$

where $\hat{F}_i$ is the day's predicted distribution and y_i the observed rainfall. In words, it asks which percentile of the forecast, the observation landed on. If the forecasts are honest, these values are spread evenly between 0 and 1, so their histogram is flat [27, 28]. A ∪-shaped histogram — too many values near 0 and 1 — means the predicted spread is too narrow and reality keeps falling outside it (under-dispersion, the deterministic model's failing); a central hump means the opposite. With eleven discrete quantiles the PIT is read off as the level the observation falls between. Uniformity is necessary but not, on its own, sufficient for full calibration [28].

Calibration at a chosen event — the reliability diagram. For a specific event, the model issues a probability each day; here the event is rainfall above the heavy-rain threshold of 64.5 mm/day. Group the days into bins of forecast probability and, for each bin, compare the average forecast probability with the fraction of days the event actually happened. Plotting observed frequency against forecast probability, perfectly reliable forecasts lie on the diagonal; points below it indicate over-confidence, above it under-confidence [29]. The PIT and the reliability diagram are complementary, not the same check in two forms. The PIT looks at the *whole* predicted distribution on

every day and asks whether it is calibrated overall; the reliability diagram singles out *one* decision-relevant event — the heavy-rain warning at 64.5 mm — and asks whether the probabilities issued for *that* event are trustworthy. A flat PIT is global, distributional calibration; a diagonal reliability curve is event-level calibration at the threshold a user would actually act on. We show the reliability curve at 64.5 mm because that is the operationally important threshold; it could in principle be drawn at any level.

Overall quality and sharpness — the continuous ranked probability score (CRPS). The CRPS compares a whole predicted distribution with the single value observed. With $\hat{F}$ the predicted cumulative distribution and $\mathbb{1}\{x \geq y\}$ the "perfect" forecast that places all its certainty on the observation y (a step rising from 0 to 1 at y),

$$\mathrm{CRPS}(\hat{F}, y) = \int_{-\infty}^{\infty} \left(\hat{F}(x) - \mathbb{1}\{x \geq y\}\right)^2 dx$$

[24, 25]. It is the squared area between the predicted S-shaped curve and the observation's step: a sharp, well-placed forecast hugs the step and scores low, a vague or biased one leaves a large gap. It is reported in millimetres, and lower is better. Its useful property is the link to point forecasts: a deterministic prediction $\hat{y}$ is a distribution with all its mass at one value, whose cumulative curve is itself a step at $\hat{y}$; the two steps then agree everywhere except between $\hat{y}$ and y, and the integral collapses to $|\, y - \hat{y}\,|$. Averaged over all days, the CRPS of a point forecast is exactly its mean absolute error — so the distribution's CRPS can be set directly against the deterministic model's mean absolute error, in the same units.

Skill at a heavy-rain threshold — the Brier score. For a yes/no event (did rainfall exceed a threshold?), with forecast probability p_i and outcome $o_i \in \{0, 1\}$,

$$\mathrm{BS} = \frac{1}{N} \sum_{i=1}^{N} (p_i - o_i)^2,$$

the mean squared error of the probability forecast [26] (lower is better). It is the single-threshold special case of the CRPS.

Turning the Brier score into skill (BSS). A Brier score is hard to judge on its own, so it is compared with a no-effort reference forecast: always predicting the event's long-run frequency $\bar{o}$ (climatology). Putting the constant forecast $p = \bar{o}$ into

the Brier formula and splitting the days into those where the event happened (a fraction $\bar{o}$, outcome 1) and those where it did not (a fraction $1-\bar{o}$, outcome 0),

$$\mathrm{BS}_{\mathrm{ref}} = \underbrace{\bar{o}\,(1-\bar{o})^2}_{\text{event days}} + \underbrace{(1-\bar{o})\,\bar{o}^{\,2}}_{\text{non-event days}} = \bar{o}\,(1-\bar{o}).$$

The skill score is the fractional improvement on this reference,

$$\mathrm{BSS} = 1 - \frac{\mathrm{BS}}{\mathrm{BS}_{\mathrm{ref}}},$$

where 0 means no better than climatology, a positive value means better, and a negative value means below it [29]. This is the precise sense in which we use the word *skill*: improvement over a no-effort reference. As a worked example, at the ≥ 35.6 mm threshold the event occurs on a fraction $\bar{o} = 0.0243$ of station-days, so $\mathrm{BS}_{\mathrm{ref}} = 0.0243 \times 0.9757 = 0.0237$; the from-scratch forecast's Brier score is 0.0183, giving $\mathrm{BSS} = 1 - 0.0183/0.0237 = 0.23$ — a 23% improvement on climatology.

9.4 Results: calibrated risk at every threshold

We report the probabilistic forecaster in two forms: trained from scratch — the clean counterpart of the deterministic model, identical except for the output and loss — and with the self-supervised pretraining of Chapter 8 added. The two differ only marginally, and we say which is which throughout.

The forecasts are well calibrated. The PIT histogram for the from-scratch model sits close to the flat, perfectly-calibrated line in its overall envelope, with its values averaging 0.51 against the ideal of one-half; the regular peaks at the eleven quantile levels are an artefact of locating each observation among a discrete set of quantiles, and the extra mass at the dry end reflects the many zero-rainfall days. The reliability curve at the heavy threshold sits close to the diagonal, running just below it (a mild over-confidence on the rarest days) (Figure 9.2).

Followed day by day through a single monsoon at one station, the predicted bands narrow on quiet days and widen through the active spells, and the upper percentiles climb far above the deterministic model's hard ceiling to track the heaviest days (Figure 9.3).

The headline is the heavy-rain skill. Against climatology, the probabilistic forecast has a positive Brier skill score at all three heavy thresholds: trained from scratch it improves on climatology by about 23%, 15%, and 8% at the rather-heavy (≥ 35.6

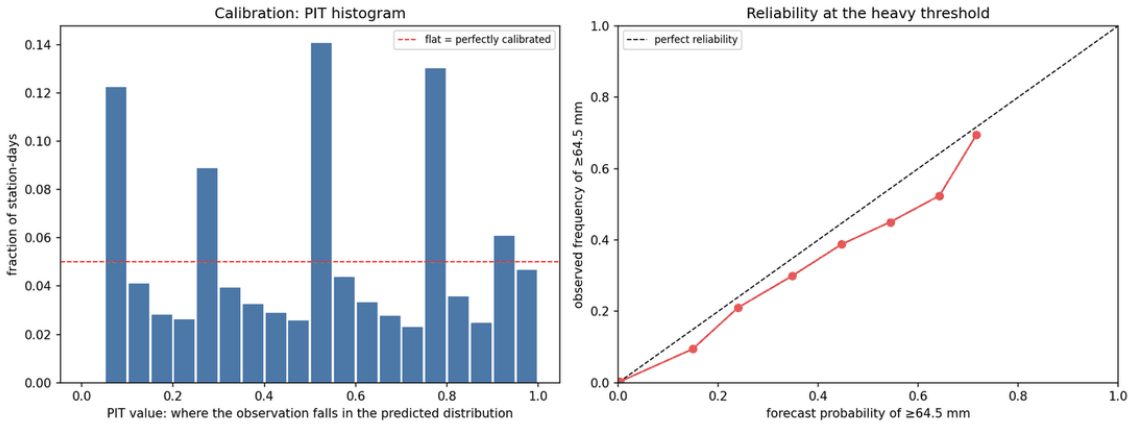


Figure 9.2. Left: the PIT histogram; the x-axis is the percentile the observation fell on (0 to 1), the y-axis is the fraction of station-days landing there, and a flat profile along the red line means calibrated. Right: the reliability diagram at ≥ 64.5 mm; the x-axis is the forecast probability, the y-axis the observed frequency, and the dashed diagonal is perfect reliability.

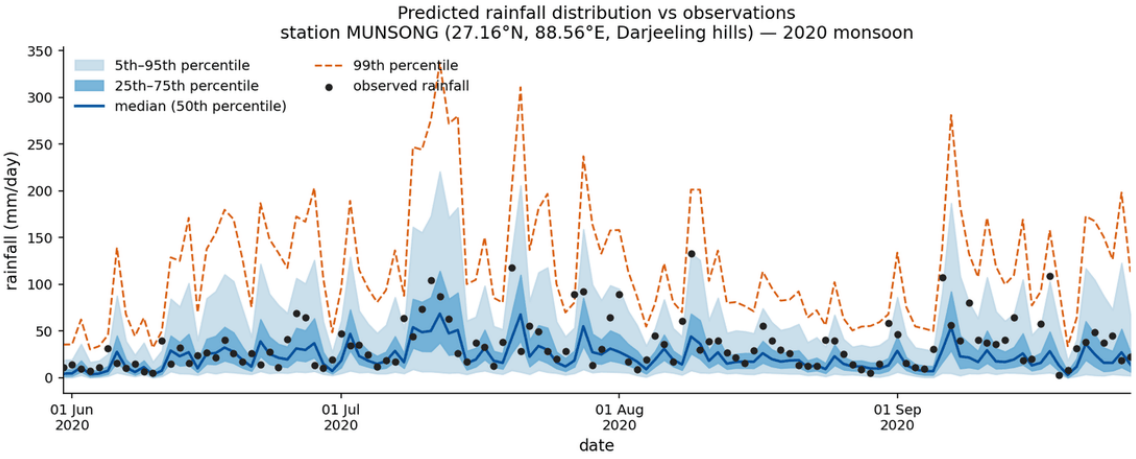


Figure 9.3. The predicted distribution against observations at one gauge: station Munsong in the Darjeeling hills of northern West Bengal (27.16°N, 88.56°E), the wettest station in the test set, shown over the 2020 south-west monsoon (31 May – 27 September 2020). This single station and window are an illustrative close-up of heavy-rain behaviour; the quantitative results elsewhere in the chapter are computed over all 291 stations and the full December 2015 – December 2023 test period. The x-axis is the date, the y-axis is rainfall in millimetres per day; the central line is the predicted median (50th percentile), the two shaded bands are the 25th–75th and 5th–95th percentile ranges, the upper dashed curve is the 99th percentile, and the dots are the observed rainfall. The forecaster shown is the from-scratch model.

mm), heavy ($\geq$ 64.5 mm) and very-heavy ($\geq$ 124.5 mm) levels, and the pretrained form is marginally higher at 24%, 18%, and 9%. The score stays positive even at the very-heavy $\geq$ 124.5 mm threshold, where the deterministic model issued no heavy forecasts at all and so scored zero (Figure 9.4). The model gives genuine, calibrated warnings exactly where the point forecast was silent.

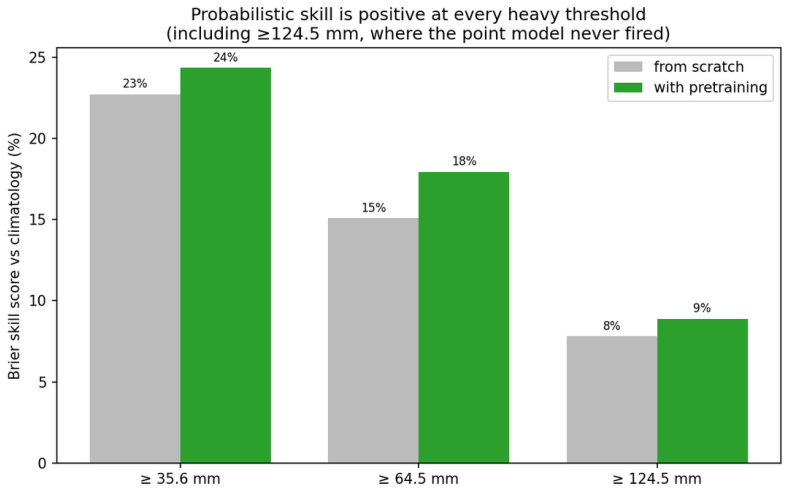


Figure 9.4. Brier skill score against climatology; the x-axis groups the three rainfall thresholds, the y-axis is the percentage improvement on climatology (higher is better, zero being the climatology baseline), and the paired bars are the from-scratch and pretrained models.

Because it outputs a full distribution, the model also offers something a single forecast cannot: a tunable warning level. We treat a chosen percentile as the value at which a heavy-rain warning is issued, which sorts every day into a hit, a miss or a false alarm and yields three standard verification scores. The hit rate, or probability of detection (POD), is the fraction of heavy days that are caught, POD = hits/(hits + misses). The overall detection score is the critical success index (CSI, introduced in Chapter 6), CSI = hits/(hits + misses + false alarms), which rewards hits while penalising both misses and false alarms, and unlike the hit rate cannot be inflated just by issuing more warnings. The frequency bias, (hits + false alarms)/(hits + misses), is how many times more often the warning fires than the event occurs — one is ideal, above one means over-warning. A cautious setting (the median) raises few false alarms but catches few heavy days; a higher percentile catches far more, at the cost of more false alarms. For heavy rain ($\geq$ 64.5 mm), with the pretrained model, the hit rate climbs from about 7% at the median to 55% at the 90th percentile and 85% at the 99th, while the frequency bias rises from well below one to about eleven (Figure 9.5). The critical success index, read

at the 90th percentile, reaches 0.187 with pretraining (0.170 from scratch) — the highest anywhere in this study, above both the deterministic baseline (0.142) and the strongest representation-learning variant of Chapter 8 (0.167).

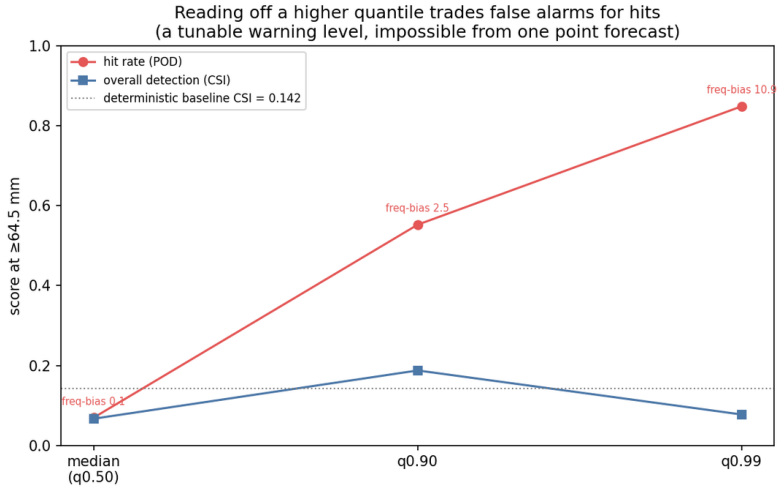


Figure 9.5. As the warning percentile is raised along the x-axis (median, 90th, 99th), the red curve is the hit rate (probability of detection) and the blue curve the critical success index for $\geq$ 64.5 mm on the y-axis; each point is labelled with its frequency bias, and the dotted line marks the deterministic baseline's critical success index of 0.142.

Looking at the warning percentile over the whole range and computing the critical success index for each threshold gives a unimodal curve(Figure 9.6). The extremes, however, must necessarily be low: not because there is something wrong with the forecasts in these areas, but because a very conservative threshold means that very few warnings will be issued, which in turn leads to very few days being captured correctly, while a very liberal one results in many false warnings. So the middle ground produces a curve that peaks and slopes down on both sides. (While the hit rate grows strictly with the number of warnings, the critical success index is maximized somewhere inside the range.) What can be faithfully inferred from the above graph is the overall profile of the function together with the large range of thresholds beating the purely deterministic case. The peak itself, about 0.21 for scratch training and 0.22 after pretraining, around the 80th percentile, was computed on the test set and is therefore somewhat optimistic. One should consider the preselected thresholds of 0.9 and their corresponding critical success indexes (0.170 for scratch training and 0.187 after pretraining) as conservative estimates and use the validation set to find the operating point in production. In these terms, the bottom line is the same: the decision-maker picks the threshold, not the algorithm.

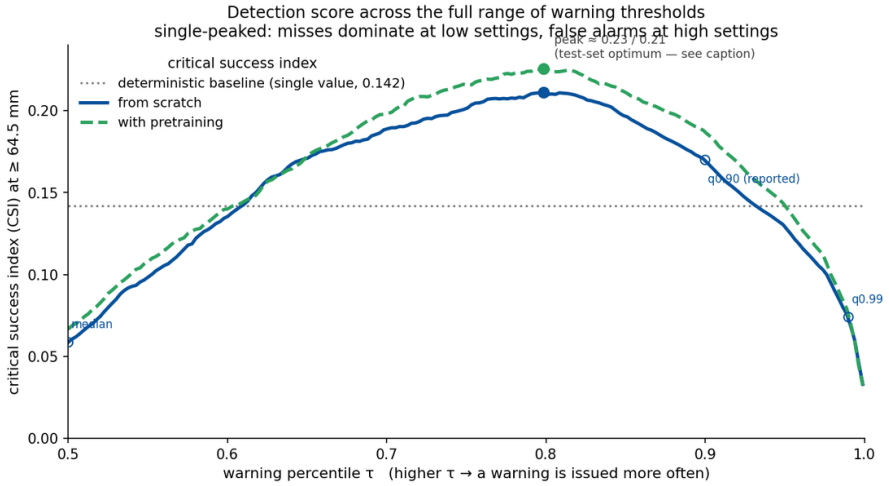


Figure 9.6. The critical success index for heavy rain ($\geq$ 64.5 mm) as the warning percentile τ is swept from the median to the 99.9th percentile (x-axis; a higher τ issues a warning more often), for the from-scratch and pretrained models; the dotted line is the deterministic baseline's single value (0.142), and the three hollow markers are the median, 90th- and 99th-percentile settings of Figure 9.5.

9.5 The point-forecast trade-off

There is an honest cost, and it is in point accuracy. Collapsing the distribution back to a single number by taking its median gives a root mean squared error of about 9.0 mm/day, a little above the deterministic 8.78 — because rainfall is right-skewed, the median sits below the mean, and root mean squared error rewards the mean. By the more robust mean absolute error the median forecast is in fact the best in the study (about 3.5 against the deterministic 3.9), but on squared error it pays a small penalty. This is illustrated in the context of the distributional score, where the CRPS amounts to 2.50 mm, significantly below the mean absolute error of the deterministic approach, which stands at 3.9. This suggests that the forecasted distribution as a whole contains much more information compared to any point prediction. This means that it comes at the cost of just a tiny bit of square error per day, but it is worth it due to its skill in predicting the worst days.

9.6 Extremes resolved as risk

This wraps up the argument. In Chapter 7, the deterministic ceiling was set by predicting a single value using squared error. Chapter 8 showed that you can't lift this ceiling just by tweaking the encoder. Then, in this chapter, we raised the ceiling

by changing how the output is formulated. Using the same graph-attention setup, but asking for a distribution instead of a point estimate and optimizing with the right loss function, gives us forecasts that actually work for heavy rain—something the old model couldn't do. These forecasts also come with a handy warning level.

Sure, heavy precipitation is still tough to pin down exactly, day to day. But now, we can give informed and truthful statements about the risk. This helps a lot for those rare but really impactful storms.

On a smaller note, the self-supervised pretraining from Chapter 8 is used here too. The results are pretty much the same: the impact is slight and a bit confusing. The pretrained model is a tad sharper and has a bit higher Brier skill. Yet, the model trained from scratch does better in terms of calibration. It scores an average of 0.51 on its Probability Integral Transform, while the pretrained one only hits 0.40. Still, this isn't clearly better or worse overall. So, like before, pretraining doesn't make much of a difference in how well the model performs.

CHAPTER 10

Conclusion

This dissertation poses one major question: Can we reliably forecast daily rainfall, particularly the extremely wet days? If not, then what is the bottleneck and what can we do about it? The answer follows from a bunch of results. Among all the models examined, a graph-based spatio-temporal model works best. Yet, it fails when forecasting the extreme events.

Surprisingly, this isn't because of the model design but due to the actual data and the way it's trained. Adjusting the interpretation component isn't the solution. The key was in redefining the prediction goal. Instead of predicting just single number, the model estimates the whole rainfall range. This change significantly improved the performance, especially for the tail events. Rather than trying to nail down a precise amount—which is super tricky—the model now accurately shows the full spectrum of expected rainfall and related risks. The three questions set out in the introduction are answered along the way.

10.1 Summary of findings

The first question is whether spatio-temporal models, which include both time dynamics and the spatial setup of IMD stations, would give a better daily rainfall forecasts than simpler models. Evidence shows they do. Chapter 4 tests this, starting with a time-only method giving 15.82 mm/day, then moving to a graph-attention model with a recurrent core that drops the error to 8.78 mm/day. Most of the boost comes from the spatial graph structure and the attention mechanism on neighboring stations. This model is the best deterministic tool in the study and the reference for everything after it.

The second question was whether the source of the input data changes what is achievable. With the usual setting, the same model produces a figure of 17.60 mm/day using the gauge method while it gives 10.03 mm/day using the reanalysis data set.

The difference cannot be due to better predictive power using reanalysis data but rather because the reanalysis data is smoother and more artificial. In particular, it tends to overestimate rainfall occurrence (on 51% of days against only 30% for gauging), underpredict extremes, and show little consistency with real daily gauge observations (correlation about 0.3).

In Chapter 6, the model got expanded to cover both flat and mountainous areas, giving mixed results. The one consistent thing was that all three regions still nailed the heaviest downpours, with nowhere nailing the very heavy forecasts correctly. Yet, the expected drop-off with tougher terrain didn't happen.

The core finding here is that in the reanalysis, the ground features signal is way weaker. Rainfall barely rises with elevation in the Himalayas (only a 0.09 correlation, while in West Bengal it's 0.48). This missing trend backs up the smoothing spotted in Chapter 5. Adding elevation back in, even just statically, brings some skill back (going from 7.99 to 7.71 mm/day in the Himalayas).

The third question was whether the heavy-rain events, which the standard approach handles badly, can be predicted more usefully. This is the heart of the thesis. Chapter 7 diagnosed the limit: the best model behaves almost exactly like an optimal estimator of each day's average rainfall, and under a squared-error objective such an estimator cannot place the rare heavy days — its predictions cap near 70 mm while observations run far higher, and roughly three-fifths of all squared error sits in the small minority of heaviest days. Chapter 8 showed the limit is not in the encoder: enriching it with static features, more depth, two forms of self-supervised pretraining, auxiliary tasks and a three-stage pipeline left the error essentially where it started, so the bottleneck is the output and the loss. Chapter 9 acted on exactly that. We keep the same encoder but change only the output to a set of ordered quantiles using the pinball loss. This results in calibrated forecasts – the probability integral transform averages at 0.51, very close to the ideal 0.5. Our setup shows positive skill against climatology across all heavy thresholds, with a Brier skill score between about 23% and 8% for models trained from scratch, and from 24% to 9% for pretrained ones. Additionally, it gets a continuous ranked probability score of 2.50 mm, way lower than the deterministic model's mean absolute error of 3.9, and allows setting a user-defined warning level. However, there's a slight rise in errors on typical days; the median root mean squared error is around 9.0, up from the deterministic 8.78. While the model doesn't precisely predict millimeter amounts during heavy rain, it shifts focus to well-calibrated risk estimates, which is much more useful for those rare yet critical rainfall events.

10.2 Limitations

The main issue revolves around the input data. The model is trained and evaluated using reanalysis data, which, as shown in Chapters 5 and 6, is just a smoothed proxy. The data set tends to have a bias toward an overestimation of event occurrences while understating their intensities. Hence, some of the errors associated with heavy rainfall do not stem from the model but from the data. The deterministic outcomes seem to be a consequence of models built using such data and not a yardstick for all rainfall predictions.

Another problem is with the computing aspect. For each setup, there was only one shot at training with a single random seed. This implies that the level of possible variance in subsequent attempts remains unclear. This means it becomes hard to tell for sure which changes are beneficial, like pre-training rather than joint training. Cross regional testing also faces challenges because it is limited by its sole dependence on reanalyzed data, and differences in number of observation stations, among other complications. Another challenge posed to inter-regional comparison is the different rain climates in different regions.

Fourthly, the matter of evaluation deserves some discussion. While it is true that probabilistic forecasts are subjected to rigorous evaluation, it would have been more advantageous to conduct a more thorough validation that includes other thresholds besides those used here, in addition to spatial analysis. In addition, it must be noted that the upper part of the warning curve is taken from the test data set, leading to potentially overly optimistic evaluation. It is best practice that an independent validation dataset be used to ascertain the operating point of the method in use.

10.3 Future work

The next clearly delineated step entails multi-seed training to obtain error bars in order to verify marginal effects. Separating out model errors and input errors becomes an important task: training and testing on gauge- or bias-corrected observations, in addition to those from the regional study, would help determine the degree to which the heavy-day limit is a problem with the input data and not with the approach. The probabilistic framework allows a broader scope of application: region-specific calibrations, other threshold values, spatial verification, and selected warning level via validation; additionally, this model fits nicely into the rest of the paper by

complementing the distribution with the elevation variable used in the Himalayas case study, and with the other loss functions (both focal and transformed). Testing under longer lead times, other regions, and different neural network architectures could assess the robustness of this approach. The key message of this paper is methodological in nature and, hence, is generalizable: if a squared-error predictor does not account for extremes, the distribution can be calibrated to treat them as risk, which may be more helpful and more honest.

References

[1] Deep learning based forecasting of Indian summer monsoon rainfall, 2021. arXiv:2107.04270; authors / venue to confirm.

[2] M. I. Khan and R. Maity. Hybrid deep-learning (1D-CNN + MLP) approach for daily rainfall prediction over India. 2020. full reference to confirm.

[3] J. Adhikary and R. Maiti. Long-Term Spatio-Temporal Forecasting of Monthly Rainfall in West Bengal Using Ensemble Learning Approaches, 2025. preprint, unpublished.

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

[5] K. Cho et al. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *EMNLP*, 2014.

[6] X. Shi, 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 *NeurIPS*, 2015. arXiv:1506.04214.

[7] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[8] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph Attention Networks. In *ICLR*, 2018.

[9] B. Yu, H. Yin, and Z. Zhu. Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In *IJCAI*, 2018.

[10] Y. Li, R. Yu, C. Shahabi, and Y. Liu. Diffusion convolutional recurrent neural network: data-driven traffic forecasting. In *ICLR*, 2018.

[11] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang. Graph WaveNet for deep spatial-temporal graph modeling. In *IJCAI*, 2019.

[12] Z. Wu et al. Connecting the dots: multivariate time series forecasting with graph neural networks (MTGNN). In *KDD*, 2020.

[13] Z. Shao et al. Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting (STEP). In *KDD*, 2022. arXiv:2206.09113.

[14] STD-MAE: Spatial-temporal-decoupled masked pre-training for spatio-temporal forecasting. In *IJCAI*, 2024. arXiv:2312.00516; authors to confirm.

[15] GPT-ST: Generative pre-training of spatio-temporal graph neural networks. In *NeurIPS*, 2023. arXiv:2311.04245; authors to confirm.

[16] W-MAE: Pre-trained weather model with masked autoencoder, 2023. authors to confirm.

[17] SpaT-SparK: Self-supervised spatial-temporal learner for precipitation nowcasting, 2024. authors to confirm.

[18] STGCL: When do contrastive learning signals help spatio-temporal graph forecasting? In *SIGSPATIAL*, 2022. arXiv:2108.11873; authors to confirm.

[19] D. S. Pai et al. Development of a new high spatial resolution ($0.25^\circ \times 0.25^\circ$) long period (1901–2010) daily gridded rainfall data set over India. *MAUSAM*, 65(1):1–18, 2014.

[20] H. Hersbach et al. The ERA5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730):1999–2049, 2020.

[21] J. Muñoz-Sabater et al. ERA5-Land: a state-of-the-art global reanalysis dataset for land applications. *Earth System Science Data*, 13(9):4349–4383, 2021.

[22] D. A. Lavers et al. An evaluation of ERA5 precipitation. 2022. full reference to confirm.

[23] R. Koenker and G. Bassett. Regression quantiles. *Econometrica*, 46(1):33–50, 1978.

[24] J. E. Matheson and R. L. Winkler. Scoring rules for continuous probability distributions. *Management Science*, 22(10):1087–1096, 1976.

[25] T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

[26] G. W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, 1950.

[27] A. P. Dawid. Statistical theory: the prequential approach. *Journal of the Royal Statistical Society: Series A*, 147(2):278–292, 1984.

[28] T. Gneiting, F. Balabdaoui, and A. E. Raftery. Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society: Series B*, 69(2):243–268, 2007.

[29] D. S. Wilks. *Statistical Methods in the Atmospheric Sciences*. Academic Press, 3rd edition, 2011.