Machine Learning with Applications 7 (2022) 100204

Contents lists available at ScienceDirect

Machine Learning with Applications

journal homepage: www.elsevier.com/locate/mlwa

# Rainfall prediction: A comparative analysis of modern machine learning algorithms for time-series forecasting

Ari Yair Barrera-Animas, Lukumon O. Oyedele *, Muhammad Bilal, Taofeek Dolapo Akinosho, Juan Manuel Davila Delgado, Lukman Adewale Akanbi

*Big Data Enterprise and Artificial Intelligent Lab (Big-DEAL), Bristol Business School, University of the West of England Bristol, United Kingdom*

ARTICLE INFO

*Keywords:*
Rainfall prediction
LSTM Networks
Multivariate time-series
Multi-step forecast
Time-series data

ABSTRACT

Rainfall forecasting has gained utmost research relevance in recent times due to its complexities and persistent applications such as flood forecasting and monitoring of pollutant concentration levels, among others. Existing models use complex statistical models that are often too costly, both computationally and budgetary, or are not applied to downstream applications. Therefore, approaches that use Machine Learning algorithms in conjunction with time-series data are being explored as an alternative to overcome these drawbacks. To this end, this study presents a comparative analysis using simplified rainfall estimation models based on conventional Machine Learning algorithms and Deep Learning architectures that are efficient for these downstream applications. Models based on LSTM, Stacked-LSTM, Bidirectional-LSTM Networks, XGBoost, and an ensemble of Gradient Boosting Regressor, Linear Support Vector Regression, and an Extra-trees Regressor were compared in the task of forecasting hourly rainfall volumes using time-series data. Climate data from 2000 to 2020 from five major cities in the United Kingdom were used. The evaluation metrics of Loss, Root Mean Squared Error, Mean Absolute Error, and Root Mean Squared Logarithmic Error were used to evaluate the models' performance. Results show that a Bidirectional-LSTM Network can be used as a rainfall forecast model with comparable performance to Stacked-LSTM Networks. Among all the models tested, the Stacked-LSTM Network with two hidden layers and the Bidirectional-LSTM Network performed best. This suggests that models based on LSTM-Networks with fewer hidden layers perform better for this approach; denoting its ability to be applied as an approach for budget-wise rainfall forecast applications.

## 1. Introduction

Rainfall remains one of the most influential meteorological parameters in many aspects of our daily lives. With effects ranging from damage to infrastructure in the event of a flood to disruptions in the transport network, the socio-economic impacts of rainfall are noteworthy (Le, Pham, Ly, Shirzadi and Le, 2020). Floods and similar extreme events are consequences of climate change that are expected to occur more frequently and have catastrophic effects in years to come (Yucel, Onen, Yilmaz, & Gochis, 2015). More interestingly, recent studies have highlighted that weather conditions can potentially increase air pollution (another major topic of discourse alongside climate change in recent times) in winter and summer periods (Czarnecka, Nidzgorska-Lencewicz, et al., 2011). It is pertinent to reiterate that increased air pollution results in health conditions such as asthma and similar problems related to the lungs (Mokrani, Lounas, Bennai, Salhi, & Djerbi, 2019; Zadtootaghaj, Mohammadian, Mahbanooei, & Ghasemi, 2019). Therefore, as a mitigation approach, many studies have investigated and proposed rainfall forecasting techniques in preparation for any eventuality. However, in order to enhance human mobility activities (Salman, Heryadi, Abdurahman, & Suparta, 2018; Xingjian et al., 2015) and enhance agriculture and industrial development (Aguasca-Colomo, Castellanos-Nieves, & Méndez, 2019; Chao, Pu, Yin, Han, & Chen, 2018; Kim, Hong, Joh, & Song, 2017; Kumar, Singh, Samui, & Jha, 2019; Poornima & Pushpalatha, 2019; Zhang, Zhu, Zhang, Ye, & Yang, 2018), these approaches must provide efficient and timely predictions.

Rainfall forecasting has been around for years using traditional methods that employ statistical techniques to assess the correlation between rainfall, geographic coordinates (such as latitude and longitude), and other atmospheric factors (like pressure, temperature, wind speed, and humidity). However, the complexity of rainfall such as its non-linearity makes it difficult to predict (Wu & Chau, 2013). Consequently, attempts have been made to reduce this non-linearity by using Singular Spectrum Analysis, Empirical Mode Decomposition, Wavelet analysis,

* Corresponding author.
*E-mail addresses:* ari.barreraanimas@uwe.ac.uk (A.Y. Barrera-Animas), l.oyedele@uwe.ac.uk (L.O. Oyedele), muhammad.bilal@uwe.ac.uk (M. Bilal), taofeek.akinosho@uwe.ac.uk (T.D. Akinosho), manuel.daviladelgado@uwe.ac.uk (J.M.D. Delgado), lukman.akanbi@uwe.ac.uk (L.A. Akanbi).

https://doi.org/10.1016/j.mlwa.2021.100204
Received 23 August 2021; Received in revised form 14 October 2021; Accepted 28 October 2021
Available online 11 November 2021

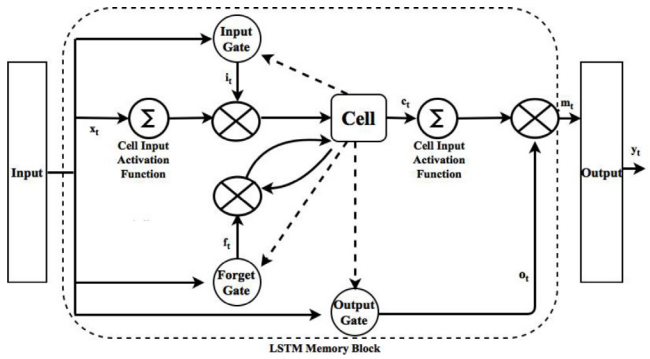



**Fig. 1.** LSTM block diagram (Singh, Chauhan, Krishnamachari, & Vig, 2015).

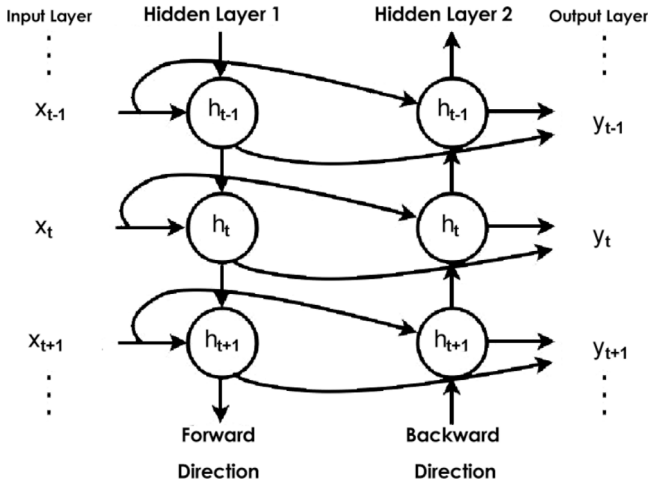



**Fig. 2.** Bidirectional-LSTM Network architecture (Singh et al., 2015).

**Table 1**
Tuples of highly correlated features in the five cities.

| Tuples of features | | | |
|---|---|---|---|
| Number | Tuple | Number | Tuple |
| 1 | Latitude - Longitude | 5 | Min Temp - Max Temp |
| 2 | Temp - Min Temp | 6 | Min Temp - Feels Like |
| 3 | Temp - Max Temp | 7 | Max Temp - Feels Like |
| 4 | Temp - Feels Like | 8 | Snow 1h - Snow ID 601 |

among others (Gan, Sun, Wang, & Wei, 2018; Xiang, Gou, He, Xia, & Wang, 2018). Nevertheless, the mathematical and statistical models employed require complex computing power (Singh & Borah, 2013) and can be time-consuming with minimal effects.

Thence, the use of Artificial Neural Networks (ANNs) as rainfall forecasting models has widely captured the attention of researchers (Liu, Zou, Liu, & Linge, 2019; Singh & Borah, 2013). One factor that played an important role in the widespread use of ANNs in this field of application was the emergence of wireless technologies, such as the Internet of Things, which accelerated the development of inexpensive and effective solutions for capturing satellite imagery and historical radar data. Another relevant factor that contributed to the growth of the use of ANNs as an approach to forecast rainfall is its capability to address non-linearity in rainfall data and that they require little or no knowledge of the relationships between the variables that are being considered (Liu et al., 2019). However, precipitation forecasting poses an even more challenging task, as it is often hindered by spatial and temporal variation of regional rainfalls (Hossain, Rasel, Imteaz, & Mekanik, 2020). In this sense, a variant of ANNs known as Recurrent Neural Networks (RNNs) is the most appropriate to face these types of challenges (Balluff, Bendfeld, & Krauter, 2020).

**Table 2**
Weather features used as predictors.

| Number | Name | Description | Units |
|---|---|---|---|
| 1 | Temperature | Temperature at the recorded hour | Kelvins (K) |
| 2 | Pressure | Atmospheric pressure at the recorded hour | Hectopascal (hPa) |
| 3 | Humidity | Amount of water vapour in the air at the recorded hour | Percentage (%) |
| 4 | Wind speed | Average air speed at the recorded hour | metres/second (m/s) |
| 5 | Wind Degrees | Direction from which air blows at the recorded hour | Degrees (°) |
| 6 | Clouds coverage | Fraction of sky covered by clouds at the recorded hour | Percentage (%) |
| 7 | Timezone | Code of the local time of a region where the measurement occurs | Real integers numbers |
| 8 | Snow 3h | Snow volume during the last 3 h before the recorded hour | millimetres (mm) |
| 9 | Rain 3h | Rain volume during the last 3 h before the recorded hour | millimetres (mm) |
| 10 | Snow 1h | Snow volume during the last 1 h before the recorded hour | millimetres (mm) |
| 11 | Rain 1h | Rain volume during the last 1 h before the recorded hour | millimetres (mm) |

**Table 3**
Structure of the feature vectors of the five weather datasets.

| Predictor features | Target feature |
|---|---|
| Features 1 to 11 of Table 2 or $X(o_n)(x_m)_{m=1}^{11}$ | Feature 11 of Table 2 shifted one position forward or $X(o_n)(x_{12})$ |

Originally adopted for natural language processing and time series modelling, RNNs are now being explored for meteorological time-series (Ramos, Del Alamo, & Zapana, 2019). RNNs are able to handle temporal dynamics by adopting feedback connections that allow them to remember information previously fed into their architecture (Elman, 1990). However, a limitation of the architecture of RNNs is its inability to learn and make long-term forecasts (Ni et al., 2020). To overcome this restriction, a variation of the ANNs called Long short-term memory (LSTM) Networks has been developed with the inclusion of memory cells that regulate the passage of information in and out of its cells (Greff, Srivastava, Koutník, Steunebrink, & Schmidhuber, 2016). Studies such as (Kratzert, Klotz, Brenner, Schulz, & Herrnegger, 2018; Yunpeng, Di, Junpeng, & Yong, 2017) suggest the superiority of LSTM for multi-step ahead predictions.

As can be noticed, although the task of forecasting the volume of rainfall can be addressed through mathematical models, the use of diverse types of ANNs stands out as an alternative that allows the development of less computationally expensive forecasting approaches. Notwithstanding, as in any other field of application of Machine Learning, there is no "free lunch"; therefore, the use and performance of forecasting models depend on a plethora of design decisions such as the available data, the aim/approach of the prediction tasks, and the models' implementation (Hutter, Kotthoff, & Vanschoren, 2019).

Therefore, this study seeks to investigate the suitability of three variants of LSTM-Network architectures for the task of forecasting rainfall volume per hour when compared to modern Machine Learning approaches. In particular, this study compares the performance of models based on LSTM, Stacked-LSTM and Bidirectional-LSTM Networks

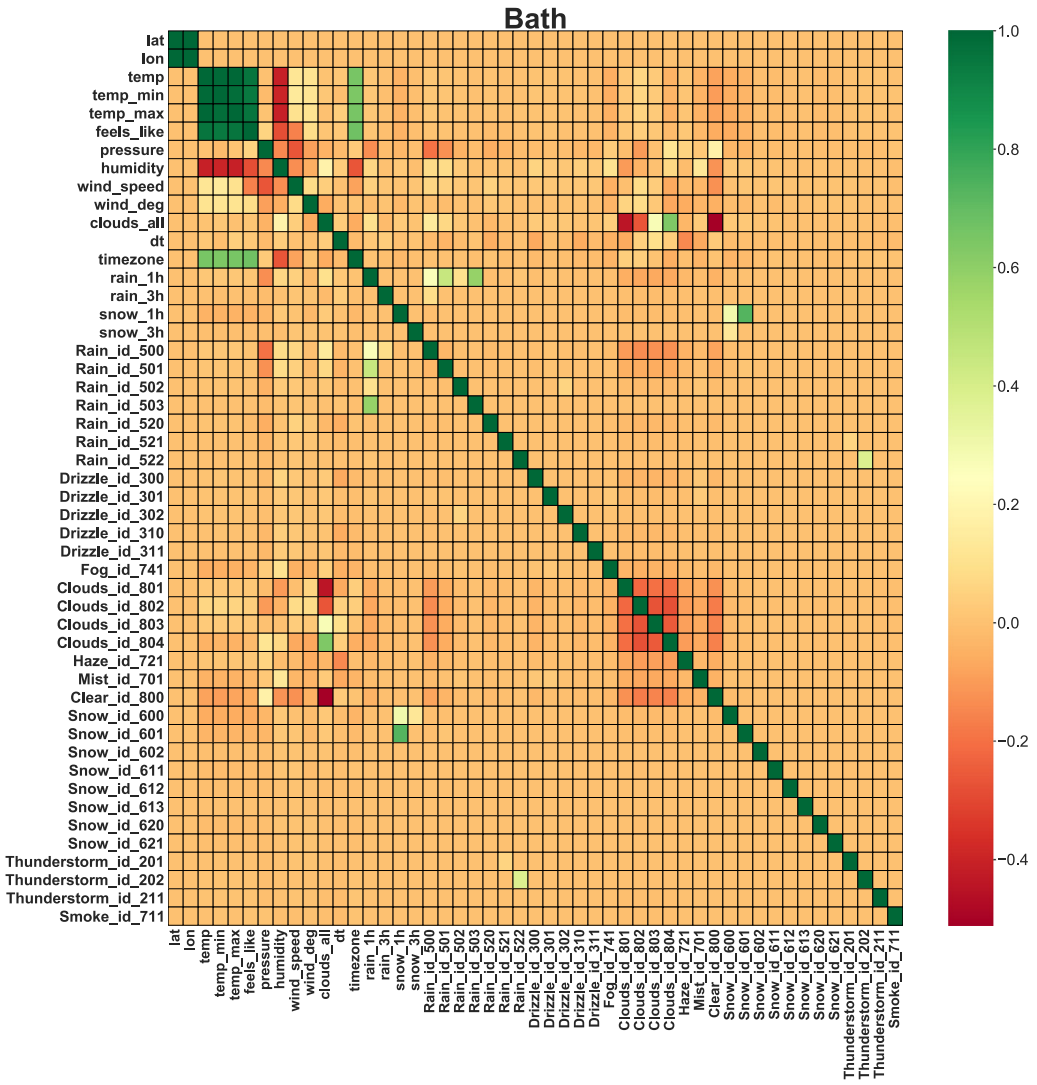



**Fig. 3.** Correlation Matrix of the city of Bath.

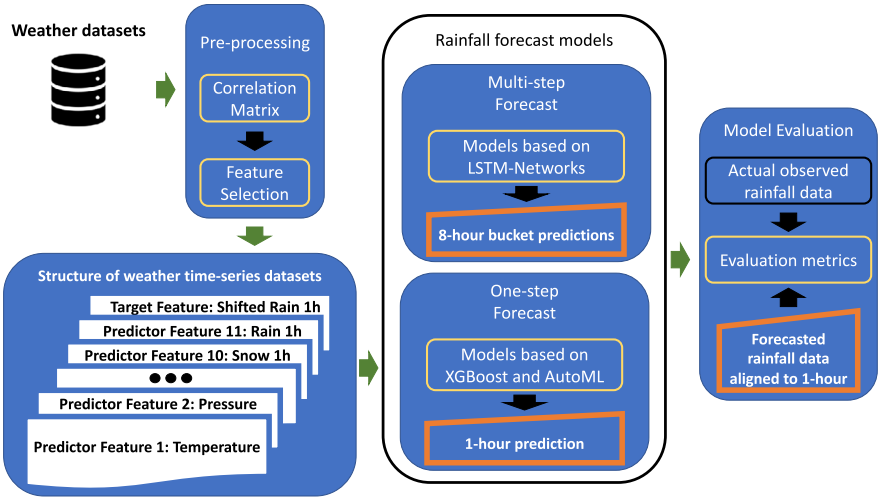



**Fig. 4.** Experimental methodology.

with an XGBoost decision trees model and a proposed model that will result from the use of an Automated Machine Learning (AutoML) tool.

AutoML (Hutter et al., 2019) is an approach that provides researchers with a tool to automatically and data-driven search for an

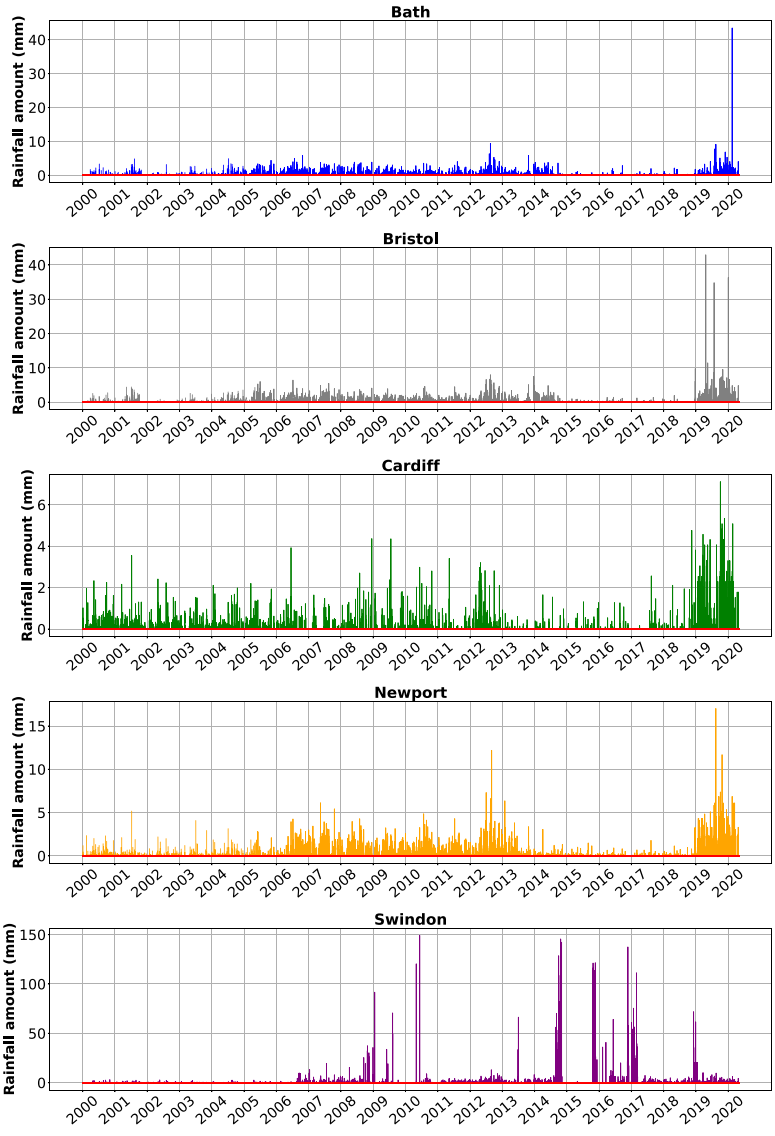



**Fig. 5.** Rainfall volume in the datasets for the five UK cities.

**Table 4**
Model 1 (a version of Stacked-LSTM Network) - fixed parameters and hyperparameters values.

| Model 1 based on Kim and Bae (2017) | | | |
|---|---|---|---|
| Parameter/hyperparameter | Value | Parameter/hyperparameter | Value |
| Training set % | 67 | Epochs steps | 200 |
| Validation set % | 17 | No. of epochs | 20 |
| Testing set % | 16 | No. of hidden layers | 10 |
| No. of memory cells | 1 | Backward | No |
| Time steps | 15 | | |

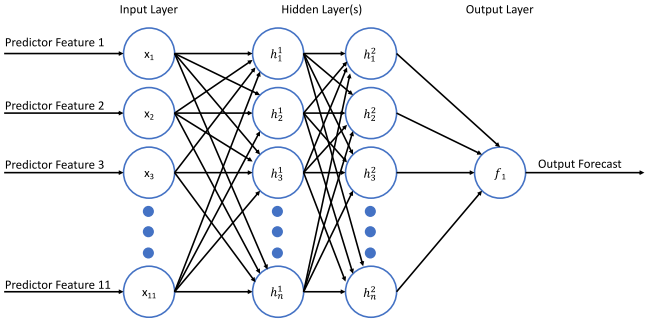



**Fig. 6.** General Neural Network architecture.

algorithm and its hyperparameter values that work best for a particular application. This tool has proven to be useful both for new scientists by giving good results in their exploratory analyses, and for expert researchers by providing new insights about the problem they are investigating (Hutter et al., 2019). Since AutoML performs an intelligent search on a diverse set of Machine Learning algorithms in a reasonable amount of time, it will be used to obtain a regression model, and its hyperparameter values, that performs better in the task of forecasting the volume of rainfall over a given data set.

On the other hand, despite the fact that AutoML tools implement the XGBoost algorithm in their test procedure; the model recommended by these tools also depends on the configured hyperparameter search space (Le, Fu and Moore, 2020). Therefore, building a baseline algorithm, in this case the XGboost model, is always part of good practice when designing experimental procedures.

Regarding the model based on Bidirectional-LSTM Networks, it is worth mentioning that this type of RNNs uses information from both the past and the future of sequential data to obtain a deeper insight

**Table 5**
Model 2 (a version of LSTM Network) - fixed parameters and hyperparameters values.

| Model 2 based on Kumar et al. (2019) | | | |
|---|---|---|---|
| Parameter/hyperparameter | Value | Parameter/hyperparameter | Value |
| Training set % | 70 | Epochs steps | 200 |
| Validation set % | 15 | No. of epochs | 500 |
| Testing set % | 15 | No. of hidden layers | 1 |
| No. of memory cells | 1 | Backward | No |
| Time steps | 12 | | |

**Table 6**
Model 3 (a version of LSTM Network) - Fixed parameters and hyperparameters values.

| Model 3 based on Aswin, Geetha, and Vinayakumar (2018) | | | |
|---|---|---|---|
| Parameter/ Hyperparameter | Value | Parameter/ Hyperparameter | Value |
| Training set % | 70 | Epochs Steps | 200 |
| Validation set % | 15 | No. of Epochs | 100 |
| Testing set % | 15 | No. of Hidden Layers | 1 |
| No. of Memory cells | 32 | Backward | Yes |
| Time steps | 12 | | |

**Table 7**
Hyperparameter values and performance of XGBoost prediction models.

| | Bath | Bristol | Cardiff | Newport | Swindon |
|---|---|---|---|---|---|
| No. of estimators | 28 | 30 | 27 | 32 | 30 |
| Subsample ratio of features | 0.5 | 0.7 | 0.5 | 0.5 | 0.9 |
| Subsample ratio of training | 0.8 | 0.5 | 0.6 | 1.0 | 0.5 |
| Minimum sum of instance weight | 10 | 12 | 12 | 12 | 12 |
| Maximum depth of a tree | 6 | 6 | 8 | 6 | 6 |
| Learning Rate | 0.3 | 0.3 | 0.3 | 0.3 | 0.3 |
| RMSE↓ | 0.41 | 0.62 | 0.23 | 0.31 | 1.15 |
| MAE↓ | 0.05 | 0.08 | 0.04 | 0.07 | 0.21 |
| RMSLE↓ | 0.10 | 0.15 | 0.12 | 0.14 | 0.24 |

**Table 8**
Performance of the ensemble prediction model.

| | Bath | Bristol | Cardiff | Newport | Swindon |
|---|---|---|---|---|---|
| RMSE↓ | 0.41 | 0.62 | 0.23 | 0.31 | 0.93 |
| MAE↓ | 0.04 | 0.07 | 0.04 | 0.07 | 0.18 |
| RMSLE↓ | 0.10 | 0.14 | 0.12 | 0.13 | 0.21 |

**Table 9**
Performance of prediction Models 1, 2 , and 3.

| Model | City | Loss↓ | RMSE↓ | MAE↓ | RMSLE↓ |
|---|---|---|---|---|---|
| 1 | Bath | 0.0001 | 0.0097 | 0.0013 | 0.0025 |
| | Bristol | 0.0002 | 0.0149 | 0.0019 | 0.0039 |
| | Cardiff | 0.0013 | 0.0360 | 0.0065 | 0.0127 |
| | Newport | 0.0005 | 0.0231 | 0.0045 | 0.0088 |
| | Swindon | 0.0001 | 0.0107 | 0.0016 | 0.0030 |
| 2 | Bath | 0.0001 | 0.0103 | 0.0011 | 0.0028 |
| | Bristol | 0.0002 | 0.0157 | 0.0020 | 0.0047 |
| | Cardiff | 0.0014 | 0.0375 | 0.0065 | 0.0138 |
| | Newport | 0.0006 | 0.0241 | 0.0046 | 0.0098 |
| | Swindon | 0.0001 | 0.0083 | 0.0014 | 0.0028 |
| 3 | Bath | 0.0001 | 0.0120 | 0.0017 | 0.0051 |
| | Bristol | 0.0003 | 0.0172 | 0.0028 | 0.0076 |
| | Cardiff | 0.0015 | 0.0383 | 0.0077 | 0.0175 |
| | Newport | 0.0006 | 0.0247 | 0.0049 | 0.0113 |
| | Swindon | 0.0001 | 0.0099 | 0.0019 | 0.0051 |

into the context surrounding a required prediction. For this reason, this RNNs architecture is suitable for applications where past and future records can be accessed when making a forecast, such as in the Natural Language Processing field. Nevertheless, this research work explores the feasibility of using future sequential data as a subsequent readjustment process once a forecast has been made during the training process.

**Table 10**
Performance of prediction models 4, 5, and 6.

| Model | City | Loss↓ | RMSE↓ | MAE↓ | RMSLE↓ |
|---|---|---|---|---|---|
| 4 | Bath | 0.0001 | 0.0104 | 0.0013 | 0.0037 |
| | Bristol | 0.0003 | 0.0158 | 0.0023 | 0.0058 |
| | Cardiff | 0.0014 | 0.0375 | 0.0071 | 0.0157 |
| | Newport | 0.0006 | 0.0241 | 0.0047 | 0.0104 |
| | Swindon | 0.0001 | 0.0084 | 0.0016 | 0.0038 |
| 5 | Bath | 0.0001 | 0.0101 | 0.0011 | 0.0028 |
| | Bristol | 0.0002 | 0.0155 | 0.0019 | 0.0046 |
| | Cardiff | 0.0014 | 0.0374 | 0.0069 | 0.0148 |
| | Newport | 0.0006 | 0.0240 | 0.0045 | 0.0096 |
| | Swindon | 0.0001 | 0.0080 | 0.0013 | 0.0027 |
| 6 | Bath | 0.0001 | 0.0109 | 0.0015 | 0.0044 |
| | Bristol | 0.0003 | 0.0162 | 0.0024 | 0.0065 |
| | Cardiff | 0.0014 | 0.0377 | 0.0072 | 0.0164 |
| | Newport | 0.0006 | 0.0243 | 0.0048 | 0.0111 |
| | Swindon | 0.0001 | 0.0099 | 0.0021 | 0.0057 |

From four commonly adopted forecasting strategies — Direct Multi-step forecast strategy, Recursive Multi-step forecast strategy, Direct Recursive Hybrid strategy, and Multiple output strategy — this study adopts the multiple output forecast strategy for the models based on LSTM Networks. This forecasting strategy allows a model developed for a particular region to forecast a sequence of values in a one-shot process. Weather data from five cities in the United Kingdom (UK) were used to train and validate the rainfall prediction models.

This study is the first step in a series of research aimed at forecasting the air quality of a region in a multi-step fashion based on weather parameters and pollutant concentration levels. A robust air pollution model would require forecasted weather parameters, emission factors, background concentration, traffic flow, and geographic terrain to improve forecast accuracy.

To the authors' knowledge, this study is the first to present a comparative analysis of the performance of rainfall forecasting models based on modern Machine Learning algorithms in predicting hourly rainfall volume using weather time-series data from cities in the United Kingdom. In summary, the main contributions to the knowledge of this study are:

- From the literature reviewed, three models based on the LSTM and Stacked-LSTM Networks were adapted for the task of forecasting hourly rainfall using time-series data from five major UK cities.
- A model based on Bidirectional-LSTM Networks was proposed for the task of forecasting rainfall on an hourly basis using time-series data from five major UK cities.
- A comparison of the performance of models based on LSTM-Networks, Stacked-LSTM Networks, Bidirectional-LSTM Networks, XGBoost, and the resulting model from AutoML was performed in the task of forecasting the amount of rainfall per hour using time-series data from five major UK cities.

The rest of the paper is structured as follows: First, research works that use Feed-forward Back-propagation Neural Networks and LSTM-Networks architectures to build rainfall prediction models are introduced in Section 2. Thereupon, in Section 3 a brief introduction of Neural Networks, Recurrent Neural Networks, and its architectures used in this research work is provided. Then, a description of the weather dataset and the pre-processing process performed to prepare it for experimentation is given in Section 4. Afterwards, in Section 5 the methodology followed to carry out the experiments is described. This

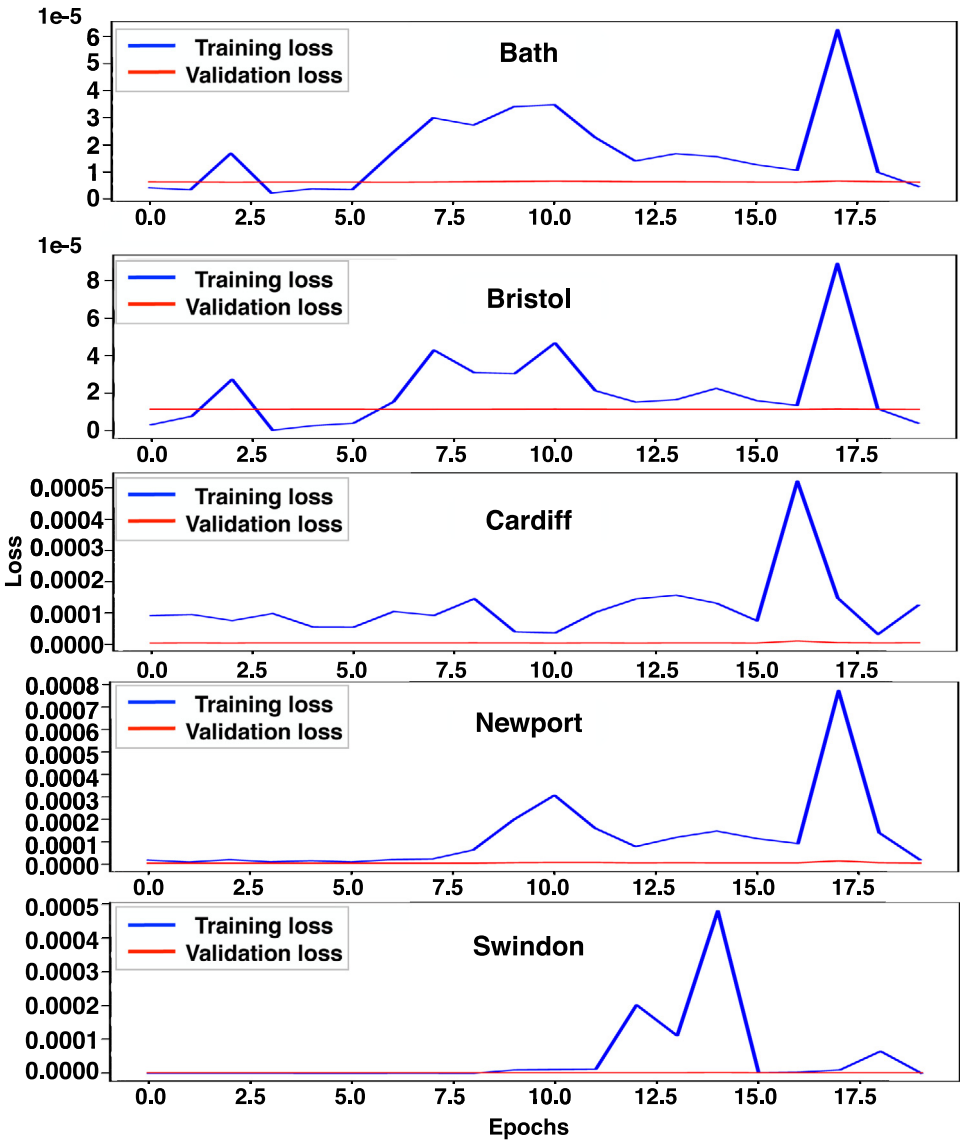



**Fig. 7.** Training and validation loss - Model 1.

description includes both the structure of the datasets used in the training and testing processes to forecast hourly rainfall values, as well as the architectures and parameters of the different implemented models based on XGBoost, AutoML, LSTM, Stacked-LSTM, and Bidirectional-LSTM networks. Later on, results obtained are presented and discussed in Section 6. Finally, in Section 7, conclusions and further research steps are highlighted.

## 2. Related work

This section gives an introduction to research works that use time-series data to train rainfall forecast models based on Feed-forward Back-propagation Neural Networks, LSTM-Networks, and Stacked-LSTM Networks.

Singh and Borah (2013), trained five architectures of a Feed-forward Back-propagation Neural Network algorithm containing only three layers (1 input, 1 hidden, and 1 output layer) to forecast the mean rainfall of the summer monsoon in India on a monthly and seasonal basis. The authors provide the prediction of rainfall amounts by combining the results given by the five trained Neural Networks. Only monthly values of rainfall and seasonal rainfall were used from two sources from the period 1871 to 2010, a dataset from a literature reviewed work and a dataset from the Indian Institute of Tropical Meteorology. Results showed that their ensemble approach performed better on the evaluation metrics of Means, Standard Deviations, Correlation Coefficient, Root Mean Square Error (RMSE), and Performance Parameter compared to a related work that uses a more complex Feed-forward Back-propagation Neural Network.

Kim and Bae proposed an LSTM-Networks model to forecast one hour of rainfall into the future (Kim & Bae, 2017). To train and validate the forecast model, weather data from 2012 from Gangneung, Gangwon-do region (Korea) was used. The climatic features that integrate the weather dataset were temperature, wind speed, humidity, and sea surface pressure. Moreover, the lag characteristics of the amount of rainfall in the current and past hours of observation were considered. In the first phase of experimentation, the proposed LSTM-Networks model was compared with an Artificial Neural Network (ANN) model. The results of this experiment showed that the LSTM-Networks model achieves better values for the RMSE evaluation metric. A second phase of experimentation was carried out including the measurement of water vapour as a feature. However, the RMSE plotted values throughout the training epochs showed an over-fitting behaviour for the training data.

Later, Chao et al. (2018) compared five models based on Auto-regressive and Moving Average, Random Forest, Back-propagation Neural Networks, Support Vector Machines, and LSTM-Networks in the

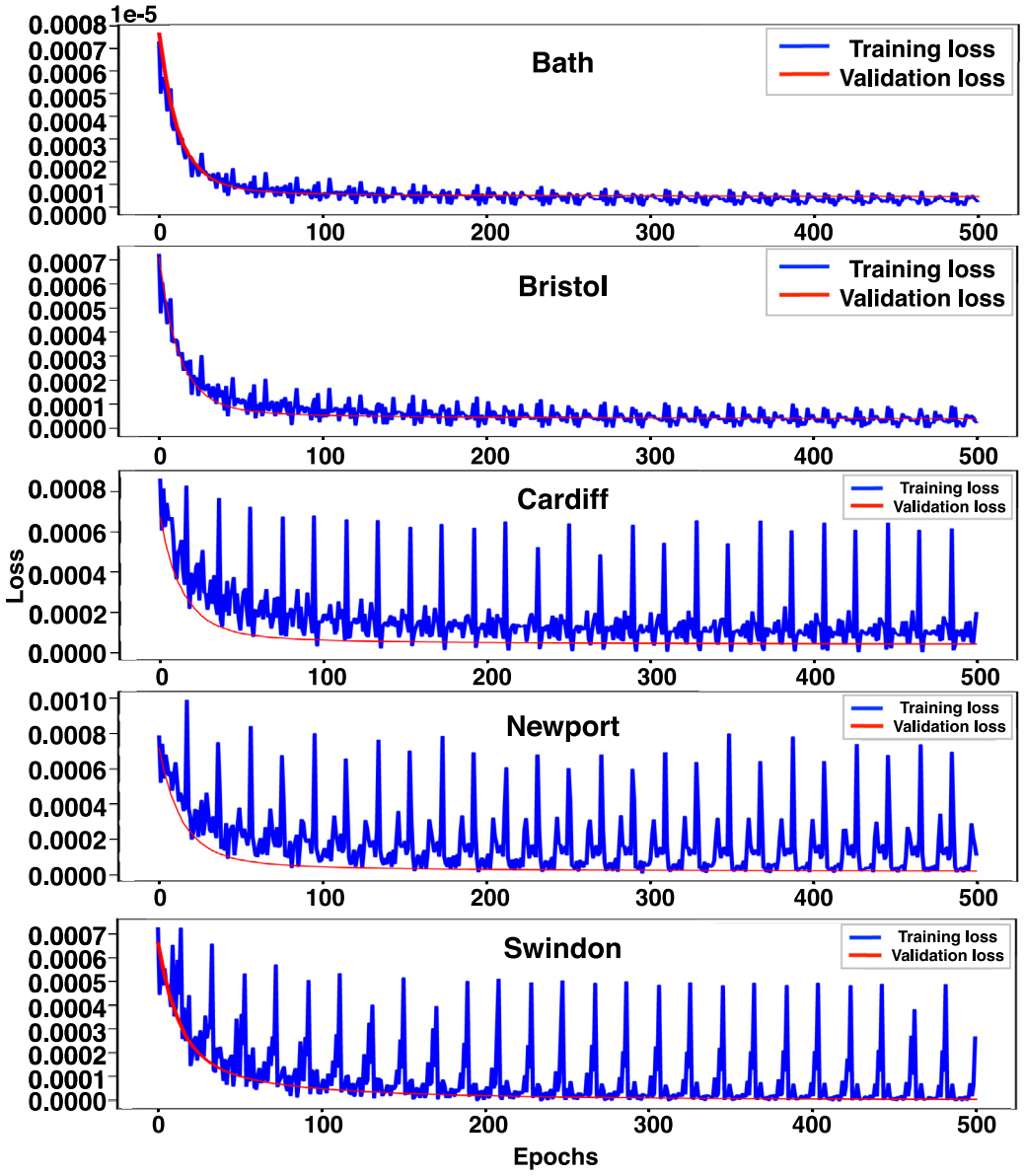



**Fig. 8.** Training and validation loss - Model 2.

task of predicting rainfall amounts in five, 10, and 15 min into the future. Weather data from 2015 and 2016 from the Wuhan region of China were used to test the forecast models. Wind speed, wind direction, temperature, humidity, pressure, amount of rainfall, and radiation were the features concentrated in the dataset. Results showed that the forecast model based on LSTM-Networks achieved the best values in the RMSE and Mean Absolute Error (MAE) metrics.

On the other hand, Aswin et al. (2018) proposed an approach that uses an LSTM-Networks model and a ConvNet model to perform monthly rainfall predictions. Microwave data, infrared data, and rain gauge measurements were used to extract precipitation estimation features. Weather data from July 1979 to January 2018 from the Global Precipitation Climatology Project was used to train and test the models. Results showed that according to the RMSE and Mean Absolute Percentage Error (MAPE) metrics, the ConvNet and the LSTM-Networks models obtained similar values.

Posteriorly, Kumar et al. (2019) compared two models based on RNNs and LSTM-Networks in the task of forecasting monthly rainfall. The climate dataset used comprised the average rainfall for each month from 1871 to 2016 across India. Lag features of rainfall for 12 past months were used to predict the rainfall value of a future month. The LSTM-Networks model performed better in the evaluation metrics of RMSE, coefficient correlation (R), Nash–Sutcliffe efficiency coefficient (NSE), and MAE.

Lately, Poornima and Pushpalatha (2019) proposed an Intensified LSTM-Networks model to forecast the amount of rainfall on one day into the future. Furthermore, a comparison of the proposed model was made with models based on Holt–Winters, Extreme Learning Machine, Auto-regressive Integrated Moving Average, LSTM-Networks, and RNNs. Weather data from the Hyderabad region, India, from 1980 to 2014 were used to perform the experiments. The climatic features included in the dataset were wind speed, sunshine, evapotranspiration, as well as the maximum and minimum values of temperature and relative humidity. The Intensified LSTM-Networks model achieved the best values for RMSE, Accuracy, Loss, and Learning Rate; followed by the LSTM-Networks model.

Previous research works show that prediction models based on LSTM-Networks outperform other models in the task of forecasting rainfall on an hourly, daily, and monthly basis. Nonetheless, these models are not exempt from the challenges in the field of weather forecasting and Machine Learning. Overfitting and distinctly variance, in

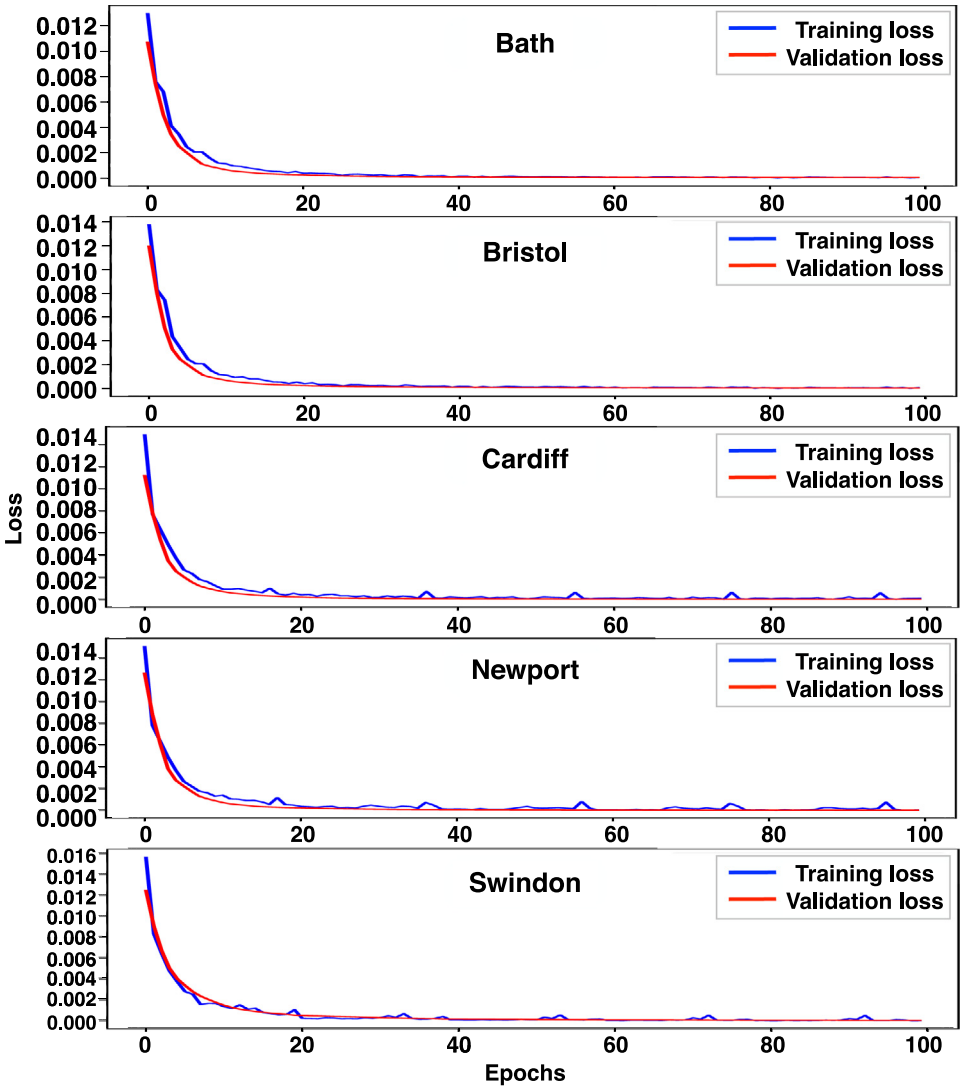



**Fig. 9.** Training and validation loss - Model 3.

time-lapse and precipitation amount, between observed and forecasted values remain as one of the major drawbacks in the complex task of rainfall forecasting.

Moreover, and as previously mentioned, the performance of rainfall forecasting models depends on the parameters and architecture of the models, as well as the characteristics of the data used for their training. Therefore, a comparative analysis of the performance of rainfall forecasting models using meteorological time series data from five major UK cities would benefit the development of budgeting applications aimed at improving quality of life and decreasing the socio-economic impacts caused by rains in the country.

## 3. Theoretical background

This section provides a brief introduction to the concepts of Neural Networks, LSTM-networks, Stacked-LSTM Networks, and Bidirectional-LSTM Networks.

As shown in Section 2, several studies highlight the performance of models based on Neural Networks (NNs) in the rainfall forecast task (Aswin et al., 2018; Balluff et al., 2020; Chao et al., 2018; Kim & Bae, 2017; Kratzert et al., 2018; Kumar et al., 2019; Poornima & Pushpalatha, 2019; Ramos et al., 2019; Yunpeng et al., 2017). The following three major aspects summarise the rationale for the suitability of NNs in the problem of weather forecasting:

- Probabilistic models: Weather forecasting is a non-trivial problem due to humungous uncertainty underlying the question. This uncertainty shall be inherent in the underlying ML approach. The Neural Network (NN) algorithm supports this uncertainty by engineering the last layer of the model to yield a probabilistic output. It is usually achieved by applying the Softmax activation to the output in the last layer, which scales these outputs between zero and one. This scaling makes the uncertainty an inherent feature of the network and allows learning humungous uncertainty in a novel fashion.
- Spatiotemporal dependencies: The weather forecast of a geographical location is highly dependent on the state and context of the area at that point as well as points in the past. This fact highlights the importance of spatiotemporal features and their inevitable role in the right determination of the weather. NNs are a great tool to harness spatiotemporal dependencies from the data and learn latent features that relate the input to target with greater accuracy.
- Discrete distributions: Discretisation of output is also crucial for the weather prediction tasks like precipitation modelling that is inherent to the NNs algorithm. Unlike continuous outputs, the output of the model is designed to be split across buckets which, in this approach, is the hourly window. To avoid recursive invocations, the problem is formulated to generate eight buckets

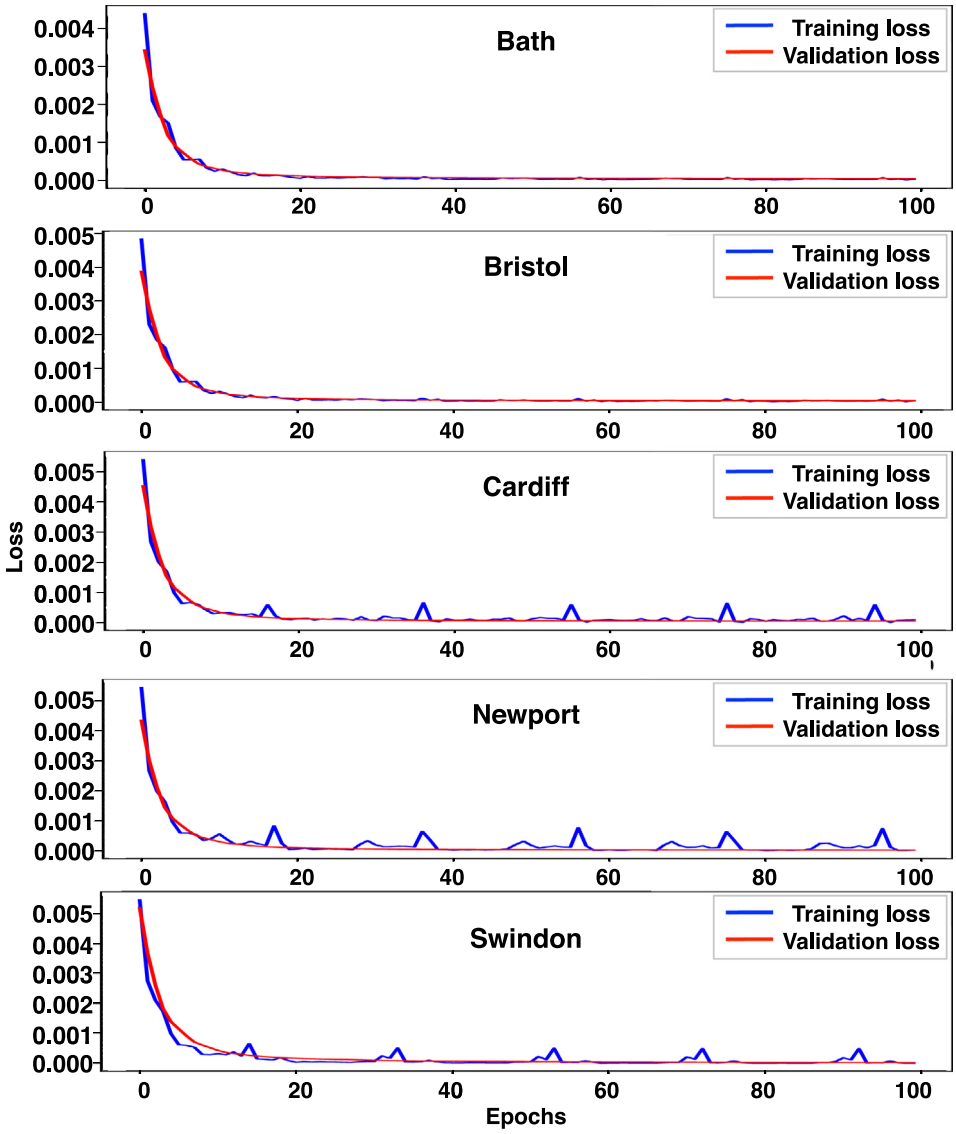



**Fig. 10.** Training and validation loss - Model 4.

that inform precipitation forecast in the next eight hours into the future. This formulation turns the problem into multi-target regression model.

### 3.1. Recurrent neural networks

A Recurrent Neural Network (RNN) is a class of NN that, due to its flexibility to exploit time-series data, has been widely used in research areas such as machine translation, sentiment analysis, speech recognition, and weather forecast, among others. RNN cells capture the dependencies that exist in the elements of data that are been stored in sequence by incorporating a hidden state. This state is what allows maintaining the relationship between the previous and current observed data. Eq. (1) shows how RNNs can preserve dependencies within time-series sequences (Akbari Asanjan et al., 2018; Goodfellow, Bengio, & Courville, 2016; Gulli, Kapoor, & Pal, 2019; Kim & Bae, 2017; Kumar et al., 2019; Poornima & Pushpalatha, 2019; Salman et al., 2018; Zhang et al., 2018; Zou, Fang, Harrison, & Djokic, 2019).

$$h_t = \phi(h_{t-1}, X_t) \tag{1}$$

where $h_t$ and $h_{t-1}$ are hidden states at a current time ($t$) and previous time ($t-1$), respectively. $X_t$ is the current input value at time $t$. This equation is used recursively to allow the network to learn the dependencies of the data.

The RNNs are sensitive to the effects of vanishing gradients when learning long-range dependencies is pursued. LSTM-Networks is an RNN cell variant that reduces this limitation (Akbari Asanjan et al., 2018; Chao et al., 2018; Goodfellow et al., 2016; Gulli et al., 2019; Kumar et al., 2019; Poornima & Pushpalatha, 2019; Singh et al., 2015; Xingjian et al., 2015; Zhang et al., 2018; Zou et al., 2019).

### 3.2. (Stacked) Long-short term memory networks

The Long-short Term Memory (LSTM) Networks are the most widely used RNN cell variant due to their ability to learning long dependencies within sequential data. Successful applications of LSTM-Networks can be found in research fields such as human trajectory prediction, traffic forecasting, speech recognition, and weather prediction. This type of RNN cell is capable of learning the dependencies of at least two previous states and the current state. The vanishing gradient effect is minimised by implementing three gates along with the hidden state. The three implemented gates are commonly referred to as input, output, and forget gates. The input gate defines the amount of information of the new state that is used. The output gate determines the amount of information that is used from previous states. The forget gate conditions

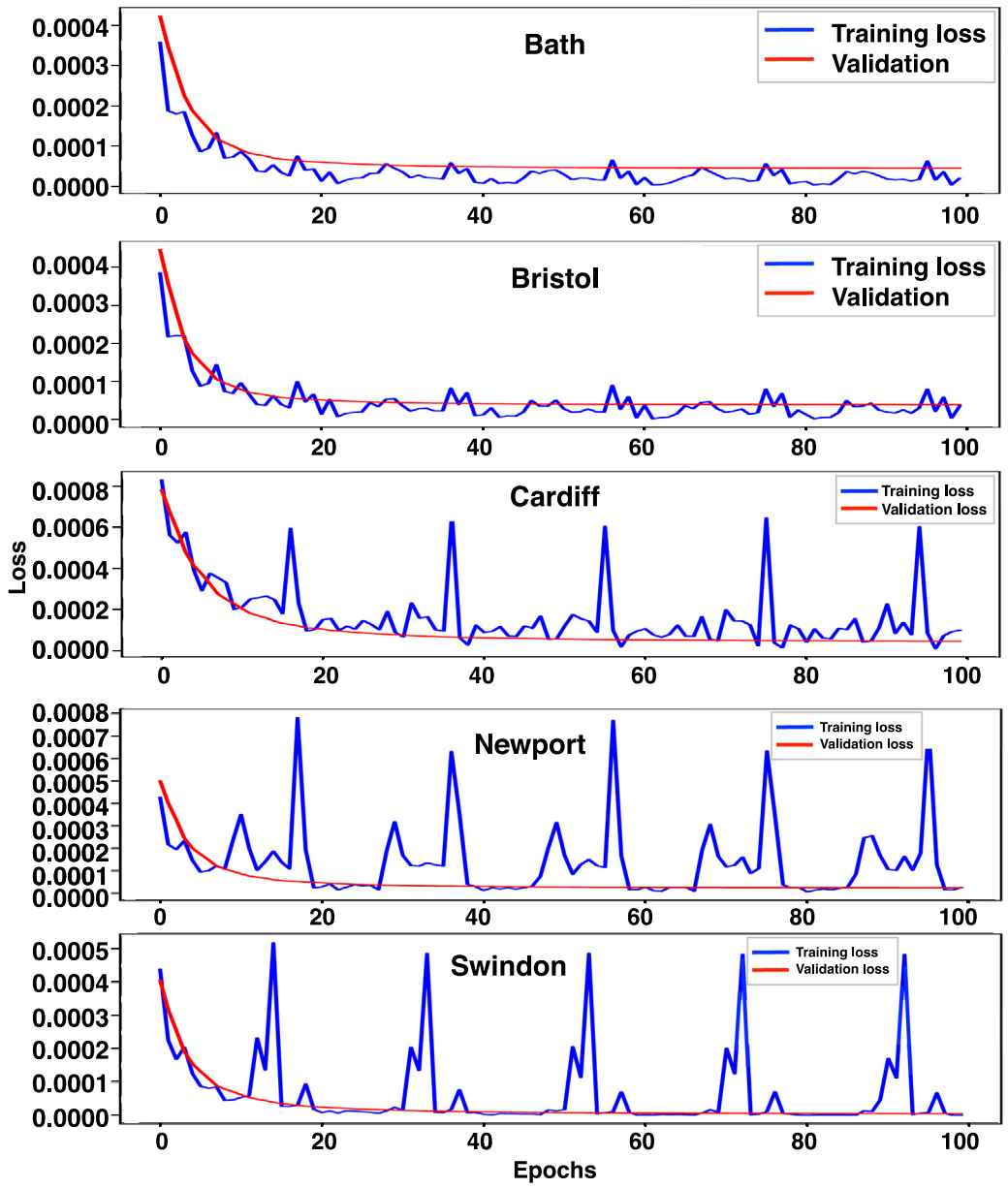



**Fig. 11.** Training and validation loss - Model 5.

the amount of information of the internal state that passes to the next layer. Eq. (2) cover the basic definitions of LSTM-Networks (Akbari Asanjan et al., 2018; Aswin et al., 2018; Chao et al., 2018; Cui, Ke, Pu, & Wang, 2018; Goodfellow et al., 2016; Gulli et al., 2019; Kim & Bae, 2017; Kumar et al., 2019; Poornima & Pushpalatha, 2019; Salman et al., 2018; Singh et al., 2015; Zhang et al., 2018; Zou et al., 2019).

$$
\begin{aligned}
&input\ gate\ (i) = \sigma(W_i h_{t-1} + U_{iX_t} + V_{iC_{t-1}})\\
&output\ gate\ (o) = \sigma(W_o h_{t-1} + U_{oX_t} + V_{oC_{t-1}})\\
&forget\ gate\ (f) = \sigma(W_f h_{t-1} + U_{fX_t} + V_{fC_{t-1}})\\
&internal\ hidden\ state\ (g) = tanh(W_g h_{t-1} + U_{gX_t})\\
&current\ cell\ state\ c_t = (f * c_{t-1}) + (g * i)\\
&current\ hidden\ state\ h_t = tanh(c_t) * o
\end{aligned}
\tag{2}
$$

where $W,U,V$ are different weight matrices, current time $t$ and previous time $t-1$. Stacked-LSTM Networks are composed of two or more LSTM Networks linked successively as hidden layers. This stacked architecture can provide, in some applications, a higher level of representation of time-series data than LSTM-Networks (Cui et al., 2018). Fig. 1 shows the architecture of a simple LSTM-Network.

### 3.3. Bidirectional RNN networks

Bidirectional RNN is a variation of RNNs that are capable of learning dependencies of previous and future states. This type of architecture has shown good results in domains such as natural language processing for speech and handwriting recognition. Bidirectional RNN involves the implementation of RNN cells that capture time-series data from left to right (standard RNN cells) and RNN cells that capture data in a reverse manner (Cui et al., 2018; Goodfellow et al., 2016; Gulli et al., 2019; Singh et al., 2015; Zou et al., 2019). Fig. 2 shows the architecture of a Bidirectional-LSTM Network with two hidden layers. Replacing the RNN cells with LSTM cells of the Bidirectional RNN results in a Bidirectional-LSTM Networks.

## 4. Weather datasets

In this section, the climatic features included in the weather datasets are first described. Subsequently, the description of the Correlation Matrix analysis and the feature selection process carried out as part

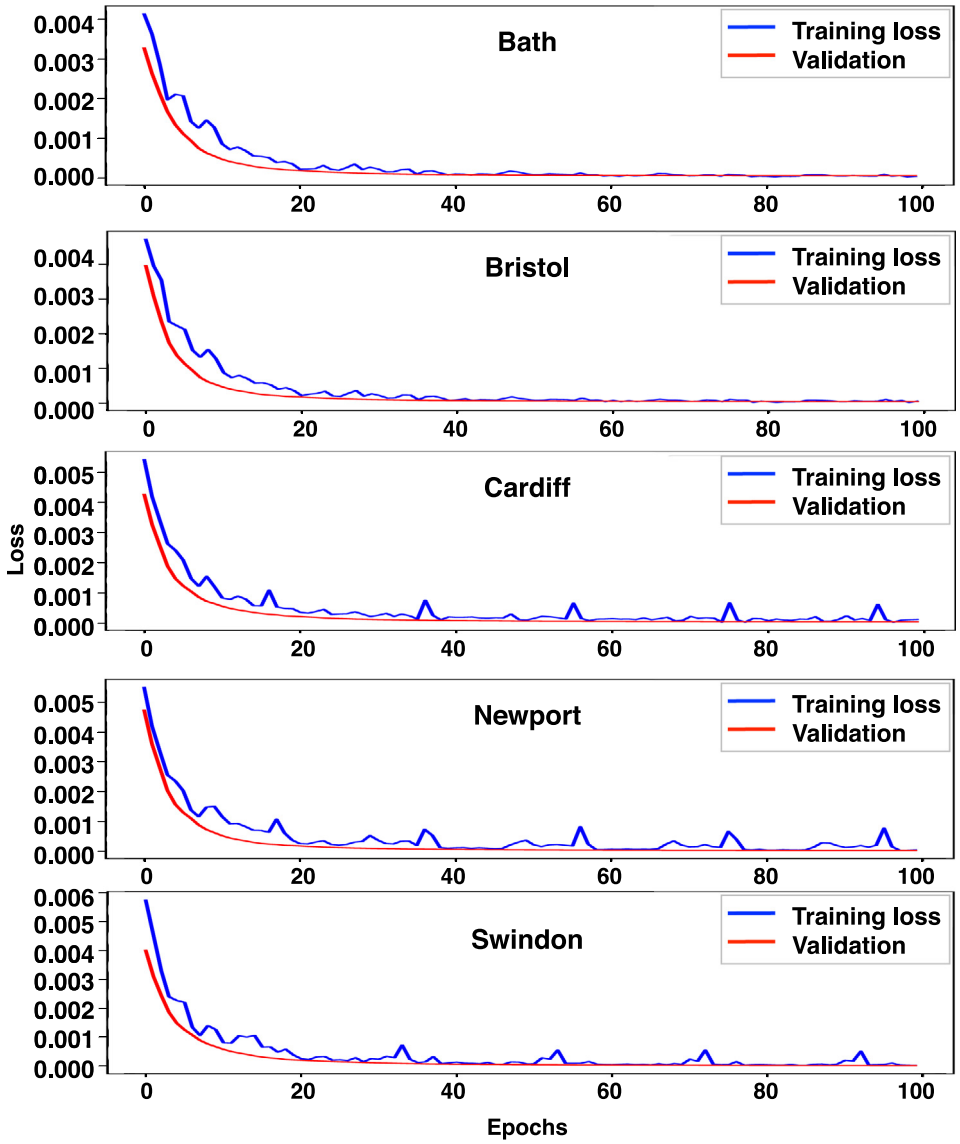



**Fig. 12.** Training and validation loss - Model 6.

of the pre-processing procedure to prepare the time-series data for use in the training of the rainfall forecast models is given.

The rationale for developing a prediction model that forecasts 8-hours of rainfall lies not only in the impact of precipitation on washing and deposition of different air pollutants but in providing prompt estimates to support decision making to diminish eventualities in several human-related activities. Therefore, the acquisition of historical weather data that includes the climatic measurements recorded every hour is imperative.

Consequently, OpenWeather[1] data from five major UK cities were selected to test the proposed models. Specifically, historical data from January 1, 2000, to April 21, 2020, from the cities of Bath, Bristol, Cardiff, Newport, and Swindon were used.

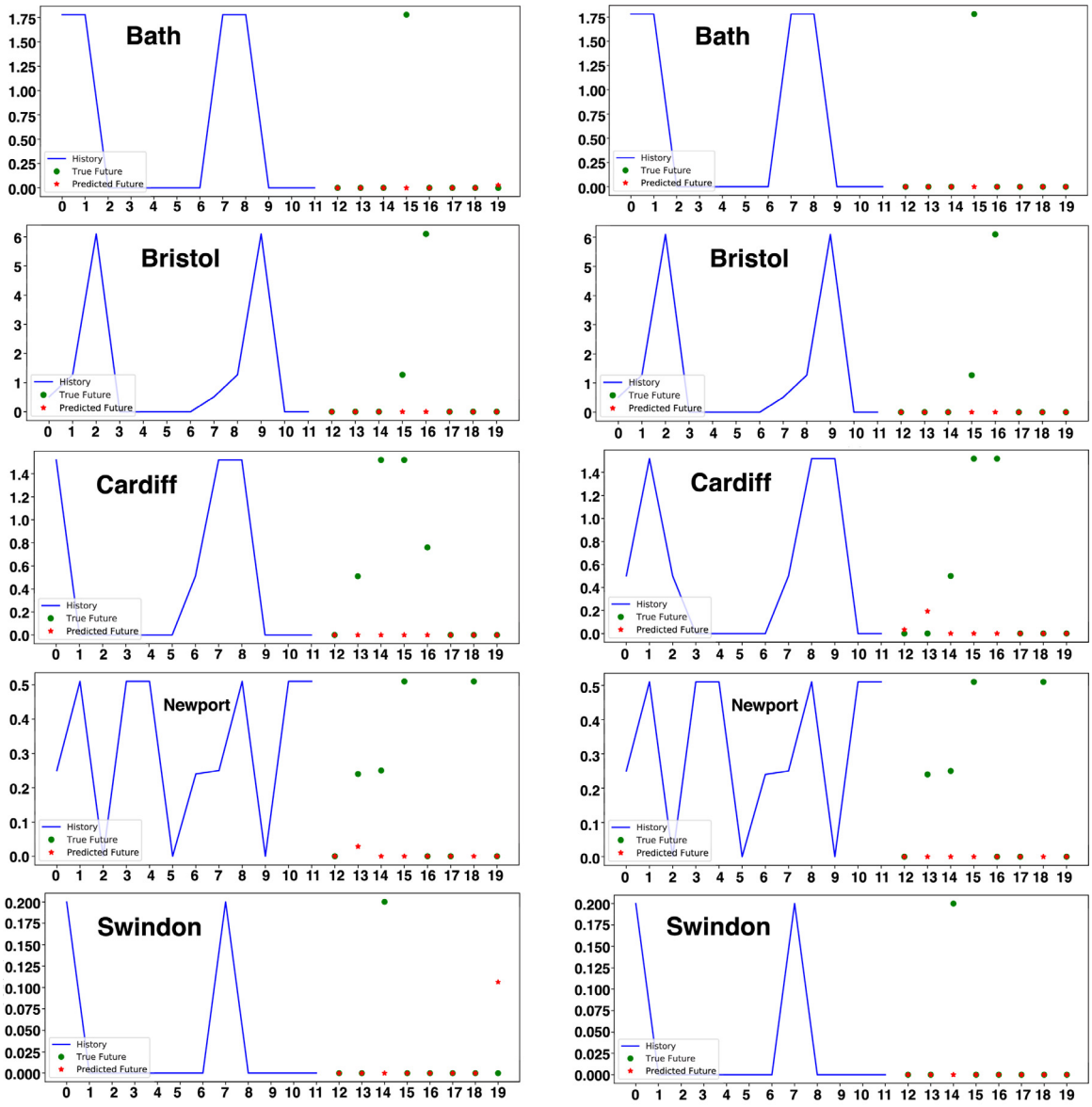
The complete dataset comprises hourly recorded weather measurements of temperature, pressure, humidity, wind speed and direction, percentage of clouds, the volume of rain, and volume of snow. Moreover, the dataset provides records of the city name, latitude and longitude coordinates, timezone code, the date the observation was recorded, and a set of codes that OpenWeather uses on its platform to label various weather conditions such as mist, thunderstorm, smoke, etc.

[1] https://openweathermap.org.

### 4.1. Dataset pre-processing

As with any Machine Learning approach, a processing procedure is required to prepare raw data for use in model training and testing processes. The pre-processing procedure carried out had as its objective the elimination of categorical data, the deletion or replenishment of incomplete data, and the structuring of one hour of observations in one vector row. Specifically, the pre-processing procedure performed is described below:

- Separate weather data by city. This step aims to preserve the weather singularities of each city during the training process.
- Obtain the nullness percentage of each feature. This step help in the decision of removing or filling up missing values of each column vector. The nullness percentage values of each feature in the five major UK cities are concentrated in Appendix A.
- Remove the sea and ground-level measurements. Both features present 100% of nullness; therefore, these features cannot be used in the experimental phase.
- Remove the city name and date-time iso features. The city name is no longer required due to the process of dividing the data for each city. The date and time values of the records are duplicated in the date-time feature; then, the date-time iso column is removed.




(a) Model 4 (b) Model 6

**Fig. 13.** Examples of rainfall predictions and true observed rainfall.

- Replenish the missing values of rain 1 h, rain 3 h, snow 1 h, and snow 3 h features. These features refer to the volume of rain and snow that fell during one and three hours. Consequently, missing values of these features were filled in with zeros. This approach follows the rationale that for any hour that any of these values are missing, there was no precipitation.
- Separate weather condition codes. The weather condition codes comprehend the weather id, weather main, weather description, and weather icon sub-features. These sub-features include one to several records depending on the types of weather recorded in an hour. The objective of the separation process is to help in the task of building vector rows of invariant length.
- Standardise the number of weather condition codes. A recorded hour can contain zero or multiple weather conditions. Two steps were implemented to obtain a dataset integrated by feature vectors of the same length. First, the weather description, weather main, and weather icon features were removed because they are represented by the weather id feature. Secondly, the weather id values that were not present in the five data sets, corresponding to the five UK cities, were removed. This step also allows the building of all five datasets with the same feature-length.
- Compute one-hot encoding for the weather id codes. This step downsizes the range of id code values of 300–900 to values of zero or one. Values of zero denote the absence of a particular weather condition and values of one denote its presence.
- Structure the feature vectors rows. One feature vector concentrates the described weather measurements observed in one hour. The weather datasets of the five major UK cities were built with this structure.

The pre-processing process of the weather dataset results in five datasets with a 43-dimensional feature vector structure. Each dataset is integrated by all previously described weather measurements, the geographical coordinates, and by common weather codes across the five datasets such as Rain ID 500, Drizzle ID 300, and Mist ID 701.

#### *4.1.1. Correlation Matrix and feature selection*

The next step involved in preparing the data that will be used to train the rainfall forecasting models is the selection and retention of

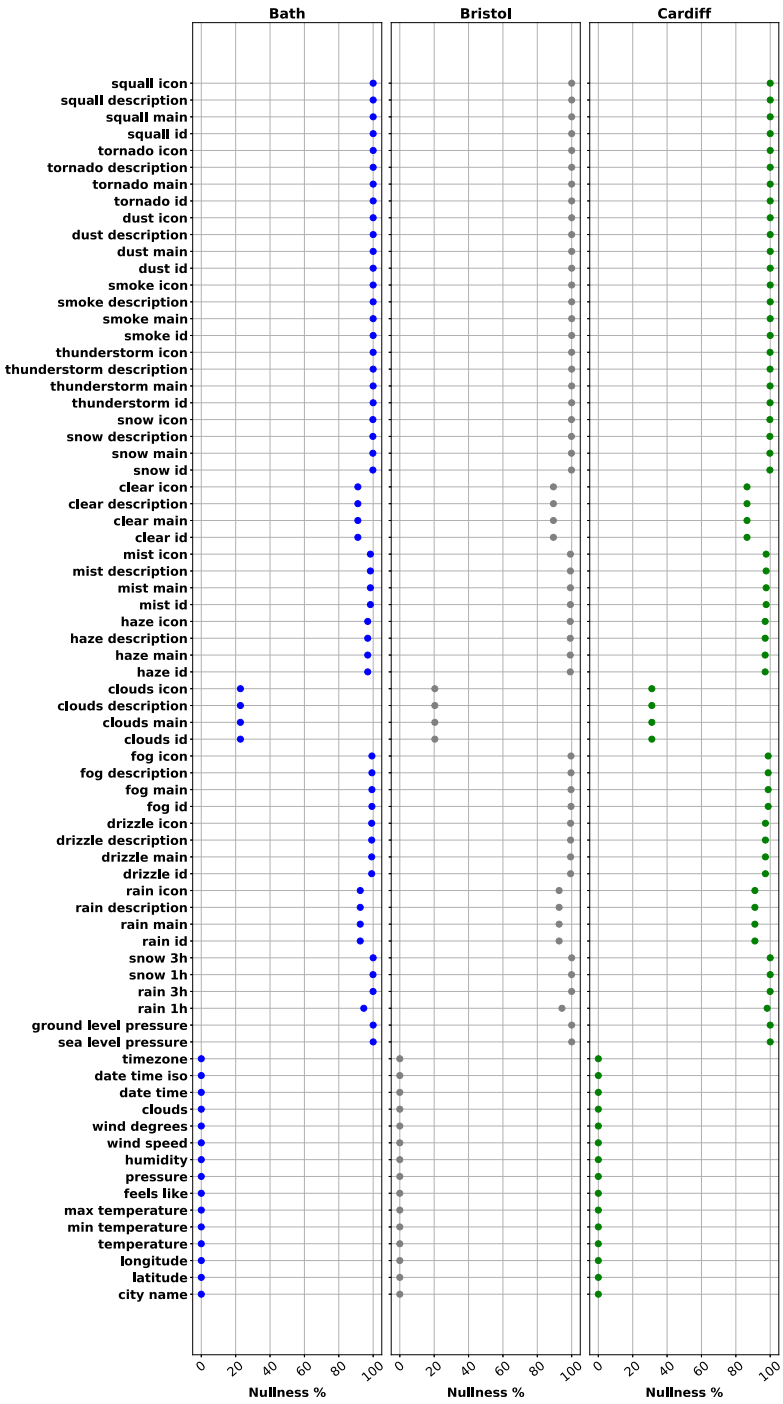



**Fig. 14.** Nullness of features in the cities of Bath, Bristol, and Cardiff.

the characteristics that best represent the variability of the dataset. To achieve this, a Correlation Matrix (CM) was computed to then perform a feature selection process in an unsupervised manner; that is, to remove redundant (highly-correlated) features without using a target feature. In this study, Pearson correlation coefficient was used to calculate the CM for each of the five 43-dimensional datasets. As an example of the CM analysis, Fig. 3 shows the result for the city of Bath.

In order to perform the feature selection process, the tuples of features that obtain a correlation value equal to or greater than $\pm 0.7$ for each city are retained for further analysis. Afterwards, the tuples that are present as highly correlated features in the five cities are considered for elimination. Table 1 shows the list of tuples that are candidates for removal.

From Table 1 the following conclusions can be obtained:

- Latitude or Longitude features cannot be considered independent features since they both represent a geographic coordinate, making a proper interpretation of one without the other impossible. Considering that the datasets are divided by each city, and the previous statement, both features are removed from the datasets.
- All features related to temperature measurements are highly correlated one to another. Thus, the main feature is retained (temperature) and the rest features that concentrates the maximum, minimum, and feels like values are removed.
- The weather code of snow ID 601 is highly correlated with the feature that measures the volume of snow in one hour. The

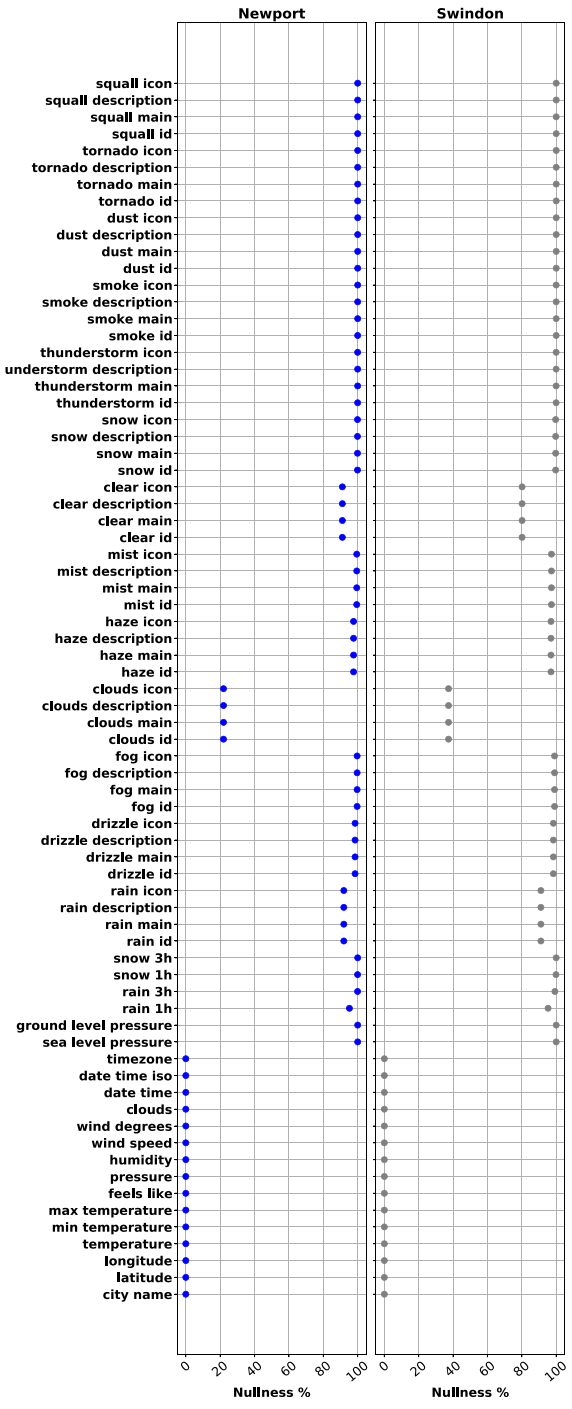



**Fig. 15.** Nullness of features in the cities of Newport and Swindon.

main weather feature is Snow 1 h and the Snow ID 601 feature represents an internal code used by the OpenWeather services. Consequently, the feature Snow ID 601 is removed.

Taking into account the last consideration of removing the feature that represents an internal ID code for the use of OpenWeather services, it was decided to remove the rest of the features that contain ID codes from the five datasets.

This study does not consider time and date values as features that models can learn. The rationale is to avoid restricting the learning process of the models to possible time hidden patterns. Therefore, the date-time feature was removed from all five datasets. After this process, the dimensionality of the datasets was downsized from 43 to 11 features.

Finally, dataset normalisation is a common approach performed in Machine Learning problems when its features have a different range of values between them (Kim & Bae, 2017; Shanker, Hu, & Hung, 1996). Implementing this process could produce better results and minimise the time required to train a model (Shanker et al., 1996). Hence, the MinMaxScaler normalisation function from the Scikit Learn library (Pedregosa et al., 2011) was used to re-scale the features of the five datasets in the range between zero and one.

The final structure of the datasets that resulted from this preprocessing procedure, and used to train and test the rainfall prediction models, are described in Section 5.2.

## 5. Methods

This section provides a detailed description of the experiments performed to develop and test different rainfall prediction models. This description includes the structure of the feature vector rows of the datasets, the procedure for building the training, validation and testing subsets of each of the five datasets, the implementation of the non-exhaustive hyperparameter search for the rainfall forecasting models based on XGBoost and LSTM-Networks, the fixed values for the rest of the parameters of the models, and the implementation and adaptations made to the tested models.

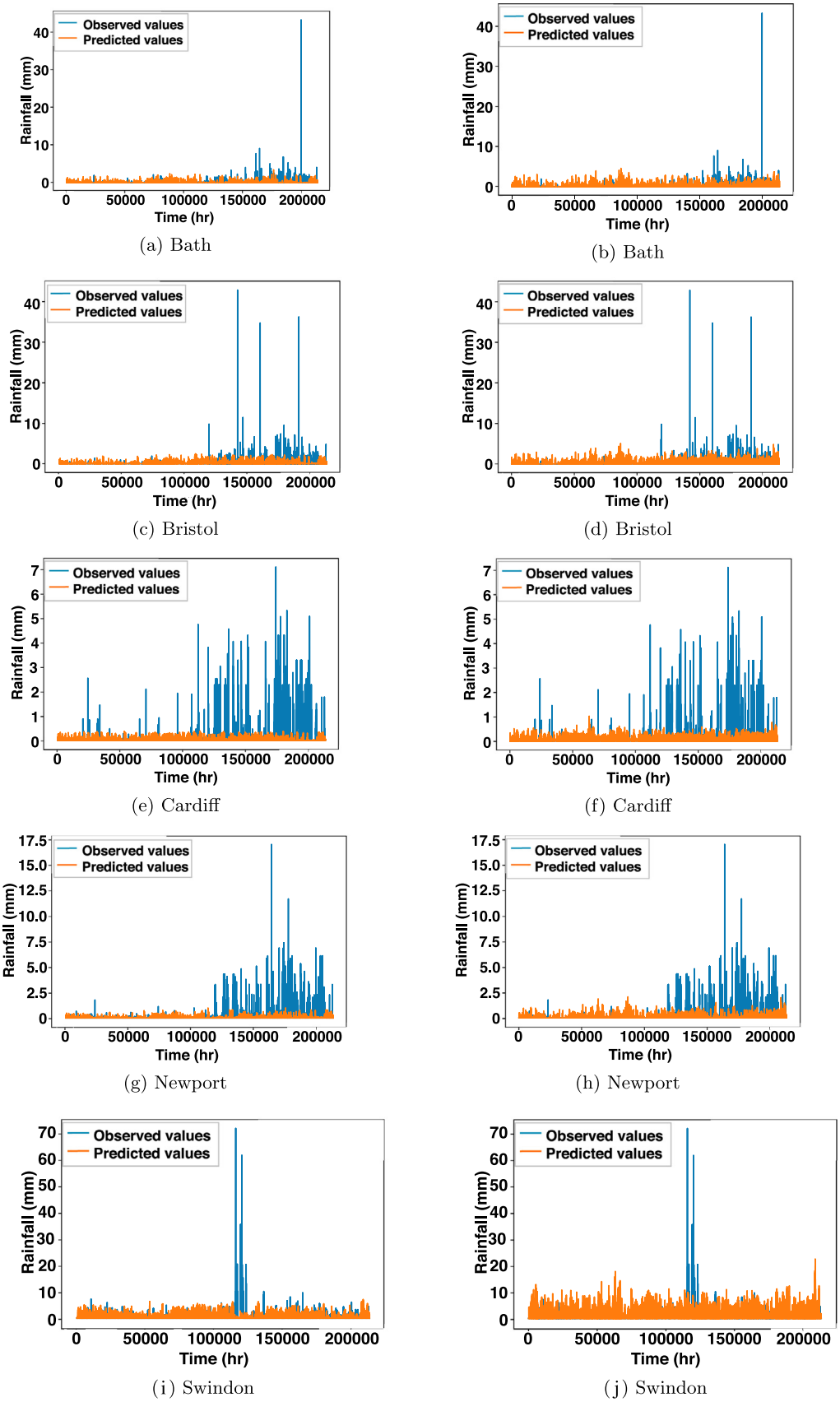
### 5.1. Rainfall prediction approach

This study seeks to investigate the suitability of three LSTM-Networks architectures in the task of predicting 8-hours of rainfall volume using time-series data from five major UK cities. In particular, this study makes a comparison between forecast models based on LSTM-Networks, Stacked-LSTM Networks and Bidirectional-LSTM Networks against an XGBoost decision tree algorithm and an algorithm resulting from the use of AutoML.

Therefore, in order to achieve a broad, non-exhaustive, comparison the following experimental procedure was followed:

1. Build an XGBoost model for each of the five datasets as a baseline. Perform a non-exhaustive hyperparameter grid search to obtain the best hyperparameter values to train each model.
2. Use an AutoML tool to explore the performance of different well-known regression algorithms in the literature. For each of the five datasets, train a model recommended by the AutoML tool.
3. From the literature review, build two models based on LSTM-Networks and one model based on Stacked-LSTM Networks. For each of the five datasets, train an individual version of the three models built. During the training process of each individual model, perform a non-exhaustive hyperparameter grid search to obtain the best hyperparameter values that boost its performance.
4. From the previous models based on LSTM-Networks and Stacked-LSTM Networks, identify the model that achieves the best performance and accuracy across all five datasets. Based on the architecture of this model, build two different model versions of Stacked-LSTM Networks and one Bidirectional-LSTM Networks model.

As mentioned earlier, the performance of algorithms is directly affected by a plethora of design decisions. These design decisions include the choice of the values of the different parameters and hyperparameters involved in each algorithm. Therefore, it is good practice to search for the values of the parameters and hyperparameters (Poornima & Pushpalatha, 2019) that optimise the performance of an algorithm. Even though a non-exhaustive hyperparameter tuning process was performed for each dataset, it is out of the scope of this research to




(a) Bath (b) Bath

(c) Bristol (d) Bristol

(e) Cardiff (f) Cardiff

(g) Newport (h) Newport

(i) Swindon (j) Swindon

**Fig. 16.** Observed rainfall vs. Predicted rainfall (left: Model 4, right: Model 6).

find the optimum values of the parameters and hyperparameters for each of the datasets and rainfall forecasting models.

The final objective of the prediction models is to forecast the volume of rainfall 8-hours into the future with 1-hour observation; that is, to

perform a multiple output forecast with a single input. Suppose $X$ is an input dataset with $o_1, o_2, \ldots, o_n$ observations where $o_i$ is a record of $x_1, x_2, \ldots, x_m$ features registered in a time $t(t = 1, 2, \ldots, j)$. Thus, $X(o_i)(t_j)$ will be used to predict the volume of rainfall $f_k$ in $t_{j+8}$.

Models based on LSTM architectures can perform a multiple-output forecast for each entry point in an straightforward manner. On the other hand, the current implementation of XGBoost, as well as most AutoML tools, do not incorporate this capability in a direct way. Consequently, even though the LSTM-based prediction models will be built with a multiple-output forecast strategy and the rest of the models with a single-output forecast strategy, the comparisons of the predictions of the models will be made under the single-input single-output basis. That is, $X(f_i)(t_j)_{j=0}^{n}$ will be compared with $X(o_i)(x_m)(t_j)_{j=0}^{n}$. The methodology used in the experimentation process is illustrated in Fig. 4.

Experiments were performed in a MacBook Pro with a 2.9 GHz Intel Core i9 processor, and 32 GB DDR4 RAM. The prediction models were implemented using Python (van Rossum, 1995) 3.6.10, Tensorflow (Abadi et al., 2015) 2.0, Keras (Charles, 2013) 2.2.4, Scikit Learn (Pedregosa et al., 2011) library version 0.22.1, TPOT (Le, Fu et al., 2020) tool version 0.11.2, and XGBoost (Chen & Guestrin, 2016) library version 0.90.

### 5.2. Structure of datasets for rainfall forecasting

In order to train the rainfall prediction models, the steps described in Section 4.1 were carried out in each of the five weather datasets. The result of the pre-processing procedure delivers a $X(o_n)(x_m)(m = 1, 2, \ldots, 11)$. Table 2 shows the features comprehended in each of the five datasets.

From Table 2 it can be noticed that feature 11, "Rain 1h" $(X(o_n)(m_{11}))$, is the actual volume of rain observed in a previous hour. That is, for each record (row) in the dataset, the weather conditions associated with the actual amount of rainfall are out of phase. Therefore, to train the rainfall prediction models, a vector containing the shifted value of "Rain 1h" was computed. This results in $X(o_n)(x_m)(m = 1, 2, \ldots, 12)$ where $X(o_n)(x_{12})$ is the actual value of rainfall amount for that $X(o_n)(t_j)$. The shifted feature $(X(o_n)(x_{12}))$ is used only as the target feature to predict by the models; that is, it is not used as a learnable predictor. Table 3 shows the structure of the weather dataset vectors used in the training process of the rainfall prediction models.

Fig. 5 shows the volume of rainfall recorded each hour in the period from 2000 to 2020 for each of the five datasets processed. In the Figure, the horizontal red lines denote the average value of rainfall throughout the period.

### 5.3. Preparation for building the training, validation, and testing sets

Each of the five datasets built in Section 5.2 is used to train and test independent models for the task of predicting rainfall. This approach will allow each model to learn the singularities of each city.

In the rainfall forecasting field, there is no established heuristic to divide the dataset into training, validation, and testing subsets. Notwithstanding, the historical weather datasets for the five major UK cities were built under the nature of time-series data for regression problems. Consequently, the weather data was sorted in ascending order. That is, $X(o_n)(t_j)_{j=0}^{m}$ where $t_0$ = 00:00:00 January-01-2000 and $t_m$ = 23:00:00 December-31-2020. Maintaining this order, the datasets were divided into training $(X(o_n)(t_j)_{j=0}^{trainingValue})$, validation $(X(o_n)(t_j)_{j=0}^{validationValue})$, and test $(X(o_n)(t_j)_{j=0}^{testingValue})$ sets. It is noteworthy to mention, that data shuffle approaches were not carried out during the partitioning of the datasets and during the training process of the rainfall prediction models.

The percentages used to build the training and test sets for the XGBoost and AutoML models were 85% and 15%, respectively. Due to the experimental methodology involves testing the LSTM and Stacked-LSTM approaches from the literature reviewed, the percentages used to construct the corresponding sets for the LSTM-Networks based models are specified in Subsection 2 5.6.

### 5.4. XGBoost model

The XGBoost regression model was selected as the baseline for this comparison. An individual model was built for each of the five datasets. The models were developed using the Python library provided in Chen and Guestrin (2016).

In this research study, a non-exhaustive hyperparameter grid search was implemented for each dataset with the aim of minimising the RMSE values. The hyperparameters and their contemplated search ranges were: an early stop of 20 iterations, a search for the best number of estimators, a subsample ratio of features and subsample ratio of the training instances within 0.5 to 1.0, values between six to 13 for the minimum sum of instance weight in a child and in the maximum depth of a tree, and Learning Rate values of 0.3, 0.1, 0.05, 0.01, and 0.007.

After the non-exhaustive hyperparameter search, the best values obtained for each dataset were used to train the corresponding individual XGBoost regression model. The default values of gamma, alpha, and lambda were used in all the models.

### 5.5. Automated machine learning

The TPOT tool (Le, Fu et al., 2020) was used to perform the AutoML experiment. The default TPOT configuration for regression problems was used in conjunction with the default values of generations and population size of 100.

The TPOT tool tests various regression models and techniques that follow the scikit-learn API. Moreover, it performs a search over the hyperparameters of these models and techniques. Therefore, experimenting with each of the five datasets would be time-consuming and computationally expensive. For this reason, due to time and computational resource constraints, the Bristol dataset was used to carry out the AutoML experiment with this tool. As a result, the TPOT tool will deliver a regression model with the corresponding hyperparameter values that achieved the best performance for the rainfall forecast task.

Subsequently, the resulting model and the hyperparameter values were used to train an independent model for each of the five datasets. No hyperparameter search was carried out for this model.

### 5.6. LSTM and Stacked-LSTM models from the literature

Three models from the literature reviewed in Section 2 were adapted and implemented using TensorFlow (Abadi et al., 2015) and Keras (Charles, 2013).

The first adapted model (Model 1) is a Staked-LSTM Network introduced in Kim and Bae (2017). The second (Model 2) and third (Model 3) adapted models are based on LSTM-Networks presented in Kumar et al. (2019) and Aswin et al. (2018), respectively. All three models present a fully connected layer structure. The general architecture of the implemented Neural Networks is shown in Fig. 6, where $x_i$ is a predictor in the feature vector of the dataset $X$ (Table 3), $h_n^m$ is a neuron $n$ in a hidden layer $m$, and $f_o$ is the output value.

All the LSTM-Networks and Stacked-LSTM Networks models implemented in this study have an input layer with 11 input neurons that correspond to the 11 weather features used as predictors $(X(o_n)(x_m)_{m=1}^{11})$, and an output layer with one neuron corresponding to the forecasted rainfall value $(X(o_n)(x_{12}))$. The specific number of hidden layers, hidden neurons, and the rest of the parameters and hyperparameters values used for Model 1, Model 2, and Model 3 are the same as described in Kim and Bae (2017), Kumar et al. (2019) and Aswin et al. (2018), respectively. The values of the hyperparameters not described in their original articles were proposed following common heuristics in the field of Deep Learning.

A non-exhaustive hyperparameter grid search was performed independently for each of the five datasets and for each of the three adapted models. The non-exhaustive hyperparameter grid search focused on finding the best values for the optimiser and the Learning Rate. The

tested optimisers were Adam and Stochastic Gradient Descent (SGD). Tested Learning Rate values were 0.3, 0.1, 0.05, 0.01, 0.007, and 0.001. Moreover, the hyperparameter grid search also included test batch size values of 15 and 32, as well as tanh, ReLU, and Swish (Ramachandran, Zoph, & Le, 2017) as activation functions of the LSTM neurons.

In all models, the ReLU activation function was used in the output layer and the Mean Square Error (MSE) was used as Loss function. No dropout or early stop approaches were implemented for any of the models.

Tables 4, 5, and 6 show the fixed values of the parameters and hyperparameters, as well as the percentage of training, validation and testing of the datasets used in Model 1, Model 2, and Model 3, respectively. The number of past observations used to predict 8-hours of rainfall was established according to the adapted approach and is denoted as the time steps parameter.

### 5.7. Proposed models based on Stacked-LSTM and Bidirectional-LSTM networks

After completing the experiments described in Section 5.6, a performance comparison of Model 1, Model 2, and Model 3 will be performed. From the three models, the one with the best performance across all five datasets will be selected as a base to develop two Stacked-LSTM Network models and one Bidirectional-LSTM Network model. The corresponding fixed values of the parameters and hyperparameters of this model will be preserved, with the exception of the number of hidden layers.

Of the two proposed models based on Stacked-LSTM Networks, one will include two hidden layers (Model 4) and the other will comprise three hidden layers (Model 5). The proposed Bidirectional-LSTM Network model (Model 6) will contain only one hidden layer.

Following the same experimental procedure as before, for each of the five data sets, a Model 4, a Model 5 and a Model 6 will be trained with the non-exhaustive hyperparameter grid search previously described.

## 6. Results and discussion

This section describes the evaluation metrics used to measure the performance of the trained rainfall prediction models, the structure of the datasets used in the experiments carried out, and the results of the prediction models tested with the best values obtained by the hyperparameter search.

### 6.1. Evaluation metrics

Following the approach of the related works to evaluate the performance of the prediction models (Akbari Asanjan et al., 2018; Aswin et al., 2018; Bell, Carrington, & Moore, 1994; Chao et al., 2018; Kim & Bae, 2017; Kim et al., 2017; Kumar et al., 2019; Poornima & Pushpalatha, 2019; Salman et al., 2018; Zhang et al., 2018; Zou et al., 2019), the metrics used in this study are:

- Loss: The error associated to the proportion of examples for which the model produces an incorrect output (Goodfellow et al., 2016) as depicted in Eq. (3).

$$loss = \begin{cases} 1 & \text{if error} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

- Root Mean Squared Error (RMSE): The sum of the square root of the mean of the squared differences between corresponding model outputs and observations (Barnston, 1992; Willmott & Matsuura, 2005).

$$RMSE_{fo} = \sqrt{\frac{\sum_{i=1}^{n}(z_{fi} - z_{oi})^2}{n}} \tag{4}$$

Formula (4) describes the RMSE evaluation metric, where $f$ is the model outputs (forecasts), $o$ is the observations, $\sum$ is summation, $n$ is the sample size, and $i$ is the $i$th element.

- Mean Absolute Error (MAE): The sum of the magnitudes of difference between the corresponding model outputs and observations divided by the total number of examples (Willmott & Matsuura, 2005). This metric can be expressed as in Eq. (5).

$$MAE_{fo} = \frac{\sum_{i=1}^{n}(|z_{fi} - z_{oi}|)}{n} \tag{5}$$

- Root Mean Squared Logarithmic Error (RMSLE): A variant of the RMSE which computes the logarithmic difference between corresponding model outputs and observations. This measure deflates the influence of a large error when the observations rate is higher than the model outputs (Bell et al., 1994). A mathematical representation of this evaluation metric is given in Eq. (6); where $log$ is the logarithm.

$$RMSLE_{fo} = \sqrt{\frac{\sum_{i=1}^{n}(log(z_{fi}+1) - log(z_{oi}+1))^2}{n}} \tag{6}$$

### 6.2. Prediction models results

#### 6.2.1. XGBoost results

The XGBoost regression model was considered to serve as the baseline for subsequent rainfall prediction models. Prior to the training process, a non-exhaustive hyperparameter search was performed for each of the five datasets. The results of the best hyperparameter values found by this search are concentrated in the first part of Table 7. Subsequently, an individual XGBoost model was trained with each dataset and its corresponding hyperparameter values. The performance results of these models are presented in the lower part of Table 7.

Results show that the Swindon dataset obtained the biggest evaluation metrics values, denoting that it was more difficult for the model to learn the singularities of the rainfall behaviour for this city compared to the rest. On the contrary, the city with the best performance values, in general, was Cardiff.

#### 6.2.2. AutoML results

Afterwards, the TPOT library was used to perform the AutoML approach. A stacked ensemble of Gradient Boosting Regressor, Linear Support Vector Regression, and an Extra-trees Regressor was the resulting model.

The resulting parameters values for the Gradient Boosting Regressor were: alpha of 0.75, a Learning Rate of 1.0, quantile as loss function, a maximum depth individual regression estimators of 1, a value of 0.5 for the number of features to consider for the best split, a minimum of 15 samples required in a leaf, a minimum of 12 samples required to split an internal node, 100 boosting stages, and 0.45 as samples fraction to fit individual base learners. Regarding the Linear Support Vector Regression, the resulting parameters were: a regularisation of 0.1, set to false the option to solve the dual optimisation problem, an epsilon value of 0.001, squared epsilon insensitive as loss function, and a tolerance stopping criteria of 0.001. Lastly, the resulting parameters values for the Extra-trees Regressor were: 100 estimators (trees), set to true the bootstrap of samples, a number of one feature to consider when looking for the best split, a minimum number of 16 samples required to be at a leaf, and a minimum number of 17 samples required to split an internal node.

Subsequently, an independent stacked ensemble model, with the values of the parameters described above, was trained with each of the five data sets. Table 8 shows the results obtained by the ensemble model in the evaluation metrics.

The results in Table 8 show that the behaviour of rainfall in Swindon was more difficult to learn than in the rest of the cities. Moreover, the Cardiff dataset achieved the best performance in the evaluation metrics.

By comparing the rainfall prediction results made by the XGBoost model (Table 7) and the stacked ensemble model (Table 8), the following observations are found. For both models, the behaviour of rainfall for the city of Swindon and the city of Cardiff was more difficult and less difficult to learn, respectively. In the city of Bath, both models obtained similar values in the RMSE and RMSLE metrics. Regarding the MAE metric, the lowest value was obtained by the ensemble model. For the Bristol and Swindon cities, the ensemble prediction model achieved lower values in all the metrics except for the RMSE in Bristol (which is the same value in both models). For the city of Cardiff, it can be noticed that all the metrics values are the same for both models. Regarding the performance with the city of Newport, the values of both prediction models in the MAE and RMSE metrics are similar; while the ensemble model achieved a lower RMSLE value.

In general, the best performance, according to the evaluation metrics, was obtained by the ensemble model of Gradient Boosting Regressor, Linear Support Vector Regression, and Extra-trees Regressor; where a considerable difference in the performance in the city of Swindon for the RMSE metric can be seen.

Despite the fact that the XGBoost model and the stacked ensemble use decision trees as final or only steps to deliver a prediction, the results show that a good approach to follow is not to rely on just a single regression algorithm, but to take advantage of the benefits offered by ensemble approaches. Consequently, this result opens the possibility of exploring joint approaches as a follow-up study.

#### *6.2.3. Results of the LSTM and Stacked-LSTM models adapted from the literature*

The results of the hyperparameter grid search performed for Model 1, Model 2, and Model 3 with each of the five datasets showed that a batch size of 32, using the tanh activation function for the LSTM cells and SGD optimiser achieved the best performance. Regarding the best values for the Learning Rate, the non-exhaustive grid search showed that: (1) a value of 0.001 was the best for all the cities when they were trained in Model 2 and Model 3, as well as for the cities of Bristol and Swindon with Model 1; and (2) a value of 0.01 is better suited to training Model 1 with the Bath, Cardiff and Newport datasets.

Table 9 shows the results obtained by Model 1, Model 2 and Model 3 after being trained with the identified hyperparameter values.

Because the stacked ensemble model performed better than the XGBoost model, a comparison of the models based on LSTM-Networks with the XGBoost model is not performed. Therefore, the first comparison carried out is between the rainfall prediction models based on LSTM-Networks and the ensemble model. To this end, the results concentrated in Tables 7 and 9 are contrasted.

From both tables, it can be noticed that all values of the RMSE, MAE, and RMSLE metrics for Model 1, Model 2, and Model 3 are lower than those obtained by the ensemble model for the five cities. These results indicate that the prediction models based on LSTM and Stacked-LSTM Networks can achieve better performance values for the evaluation metrics RMSE, MAE and RMSLE than the model based on an ensemble of Gradient Boosting Regressor, Linear Support Vector Regression, and Extra-trees Regressor for the task of forecasting the volume of rainfall. Consequently, a comparison between the three models based on LSTM-Networks is required to determine which model performs better in the rain prediction task.

The results in Table 9 show that for the RMSE metric, Model 1 achieves the lowest values except for the city of Swindon where the lowest value is obtained by Model 2. Regarding the MAE evaluation metric, Model 1 and Model 2 have the same number of cities with the lowest values. Model 1 achieves the lowest MAE values for the cities of Bristol and Newport, while Model 2 has lowest MAE values for the cities of Bath and Swindon. Both models show the same MAE value for the city of Cardiff. As for the RMSLE and the Loss metrics, Model 1 achieved the lowest values for the five cities among the three models.

However, analysing the evaluation metrics themselves is only one part of the Deep Neural Networks evaluation. The other equally relevant part of the evaluation is the visual analysis of the training and validation loss curves. This visual evaluation allows the inspection of the performance of the Neural Network during the training process, revealing aspects such as overfitting, underfitting and diagnosis of non-representative datasets. The aim of the evaluation is to find an optimal learning curve, that is, the training and validation loss curves decrease to a point of stabilisation while the generalisation gap is minimal (Goodfellow et al., 2016; Gulli et al., 2019).

The evaluation of the models resulting from the hyperparameter grid search also considered this second part of the analysis to identify those models with the best training and validation loss curves for each of the five datasets. That is, the models with the best performance in the evaluation metrics and with the best training and validation loss curves of all the models tested in the hyperparameter grid search were selected and identified as Model 1, Model 2 and Model 3 as appropriate. Figs. 7–9 show the training and validation loss plots of Model 1, Model 2 and Model 3, respectively.

From Figs. 7, 8, and 9, it can be noticed that Model 3 has better training and validation loss curves than Model 1 and Model 2. Although Model 1 achieved the lowest values in the evaluation metrics for most of the five cities, it can be seen that the Stacked-LSTM Network model did not learn properly. This drawback in Model 1 may be due to the fact that the model is a stack with 10 hidden LSTM layers, while Model 2 and Model 3 only have one hidden layer; suggesting that the complexity of the five datasets can be better learned with shallower Neural Networks. Furthermore, Model 3 shows that configuring a greater number of memory cells and allowing the backwards processing of the input sequence while training with a greater number of epochs helps the network learn better.

#### *6.2.4. Results of the proposed models based on Stacked-LSTM and Bidirectional-LSTM networks*

Due to Model 3 presents the best training and validation loss curves, its parameters and hyperparameters were used as the basis to build Model 4, Model 5 and Model 6 following the experimental procedure described in Section 5.7.

The results of the hyperparameter grid search performed for Model 4, Model 5 and Model 6 with each of the five datasets corroborate that a batch size of 32, using tanh as the activation function for the LSTM cells and SGD as an optimiser, makes all three models achieve better performance. On the other hand, the results show that for all three models, the best Learning Rate value was 0.001.

Table 10 shows the results of the evaluation metrics obtained by Model 4 (Stacked-LSTM with two hidden layers), Model 5 (Stacked-LSTM with three hidden layers), and Model 6 (Bidirectional-LSTM Networks) after being trained with the identified hyperparameter values.

The results in Table 10 show the following: (1) The three models achieved equal Loss values for all cities with the exception of the city of Bristol with Model 5, where the Loss value is lower; and (2) In the evaluation metrics RMSE, MAE, and RMSLE, Model 5 obtained lower values in the five cities, while Model 6 presents the highest values. These results indicate that the performance of Model 5 is better than Model 4 and Model 6 according to the evaluation metrics. Nevertheless, training and validation loss curves should also be analysed.

Similar to the approach followed to present the results of the previous models, Model 4, Model 5 and Model 6 are the models with the best performance in the evaluation metrics and with the best training and validation loss curves of all models tested in the hyperparameter grid search. Training and validation loss plots for the five cities for Model 4, Model 5, and Model 6 are shown in Figs. 10, 11, and 12, respectively.

Figs. 10–12 show that Model 4 presents better training and validation loss curve behaviour for the five cities tested. Between Model 4

and Model 5, it can be noticed that the addition of an LSTM hidden layer (Model 5) impacts the learning ability of the rainfall prediction model in the five cities. This impact on the learning behaviour of Model 5 causes that (1) training and validation loss is reduced by a factor of one, and (2) training loss curves do not stabilise through the epochs. Regarding the comparison between Model 4 and Model 6, it can be noticed that both models show similar training and validation loss curves behaviour and values.

Last, but not least, it is performed a comparison between Model 3, Model 4, Model 5, and Model 6. From Tables 9 and 10 it can be seen that in general Model 4 and Model 6 present better performance than Model 3. Between Model 3, Model 4 and Model 6, the Loss values are similar with the exception of the city of Cardiff in Model 3, where the value is higher. Regarding the RMSE evaluation metric, the values of Model 4 and Model 6 are lower for all cities, with the exception of Swindon in Model 6 where the values are equal to Model 3 for this city. For the MAE metric, Model 4 and Model 6 reached lower values than Model 3 except for the city of Swindon in Model 6, where Model 3 has a lower value. Model 4 and Model 6 obtained lower RMSLE values than Model 3 for the five major UK cities. Consequently, the comparison of Model 3, Model 4 and Model 6 denotes that, in general, Model 4 and Model 6 perform better than Model 3 for the rainfall prediction task.

Fig. 13 shows examples of an 8-hour rainfall forecast for the five major UK cities using Model 4 and Model 6. The complete comparison between the actual rainfall observations and the forecasts made by Model 4 and Model 6 for the five cities can be consulted in Appendix B.

Despite the fact that the performance of Model 4 and Model 6, in the evaluation metrics, is better than the rest of the models based on LSTM-Networks, XGBoost and the ensemble; the predictions presented in Fig. 13 show how the models struggle to adapt to abrupt variations in the precipitation pattern. An explanation for this shortcoming may lie in the selection of the values of the hyperparameters, where the non-exhaustive grid search shows that a small change, such as passing from a batch size of 15 to 32 or using different batches of observations to make the predictions, have a great impact on the performance of the LSTM-Networks.

## 7. Conclusions

This study set out to compare the prediction performance of rainfall forecasting models based on LSTM-Networks architectures with modern Machine Learning algorithms. To achieve this objective, 2 models based on LSTM-Networks, 3 models based on Stacked-LSTM, and 1 Bidirectional-LSTM Networks model were compared with an XGBoost (baseline model) and an ensemble model that resulted from carrying out an Automated Machine Learning approach.

In order to evaluate the performance of the implemented rainfall forecasting models, historical weather data of two decades from the UK cities of Bath, Bristol, Cardiff, Newport, and Swindon were used. Good practices were implemented when conducting Machine Learning experiments such as (a) a pre-processing procedure to eliminate categorical and incomplete data within the datasets, (b) a feature selection process carried out through a Correlation Matrix analysis, (c) implementation of a non-exhaustive hyperparameters grid search to obtain the best performance of models based on XGBoost and LSTM-Networks architectures.

To the authors' knowledge, this study is the first to present a comparative analysis of the performance of rainfall forecasting models based on modern Machine Learning algorithms in predicting hourly rainfall volume using weather time-series data from cities in the United Kingdom. Specifically, the main contributions of this study are highlighted below:

- From the literature reviewed, three models based on the LSTM and Stacked-LSTM Networks were adapted for the task of forecasting hourly rainfall using time-series data from five major UK cities.
- A model based on Bidirectional-LSTM Networks was proposed for the task of forecasting rainfall on an hourly basis using time-series data from five major UK cities.
- A comparison of the performance of models based on LSTM-Networks, Stacked-LSTM Networks, Bidirectional-LSTM Networks, XGBoost, and an ensemble model resulted from performing an AutoML approach was performed in the task of forecasting the amount of rainfall per hour using time-series data from five major UK cities.

Results showed that a Bidirectional-LSTM Network can be used as a rainfall forecast model with comparable performance to a Stacked-LSTM Network with two hidden layers. Among all the rainfall prediction models tested in the comparison, which included three models adapted from the literature, the Stacked-LSTM (Model 4) and Bidirectional-LSTM (Model 6) models achieved, in general, lower values in the evaluation metrics RMSE, MAE, and RMSLE. Model 4 achieved Loss values between 0.0014–0.0001, RMSE values in the range of 0.0375–0.0084, MAE values between 0.0071–0.0013, and RMSLE values ranging between 0.0157–0.0037. On the other hand, Model 6 obtained values ranging between 0.0014–0.0001, 0.0377–0.0099, 0.0072–0.0015, and 0.0111–0.0044 for Loss, RMSE, MAE, and RMSLE, respectively.

The Stacked-LSTM Network with 10 hidden layers (Model 1) presented the worst performance in the training and validation loss curves, among all the rainfall prediction models. Moreover, Model 5 (a Stacked-LSTM with three hidden layers) showed worst training and validation loss curves than Model 4 (a Stacked-LSTM with two hidden layers) and Model 5 (a Bidirectional-LSTM Network). This suggest that LSTM Neural Networks with a large number of hidden layers are less suitable to learn the singularities of weather time-series to forecast hourly rainfall volume values.

Although these results are encouraging and even suggest that it is possible to use Bidirectional-LSTM Networks as a rainfall prediction model, the tested models share one major drawback. This major drawback, similar to other approaches that use models based on LSTM-Networks, is the inability to generalise adequately. For the most part, models overfit training data and cannot record accurate predictions in test and validation sets.

Future work would be directed at fine-tuning the parameters and hyperparameters of the prediction models with the aim of further closing the gap between the predicted values and the observed rainfall volumes. In addition, different approaches can also be considered to deal with missing values of features in time-series data, such as moving average (Karasu & Altan, 2019) and to calculate lag features. On the other hand, recent related approaches in forecasting using fuzzy time-series (E.g. Singh, 2021) or hybrid models of LSTM-Networks, decomposition methods and grey wolf optimiser (Altan, Karasu, & Zio, 2021) will be investigated. In addition, it is worth considering a comprehensive analysis of the importance of the features and the inclusion of other weather factors to achieve better performance of the rainfall forecasting models.

## CRediT authorship contribution statement

**Ari Yair Barrera-Animas:** Conceptualisation, Methodology, Software, Validation, Writing – original draft, Writing – review & editing, Resources, Data curation, Visualisation. **Lukumon O. Oyedele:** Writing – review & editing, Supervision, Funding acquisition. **Muhammad Bilal:** Conceptualisation, Methodology, Validation, Writing – review & editing, Resources, Data curation, Visualisation, Supervision, Project administration. **Taofeek Dolapo Akinosho:** Methodology, Writing – review & editing, Resources, Visualisation. **Juan Manuel Davila Delgado:** Methodology, Writing – review & editing, Visualisation. **Lukman Adewale Akanbi:** Writing – review & editing, Supervision.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgements

The authors would like to express their sincere gratitude to the Engineering and Physical Science Research Council (EPSRC) grant ref: EP/S031480/1 and the Innovate UK (Grant Application No 10137 and File No 104367) for providing the financial support for this study.

## Appendix A. Nullness of features contained in the raw weather datasets of the major cities of UK

See Figs. 14 and 15

## Appendix B. Comparison of rainfall predictions and true observed rainfall

See Fig. 16

## References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. URL http://tensorflow.org/, Software available from tensorflow.org.

Aguasca-Colomo, R., Castellanos-Nieves, D., & Méndez, M. (2019). Comparative analysis of rainfall prediction models using machine learning in islands with complex orography: Tenerife island. *Applied Sciences*, *9*(22), 4931.

Akbari Asanjan, A., Yang, T., Hsu, K., Sorooshian, S., Lin, J., & Peng, Q. (2018). Short-term precipitation forecast based on the PERSIANN System and LSTM recurrent neural networks. *Journal of Geophysical Research: Atmospheres*, *123*(22), 12–543.

Altan, A., Karasu, S., & Zio, E. (2021). A new hybrid model for wind speed forecasting combining long short-term memory neural network, decomposition methods and grey wolf optimizer. *Applied Soft Computing*, *100*, Article 106996.

Aswin, S., Geetha, P., & Vinayakumar, R. (2018). Deep learning models for the prediction of rainfall. In *2018 International Conference on Communication and Signal Processing (ICCSP)* (pp. 0657–0661). IEEE.

Balluff, S., Bendfeld, J., & Krauter, S. (2020). Meteorological data forecast using RNN. In *Deep Learning and Neural Networks: Concepts, Methodologies, Tools, and Applications* (pp. 905–920). IGI Global.

Barnston, A. G. (1992). Correspondence among the correlation, RMSE, and heidke forecast verification measures; refinement of the heidke score. *Weather and Forecasting*, *7*(4), 699–709.

Bell, V., Carrington, D., & Moore, R. (1994). Rainfall forecasting using a simple advected cloud model with weather radar, satellite infra-red and surface weather observations: an initial appraisal under UK conditions.

Chao, Z., Pu, F., Yin, Y., Han, B., & Chen, X. (2018). Research on real-time local rainfall prediction based on MEMS sensors. *Journal of Sensors*, *2018*.

Charles, P. (2013). Project title. https://github.com/charlespwd/project-title.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining* (pp. 785–794).

Cui, Z., Ke, R., Pu, Z., & Wang, Y. (2018). Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction. arXiv preprint arXiv:1801.02143.

Czarnecka, M., Nidzgorska-Lencewicz, J., et al. (2011). Impact of weather conditions on winter and summer air quality. *International Agrophysics*, *25*(1), 7–12.

Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, *14*(2), 179–211.

Gan, K., Sun, S., Wang, S., & Wei, Y. (2018). A secondary-decomposition-ensemble learning paradigm for forecasting PM2. 5 concentration. *Atmospheric Pollution Research*, *9*(6), 989–999.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2016). LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(10), 2222–2232.

Gulli, A., Kapoor, A., & Pal, S. (2019). *Deep Learning with TensorFlow 2.0 and Keras: Regression, ConvNets, GANs, RNNs, NLP & more with TF 2.0 and the Keras API*. Packt Publishing Ltd.

Hossain, I., Rasel, H., Imteaz, M. A., & Mekanik, F. (2020). Long-term seasonal rainfall forecasting using linear and non-linear modelling approaches: A case study for western Australia. *Meteorology and Atmospheric Physics*, *132*(1), 131–141.

Hutter, F., Kotthoff, L., & Vanschoren, J. (2019). *Automated Machine Learning: Methods, Systems, Challenges*. Springer Nature.

Karasu, S., & Altan, A. (2019). Recognition model for solar radiation time series based on random forest with feature selection approach. In *2019 11th International Conference on Electrical and Electronics Engineering (ELECO)* (pp. 8–11). IEEE.

Kim, H.-U., & Bae, T.-S. (2017). Preliminary study of deep learning-based precipitation prediction. *Journal of the Korean Society of Surveying, Geodesy, Photogrammetry and Cartography*, *35*(5), 423–429.

Kim, S., Hong, S., Joh, M., & Song, S.-k. (2017). Deeprain: Convlstm network for precipitation prediction using multichannel radar data. arXiv preprint arXiv:1711.02316.

Kratzert, F., Klotz, D., Brenner, C., Schulz, K., & Herrnegger, M. (2018). Rainfall–runoff modelling using long short-term memory (LSTM) networks. *Hydrology and Earth System Sciences*, *22*(11), 6005–6022.

Kumar, D., Singh, A., Samui, P., & Jha, R. K. (2019). Forecasting monthly precipitation using sequential modelling. *Hydrological Sciences Journal*, *64*(6), 690–700.

Le, T. T., Fu, W., & Moore, J. H. (2020). Scaling tree-based automated machine learning to biomedical big data with a feature set selector. *Bioinformatics*, *36*(1), 250–256.

Le, T.-T., Pham, B. T., Ly, H.-B., Shirzadi, A., & Le, L. M. (2020). Development of 48-hour precipitation forecasting model using nonlinear autoregressive neural network. In *CIGOS 2019, Innovation for Sustainable Infrastructure* (pp. 1191–1196). Springer.

Liu, Q., Zou, Y., Liu, X., & Linge, N. (2019). A survey on rainfall forecasting using artificial neural network. *International Journal of Embedded Systems*, *11*(2), 240–249.

Mokrani, H., Lounas, R., Bennai, M. T., Salhi, D. E., & Djerbi, R. (2019). Air quality monitoring using iot: A survey. In *2019 IEEE International Conference on Smart Internet of Things (SmartIoT)* (pp. 127–134). IEEE.

Ni, L., Wang, D., Singh, V. P., Wu, J., Wang, Y., Tao, Y., et al. (2020). Streamflow and rainfall forecasting by two long short-term memory-based models. *Journal of Hydrology*, *583*, Article 124296.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Poornima, S., & Pushpalatha, M. (2019). Prediction of rainfall using intensified LSTM based recurrent neural network with weighted linear units. *Atmosphere*, *10*(11), 668.

Ramachandran, P., Zoph, B., & Le, Q. V. (2017). Swish: a self-gated activation function. (p. 7). arXiv preprint arXiv:1710.05941.

Ramos, M. M. P., Del Alamo, C. L., & Zapana, R. A. (2019). Forecasting of meteorological weather time series through a feature vector based on correlation. In *International Conference on Computer Analysis of Images and Patterns* (pp. 542–553). Springer.

van Rossum, G. (1995). *Python tutorial*: *Technical Report CS-R9526*, Centrum Voor Wiskunde En Informatica (CWI), Amsterdam.

Salman, A. G., Heryadi, Y., Abdurahman, E., & Suparta, W. (2018). Single layer & multi-layer long short-term memory (LSTM) model with intermediate variables for weather forecasting. *Procedia Computer Science*, *135*, 89–98.

Shanker, M., Hu, M. Y., & Hung, M. S. (1996). Effect of data standardization on neural network training. *Omega*, *24*(4), 385–397.

Singh, P. (2021). FQTSFM: A fuzzy-quantum time series forecasting model. *Information Sciences*, *566*, 57–79.

Singh, P., & Borah, B. (2013). Indian summer monsoon rainfall prediction using artificial neural network. *Stochastic Environmental Research and Risk Assessment*, *27*(7), 1585–1599.

Singh, U., Chauhan, S., Krishnamachari, A., & Vig, L. (2015). Ensemble of deep long short term memory networks for labelling origin of replication sequences. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)* (pp. 1–7). IEEE.

Willmott, C. J., & Matsuura, K. (2005). Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate Research*, *30*(1), 79–82.

Wu, C., & Chau, K.-W. (2013). Prediction of rainfall time series using modular soft computingmethods. *Engineering Applications of Artificial Intelligence*, *26*(3), 997–1007.

Xiang, Y., Gou, L., He, L., Xia, S., & Wang, W. (2018). A SVR–ANN combined model based on ensemble EMD for rainfall prediction. *Applied Soft Computing*, *73*, 874–883.

Xingjian, S., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., & Woo, W.-c. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems* (pp. 802–810).

Yucel, I., Onen, A., Yilmaz, K., & Gochis, D. (2015). Calibration and evaluation of a flood forecasting system: Utility of numerical weather prediction model, data assimilation and satellite-based rainfall. *Journal of Hydrology*, *523*, 49–66.

Yunpeng, L., Di, H., Junpeng, B., & Yong, Q. (2017). Multi-step ahead time series forecasting for different data patterns based on LSTM recurrent neural network. In *2017 14th Web Information Systems and Applications Conference (WISA)* (pp. 305–310). IEEE.

Zadtootaghaj, P., Mohammadian, A., Mahbanooei, B., & Ghasemi, R. (2019). Internet of things: A survey for the individuals' E-health applications. *Journal of Information Technology Management*, *11*(1), 102–129.

Zhang, J., Zhu, Y., Zhang, X., Ye, M., & Yang, J. (2018). Developing a long short-term memory (LSTM) based model for predicting water table depth in agricultural areas. *Journal of Hydrology*, *561*, 918–929.

Zou, M., Fang, D., Harrison, G., & Djokic, S. (2019). Weather based day-ahead and week-ahead load forecasting using deep recurrent neural network. In *2019 IEEE 5th International Forum on Research and Technology for Society and Industry (RTSI)* (pp. 341–346). IEEE.