DOI: 10.14529/jsfi220104

Machine Learning Approaches to Extreme Weather Events Forecast in Urban Areas: Challenges and Initial Results

***Fabio Porto*[1], *Mariza Ferro*[1], *Eduardo Ogasawara*[2], *Thiago Moeda*[3], *Claudio D. T. Barros*[1], *Anderson Chaves Silva*[1], *Rocio Zorrilla*[1], *Rafael S. Pereira*[1], *Rafaela Castro*[2], *João Victor Silva*[2], *Rebecca Salles*[2], *Augusto Fonseca*[2], *Juliana Hermsdorff*[4], *Marcelo Magalhães*[5], *Vitor Sá*[6], *Antonio Adolfo Simões*[1], *Carlos Cardoso*[1], *Eduardo Bezerra*[2]**

Weather forecast services in urban areas face an increasingly hard task of alerting the population on extreme weather events. The hardness of the problem is due to the dynamics of the phenomenon, which challenges numerical weather prediction models and opens an opportunity for Machine Learning (ML) based models that may learn complex mappings between input-output from data. In this paper, we present an ongoing research project which aims at building ML predictive models for extreme precipitation forecast in urban areas, in particular in the Rio de Janeiro City. We present the techniques that we have been developing to improve rainfall prediction and extreme rainfall forecast, along with some initial experimental results. Finally, we discuss some challenges that remain to be tackled in this project.

Keywords: machine learning, rainfall forecast, extreme events.

Introduction

Precipitation nowcasting, a few hours ahead forecast for extreme rainfall events, is a relevant research topic with important impact on urban areas monitoring decision-making. In large cities, such as Rio de Janeiro, heavy precipitations events have been registered, especially during summer, causing property damage, citizens mobility disruption, and even deaths. A single extreme rainfall event, occurred in 2011 in a Rio de Janeiro nearby town, claimed the lives of more than 900 people. As the effects of climate change become stronger, more frequent episodes such as this are likely to be observed [12]. Despite the efforts dispensed to improve strong events forecast accuracy, results of the Rio Operation Center[7] (COR), a department of the municipality responsible for throwing alerts of extreme events, need improvements. In a 2019 internal study, da Silva [10] analyzed 168 rain alerts, from February 2019 to May 2019, emitted by COR. On the total forecasts emitted by COR classified as *strong rain*, only 12% did materialize as such. More interestingly, 35% of the these alerts corresponded to actual *no rain* observed. Conversely, considering the total forecast events for *moderate rain*, 49% had *no rain* and 2% faced *heavy rain*. Thus, there is a clear need to improve on extreme weather forecasts in urban areas, and in particular for the Rio de Janeiro city. Current approach followed by COR involves the interpretation by meteorologists of the results of numerical weather predictions (NWPs), such as: COSMO [1] and the Weather Research and Forecasting (WRF) model [2]; and the follow-up on radar images, electromagnetic discharge and other meteorology related sensors. In this context,

[1]National Laboratory of Scientific Computing, Petropolis, Brazil
[2]Federal Center for Technological Education, Rio de Janeiro, Brazil
[3]National Observatory, Rio de Janeiro, Brazil
[4]Sistema de Alerta Rio da Prefeitura do Rio de Janeiro, Rio de Janeiro, Brazil
[5]Fundação Instituto de Geotécnica do Município do Rio de Janeiro, Rio de Janeiro, Brazil
[6]Centro de Operações Rio, Rio de Janeiro, Brazil
[7]http://cor.rio/

a recent research track has explored the opportunities of applying machine learning models in the prediction of extreme weather events [11, 16, 18]. While the prediction of normal weather condition benefits from the huge history of recorded weather related observations, which can be directly used for model training, a major challenge in the extreme weather context is the small number of events. Fortunately, extreme weather events are still rare, specially in a particular region, such as the Rio de Janeiro city. As a result, few data points are available recording the data patterns of these events. Learning under such a constrained scenario has been studied under the umbrella of *learning with small data* [5, 23].

In this paper, we describe our initial contributions towards using Machine Learning (ML) models for predicting extreme weather events. We focus on a particular type of extreme event, namely, rainfall. We present three alternative approaches we have been investigating to solve this forecasting problem, along with initial validation experiments for each one of them. The first approach aim at building deep learning models that learn spatio-temporal signals obtained from precipitation observations, captured from weather stations, as well as weather forecast data computed by NWPs. The second approach leverages DL models built using the first approach but focusing on a particular spatial forecasting region of interest. Finally, the third one, proposes an extreme weather forecast dataflow.

The remainder of this paper is structured as follows. Section 1 formalizes the problem. Next, Section 2 contextualizes the problem presenting the geographic regions of interest and available data sources. In Section 3, we describe the ML approaches we have been investigating for weather forecast. Section 4 discusses experiments evaluating the presented approaches. Section 5 presents related work. Then, in Section 6, we describe the challenges we foresee in this project. Finally, we conclude highlighting some final remarks and pointing to future work.

1. Problem Statement

In this paper, we deal with the problem of precipitation forecasting. We take a machine learning approach to it. We assume the existence of several spatiotemporal data sources from which the parameters of the forecasting models can be fitted.

Informally, a spatiotemporal data source provides measurements that have stamps in both space and time. Formally, we define a spatiotemporal data source as a sequence of observations made at regular time intervals: $\widetilde{X} = [X_1, X_2, \ldots, X_T]$. The interval Δt between two observations defines the *temporal resolution* of the data source. Each observation $X_i \in \mathbb{R}^{H \times W \times C}$, for $i = 1, 2, \ldots, T$ consists of a regular grid that determines the spatial location of interest, where H and W specify the number of horizontal and vertical elements (i.e., subdivisions) in the grid, respectively. We call each of the $H \times W$ subdivisions in the grid a *cell*. The values H and W control the *spatial resolution* of the data source, since they determine the height and width of each cell, which we denote by Δh and Δw, respectively. For each element in the grid map, C represents how many meteorological variables (e.g., precipitation, temperature, humidity) are measured simultaneously. Each data source has its particular set of observed meteorological variables.

In a machine learning-based spatiotemporal forecasting model, its parameters are fitted using a set of one or more spatiotemporal data sources as training data. Such a model can be viewed a function f that maps an input sequence of past observations $\widetilde{X}_{in} = [X_{t-T_{in}+1}, \ldots, X_{t-1}, X_t]$

to an output sequence of predicted observations $\widetilde{X}_{out} = [\hat{X}_{t+1}, \hat{X}_{t+2}, \ldots, \hat{X}_{t+T_{out}}]$ (see Eq. 1). The lengths of these sequences (denoted by T_{out} and T_{in}) may differ. However, while $C_{in} \geq 1$ (i.e., several meteorological variables can be used as predictors), $C_{out} = 1$ (since there is only one target variable, namely, precipitation).

$$\hat{X}_{t+1}, \hat{X}_{t+2}, \ldots, \hat{X}_{t+T_{out}} = f(X_{t-T_{in}+1}, \ldots, X_{t-1}, X_t) \tag{1}$$

If it is the case that more than one spatiotemporal data source is available, we assume that, as a preprocessing step, these data sources are conciliated with relation to spatial and temporal resolutions to form an aggregated data source in which each element in a grid map contains the union of all predictor variables in the component data sources.

The problem of precipitation forecasting is very challenging, and several factors contribute to this. For example, some of the data sources may have missing data, some elements in the grid maps may not be observable, conciliation of different spatial and/or temporal resolutions may be needed, inherent sparseness of precipitation data, just to mention a few.

2. Available Data Sources

As we discuss in Section 5, a number of ML based models for extreme events have been proposed lately. A common understanding is that extreme events nowcasting considers temporal extrapolation of trends derived from several types of sensors informing about local weather conditions, such as radar, satellite, meteorological stations, among others [33]. Indeed, while investigating approaches to extreme rainfall events in the city of Rio de Janeiro, we are confronted with a set of data sources that can contextualize each extreme event, compensating for the scarcity of samples and for resulting data imbalance during training.

In Fig. 1, we depict the geographical area of interest for our work. This region was delimited with the help of meteorologists that make part of our team. The total corresponding area is 84.23 Km × 48.42 Km. The upper left and lower right coordinates of the rectangle corresponding to the region are (lat = –23.1339033365138, lon = –43.8906028271505) and (lat = –22.649724748272934, long = –43.04835145732227), respectively.

In this figure, one can find markers for the location of the 33 weather stations, monitored by COR. These station provide online real-time information about weather conditions in the city. Most of these stations are only pluviometric (i.e., measure only precipitation), while seven of them are meteorological (i.e., besides precipitation, they also gather data on temperature, humidity, wind and pressure). The temporal resolution of these stations is fifteen minutes.

By analysing the small dots and triangles marked in Fig. 1, we can also observe a great imbalance in the distribution of these stations throughout the city. Moreover, the considered area contains a chain of mountains that crosses the city with significant coverage by the tropical forest, which brings humidity to the city and, at the same time, offers obstacles to the dynamics of weather systems. Rio de Janeiro is a coastal city, and is under the influence of SACZ (South Atlantic Convergence Zone), two factors that add to the complex climatic scenario observed in the city.

Most of the weather stations available in our study have been collecting data since 1997 (with some interruptions due to maintenance events). Table 1 presents summary information about the distribution of rainfall (precipitation) severity levels, considering the time series of

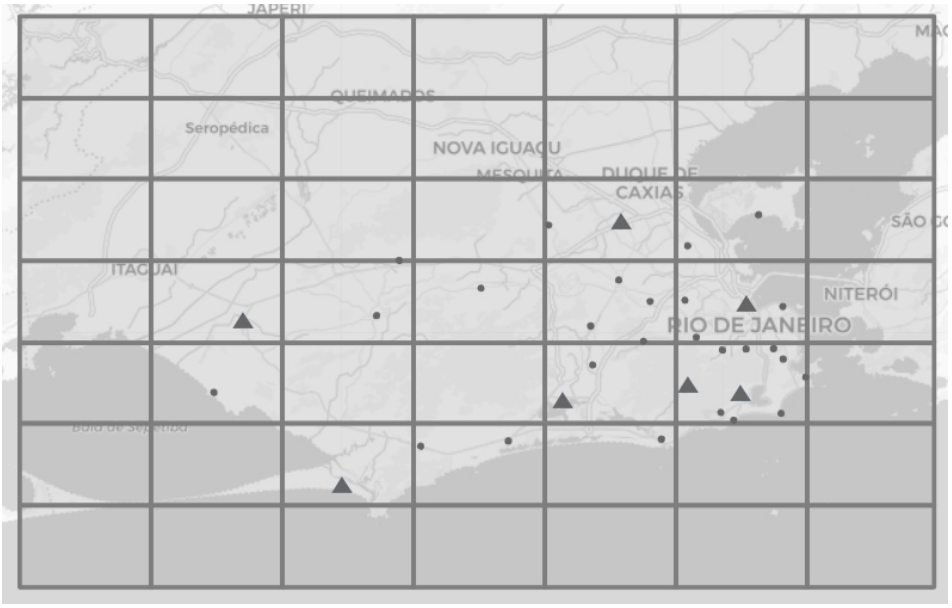

Figure 1. Spatial region of interest for our study in the Rio de Janeiro city. The light grey area correspond to land. The dark grey area correspond to sea. Small black dots mark the location of pluviometric stations. Black triangles mark the location of meteorological stations

all thirty three stations in the period from 2018-01-01 to 2021-12-31. These levels were defined by a group of meteorologists at AlertaRio, an organization linked to the city's municipality and responsible for providing weather bulletins to the population. One can notice that this data source corresponds to a very unbalanced dataset. Specifically, only 0.17% of the observations are considered extreme events, according to the AlertaRio's severity level classification scheme. However, the occurrence of these kind of events has historically caused great negative impact to the metropolitan area of Rio de Janeiro, and their precise forecasting would be of paramount importance to the city.

Table 1. Rain rate statistics of data provided by the 33 weather stations monitored by COR in the period from 2018-01-01 to 2021-12-31

Rain Rate (mm/h)	Proportion (%)	Severity Level
$0 \leq x < 5$	97.31	No / Light
$5 \leq x < 25$	2.19	Moderate
$25 \leq x < 50$	0.33	Heavy
$x \geq 50$	0.17	Rainstorm

In addition to the weather stations, there is a list of other complementary data sources that we plan to use to train ML models for extreme precipitation forecast. In Tab. 2, we depict the different data sources we aim at using to contextualize extreme weather events, and their corresponding responsible organization. Bellow, we list additional relevant information about these other data sources.

Table 2. Available data sources and their providers

Simulations from meteorological models	IAG[8]
Fluviometric stations	INEA[9]
Radars	INEA, AlertaRio[10]
Weather buoys	Brazilian Navy[11], Portal SIMCosta[12]
Electromagnetic activity	Furnas[13], SOS Raios[14]
Pluviometric/meteorologial stations	GeoRio
Satellite data	INMET[15]
Weather balloon	Tom Jobim Airport

- *Simulations from meteorological models.* We have available data from simulations produced by NWP systems, namely, COSMO and GFS25. The geographical area filtered from the two datasets covers the Rio de Janeiro city and was fixed at Latitude $[-19.5, -24.5]$ and Longitude $[-40.5, -45.5]$. The time interval was set to the period from January 2015 to March 2021. From the data obtained in the spatio-temporal region defined above, we computed a new more fine grained grid dataset. The latter includes a grid of 300×300 meters within the limits of the area defined by the original filtered dataset. The values for each new grid point are the fruit of an interpolation computing an average among the three nearest points in the filtered dataset inversely weighted by their distance to the new point.
- *Fluviometric stations.* INEA provides twenty three fluviometric stations that record river levels every 15 minutes. This data has been collected since 2008.
- *Radar data.* INEA has two meteorological radars, both with a radius do 250 Km covering the Rio de Janeiro area and their surroundings. Their coordinates are (Lat: $-22°59'35.81077"$S, Lon: $-43°35'16.65427"$W) and (Lat: $-22°24'20.99917"$S, Lon: $-41°51'37.65632"$W). This data is being collects since 2015. Each one of them produces data every five minutes.
- *Weather buoys.* These sensors collect variables such as dew point, water temperature, pressure, humidity, wind speed and direction. New data is available every one hour. This data source is available since 2019.
- *Satellite data.* We also collect electromagnetic activity data from the GOES satellite through Amazon API. This data source is available since 2018. This data source produces a data file every twenty seconds.
- *Weather baloon.* Every 6 hours, a weather balloon (filled with hydrogen or helium gas) equipped with a radiosonde raises to the atmosphere from the Tom Jobim International airport, located in Rio de Janeiro. Its goal is to collect meteorological data (humidity, temperature, air pressure) at different altitudes. This balloon reaches an altitude of 25 Km. While the balloon is suspended, its radiosonde sends the data via radio to a ground station.

[8] https://www.iag.usp.br/
[9] https://alertadecheias.inea.rj.gov.br/
[10] http://alertario.rio.rj.gov.br/
[11] https://www.marinha.mil.br/
[12] https://simcosta.furg.br/
[13] https://www.furnas.com.br/
[14] https://detectorderaios.com.br/
[15] https://portal.inmet.gov.br/

In addition to the available data sources describing real-time weather condition, COR meteorologists provided us with information regarding previous extreme rainfall events, with the dates and time of their occurrence.

3. ML Approaches for Precipitation Forecast

To face the challenges involving extreme rainfall prediction in urban areas using ML models, we count with our previous experiences in building spatiotemporal ML models for weather predictions. Our expectation is that we can build on these experiences to tackle the particularities of extreme weather events. In particular, we are currently investigating three approaches to produce ML models for precipitation forecast. In this section, we briefly describe each of these approaches. In Section 4, we conduct some initial experiments we have run to validate these approaches.

3.1. Regression Approach

We start by describing the first approach, which considers the target variable as a continuous one. Hence, in this approach we are trying to fit regression models. We investigate the application of two deep learning architectures in this approach, namely, YConvLSTM [34] and STConvS2S [7]. Both models, which are results of recent investigations carried out by the authors, are suited to capture spatial and temporal features by design and automatically make a connection between temporal and spatial features. As a result, these models can capture complex dependencies, both intra- (time only or space only) and inter-dimensional (time and space) patterns. In this section, we provide a general view of these two architectures. In Section 4, we describe how these generic schemes are instantiated to be used with the data available in our initial experiments.

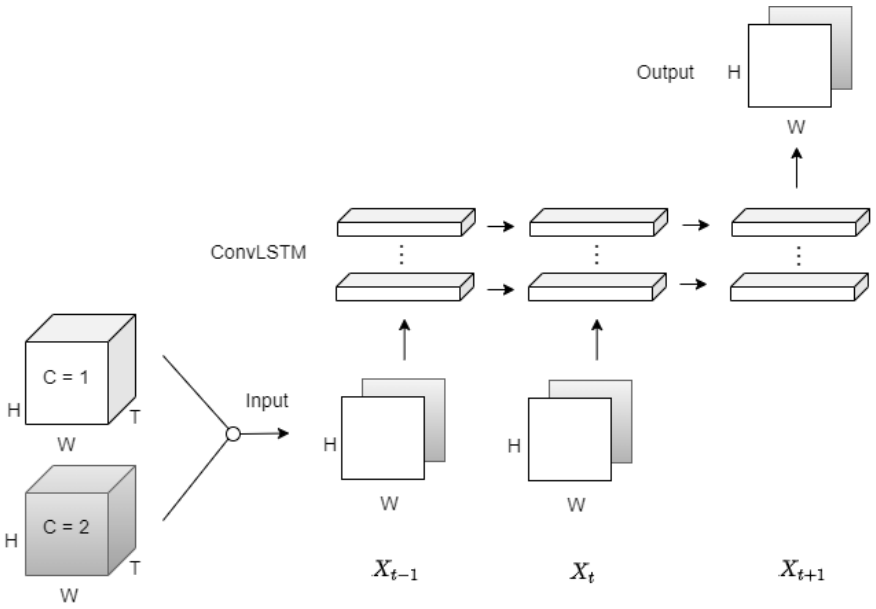

Figure 2. YConvLSTM architecture uses ConvLSTM layers to generate $X_{t+T_{out}}$ predictions that are a combination of C components, with dimension $H \times W$, applied as a channel in the convolutional layer. In this work C represents several NWP models used as input

Figure 2 depicts the YConvLSTM architecture. The latter is an adaptation of the CONVLSTM model [30], which was proposed to improve rainfall nowcasting. The main novelty introduced by the YConvLSTM deep learning model is to adopt a learning process that combines predictions from multiple numerical weather prediction models. In YCONVLSTM, the different predictions produced by the component numerical models are conciliated into a single spatial and temporal grid. Thus, at each point of the grid and time instant, a list of prediction values produced by the NWP models is associated. The slicing of the 3D grid at each time-instants produces frames that are input to the deep learning model. The list of prediction values associated at each 2D frame is mapped to the channel structure expected by the convolution input frame. The prediction process is qualified as auto-regressive, as it reads a single variable type as input (i.e. precipitation volume) and produces future predictions of the same variable type.

This can be seen as a multivariate regression model, since the variable of interest is predicted by a combination of predictors.

Figure 3 depicts the architecture of the STConvS2S model. STConvS2S [7] is a sequence-to-sequence architecture comprised exclusively of convolutional layers. Convolutions are performed with factorized 3D kernels. The architecture is comprised of three component blocks, each one in turn is a sequence of factorized convolutional layers. These blocks are applied sequentially to a given input. The first component is the *temporal block*, responsible for extracting temporal features through the temporal kernels. The temporal block uses causal convolutions to maintain temporal coherency during prediction. The second component is the *spatial block*, responsible for receiving the output of the previous block and extracting spatial features through the spatial kernels. Following the spatial block is the *temporal generator block*, designed to increase the sequence length T_{in} if the task requires a longer predictive horizon, where $T_{out} \geqslant T_{in}$. Finally, the output of this last block is further fed into a final convolutional layer to complete the prediction. Different from YConvLSTM, STConvS2S is an autoregression model, because the target variable is predicted based on its own past states.

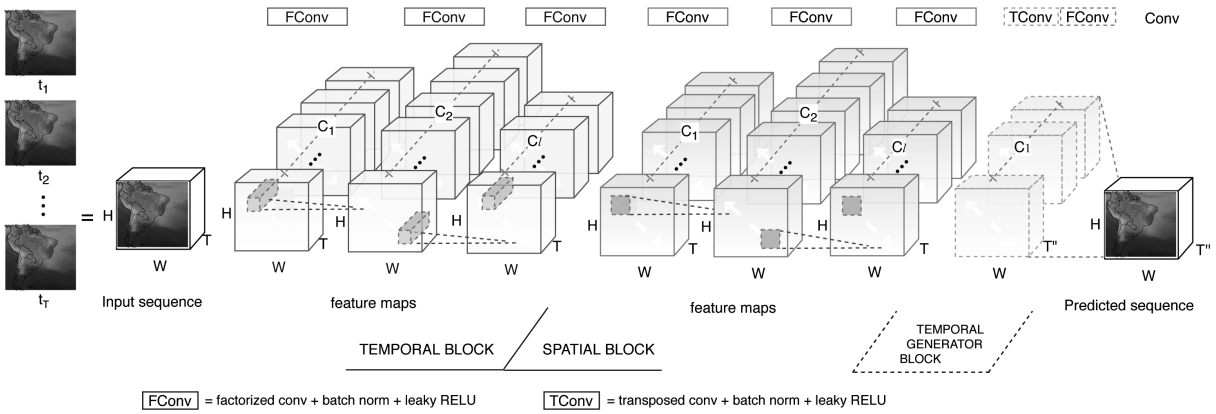

Figure 3. STConvS2S architecture is composed only of 3D CNNs, which are divided into three blocks to predict an output with temporal size T_{out}. The first block learns a temporal representation of the input sequence followed by a block that extracts only spatial features. This architecture can uses the channel in the first convolutional layer to combine several spatiotemporal inputs with dimension $H \times W \times T$. Source: [7]

3.2. DJEnsemble Approach

Another initiative developed within the DEXL Lab[16] builds on the traditional ensemble of models to achieve improved local weather forecasts. We start by taking into account the well known "No-Free-Lunch Theorem" [29, 36], which, informally, states that *no learner can succeed on all learning tasks*. A corollary to this theorem is that for a given problem there will be a region of the data space where an optimal model trained with a sampled distribution of the domain will produce predictions that are sub-optimal. We built on this result and introduced a data pre-processing step in ML model selection for weather prediction [24].

3.2.1. Data pre-processing

This pre-processing step aims at identifying regions of the data space that share similar data patterns. We consider the spatial domain of interest discretized in a set of point locations, for instance the positions of weather sensors, and model the observations at each point as time-series.

Thus, let us consider a discretized spatiotemporal domain $\mathcal{D} = \{d_1, d_2, ..., d_n\}$, where $d_i = \{p_i = (x, y), s_i\}, 1 \leq i \leq n, p = (x, y)$ is a spatial location (longitude, latitude) and $s_i =< v_1, v_2, .., v_k >$ is a time-series of observations. We accounted for seasonality effects on time-series of weather observations by splitting them into annual periods. Thus, each time-series corresponds to the observations of one location during one year. We partition the domain $\mathcal{D}$ into disjoint sub-domains $\mathcal{S}, S \subseteq \mathcal{D}$. Such partitioning is obtained through the clustering of the set of time-series s_i at each domain point $p_i \in D, 1 \leq i \leq n$. The *k-means* clustering algorithm was used to compute the disjoint partitioning of the domain. In order to compute the distance between time-series, we adopted the *Dynamic Time Warping (DTW)* algorithm [27]. Thus, after clustering, each time-series s_i is annotated with the cluster-id it was associated to. The choice involving the number (ie. k) of partitions for a given time-series dataset is obtained by computing partitions for different values of k and using the *Silhouette* score [26] to rank the resulting clustering. The *Silhouette* score is obtained by first computing a per time-series mean internal clustering DTW distance (a) divided by the mean extra-cluster DTW distance, for the nearest extra-cluster (b). The *Silhouette* score is obtained by computing the mean over all per time-series scores. Finally, in order to define rectangular spatial areas that conform to the expected frame structure of data inputs to Deep Learning (convolutional) models, we structured the domain time-series into disjoint rectangular tiles, $T = \{t_1, t_2, \ldots, t_t\}$. In this setting, each time-series $s_i \in D$ is associated to a single tile $t_j \in T$. Moreover, tiles are constructed such that a maximum ϵ of variation in different clusters is observed. Thus, given $s_i \in t_j$, the set of time-series associated to a tile t_j, we define $tx = \arg\max_k sum(x_i, x_i = 1 \; if \; s_i \in c_k)$ as the maximum number of time-series associated to the same cluster in tile t_j, with c_k identifying a cluster. Then, the tiling algorithm assures the constraint for all tiles $t_j \in T, 1 \leq j \leq t$: $\frac{tx}{|t_j|} \leq \epsilon$, where $|t_j|$ is the number of time-series in tile t_j. Therefore, at the end of the pre-processing stage, we have a tiled spatiotemporal domain, where each tile exhibits similar data behaviour corresponding to its weather history. As a final step in data pre-processing, we compute for each tile t_i its time-series representative, s_r. The representative in a tile t_i is a time-series $s_r \in S$ whose average DTW distance, $\mathrm{dtw}(s_r, s_j)$, between s_r and all time-series $s_j \in t_j$ is the smallest. Finally, we expect that the structuring of spatiotemporal domain into tiles to provide a more precise learning experience such that a model optimized to a tile is less prompt to suffer from the effect of the *non-free-lunch* theorem when applied to tiles with similar data distribution.

[16] http://dexl.lncc.br

3.2.2. Model selection

Once the spatiotemporal domain $\mathcal{D}$ has been structured into tiles, the DJEnsemble approach is ready to compute predictions. We consider spatiotemporal predictive queries $Q = \{R, M\}$, where $R \subseteq \mathcal{D}$ is a spatiotemporal region where weather predictions are to be computed and $M = \{m_1, m_2, ..., m_m\}$ is a set of pre-trained candidate ML models. Each m_j model has been trained in a subset $\mathcal{S}_j$ of the domain $\mathcal{D}$. Therefore, for each m_i model, there is a set of tiles $\{t_1, ..t_k\}$ that were used for the model m_i training. We can interpret that as building specialized ML models for a region of particular weather characteristics. Thus, we could have a model built on time-series of the south region of the Rio de Janeiro city, which is a coastal area between the sea and the mountains, and another model built on the north of the city where weather tends to be hotter and less influenced by the sea humidity. The DJEnsemble approach works by finding for each predictive query Q the set of models in $M' \subseteq M$ whose training data, induced by the data distribution of tiles used for their training, most closely approximates the data distribution of the tiles covered by the query region $Q.R$. This is obtained by comparing the distances between time-series representatives among the query tiles and the model training tiles. Observe that the set of candidate models M may not have been trained on data from query region $Q.R$. Nevertheless, DJEnsemble is still able to indicate the best candidate model whose data distribution most closely approximates that of the query region. Finally, we want to highlight that the improvements on extreme weather events precision forecasts may involve the ability to use specialized models whose training capture nuances of the weather behavior of a region. We expect that the ideas induced by the DJEnsemble may contribute to a more precise ML based extreme weather forecast.

3.3. Classification Approach

In this approach, we frame the problem of forecasting extreme rainfall events as a binary classification task. It considers extreme precipitation as an anomaly on the time-series of rainfall observations. We consider a particular type of spatiotemporal data source, namely, data coming from a set of meteorological stations, each one of them having collected observations about multiple weather variables over a given period of time, one of them necessarily being precipitation. We also assume the times series coming from these meteorological stations all have the same temporal resolution. For the particular data source we used, this temporal resolution was fifteen minutes.

3.3.1. Construction of the labelled dataset

As a first step in this approach, we had to build a classification dataset, in which the target variable is binary: the value 1 indicates the occurrence of an extreme precipitation, and the value 0 indicates otherwise. Hence, we had to map continuous values of precipitation to discrete (binary) values. For this, we built a semi-automatic instance labelling system. In particular, we used a particular type neural network, the Self-organizing Map (SOM) [19]. SOM neural networks are models based on the functioning of neurons in the human brain's cortex, in which learning happens through a process of competition among its neurons. During training, a multidimensional data point is mapped to a 2D topological map. The latter suggests a visual interpretation of the input data, where discrepant points can be highlighted [20]. To define a

SOM architecture, several hyperparameters must be defined, such as dimensions and shape of the topological map, and the neighborhood function.

To train the SOM model, we concatenated all the time series coming from the meteorological stations. As a result, a multivariate time series was built containing observations from all stations. Notice that, when building this dataset, we completely disregarded the spatial locations of the meteorological stations. The rationale for that is that we wanted the SOM model to capture patterns of anomaly (with respect to the precipitation variable) irrespective of the spatial location.

During training, the data instances were processed in incremental windows of 96 instances, each window corresponding to one day of measurements of the meteorological stations. When processing the set of windows, say w_i, the model also receives the information provided by the previous map on w_{i-1}, in order to form the map solution set S.

As a last activity in this step, the solution set of generated maps S is visually analyzed to assist in labelling the instances. The descriptive method we established for the visual analysis of SOM maps to detect a shape anomaly consists of interpreting the amplitudes of the displacement rates of the distances between the spatial components x, y and z in relation to the sequence of maps $s_i \in S$. A shape anomaly is detected and annotated in Y if there is a discrepant amplitude of the component z, correlated to (x, y), whose displacement varies in a discrepant way in relation to the adjacent maps s_{i-1} and s_{i+1}, where (x, y) represents the topographic ordered pair in the map and the component z the topographical error.

3.3.2. Assessment of the labelled dataset

The visual interpretation of the SOM model in the first step was helpful because it assisted in the labelling of observations as either normal or anomalous. The goal of this second step was to assess the quality of the labelling conducted in the first step. For this, we fitted a decision tree (DT) model to the labelled dataset.

A DT is a symbolic classification method applied in this work as a tool to assess the quality of the annotations of detected events. In the structure of a DT, each data instance follows a unique path for its classification, depending on a set of comparison rules, of the *If-Then* type, which are performed in each node of the tree, determining whether the result should proceed left or right, thus building the DT [31]. The DT learning algorithm splits the training example into subsets, represented in its leaves, in which points falling in the same leave are explained by a conjunction of predicates over independent features of the dataset.

In our scenario, the dependent variable is the label indicating the value status as normal or anomalous, whereas the independent variables are the features describing each time instant: precipitation, humidity, etc. If we consider that the independent variable observe a certain state when anomalous phenomenon occurs, which differs from the state of a normal status, then we can expect leaves comprising a single value. Therefore, we propose to compute the entropy of a DT classification. DTs with lower entropy would indicate that leaves share similar values and the annotations of anomaly would tend to be correct. Thus, as a criterion for evaluating labelling step, we established that the more accurate the adjustment of the DT algorithm, the better would be the qualitative labeling process of anomaly detection. If the results obtained after fitting the DT model were not satisfactory, the process is restarted with a new hyperparameter configuration for training the SOM model. Otherwise, we proceed to the third (and last) step to fit a classification model to the previously labelled dataset.

3.3.3. Model fitting through LSTMs

In this last step, the goal was to fit the binary classification model to predict the label (either extreme/anomalous or normal) for a given instance. For this, we used a Long Short-term Memory (LSTM) neural network, due to the capacity of its units to learn long sequential patterns of data behavior. The LSTM architecture is a type of recurrent neural network, which are designed to analyze the behavior of data sequences over time. As stated in [13], an LSTM model addresses the problem of vanishing gradients, incorporating functions *(gates)* into its state dynamics to maintain or discard information. The original LTSM formulation features three gates: *input*, *forget* and *output*.

After model fitting, the model is evaluated. If the obtained results are not satisfactory, the process is restarted from step 1 (dataset labelling). Otherwise, the resulting classification model is ready to be used for inference.

3.4. Discussion

The YConvLSTM model [34] implements a post-processing ensemble approach on the output of NWPs. It builds on the architecture proposed by Shi et al. [30] extending it to cope with the input coming from n NWP simulators. We also discussed STConvS2S, originally presented in [7]. The model comprises a autoregressive CONV2D+1 architecture suited for spatiotemporal prediction. Although these two models were able to minimize rainfall prediction error with respect to individual NWPs, they were not designed for the forecasting of extreme weather events. In fact, as discussed in Ding et al. [11], deep learning models tend to either *underfit* or *overfit* on extreme weather predictions, due to the imbalance on training datasets. Thus, as an initial attempt to close this gap, we presented a dataflow of machine learning models, combining *Self Organized Maps (SOM)*, *Decision Tree* and *LSTM*, specifically designed for extreme precipitation detection. The task is modeled as a binary classification where extreme events are annotated to the observational data. Finally, we presented the *DJEnsemble* approach that considers a spatiotemporal ensemble of ML models.

4. Experiments

In this section we report initial experiments with all approaches described in Section 3. These experiments aim at highlighting the potential of the different approaches towards the challenges in extreme weather events forecast.

4.1. Validating the Deep Learning Approaches

In this section, we present some preliminary experiments we conducted to train and validate out two Deep Learning models, YConvLSTM and STConvS2S (see Section 3). In these experiments, we trained both models using the same dataset. This dataset contains precipitation simulation data obtained from two numerical models, COSMO and GFS25. The temporal range of the observations is from January 2018 to April 2021. The data are arranged in a 7×7 grid covering the city of Rio de Janeiro (see Fig. 1). The time distance between each grid (i.e., the temporal resolution) is 3 hours.

For training the YConvLSTM and STConvS2S models, we set the following learning task: we use the previous ten observations (at timesteps $t-9, t-8, \ldots, t$) to predict the next observation

(at timestep $t + 1$). We use as predictors the outputs of the above mentioned NWP models. Our target variable is precipitation. Furthermore, we used data in the temporal range between years 2018 and 2019. Data from 2020 were used for validation. Finally, we set data collected in 2021 for testing their generalization performance, which we estimated using Mean Squared Error (MSE). The sizes (i.e., number of observations) of the training, validation and test sets are 5840, 2928, and 849, respectively.

The hardware infrastructure used in training both models has the following hardware settings: CPU Intel(R) Core(TM) i7-7740X CPU @ 4.30GHz; GPU: GeForce GTX 1080 Ti 11GB; RAM: 32GB.

We conducted 10 runs of training using the STConvS2S architecture. For these ten runs, we got an average training time of approximately 14 min, with a standard deviation of 3.91 seconds. Also for these 10 runs, the resulting average MSE (on the test dataset) was 2.0052, with a standard deviation of 0.2145. Figure 4 presents the learning curves resulting from training the STConvS2S model in one of the runs. The hyperparameters used to fit the STConvS2S model were the following: learning rate = 5e-6; optimizer: RMSProp, with alpha set to 0.99; eps: 1e-6; weight decay set to 1e-1. We trained the model for 200 epochs, and used a batch size of 64. The resulting MSE on the test set was 1.802.

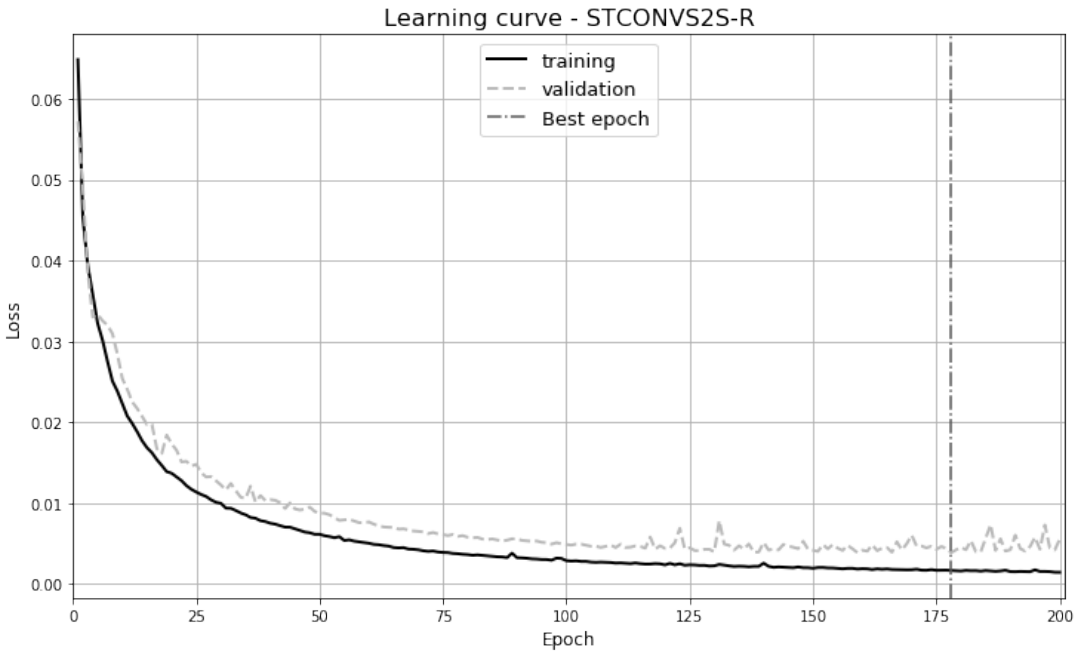

Figure 4. Learning curves obtained during one of the runs of STConvS2s model training. The losses are MSE values measured over the rainfall domain. Hence their unit is *squared millimeters* (mm^2) of rain over the specified period. The vertical dashed line indicates the epoch in which the best model (measured in the validation set) was obtained

To assess the difficulty of detecting extreme events, we manually picked ten events between January and April of 2021 that were considered extreme in our test dataset. We wanted to know if the model was capable of making good predictions of these picked up extreme events. We analysed two scenarios. We first considered all the spatial observations of these ten extreme events that presented the highest precipitation. The resulting MSE in this case was 22.20. In the second scenario, we considered the ten largest precipitation values in the same time period. In this case, the MSE value increased to 174.88. As a conclusion, the model error considering

only extreme events was much higher when compared to the average MSE value measured in the whole test set. We attribute this difference in predictive quality to the sparseness of extreme events, one of the many challenges we face in this project (see Section 6 for further discussion on this subject).

In a second validating experiment, we trained the YConvLSTM model using the same hardware infrastructure described above. The model was trained for 200 epochs with a batch size of 64, as its training time is significantly higher than the STConvS2S. The architecture consists of two ConvLSTM layers with 32 and 64 filters each and was optimized using the Adam optimizer and Keras default hyperparameters. We achieved the loss of 1.773 MSE on the testing set. The average training time was 50 minutes. Figure 5 presents the learning curves resulting from training the YConvLSTM model in one of the runs.

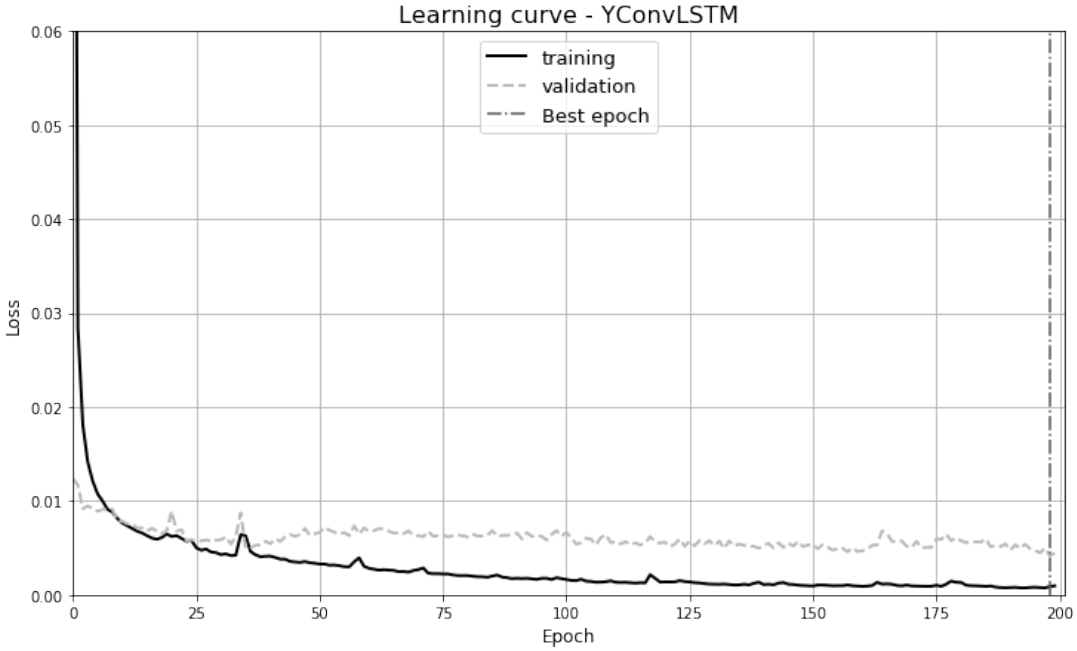

Figure 5. Learning curves obtained during one of the runs of YConvLSTM model training. The losses are MSE values measured over the rainfall domain. Hence their unit is *squared millimeters* (mm^2) of rain over the specified period. The vertical dashed line indicates the epoch in which the best model (measured in the validation set) was obtained

We also conducted the same experiment regarding extreme events on the predictive model produced by the YConvLSTM architecture (we used the same manually picked 10 extreme events in our test dataset). In the first scenario, the resulting MSE was 100.23, and in the second scenario the value increased to 534.18. As already happened with STConvS2S, the YConvLSTM model also had much difficulty to correctly predict extreme events.

4.2. Validating the DJEnsemble Approach

In this section, we depict an experiment comparing the DJEnsemble approach against other ensemble types. In the traditional ensemble approach, each available model is run over the entire query region and the prediction results are linear combinations of each prediction. In the stacking ensemble approach, a model is trained with data produced by a set of component models. The new model can be a linear regression learning algorithm, for example. The predictions are

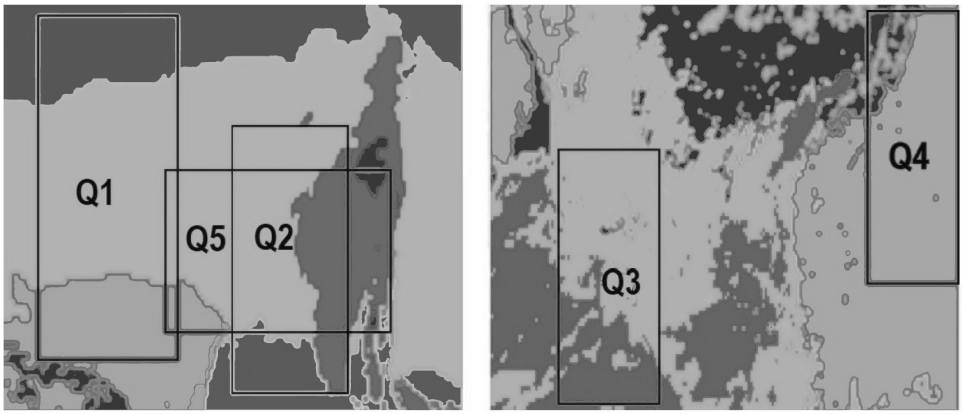

Figure 6. Queries over the temperature (left) and rainfall (right) domains

obtained by invoking the stacking model (ie. the predictor) on the outcome of the component models' predictions.

The data used in the experiments is a subset of the Climate Forecast System Reanalysis (CFSR) dataset that contains air temperature observations from January 1979 to December 2015 covering the space between 8N-54S latitude and 80W-25W longitude (temperature dataset) [22]. Additionally, we use a subset of the rainfall dataset from NASA's TRMM and GPM missions, with rainfall collected for the same spatial region as in the CSFR dataset over 22 years (rainfall dataset) [14]. The inclusion of experiments with temperature and precipitation aim to assess the approach prediction quality under different spatio-temporal patterns. We present the results of five queries executed on these data. Temperature is considered by meteorologists as an easy variable to model as opposed to rain, which is considered one of the most complex variables to predict. The temperature is homogeneously distributed in large spatial regions, while the rainfall is more heterogeneous and concentrated in small areas. Based on these, we can argue that we define queries on data with different behavior (i.e. data distribution). Figure 6 depicts the regions corresponding to every query. Each color represents a region with different data distribution (i.e. cluster). Table 3 summarizes the results for a traditional ensemble, a stacking ensemble, and DJEnsemble. DJEnsemble achieves the best accuracy for all the queries. The gap between DJEnsemble and the other ensembles is as much as a factor of 9.

Table 3. RMSE over the temperature (Q_1, Q_2 *and* Q_5) and precipitation (Q_3 *and* Q_4) domains. Precipitation values represent an estimate for the volume of rainfall, in millimeters, in a global 0.1^o spatial grid and half hourly intervals. The temperature values correspond to a resolution of 6 hours intervals and a spatial grid of 0.5^o, in Celsius

Query	R[lat, lon]	Trad. ensemble	Stacking	DJEnsemble
Q1	[70:130, 95:140]	25.01	7.28	3.35
Q2	[60:110, 40:80]	27.88	5.52	4.29
Q3	[125:175, 25:90]	14.28	13.96	5.34
Q4	[0:40, 60:130]	8.80	12.94	6.04
Q5	[59:100, 25:100]	29.10	5.04	3.18

These results, originally presented in [24], are reproduced here to illustrate the potential of the DJEnsemble approach towards more precise data driven ML model predictions of meteorological variables.

4.3. Validating the Classification Approach

In this section, we present an experimental evaluation of the method introduced in Section 3.2.2. The hardware configuration used in these experiments is the following: CPU: 280 × Intel(R) Xeon(R) Gold 5220 CPU @ 2.20GHz; RAM: 512GB.

We used data from the seven meteorological stations (see Fig. 1). Specifically, we considered the periodic measurements of three variables, namely *precipitation*, *temperature* and *humidity*. We selected a subset of these time series covering the period from December 1st, 2015 to March 17th, 2021. To prepare the data, some pre-processing steps were required. Initially, we applied data normalization and interpolation of missing values through the mean value. In the results reported in this study, there was no treatment for removing noise in the data, as this operation had a negative influence on the dataset labelling step (Section 3.3.1). The normalization of the input data for an interval between [0, 1] was obtained through the method *MinMaxScaler* [28]. Working with data at the same scale is a good practice when it comes to distance calculation algorithms. It also speeds up the optimization process for generating models via gradient-based learning algorithms, as in the case in SOM neural networks.

In the first step (see Section 3.3.1), we started by training the SOM model. For this, 100.000 iterations for training were defined, the learning rate starting at 0.6 with the radius of the training area starting at 0.5. Then, the visual analysis of the generated maps and the subsequent annotation of the instances were performed on the data set.

After the first step, the resulting labelled dataset was subdivided into six disjoint subsets (A1 to A6). Table 4 summarizes these subsets.

Table 4. Information about Meteorological datasets

Subset	Number of Instances	Period
A1	38.112	01/12/2015 a 31/12/2016
A2	35.040	01/01/2017 a 31/12/2017
A3	35.040	01/01/2018 a 31/12/2018
A4	35.040	01/01/2019 a 31/12/2019
A5	35.136	01/01/2020 a 31/12/2020
A6	7.296	01/01/2021 a 17/03/2021

To exemplify the way in which the maps produced by the SOM model were visually analyzed to pinpoint extreme precipitation events, the SOM output for subset **A6** is presented in Fig. 7. (However, this process was applied to all subsets.). The amplitudes of the component $\boldsymbol{z}$ correspond to the similarity errors of the instances in the accommodation in the node in the topographic mesh of the produced map. High amplitudes represent extreme events. Figure 8, in addition to the $\boldsymbol{z}$ component, the $\boldsymbol{x}$ and $\boldsymbol{y}$ components correspond to the topographic coordinates of the map. Vertical lines in red correspond to the beginning of the annotation of detected anomalies. The correlation between the three components represents the similarities, as well as the breaking of the similarity in relation to the input space. Thus, these correlations between

the peaks of the z component and the sudden and prolonged change in the x and y components were interpreted as anomalies of form.

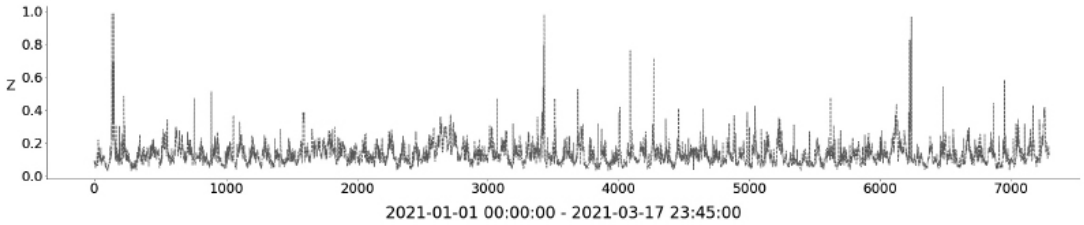

Figure 7. z z-component produced by SOM for subset **A6**

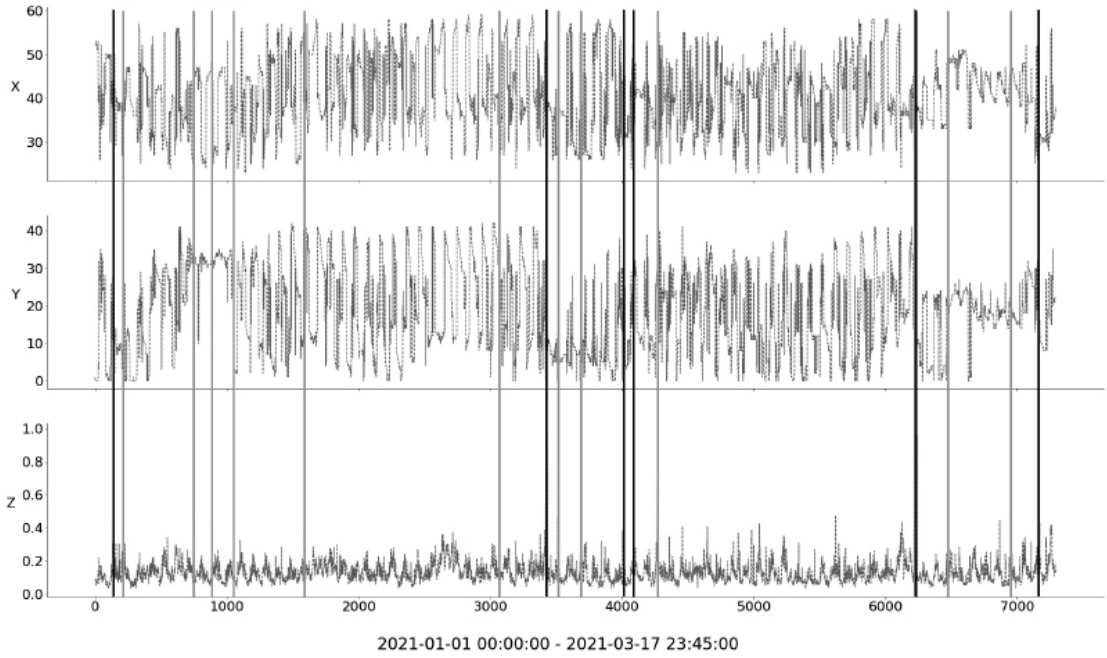

Figure 8. z components x, y, and z produced by the SOM for the subset **A6**. Vertical lines in black correspond to major shape anomalies detected and vertical lines in light gray correspond to minor anomalies

To validate the accuracy of the classification performed in the previous step, seeking to reduce subjectivity inherent to the visual analysis, a DT algorithm was applied and the result evaluated for each subset. The annotated dataset was divided into 70% for training and 30% for testing. We evaluated the labels annotated in the datasets by analyzing the test of the models fitted with such data, using the cross validation, ROC curve, and F1-score metrics. Obtained results are shown in Tab. 5.

Table 5. Summary of decision tree results for the entire dataset

Dataset	ROC	Precision	Sensitivity	F1-score	Cross Validation (10x)
A1 to A6	0.8976	0.98/0.85	0.99/0.81	0.98/0.83	0.9700 (+/– 0.01)

Finally, the step described in Section 3.3.3 was carried out to generate the binary classification model for predicting the occurrence of extreme events. An LSTM neural network model was trained for each subset. Similar to what we did in the second step, the data instances in each subset were also split in 70% used for training and 30% for testing. An *Embedding* layer was used in order to make the data streams continuous and dense.

Due to the imbalance of binary labels, experiments were performed with different balance proportions using the Synthetic Minority Over-sampling Technique (SMOTE) class from the imbalanced-learn library [15]. As mentioned by the documentation of this library, SMOTE and its variations work best if done in combination with some major class subsampling method. Thus, the applied methods of oversampling and undersampling are applied consecutively through a pipeline on the data provided by SMOTE and the random method *RandomUnderSampler*. Thus, all LSTM models were trained, tested and evaluated using the SMOTE methodology, according to the aforementioned strategy.

The network architecture was defined with a single hidden layer with 21 LSTM units. The *Adam* optimizer with a learning rate of 0.001 was used. The number of training epochs was set to 100, and the batch size equal to 32. The loss function used was the *binary_crossentropy*. The value of 252 was determined for the length of the string that returns to the model. In order to reduce *overfitting*, one *Dropout* layer with value of 0.3 were used. Finally, an output layer of just one unit was defined with the *sigmoid* activation function providing the result as a probability value. Each classification model trained with one given subset was validated against the other subsets.

We built ROC curves to evaluate the classification performance of the models fitted with each subset. Figure 9 illustrates the results obtained for the model trained on subset **A6** (we present just one curve for lack of space).

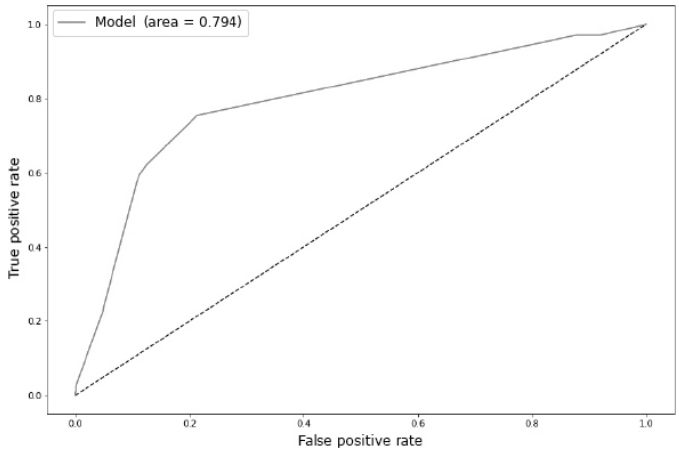

Figure 9. ROC plot of trained model with **A6** dataset

Comparisons between classification performances are shown in Tab. 6. Each model was trained with a given subset and then applied to the other subsets for inference. The first column corresponds to the models generated by the subset in parentheses. The best results are highlighted in **bold**. We considered the union of the subsets as a sample of the complete data domain. Under this assumption, and considering the proportion to the total of samples at each subset, we computed the standard deviation (StdDev) for the F_1 score for each model. The standard deviation average for all models is 0.07764.

Table 6. Summary of LSTM model validation results. F_1-score metric

Model/Data	A1	A2	A3	A4	A5	A6	StdDev
m(A1)	–	0.4659	**0.6779**	0.4476	**0.6152**	**0.6590**	0.0441
m(A2)	0.5877	–	**0.6797**	0.4490	**0.6141**	**0.6598**	0.0367
m(A3)	0.4116	0.0964	–	0.2919	**0.6110**	**0.6571**	0.0945
m(A4)	0.3924	0.4620	**0.6811**	–	0.1102	0.0903	0.1021
m(A5)	0.5898	0.4697	**0.6796**	0.4479	–	**0.6607**	0.0433
m(A6)	**0.6267**	0.0025	0.0125	**0.7743**	0.5609	–	0.1450

5. Related Work

Nowcasting deals with the problem of short-term forecasting of extreme weather events. These events may include the determination of the future track of a particularly severe storm for warning purposes, the estimation of the amount of additional precipitation that will fall in a given area in the next few hours, severe wind, winter precipitation types, etc. In addition, various types of meteorological data could be used to forecast severe weather events, including radar measurements, satellite data, and weather stations' observations [35]. This section presents a literature review of recent solutions for weather nowcasting applied on different source of data. The works presented here focus on Artificial Intelligence, mainly in Machine Learning based solutions.

Ravuri et al. [25] developed a Deep Generative Model of rainfall (DGMR) for radar's probabilistic nowcasting of precipitation. Their nowcasting algorithm is a conditional generative model that predicts N future radar fields given M past radar fields using radar-based estimates of surface precipitation at a given time point. Learning is framed in the algorithmic framework of a conditional generative adversarial network (GAN), which is specialized for the precipitation prediction problem and is driven by two loss functions. These functions guide the parameter adjustment by comparing real radar observations to those generated by the model. The first loss is defined by a spatial discriminator, a CNN that aims to distinguish individual observed radar fields from generated fields. The second loss is defined by a temporal discriminator, a 3D CNN that aims to distinguish observed and generated radar sequences. The model is trained on a large corpus of precipitation events, 256×256 crops extracted from the radar stream, of length 110 min (22 frames). All models are trained on radar observations for the UK from 2016–2018 and evaluated on a test set from 2019. DGMR performance ranked first for its accuracy and usefulness in 89% of cases against two state-of-art methods (PySTEPS and Unet). According to the authors, their model produces realistic and spatiotemporally consistent predictions over regions up to 1,536 km $\times$ 1,280 km and with lead times from 5–90 min ahead. Despite the accuracy and operational utility, the prediction of heavy precipitation at long lead times remains difficult for all approaches.

The work of [8] proposes a classifier named RadRAR (Radar products values prediction using Relational Association Rules) for convective storms nowcasting based on radar data. In addition, RadRAR is intended to prove that relational association rule mining applied on radar data helps discriminate between severe and normal weather conditions. RadRAR is trained on radar data collected from normal weather conditions and learns to predict whether the radar echo values will be higher than 35dBZ, indicating the occurrence of a storm. Experiments are

conducted on real radar data provided by the Romanian National Meteorological Administration for the 5th of June 2017. It was a day with moderate atmospheric instability manifested through thunderstorms accompanied by heavy rain and medium-size hail. Analyzing the experimental results and the comparison to existing approaches, the authors conclude that the RadRAR is effective for predicting if the radar echo values are higher than 35dBZ, obtaining performances better than the results from the literature and a Critical Success Index of 61%.

Agrawal et al. [3] present an application of DL to the problem of predicting the instantaneous rate of precipitation one hour into the future from radar data with high-resolution (1 km $\times$ 1 km). More specifically, the model uses the ubiquitous U-Net Convolutional Neural Network (CNN). Dataset is from multi-radar multi-sensor (MRMS), removing non-meteorological artifacts and projects the combined observations onto a rectangular grid. Forecasting is treated as an image-to-image translation problem where is given a sequence of n input radar images that start at some point of time, t_{in1}, and end at t_{inn}. The task is to generate the radar image at some point in the future, t_{out}. Before training the model, data is transformed, labeling images by quantifying precipitation rates into four discrete ranges, based on three thresholds of millimeters of rain per hour. This step provides three binary classifications that indicate whether the rate exceeds thresholds that roughly correspond to trace rain, light rain, and moderate rain. In sequence, they partition the US into 256 km $\times$ 256 km tiles and independently make predictions for each tile. Model is trained on data collected in 2018 and tested on data for 2017 and 2019. Results show their model performs better than MRMS persistence, optical flow (OF) and, the High Resolution Rapid Refresh (HRRR) system (the current best operational NWP available from NOAA) one-hour forecast methods. However, once the prediction window is increased to approximately 5 hours, the HRRR models consistently outperform their approach.

Sønderby et al. [32] introduce MetNet, a Neural Weather Model (NWM) that forecasts rates of precipitation with a lead time of up to 8 hours at the high spatial resolution (1 km $\times$ 1 Km) and at the temporal resolution of 2 minutes using radar images and spectral bands of satellite as input data. MetNet covers a 7000×2500 km geographical area corresponding to the continental United States. MetNet relies on mosaicked ground based radar and satellite imagery as input and the predictions take in the order of seconds independently of lead time and can be done in parallel. They cast precipitation forecasting as a structured prediction problem where the output comes in the form of a three-dimensional tensor. Each value of the tensor corresponds to a time and a location and indicates the corresponding rate of precipitation measured in mm/h. Target precipitation rates are estimated by the MRMS ground based radars as a function of the returned radar echoes. MetNet architecture combines a CNN, which encodes the time slices sampled every 15 minutes with a Convolutional LSTM to processes the time slices in the direction of time. Results show the performance of MetNet at various precipitation thresholds and find that MetNet outperforms HRRR Numerical Weather Prediction at forecasts of up to 7 to 8 hours on the scale of the continental United States.

The work [9] explores SOM to detect patterns on real radar data from Romenia National Meteorological Administration. The main goal is to provide better insight regarding how the values of weather radar products are evolving in time, both in calm and severe weather conditions, with the broader goal of using these findings for weather nowcasting. In addition, they investigated if SOM can distinguish severe weather conditions from radar data. The exported raw data collected through the radar scans during one day (24 h) on a particular geographic region is provided as a sequence of dimensional matrices. A matrix corresponds to a specific

timestamp t and a meteorological product. For each time moment t, a sequence of matrices (3D data grid) is available, containing the values for various radar products at time t. Two data sets were constructed for representing the radar data collected during the timestamps on a day. One dataset uses the entire set of meteorological products provided by the radar (i.e., 24). Another one employs only 13 products: base reflectivity of particles on six elevations, velocity on six elevations, and the estimated quantity of water contained by a one square meter column of air. To detect the underlying structure of the datasets, the SOM model is applied to obtain an unsupervised two-dimensional representation of the datasets . The corresponding U-matrices are analyzed and compared for assessing the relevance of the meteorological products used in detecting the timestamps when a certain meteorological event occurred. Through this methodology, SOM detects temporal intervals where certain meteorological phenomena occur. The U-matrix visualization for the SOM built using the proposed data model shows readable changes in the meteorological products almost 2 hours before the event, which might help forecast the start of the phenomena. In general, they conclude that there is a slow change in the values over time, except when certain severe phenomena occur.

RainNet [4] introduces a deep neural network that aims at learning representations of spatiotemporal precipitation field movement and evolution from radar data to provide skillful precipitation nowcasts. RainNet follows an encoder-decoder architecture, as U-Net [3]. The encoder progressively downscales the spatial resolution using pooling, followed by convolutional layers. The decoder progressively upscales the learned patterns to a higher spatial resolution using upsampling, followed by convolutional layers. RainNet was trained to predict precipitation at a lead time of 5 min, using several years of weather radar. Dataset covers Germany with a spatial domain of 900 km $\times$ 900 km and has a spatiotemporal resolution of 1 km in space and 5 min in time, respectively. Verification experiments were carried out on 11 summer precipitation events from 2016 to 2017. Since RainNet uses a convolutional architecture and does not use LSTM layers to propagate information through time, it was only trained to predict precipitation at 5 min lead times. So, a recursive approach was implemented using RainNet predictions to achieve a lead time of 60 min. Results showed that RainNet competes against a conventional nowcasting model based on optical flow. This work confirms [3] that CNNs provide a firm basis for competing with conventional nowcasting models based on optical flow.

Song et al. [33] present an Artificial Neural Network (ANN) nowcasting model for hourly summer precipitation over the Eastern Alps. The ANN model, a Multilayer Perceptron (MLP), is developed for the nowcasting of hourly precipitation for a one-hour lead time. Several predictors from diverse sources are used as input, including QPE, QPF, three INCA convective analysis data (CAPE, CIN, MCONV), and three radar data composite measurements of five C-band radars (MAXCAPPI, ECHOTOP, VIL). The performance of the MLP model is compared with the traditional INCA extrapolation technique and a multiple linear regression approach (MLR) model. The available period for both radar products and raw INCA forecasts is the summer period of 2012–2014 (June to September) yielding 12 months in total. The dataset is divided into the training set, containing ten randomly chosen months, and the verification set, containing the remaining two months. The independent verification months, August 2013 and July 2014, are used to evaluate the performance of the proposed nowcasting methodology. Precipitation case studies were analyzed, and the results showed that both the MLR and MLP models could predict the precipitation similarly to the INCA extrapolated forecast. However, the forecast by MLP model is better than INCA, providing more detailed small-scale structures. Also, the

precipitation intensity of the MLP model forecast is closer to gridded precipitation observations than the INCA.

The study presented in [6] uses video prediction deep learning (VPDL) algorithms applied in precipitation nowcasting for São Paulo, Brazil. They use the PredRNN++ as a VPDL model to predict reflectivity images and precipitation edges from weather radar images for up to 1-h lead time and compare the results with ENCAST, an extrapolation-based model used for precipitation nowcasting. PredRNN++ utilizes causal LSTM to capture complex dependencies and variations and gradient highway unit (GHU) to keep the gradients during training. This study uses a dual-polarization S-band (SPOL) Doppler weather radar from the Department of Water and Electric Energy, manufacture Selex ES GmbH, installed at Ponta Nova, State of São Paulo, Brazil. These radar measurements are restricted to 100 km of range for the period of March 2015 to December 2019 with a time interval of 5 min. The results have some limitations, and the authors propose several improvements; despite this, they advocate that VPDL model has good potential as an additional tool to assist nowcasting.

Jiang, Ma and Wu [37] main purpose is to propose nowcasting precipitation prediction at target station within 1–2 hours in the future using the radar historical map sequence in the past 1.5 hours. Also, a solution to deal with missing data in this kind of dataset. For this purpose, they train a CNN and two auxiliary ML models (NN and gradient boosting decision tree – GBDT) using different features from the real Hubei dataset. This dataset comprises radar reflectivity maps, precipitation and rainfall time recorded by ground monitoring stations, and stations' altitude, latitude, and longitude. For training are used 2300 samples and the remaining 200 samples for evaluation. Results showed that the precipitation nowcasting mechanism was suitable for radar reflectivity data with single or cumulative altitudes. The models showed better performance than traditional optical flow methods, and the CNN model performed best compared with the other 2 ML models. Finally, the authors conclude that their way of dealing with missing data on the experiments does not have a large negative impact on prediction performance.

6. Challenging Aspects

In this section, we present some of the challenges we are facing in the current project, with emphasis on our geographic region of interest, the Rio de Janeiro municipality. For some of them, we also present the lines of investigation we plan to follow.

- *Missing data.* An important line of research in this project has to do with an investigation of the procedure for handling missing data. As an example, we have the case of simulation data from meteorological models. In this case, not all models present data for the entire time series. It would be interesting to investigate how incomplete data for certain criteria can be used to improve training done with data from periods where there is complete observation.
- *Relief influence.* One challenging aspect that was raised by the specialists is that forecast of meteorological variables in the city of Rio de Janeiro is directly influenced by the geography of the location. Hence, we plan to investigate how information about the municipality's relief can be used to increase the predictive power of forecasting models.
- *Data sparsity.* Another complicating factor found concerns the sparseness of observed data related to a combination. We actually empirically observed the result of this data property in our preliminary experiments with Deep Learning models (see Section 4.1). A possible

solution to this problem involves an investigation of data subsampling and augmentation techniques. Another envisioned solution is the training of prediction models using Physics Informed Machine Learning [17]. This approach allows the training of models to take into account restricted the evolution of the underlying physical phenomena, as a way of compensating for the sparseness of the observed data.

- *Spatially non-uniformity of observations.* Another interesting aspect worthy of scientific investigation concerns the fact that observations from various data sources planned to be used in model training are collected in regions of space that do not form a regular grid. For example, the 33 meteorological stations under AlertaRio's responsibility are distributed in such a way that most of them are located in the eastern region of the municipality (see Fig. 1). Another example concerns observations from metoceanographic buoys. This is a complicating factor when considering the use of classical convolutional neural network architectures, which assume that training data is represented as regular lattices. Possible solutions are the use of neural network architectures for graphs, such as Graph Neural Networks [21].
- *Adding other data sources.* During the first phase of the project, the team identified several data sources potentially relevant for tuning the prediction models (see Tab. 2). It is natural to think that there must be other sources not yet considered that might be equally relevant. We have already mentioned the issue of relief before. Other data sources that are on the team's investigation horizon are related to aerosols, COR cameras located in key positions of the city, and radar images all potentially useful at the *nowcasting* prediction level.
- *Consideration of the dynamics of events.* At this point of investigation, we are interested in evaluating how we can capture the fine dynamics of events in space-time. We consider that this real-time dynamic must interact with the predictions, correcting them. As an example, we have the effect of side winds influencing the occurrence of convective rain and impacting its location and timing. Other elements are the so-called meteorological phenomena, such as Trough[17], and how to consider them in the learning process.

Conclusion

Extreme weather events have become stronger and more frequent worldwide. The concentration of the population in large urban areas places the occurrence of extreme weather events, such as strong rain, as a risk for the population. Governments have been investing in apparatus to monitor weather sensors and in organizations responsible for alerting the population and taking measures ahead of the extreme event. In this context, this paper describes an initiative we have been developing with the Rio de Janeiro municipality in building ML models to forecast extreme rainfall events a few hours ahead. Our initial investigations show that building ML models to predict extreme events is a very challenging problem, due to their scarcity of occurrences and fast dynamics.

As for the computational experiments, we presented initial results in this paper. We plan to do a more thorough experimental comparison using stronger baselines in future work. However, our initial computational experiments have led us to believe that, to achieve more precise forecasts, we will need to combine multiple data modalities. Hence, we plan to enrich the data we learn from with other available weather sensors we allude to in Section 2. We also envision the need of need fundamental ML research to cope with the challenges aspects of the problem we pointed out in Section 6.

[17] An elongated geographic region presenting relatively low atmospheric pressure.

Acknowledgements

The authors are funded by CNPq productivity research fellowship. The authors thank the Centro de Operações Rio, GeoRio, and INEA for providing us meteorological expertise and the several data sources we investigate in our project. The authors also thank professor Pedro Dias and professor Demerval Moreira, both from IAG, for providing us with NWP simulation data used in our experiments. Finally, experiments were run in the Petrus cluster at DEXL/LNCC.

References

1. https://www.cosmo-model.org/, accessed: 2021-09-20
2. https://www.mmm.ucar.edu/weather-research-and-forecasting-model, accessed: 2021-09-20
3. Agrawal, S., Barrington, L., Bromberg, C., et al.: Machine learning for precipitation nowcasting from radar images. CoRR abs/1912.12132 (2019), http://arxiv.org/abs/1912.12132
4. Ayzel, G., Scheffer, T., Heistermann, M.: Rainnet v1.0: a convolutional neural network for radar-based precipitation nowcasting. Geoscientific Model Development 13(6), 2631–2644 (2020). https://doi.org/10.5194/gmd-13-2631-2020
5. Bendre, N., Marn, H.T., Najafirad, P.: Learning from few samples: A survey (2020). https://doi.org/10.48550/ARXIV.2007.15484
6. Bonnet, S.M., Evsukoff, A., Morales Rodriguez, C.A.: Precipitation Nowcasting with Weather Radar Images and Deep Learning in São Paulo, Brasil. Atmosphere 11(11) (2020). https://doi.org/10.3390/atmos11111157
7. Castro, R., Souto, Y.M., Ogasawara, E., Porto, F., Bezerra, E.: STConvS2S: Spatiotemporal Convolutional Sequence to Sequence Network for weather forecasting. Neurocomputing 426, 285–298 (2021). https://doi.org/10.1016/j.neucom.2020.09.060
8. Czibula, G., Mihai, A., Czibula, I.: RadRAR: A relational association rule mining approach for nowcasting based on predicting radar products values. Procedia Computer Science 176, 300–309 (2020). https://doi.org/10.1016/j.procs.2020.08.032
9. Czibula, G., Mihai, A., Mihuleţ, E., Teodorovici, D.: Using Self-Organizing Maps for Unsupervised Analysis of Radar Data for Nowcasting Purposes. Procedia Comput. Sci. 159(C), 48–57 (2019). https://doi.org/10.1016/j.procs.2019.09.159
10. daSilva, F.: Projeto pesquisa operacional (2019), internal Report, in PT
11. Ding, D., Zhang, M., Pan, X., et al.: Modeling extreme events in time series prediction. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1114–1122. KDD '19, ACM, New York, NY, USA (2019). https://doi.org/10.1145/3292500.3330896

12. Gates, B.: How to avoid a Climate Disaster: The Solutions We Have and the Breakthroughs We Need. Random House Large Print Publishing (2021)

13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation pp. 52–65 (1997)

14. Huffman, G., Bolvin, D., Braithwaite, D., et al.: NASA Global Precipitation Measurement (GPM) Integrated Multi-satellitE Retrievals for GPM (IMERG) v5.2. NASA (2014)

15. Imbalanced_Learn: The imbalanced-learn (2021), https://imbalanced-learn.org/stable/

16. Jaye, A., Bruyère, C.L., Done, J.M.: Understanding future changes in tropical cyclogenesis using Self-Organizing Maps. Weather and climate extremes 26, 100235 (2019)

17. Karniadakis, G.E., Kevrekidis, I.G., Lu, L., et al.: Physics-informed machine learning. Nature Reviews Physics 3(6), 422–440 (2021). https://doi.org/10.1038/s42254-021-00314-5

18. Kim, S., Kim, H., Lee, J., et al.: Deep-Hurricane-Tracker: Tracking and Forecasting Extreme Climate Events. In: 2019 IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 1761–1769 (2019). https://doi.org/10.1109/WACV.2019.00192

19. Kohonen, T.: The self-organizing map. Proceedings of the IEEE 78(9), 1464–1480 (1990). https://doi.org/10.1109/5.58325

20. Kohonen, T.: Essentials of the self-organizing map. Elsevier - Neural Networks 37, 52–65 (2013). https://doi.org/10.1016/j.neunet.2012.09.018

21. Liu, Z., Zhou, J.: Introduction to Graph Neural Networks. Morgan & Claypool (2020)

22. NCAR: NCEP Climate Forecast System Reanalysis (CFSR) 6-hourly Products, January 1979 to December 2010 (2010). https://doi.org/10.5065/D69K487

23. Pereira, R.: Strategies and techniques for deep learning on small data. Ph.D. thesis, National Laboratory of Scientific Computing (2020)

24. Pereira, R., Souto, Y., Chaves, A., et al.: DJEnsemble: A Cost-Based Selection and Allocation of a Disjoint Ensemble of Spatio-Temporal Models, pp. 226–231. ACM, New York, NY, USA (2021). https://doi.org/10.1145/3468791.3468806

25. Ravuri, S., Lenc, K., Willson, M., et al.: Skilful precipitation nowcasting using deep generative models of radar. Nature 597(7878), 672–677 (2021). https://doi.org/10.1038/s41586-021-03854-z

26. Rousseeuw, P.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics 20, 53–65 (1987), https://wis.kuleuven.be/stat/robust/papers/publications-1987/rousseeuw-silhouettes-jcam-sciencedirectopenarchiv.pdf

27. Sakoe, Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing 26(1), 43–46 (1978)

28. Scikit_Learn: Scikit-learn: Machine Learning in Python (2011), https://scikit-learn.org/stable/index.html

29. Shalev-Shwartz, S., Ben-David, S.: Understanding Machine Learning: From Theory to Algorithms. Cambridge University Press (2017)

30. Shi, X., Chen, Z., Wang, H., et al.: Convolutional lstm network: A machine learning approach for precipitation nowcasting. In: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1. pp. 802–810. NIPS'15, MIT Press, Cambridge, MA, USA (2015)

31. Skiena, S.S.: The Data Science Design Manual, vol. 1. Springer, New York (2017)

32. Sønderby, C.K., Espeholt, L., Heek, J., et al.: MetNet: A Neural Weather Model for Precipitation Forecasting. CoRR abs/2003.12140 (2020), https://arxiv.org/abs/2003.12140

33. Song, L., Schicker, I., Papazek, P., et al.: Machine Learning Approach to Summer Precipitation Nowcasting over the Eastern Alps. Meteorologische Zeitschrift 29(4), 289–305 (2020). https://doi.org/10.1127/metz/2019/0977

34. Souto, Y.M., Porto, F., Moura, A.M., Bezerra, E.: A Spatiotemporal Ensemble Approach to Rainfall Forecasting. In: Proceedings of the International Joint Conference on Neural Networks. pp. 574–581 (2018). https://doi.org/10.1109/IJCNN.2018.8489693

35. Wang, Y., Coning, E., Harou, A., et al.: Guidelines for Nowcasting Techniques (2017)

36. Wolpert, D., Macready, W.: No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation 1(1), 67–82 (1997). https://doi.org/10.1109/4235.585893

37. Xiang, Y., Ma, J., Wu, X.: A precipitation nowcasting mechanism for real-world data based on machine learning. Mathematical Problems in Engineering 2020, 1–11 (2020). https://doi.org/10.1155/2020/8408931