

# Remote-Sensing Based Precipitation Detection using Conditional GAN and Recurrent Neural Networks

Pablo Negri<sup>1,2</sup>[0000-0003-0250-5208], Alejo Silvarrey<sup>3</sup>[0000-0002-8719-9029], Sergio Gonzalez<sup>4,5,7</sup>[0009-0009-1531-2182], Juan Ruiz<sup>4,5,6</sup>[0000-0002-5079-641X], and Luciano Vidal<sup>7</sup>[0000-0001-6715-2648]

- <sup>1</sup> Instituto de Investigación en Ciencias de la Computación (ICC) UBA-CONICET, Buenos Aires, Argentine  
<sup>2</sup> Departamento de Computación, FCEyN, UBA, Buenos Aires, Argentine  
<sup>3</sup> Universidad Católica del Uruguay, Punta del Este, Uruguay  
<sup>4</sup> Departamento de Ciencias de la Atmósfera y los Océanos, FCEyN, UBA, Buenos Aires, Argentine  
<sup>5</sup> Centro de Investigaciones del Mar y la Atmósfera (CIMA) UBA-CONICET, Buenos Aires, Argentine  
<sup>6</sup> Institut Franco-Argentin d'Études sur le Climat et ses Impacts (IRL IFAECI / CNRS-IRD-CONICET-UBA), Buenos Aires, Argentine  
<sup>7</sup> Servicio Meteorológico Nacional (SMN), Buenos Aires, Argentine

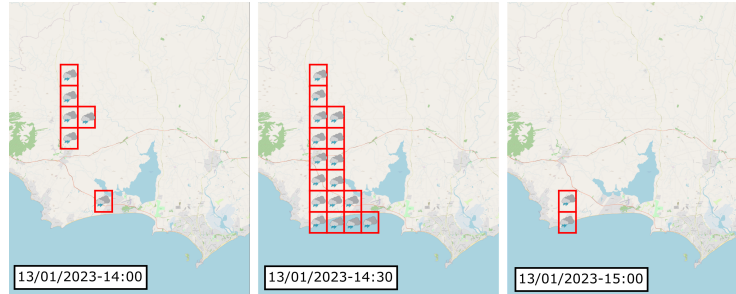
**Abstract.** Precipitation detection using infrared (IR) brightness temperature (BT) temporal flux data is a challenging problem. Other sensors, such as microwave (MW), have reliable and more robust predictive performance, but lack land coverage and temporal availability. IR-BT provides high-frequency data (from half an hour to 10 minutes) at very low resolution (4 km). However, automatic precipitation detection frameworks should face the simple nature of this variable on the one hand, and the very low number of rain events occurring in nature on the other hand. This paper addresses this challenge by proposing a conditional GAN framework using recurrent neural networks, which transforms the unbalanced problem into a small (short) pattern detection algorithm. Several tests allow the identification of robust architectures and useful loss functions that enable promising results, minimize false alarms, and improve the overlap of positive events.

**Keywords:** Remote-Sensing · Precipitation Detection · cGAN · Recurrent Neural Networks · Invert Dice Loss

## 1 Introduction

This study proposes the use of machine learning models to detect precipitation over land areas that are not covered by rain gauges. It aims to map a spatio-temporal distribution of precipitation near water resources to analyze the local climatic impact in the development of cyanobacterial harmful algal blooms

(CyanoHABs). The vulnerability of water resources to the effects of global warming is influenced by several factors, and some of them may produce CyanoHABs as increases in temperature, and changes in wind and rainfall patterns [19]. Both aspects increase the frequency of blooms and can lead to CyanoHABs due to enhanced stratification conditions and increased nutrient loading to water sources [3, 4]. Fig. 1 is an example of precipitation detection around Laguna Del Sauce,



**Fig. 1.** Precipitation detection around Laguna del Sauce, in Uruguay. Red boxes represent local places where the model detects precipitation using remote-sensing signals based on IR and BT.

in the state of Marldonado, Uruguay, where this type of bloom is habitual in the months of December and January. As the image shows, the rain was detected in some places around the lake, but not in the whole region. This local information at precise times could help to understand some variables that can trigger the bloom.

### 1.1 Detecting precipitation using remote sensors

In many regions of the world, precipitation variability across spatial and temporal scales cannot be fully resolved by conventional rain gauges. Geostationary satellites (GEO) have large spatial coverage and real-time availability with high spatio-temporal resolution in the order of 2-4 km and 10-30 minutes, respectively. These characteristics are highly relevant for detecting and monitoring the life cycle of precipitating clouds. For example, many previous studies have demonstrated the relationship between infrared (IR) brightness temperature (BT) and precipitation by providing information on cloud top properties [2, 10, 9, 17]. Other sensors, such as microwave (MW) data, which capture the internal structure of clouds, provide a better quantitative estimate of precipitation, but have very poor coverage and observation frequency.

Satellite quantitative precipitation estimate (SQPE) methodologies can be classified based on how they use precipitation dynamics information. Those using instantaneous information (or snapshots) will be referred to as static methods. Others, namely dynamic methods, integrate available information on a time window and explicitly consider the evolution of the precipitation system.

Static SQPE methods are characterized by a single or a set of satellite data corresponding to the same time, in which the objective is to integrate such information to obtain precipitation estimation. Among them, there are most of the products of the Precipitation Estimation from Remotely Sensed Information Using Artificial Neural Networks (PERSIANN) family [10].

More current versions, such as PERSIANN-CNN [22], implements an Deep Convolutional Neural Network (CNN), and use IR-BT and water vapor (WP) channels. The use of CNNs improves the extraction of 2d features such as shape, texture, and extent of precipitating clouds, aspects that ANNs cannot capture directly.

Dynamic methods use a sequence of satellite data with temporal order, which can be composed of one or more input variables. Among these methods, we can name those based on optical flow, such as the Climate Prediction Center morphing method (CMORPH) [13], Global Satellite Mapping of Precipitation (GSMaP) [14], and IMERGE [11], which are characterized by propagating the precipitation field estimated with MW sensors through the IR data. [25] uses conditional GANs to integrate estimations based on IR and MW over a time window. Recurrent neural networks (RNNs) are particularly suited for extracting information and predicting patterns in sequential data such as satellite observations. In [1] they propose a PERSIAN architecture and Long Short Time Memory (LSTM), a widely employed RNN, for estimating accumulated precipitation from a sequence of images over a time window.

## 1.2 Adversarial learning on sequential time-series

This paper combines an adversarial learning framework and recurrent network architectures inspired by the work of Rezaei et al. [21, 20]. Their approach, called RNN-GAN, addresses the semantic segmentation of medical images that face a highly imbalanced data problem. The tumor segmentation system applies a Conditional Generative Adversarial Network (cGAN) [15] architecture to two cardiac magnetic resonance (MR) images and one abdominal computed tomography (CT) dataset. The input stream consists of a 2D sequence of images and slices where the tumor and lesions are manually segmented at the pixel level. In such cases, the number of pixels representing a tumor is significantly lower than the number of pixels representing the background. The effect of unbalanced pixel distribution is mitigated by using complementary mask classes, some of which may overlap.

In this work, the precipitation detection system evaluates a remote sensing based IR-BT input signal of length  $L$  to diagnose a rainfall event. This signal is sampled at a frequency of 30 minutes, and the detection output corresponds to a binary stream of length  $L$ , where 1 is the value if it is raining and 0 otherwise. Since precipitation events do not usually last more than a few hours or less, the binary target is mostly populated with zeros even for a rainy day (see Fig. 1).

A traditional prediction model that forecasts rainfall at time  $t + 1$  from a signal going from 0 to  $t$  have the problem of positive collection of samples, and the dataset can finally be highly unbalanced. On the other hand, facing

the problem in a similar way as Rezaei et al., the proposed system trains a conditional Generator following a sequence-2-sequence approach that estimates the most probably rainfall output from a IR-BT input signal. Working with cGAN on target segments with at least one precipitation event within the stream, changes the learning approach, correctly detecting rainfall and minimizing false alarms.

The objective of this paper involves precipitation detection using satellite IR-BT information streaming data using a cGAN framework and recurrent neural networks. It proposes:

- A study of Generator and Discriminator architectures to tackle this particular problem,
- Evaluate different losses to improve regularization and convergence,
- Analyze the neighbor influence to detect precipitation.

Next section introduces materials and methods, section 3 present different experiments and results. Conclusions and perspective are proposed at section 4.

## 2 Materials and methods

The Global full-resolution infrared dataset is provided by the Climate Prediction Center (CPC)<sup>8</sup>. It merges IR-BT data between  $10\mu$  to  $11\mu$  from all available GEO satellites (GOES-8/9/10/11/12/13/14/15/16, METEOSAT-5/7/8/9/10, and GMS-5/MTSat-1R/2/Himawari-8) every 30 minutes with a spatial resolution of 4 km [12]. Spatial coverage is between 60S-60N and 180W-180E.

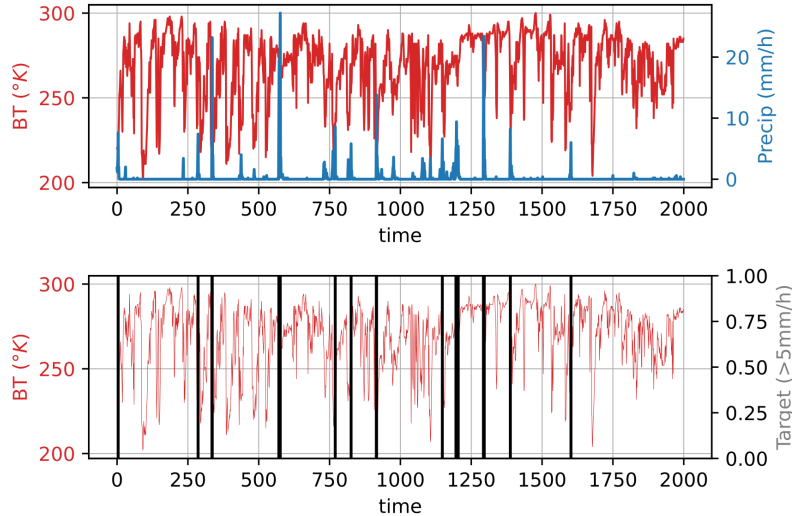
Brazil’s INMET<sup>9</sup> provides the precipitation event targets from more than 350 automated rain gauges deployed throughout their country. The dataset consists of hourly automated rain gauge measurements of accumulated precipitation. The location (latitude and longitude) of each rain gauge station is used to retrieve the corresponding IR-BT data from GEO satellites. Precipitation values are interpolated to half-hourly measures to obtain the same length of IR-BT values.

The dataset comprises measurements collected during the years 2010 and 2011. The data set has been divided into four semesters, with the initial semester of 2010 and the subsequent two semesters of 2011 designated for training and validation purposes, while the second semester of 2010 has been reserved for testing. The IR-BT and observed rainfall sequence for the years 2010 and 2011 are 8,832 and 17,520, respectively. The number of valid rain gauges for each year is 376 for 2010 and 368 for 2011.

Fig. 2 shows an example of the rain gauge accumulated precipitation in mm/h as the blue line and the IR-BT data in Kelvins degrees ( $K$ ) at the rain gauge site as the red line. The figure below shows a sequence binary target consisting of the instantaneous points where the precipitation was greater than 5 mm/h. It can be seen in the figure 2 that there is a correlation between precipitation

<sup>8</sup> <https://search.earthdata.nasa.gov/>

<sup>9</sup> <https://portal.inmet.gov.br/>



**Fig. 2.** Examples of IR-BT at the location of the rain gauges and the accumulated precipitation. The red line shows the IR-BT sensor data, and the blue line shows the accumulated rainfall. In the bottom figure, we produce a binary target sequence of those instants where precipitation was greater than 5 mm/h.

and low temperature values, but this is not a rule that can be solved with a simple threshold. At the same time, it is clear that we face a problem where precipitation events are rare in the input sequence. In fact, on average, 7.6% of the points in the 2011 dataset have precipitation greater than 0 mm/h, and only 0.6% of the points have precipitation greater than 5 mm/h. In the following, we fix  $T_{pr} = 5\text{mm/h}$  as the threshold for precipitation events, because this amount of precipitation can be considered significant for the generation of CyanoHABs.

## 2.1 Recurrent Neural Networks and Conditional GAN framework

Recurrent Neural Networks (RNN) are a potentially accurate prediction model for precipitation detection. RNNs are neural networks that compute current variables based on their prior states, giving them a "dynamic memory" [6]. This is extremely useful for prediction within a time series, where each element fed into the model is related to the previous and next values.

Temporal series denoted as  $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(T)})$  are usually the inputs of RNN models. Similarly, the target sequences corresponding to precipitation detection are given as a binary sequence  $(\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \dots, \mathbf{y}^{(T)})$ , where  $\mathbf{y}^{(t)} = 1$  if the accumulated precipitation is greater than  $T_{pr}$ , and  $\mathbf{y}^{(t)} = 0$  otherwise (see Fig. 2). The predictions produced by the recurrent model are denoted as  $\hat{\mathbf{y}}^{(t)}$ .

We express the estimation of target  $\mathbf{y}$  at time  $t$  as a dependent function  $R$  with internal parameters  $\theta_{RNN}$ :

$$\hat{\mathbf{y}}^{(t)}, \theta_{RNN}^{(t)} = R(\mathbf{x}^{(t)} | \theta_{RNN} = \theta_{RNN}^{(t-1)}) \quad (1)$$

Modern RNN architectures introduce several improvements to overcome traditional training problems. The Long-Short Term Memory model [8] (LSTM) is one of the most successful networks, widely used in several applications such as natural language processing. We will use LSTM in the architecture of our precipitation diagnosis framework.

Generative adversarial networks consist of a generator  $G$  and a discriminator  $D$  model, which are both trained simultaneously according to the two-player min-max game with value function  $V(G, D)$ :

$$\min_G \max_D V(G, D) = \mathbf{E}_{\mathbf{y} \sim p_{data}(\mathbf{y})} [\log(D(\mathbf{y}))] + \mathbf{E}_{\mathbf{z} \sim p_z(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))] \quad (2)$$

$D$  estimates whether the data is real or generated by  $G$  (fake). On the other hand,  $G$  learns how to fool  $D$  improving the quality of the output. Formally,  $D(\mathbf{y}; \theta_d)$  outputs the probability that  $\mathbf{y}$  comes from the training data instead of  $p_y$ , a generator distribution over the data  $\mathbf{y}$  that  $G$  maps from a prior noise distribution  $p_y(\mathbf{z})$  to the data space as  $G(\mathbf{y}, \theta_g)$ . The framework adjusts the parameters of  $D$  to minimize  $\log(D(\mathbf{y}))$ , the first part of eq. 2, and simultaneously adjusts the parameters of  $G$  to minimize  $\log(1 - D(G(\mathbf{z})))$ .

GANs are extended to a conditional model when  $G$  and  $D$  are conditioned with additional data  $\mathbf{x}$ . For  $G$ , the input noise  $p_y(\mathbf{z})$  and  $\mathbf{x}$  are combined in a joint hidden representation, and for  $D$ ,  $\mathbf{y}$  and  $\mathbf{x}$  are presented as inputs to the discriminator. The objective functions  $\mathcal{L}_{adv}$  would be [15]:

$$\min_G \max_D V(G, D) = \mathbf{E}_{\mathbf{y} \sim p_{data}(\mathbf{y})} [\log(D(\mathbf{y}|\mathbf{x}))] + \mathbf{E}_{\mathbf{z} \sim p_z(\mathbf{z})} [\log(1 - D(G(\mathbf{z}|\mathbf{x})))] \quad (3)$$

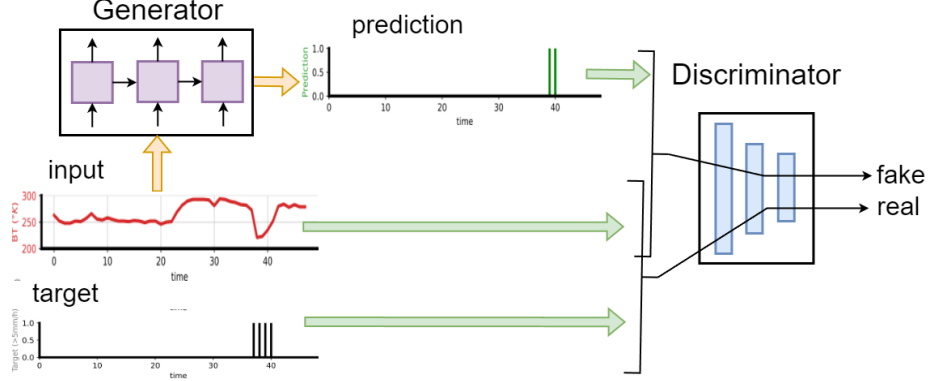
In our framework, the  $\mathbf{y}$  distribution is the binary stream of the precipitation diagnosis, and the  $\mathbf{x}$  data is the normalized IR BT in Kelvin degrees. To normalize  $\mathbf{x}$ , we subtract the overall mean and divide it by the standard deviation of the IR BT values. The framework is trained to approximate  $G(\mathbf{z}|\mathbf{x}) = \hat{\mathbf{y}}$  to  $\mathbf{y}$ .

Fig. 3 shows the cGAN framework. The generator  $G$  follows a sequence-2-sequence framework, arranged in a deep network composed of LSTM cells, followed by fully connected layers with a single sigmoid output. The discriminator  $D$  uses a one-dimensional convolutional neural network architecture with a single sigmoid output.

## 2.2 Loss functions

In this work, we evaluate the following loss functions as regularizers to train  $G$ :

$$\mathcal{L}_{L1} = \|\mathbf{y}_{target} - \hat{\mathbf{y}}\| \quad (4)$$



**Fig. 3.** Proposed framework for precipitation detection cGAN.

$$\mathcal{L}_{Ldice} = 1. - \frac{2 \cdot \sum_{t=1}^L (\mathbf{y}_{target}^{(t)} \cdot \hat{\mathbf{y}}^{(t)}) + \epsilon}{\sum_{t=1}^L (\mathbf{y}_{target}^{(t)} + \hat{\mathbf{y}}^{(t)}) + \epsilon} \quad (5)$$

$$\mathcal{L}_{Linodice} = \frac{\sum_{t=1}^L (\mathbf{y}_{target}^{(t)} + \hat{\mathbf{y}}^{(t)}) + \epsilon}{2 \cdot \sum_{t=1}^L (\mathbf{y}_{target}^{(t)} \cdot \hat{\mathbf{y}}^{(t)}) + \epsilon} - 1. \quad (6)$$

were  $\epsilon$  is a low value number to avoid division by zero.

**Table 1.** Table evaluating Dice loss functions for a target sample and  $\epsilon = 0.1$ .

	target (0,0,1,1,0)	description	L1 loss	Dice Loss	Inverted Dice Loss
predictions	(0,0,0,0,0)	2 fn	2	0.95	20.0
	(0,0,0,1,0)	1 fn	1	0.32	0.47
	(1,0,1,1,0)	1 fp	1	0.19	0.24
	(1,1,1,1,0)	2 fp	2	0.32	0.48
	(1,1,0,0,1)	all wrong	5	0.98	50.0

$\mathcal{L}_{L1}$  refers to the  $\ell_1$  distance and measures the deviation of  $G(z|y) = \hat{\mathbf{y}}$  from a binary output. This loss not only highlights prediction errors, but also forces  $G$  to produce binary values close to 0 and 1, which improves the quality of the prediction. This step is very important because this output is later evaluated by the discriminator  $D$ .

The equations 5 and 6 describe the dice function loss and the inverse dice loss, respectively. A dice score is usually used to measure the overlap between distributions. Table 1 compares the results of the three loss functions on a target

sample similar to the goal and predictions of our framework. As can be seen, Inverse Dice (iDice) generally produces larger loss values if the prediction  $\hat{y}$  does not trigger a rainfall event, which is considered a false negative (fn). It is expected that the generator  $G$  learns to trigger a rainfall event at a very precise position to avoid this negative reward. Empirical results showed that only using  $\mathcal{L}_{L1}$  cGAN train a generator  $G$  that does not trigger rainfall events. This is because the loss of false negatives for this score is not important. Dice and inverse Dice losses, on the other hand, forces the generator to produce a distribution similar to the binary target.

The final objective function  $\mathcal{L}_{pd-gan}$  of the precipitation detection generator is:  $\mathcal{L}_{pd-gan} = \mathcal{L}_{adv} + \mathcal{L}_{L1} + \mathcal{L}_{xdice}$ .

### 3 Experiments and Results

#### 3.1 Evaluation scores

The evaluation of the precipitation detection task is similar to the tumor segmentation in [20]. The idea is to obtain a measure of the overlap between the ground truth and the model prediction distributions. We then define:  $tp$ , true positives, which is the number of points correctly classified as precipitating;  $fn$ , false negatives, the number of points incorrectly classified as non-precipitating; and  $fp$ , false positives, those points classified as precipitating but it is not.

We propose the following scores:

$$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \quad (7)$$

$$IoU = \frac{intersection}{union} \quad (8)$$

where  $precision = \frac{tp}{(tp+fp)}$ ,  $recall = \frac{tp}{(tp+fn)}$ ,  $intersection = tp$ , and  $union = tp + fn + fp$ . The **F1** score (Eq. 7) measures the exact overlap between the target distribution and the prediction distribution. The **IoU** (Intersection over Union) score (eq. 8) calculates the ratio of true rainfall events detected to the total number of events triggered and the times the system should detect rainfall. **F1** and **IoU** range from 0 to 1, with 1 being optimal. Since precipitation events are usually very short, the system sometimes predicts precipitation at time  $t \pm 1$  instead of time  $t$ . For these cases, we also define a  $F1 - ext$  score, which is calculated as the correct detection of predictions at times  $t \pm 1$  of the correct positive prediction.

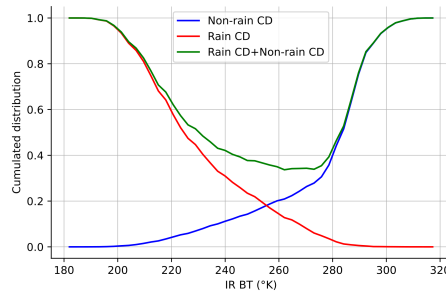
#### 3.2 BT-PR-dataset

To train and validate the PD-GAN framework, we set tuples  $(bt_k, pr_k)_{k=1, \dots, N}$ . Each  $bt_k^{(t)}$  is a sequence of IR BT, and  $pr_k^{(t)}$  is the corresponding sequence of precipitation values at the same location, both of length  $L$ :  $t = 1, \dots, L$ . We fix the length of the sequences to  $L = 48$ .



This set is populated by **positive** and **negative** tuples. The **positive** ones have at least one position  $pr_k^{(t)}$  equal to 1. This means that the precipitation accumulated by the rain gauge was greater than  $T_{pr} = 5$  mm/h. **negative** tuples have all values of  $pr_k^{(t)}$  equal to 0, which means that the accumulated precipitation was less than  $T_{pr}$  or did not rain at all. We will refer to this dataset as **BT-PR-dataset** and will use it later to train the PD-GAN framework. For  $T_{pr} = 5$  mm/h, **BT-PR-dataset** collects 64,755 tuples, with 27,969 **positive** tuples and 36,786 **negative** tuples.

### 3.3 Naive detection



**Fig. 4.** Cumulative Distribution (CD) of BT for raining and non-raining events. The green curve defines two thresholds:  $T_{naive-cross} = 256K$  and  $T_{naive-otsu} = 262K$ .

Naive precipitation detection classifies pixels as raining or non-raining by applying a fixed threshold to the IR BT. From **BT-PR-dataset** we obtain the distribution of those IR BT values  $bt_k^{(t)}$  that correspond to effective precipitation  $pr_k^{(t)} = 1$ . We also compute the distribution of IR BT values without precipitation. Both distributions are then plotted on cumulative histograms, as shown in Fig. 4, to obtain two IR BT thresholds. The first threshold  $T_{naive-cross} = 256K$  indicates the point where the two curves coincide. The other threshold  $T_{naive-otsu} = 262K$  follows the idea of the Otsu methodology [18].

Table 2 presents the result of applying this threshold to the whole 2010 dataset.

### 3.4 Precipitation Detection GAN

This section develops an ablation study on the different hyperparameters of the system and different architectures. We use a 5-fold cross-validation evaluation. The reported results are computed as the average within the folds of the performance scores. The models were trained for up to 50 epochs with a batch size of 64. We choose the Adam optimizer with a learning rate of  $2e - 4$ . The length  $L$

of the sequence of training tuples in **BT-PR-training** is set to  $L = 48$ , which means a 1-day history.

As explained in sec. 2.1, the architecture of the discriminator  $D$  is fixed. It consists of two branches of 1d CNN networks:

- $\text{br1}(bt_k) = [\text{1dConv}(L, 32, 5), \text{MaxPool}(9), \text{1dConv}(32, 64, 5), \text{BN}(), \text{MaxPool}(1), \text{BN}(), \text{FC}(64, 32)]$
- $\text{br2}(pr_k) = [\text{1dConv}(L, 32, 3), \text{MaxPool}(9), \text{1dConv}(32, 32, 3), \text{BN}(), \text{MaxPool}(1), \text{BN}(), \text{FC}(32, 32)]$

where  $\text{1dConv}(A,B,K)$  is a 1d convolutional layer with input  $A$ , output  $B$ , and 1d kernel length  $K$ ,  $\text{MaxPool}(C)$  is a max-pooling computed to length  $C$ ,  $\text{BN}()$  is a batch normalization layer, and  $\text{FC}(D,E)$  is a fully connected layer with input size  $D$  and output size  $E$ . The input of  $\text{br1}$  is the BT sequence, and the input of  $\text{br2}$  is the precipitation binary target. Then the outputs of both branches  $\text{br1}$  and  $\text{br2}$  are concatenated and feed the output branch:  $\text{ob} = [\text{FC1}(64, 32), \text{FC}(32,1)]$ .  $\text{ob}$  has a binary output that indicates whether the input is real or fake.

Table 2 shows the results using **BT-PR-dataset**, where each  $bt_k$  is the stream of IR BT at the location of the rain gauge that gives the target  $pr_k$ . In this test,  $bt_k$  is a one-dimensional variable of normalized IR BT values that feeds the training and validation framework.

**Table 2.** Results of naive thresholding and different PD-GAN architectures. Best results are highlighted in bold.

Naive threshold							
	F1	IoU	precision	recall	F1-ext	FAs	
$T_{naive-cross} = 256 K$	0.0750	0.0394	0.0399	0.7551			704.05
$T_{naive-otsu} = 262 K$	0.0631	0.0328	0.0331	<b>0.8006</b>			899.4
Hidden Units   dLoss   F1   IoU   precision   recall   F1-ext   FAs							
8	d	0.0088	0.0044	0.0044	0.7716	0.0188	5184.6
16		0.0362	0.0308	0.0409	0.0709	0.0667	126.6
32		0.0074	0.0037	0.0037	0.7739	0.0158	6405.9
64		0.0341	0.0238	0.0224	0.0899	0.0690	118.2
8	id	0.1299	0.0731	0.0959	0.2642	0.2213	<b>79.1</b>
16		0.1344	0.0755	0.0901	0.3434	0.2323	112.3
32		0.1334	0.0749	0.0902	0.3292	0.2300	105.5
64		<b>0.1423</b>	<b>0.0806</b>	<b>0.0976</b>	0.3405	<b>0.2455</b>	103.4

Table 2 compares different numbers of hidden units in the LSTM cells and the use of Dice Loss and Inverted Dice Loss. As can be seen, the use of Dice Loss does not provide the framework for correct learning of precipitation detection. High values of FAs and low values of F1 show that some of the K-fold models fall into non-optimal working states. In the former case, the generator predicts ones most of the time, or in the latter case, the generator misses precipitation events and often predicts zeros.

The inverted dice loss, on the other hand, shows robust learning for the precipitation detection framework. The best results are obtained with 64 hidden units in the LSTM cell. However, using 8 hidden units produces a minimum number of FAs.

F1-ext is another interesting score that shows a large increment compared to F1. In general, using the extended target, the number of TPs doubles, while the number of FPs decreases in the same proportion. This clearly shows that the detection framework is sensitive to the temporal evolution of the IR BT associated with the precipitation, but triggers the signal once after or before the actual precipitation time.

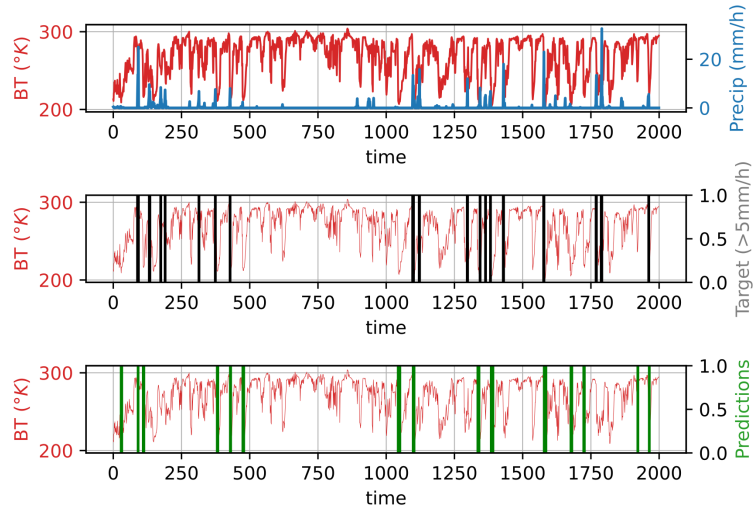
**Table 3.** Results of stacked LSTM architectures using inverse dice loss.

Nr. Layers	Hidden Units	F1	IoU	precision	recall	F1-ext	FAs
2Lay	8	<b>0.1395</b>	<b>0.0787</b>	0.0924	0.3646	<b>0.2427</b>	116.4
	16	0.1320	0.0740	0.0847	<b>0.3789</b>	0.2309	131.8
	32	0.1189	0.0658	0.0748	0.3695	0.2087	144.0
	64	0.1151	0.0635	0.0716	0.3705	0.2025	150.8
3Lay	8	0.1388	0.0782	<b>0.0927</b>	0.3577	0.2407	114.1
	16	0.1373	0.0773	0.0773	0.3749	0.2387	123.4
	32	0.1309	0.0737	0.0922	0.3252	0.2268	<b>113.5</b>
	64	0.1193	0.0662	0.0769	0.3550	0.2093	138.2

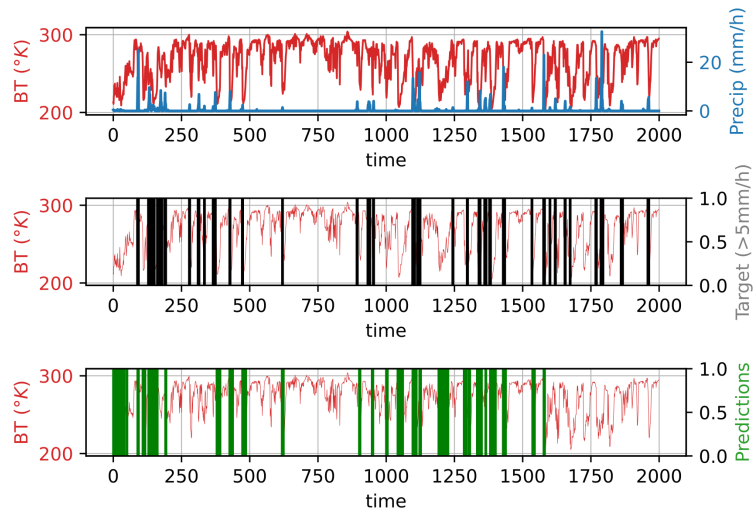
We also use a stacked LSTM network [26] for the generator. In practice, an easy way to increase the depth of the recurrent network is to stack the cells into  $D$  layers, trying to better capture the dynamics of the time-dependent signal. This architecture has been shown to improve efficiency and performance in problems such as vehicle-to-vehicle communication [5], regional commodity price estimation [16], and French-English translation [23]. However, the results in table 3 do not show a significant improvement in precipitation detection.

Next experiments increase the context of the information around the position of interest. Instead of only using the IR-BT value at the rain gauge location, we extract a grid from the EarthData around that point, thus the samples  $bt_k^{(t)}$  became an  $R \times R$  grid centered on that rain gauge location. The horizontal resolution of these grids is 4 km, which means that a  $3 \times 3$  grid covers a field of  $12 \text{ km}^2$ . Thus, the framework now has the neighborhood information that can be used to better estimate a precipitation event, but, at the same time, we have increased the number of parameters in the network.

Table 4 shows the results using a 1-layer architecture and two grid sizes:  $3 \times 3$  and  $5 \times 5$ . As expected, additional information about IR-BT at neighboring locations allows us to improve the results in both F1 and FAs for 8 hidden units architecture.



**Fig. 5.** Precipitation detection on the testing dataset 2010, 8 hidden units architecture.



**Fig. 6.** Precipitation detection 2010 dataset using a  $T_{pr}=1$  mm/h, 16 hidden units architecture.

Table 4 shows the results of the framework using the bidirectional LSTM [7] in the generator, which is also used in [20]. However, the performance of this modified LSTM cell drops in F1 and FAs scores.

The final test involves a different choice for the precipitation threshold. Using a lower value of 1 mm/h, the new binary target represents a higher number of

**Table 4.** Results using a grid region around pluviometer position.

Hidden Units	R	F1	IoU	precision	recall	F1-ext	FAs
8	3	0.1443	0.0828	0.1338	0.2549	0.2433	63.1
16		0.1415	0.0800	0.0988	0.3360	0.2429	97.3
32		0.1211	0.0672	0.0775	0.3531	0.2113	125.1
64		0.1214	0.0673	0.0783	0.3474	0.2123	123.5
8	5	0.1258	0.0702	0.0842	0.3410	0.2180	118.2
16		0.1141	0.0630	0.0723	0.3441	0.2010	136.5
32		0.1123	0.0618	0.0709	0.3384	0.1976	133.1
64		0.1123	0.0618	0.0697	0.3647	0.1976	143.9
Bidirectional LSTM							
8	3	0.1141	0.0632	0.0786	0.2690	0.1964	91.5
16		0.1250	0.0695	0.0796	0.3770	0.2184	136.2
32		0.1166	0.0644	0.0751	0.3346	0.2023	123.2
64		0.1174	0.0650	0.0743	0.3791	0.2057	154.6
8	5	0.1202	0.0666	0.0809	0.2964	0.2088	99.1
16		0.1244	0.0695	0.0845	0.3263	0.2154	114.6
32		0.1203	0.0666	0.0752	0.3771	0.2106	139.2
64		0.1200	0.0666	0.0787	0.3221	0.2091	115.7

**Table 5.** Results using a precipitation threshold of 1 mm/h.

Naive threshold	F1	IoU	precision	recall	F1-ext	FAs
257 $K$	0.2080	0.1193	0.1278	0.6687		736.5
Hidden Units	F1	IoU	precision	recall	F1-ext	FAs
8	0.2036	0.1208	0.1509	0.3785	0.2862	255.4
16	0.2120	0.1267	0.1606	0.3821	0.2964	239.5
32	0.2102	0.1255	0.1548	0.3989	0.2954	264.1
64	0.2009	0.1188	0.1431	0.4200	0.2840	310.6

precipitation events. Table 5 shows the results of the cGAN framework using a naive BT threshold. The performance shows a better F1 score compared to a threshold of 5 mm/h, but more than twice as many FAs. This is the expected behavior. In the training loop, the generator learns to trigger many more precipitation events, even if the BT corresponds to warm values. In this way, the framework loses generalization power.

The figures 5 and 6 show the results of the predictions for the two best architectures with  $T_{pr} = 5$  and  $T_{pr} = 1$  respectively.

## 4 Conclusions

This paper develops a framework to tackle local detection of precipitation using IR-BT temporal sequence signal, a cGAN framework and recurrent neural networks to produce the output sequence. The generator  $G$  of the cGAN is particularly suitable for producing binary temporal sequence outputs where the target has small (short) positive events, while it is mostly populated by zeros, using the IR-BT as input signal.

An inverse Dice function allows correct learning in the adversarial game, improving the performance of the traditional  $L1$  norm applied to binary outputs. Considering the simple nature of the IR-BT data signal, the results are promising and the system in its current form is ready to be incorporated into the framework of CyanoHABs prediction and to help set social policies for water resource management.

Further research could be directed at optimizing the forecast diagnosis by regionalizing the rain gauges, using categorical precipitation with multiple thresholds, or other architectures for the generator and discriminator, such as transformers [24].

**Acknowledgements** This research was supported by The National Research and Innovation Agency of Uruguay (ANII), International Development Research Centre of Canada (IDRC) and National Scientific and Technical Research Council of Argentina (CONICET). Grant Number: ANII-IA\_2021\_1\_1010782. Sergio Gonzalez's scholarship is supported by CONICET and the National Meteorological Service of Argentina.

## References

1. Akbari Asanjan, A., et al.: Short-term precipitation forecast based on the persiann system and lstm recurrent neural networks. *Journal of Geophysical Research: Atmospheres* **123**(22), 12,543–12,563 (2018)
2. Arkin, P.A., Meisner, B.N.: The relationship between large-scale convective rainfall and cold cloud over the western hemisphere during 1982-84. *Monthly Weather Review* **115**(1), 51 – 74 (1987)
3. Barruffa, A.S., Sposito, V., Faggian, R.: Climate change and cyanobacteria harmful algae blooms: adaptation practices for developing countries. *Marine and Freshwater Research* **72**(12), 1722–1734 (2021)
4. Burford, M., et al.: Perspective: Advancing the research agenda for improving understanding of cyanobacteria in a future of global change. *Harmful Algae* **91**, 101601 (2020)
5. Du, X., Zhang, H., Van Nguyen, H., Han, Z.: Stacked lstm deep learning model for traffic prediction in vehicle-to-vehicle communication. In: 2017 IEEE 86th Vehicular Technology Conference (VTC-Fall). pp. 1–5. IEEE (2017)
6. Elman, J.L.: Finding structure in time. *Cognitive science* **14**(2), 179–211 (1990)
7. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks* **18**(5-6), 602–610 (2005)

8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural computation* **9**(8), 1735–1780 (1997)
9. Hong, Y., Hsu, K.L., Sorooshian, S., Gao, X.: Precipitation estimation from remotely sensed imagery using an artificial neural network cloud classification system. *Journal of Applied Meteorology* **43**(12), 1834 – 1853 (2004)
10. lin Hsu, K., Gao, X., Sorooshian, S., Gupta, H.V.: Precipitation estimation from remotely sensed information using artificial neural networks. *Journal of Applied Meteorology* **36**(9), 1176 – 1190 (1997)
11. Huffman, G.J., et al.: Integrated multi-satellite retrievals for the global precipitation measurement (gpm) mission (imerg). *Satellite Precipitation Measurement: Volume 1* pp. 343–353 (2020)
12. Janowiak, J.E., Joyce, R.J., Yarosh, Y.: A real-time global half-hourly pixel-resolution infrared dataset and its applications. *Bulletin of the American Meteorological Society* **82**(2), 205 – 218 (2001)
13. Joyce, R.J., et al.: Cmorph: A method that produces global precipitation estimates from passive microwave and infrared data at high spatial and temporal resolution. *Journal of Hydrometeorology* **5**(3), 487 – 503 (2004)
14. Kubota, T., et al.: Global satellite mapping of precipitation (gsmap) products in the gpm era. *Satellite Precipitation Measurement: Vol. 1* pp. 355–373 (2020)
15. Mirza, M., Osindero, S.: Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784* (2014)
16. Negri, P., Ramos, P., Breikopf, M.: Regional commodities price volatility assessment using self-driven recurrent networks. In: *25th Iberoamerican Congress on Pattern Recognition (CIARP)*, Portugal, May 10–13. pp. 361–370. Springer (2021)
17. Nguyen, P., et al.: Persiann dynamic infrared–rain rate model (pdir) for high-resolution, real-time satellite precipitation estimation. *Bulletin of the American Meteorological Society* **101**(3), E286 – E302 (2020)
18. Otsu, N.: A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics* **9**(1), 62–66 (1979)
19. Paerl, H.W., Hall, N.S., Calandrino, E.S.: Controlling harmful cyanobacterial blooms in a world experiencing anthropogenic and climatic-induced change. *Science of the total environment* **409**(10), 1739–1745 (2011)
20. Rezaei, M., Yang, H., Meinel, C.: Recurrent generative adversarial network for learning imbalanced medical image semantic segmentation. *Multimedia Tools and Applications* **79**(21-22), 15329–15348 (2020)
21. Rezaei, M., et al.: Conditional generative refinement adversarial networks for unbalanced medical image semantic segmentation. *arXiv:1810.03871* (2018)
22. Sadeghi, M., et al.: Persiann-cnn: Precipitation estimation from remotely sensed information using artificial neural networks–convolutional neural networks. *Journal of Hydrometeorology* **20**(12), 2273 – 2289 (2019)
23. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215* (2014)
24. Vaswani, A., et al.: Attention is all you need. *Advances in neural information processing systems* **30** (2017)
25. Wang, C., et al.: Precipgan: Merging microwave and infrared data for satellite precipitation estimation using generative adversarial network. *Geophysical Research Letters* **48**(5), e2020GL092032 (2021)
26. Yu, Y., et al.: A review of recurrent neural networks: LSTM cells and network architectures. *Neural computation* **31**(7), 1235–1270 (2019)