Deep learning models for generation of precipitation maps based on Numerical Weather Prediction

Adrian Rojas-Campos[1], Michael Langguth[2], Martin Wittenbrink[3], and Gordon Pipa[1]

[1]Institute of Cognitive Science, Osnabrück University, 49069 Osnabrück, Germany
[2]Juelich Supercomputing Centre, Forschungszentrum Juelich, 52425 Jülich, Germany
[3]Deutscher Wetterdienst, 63067 Offenbach, Germany

Correspondence: Adrian Rojas-Campos (rrojascampos@uos.de)

Abstract.

Numerical Weather Prediction models (NWP) are atmospheric simulations that imitate the dynamics of the atmosphere and provide high-quality forecasts. One of the most significant limitations of NWP is the elevated amount of computational resources required for its functioning, which limits the spatial and temporal resolution of the outputs. Traditional meteorological techniques to increase the resolution are based uniquely on information from a limited group of interest variables. In this study, we offer an alternative approach to the task where we generate precipitation maps based on the complete set of variables of the NWP to generate high-resolution and short-time precipitation predictions. To achieve this, five different deep learning models were trained and evaluated: a baseline, U-Net, two deconvolutional networks, and one conditional generative model (CGAN). A total of 20 independent random initializations were performed for each of the models. The predictions were evaluated using MAE and LEPS-based skill scores, ETS, CSI, and frequency bias after applying several thresholds. The models showed a significant improvement in predicting precipitation showing the benefits of including the complete information from the NWP. The algorithms increased the resolution of the predictions and corrected an over-forecast bias from the input information. However, some new models presented new types of bias: U-Net tended to mid-range precipitation events, and the deconvolutional models favored low rain events and generated some spatial smoothing. The CGAN offered the highest quality precipitation forecast generating realistic outputs and indicating possible future research paths.

1 Introduction

Precipitation prediction is a fundamental scientific and social problem. Accurate rain forecasts play a crucial role in sectors like agriculture, energy, transportation, and recreation and help to prevent human and material losses in extreme weather events such as floods or storms. Nevertheless, it remains an unsolved problem due to its high complexity, the large number of atmospheric variables involved, and the complex interactions between them. This complexity and the rarity of high precipitation events make it a challenging phenomenon to predict.

Meteorology has developed different models to provide weather estimations. Among the most successful methods are the Numerical Weather Predictions models (NWP), which consist of systems of equations that simulate the dynamics of the atmosphere and provide highly accurate weather forecasts over long periods (Kimura, 2002). The performance of these models has

25 presented a constant improvement over time, and they are the standard operational systems in many meteorological agencies all over the world (Bauer et al., 2015).

However, NWP models still preserve some limitations, the most important being the large number of computational resources needed to generate forecasts~~. This characteristic~~ , which limits their outputs' temporal and spatial resolutions~~(Serifi et al., 2021)~~ , shrinking the possibility of offering ~~high detailed forecast~~ highly detailed forecast (Serifi et al., 2021). This shortcoming has

30 given place to two scientific tasks in meteorology: providing short-time (between 5 min and 6 hours) forecasts (nowcasting) and generating high-resolution forecasts from low-resolution (downscaling).

Deep learning algorithms are starting to be integrated as part of the weather forecast workflow and help them to overcome its limitations (Schultz et al., 2021). In the case of the short-time forecast, Shi et al. (2015) developed a first convolutional long short-term memory network (LSTM) that outperformed state-of-the-art optical flow models. Two years later, Shi et al. (2017)

35 improved previous benchmarks by presenting a new trajectory ~~GRU~~ recurrent model for nowcasting. Following the same goal, Agrawal et al. (2019) used the U-Net architecture (Ronneberger et al., 2015) to nowcast categorical radar images, with results that outperformed traditional nowcasting methods. Later after some exploratory work (Ayzel et al., 2019), Ayzel et al. (2020) introduced the RainNet v1.0, a convolutional network based on the U-Net for radar-based precipitation nowcasting. The results of the RainNet significantly overcame the performance of previous models but generated rain maps with remarkable spatial

40 smoothing and difficulty in predicting high rain events. Considering these limitations, ~~Ravuri et al. (2021)~~ Ravuri presented a deep generative model based on generative adversarial networks (GANS) (Goodfellow et al., 2014), which improved the accuracy and realism of the outputs.

The second significant area of application of deep learning for the generation of weather maps is spatial downscaling, which consists of generating high-resolution forecasts from low resolution (known in the machine learning domain as a Super

45 Resolution task (Park et al., 2003)). Stengel et al. (2020) used generative models to downscale wind velocity and solar irradiance data from global climate models, showing the usability of deep learning in climatological data. In the case of precipitation Sha et al. (2020) developed a U-Net-based model to downscale daily precipitation forecasts from a low to a higher resolution, obtaining results that overcome the performance of the statistical downscaling methods. One year later, Serifi et al. (2021) used generative models to downscale temperature and precipitation maps from weather simulations and meteorological observations,

50 obtaining improved-quality high-resolution maps while avoiding blurred results typical of the deconvolution.

Despite recent advancements, nowcasting and downscaling precipitation preserve significant room for improvement. In our opinion, one of the main shortcomings of the used approaches is the limited amount of atmospheric information integrated into the generation of predictions. Most models are developed using data from previous observations of the same variable (wind speed, precipitation, temperature, etc.), omitting information from additional meteorological states involved in the physical

55 phenomena. In the case of spatial downscaling, a deep learning approach increases the resolution of the NWP models, but it does not correct their imperfections, generating high-resolution inaccurate images.

In this matter, we propose ~~an~~ a new and alternative solution: to use the complete low-resolution weather simulations (NWP) as an informed prior for training a system that produces high-resolution precipitation maps. This translates into using deep learning algorithms to directly map the low-resolution meteorological forecast with high-resolution radar observations, gener-

60 ating short-term (3h) and high-resolution (1.4 km) precipitation maps based on the non-linear combination of all variables of the numerical weather simulations (NWP), correcting forecast inaccuracies in the process.

~~We achieve this~~ Our research goal is to achieve this mapping by developing and testing different deep learning algorithms in their ability to generate precipitation maps while increasing the resolution of the output. We will start by describing the data we worked on: the COSMO-DE-EPS simulations as input and precipitation radar images as output. Later we present the

65 different algorithms we trained and tested, their implementation and optimization procedures, and the metrics used to evaluate the generated predictions. The results obtained by each model are examined using the current literature.

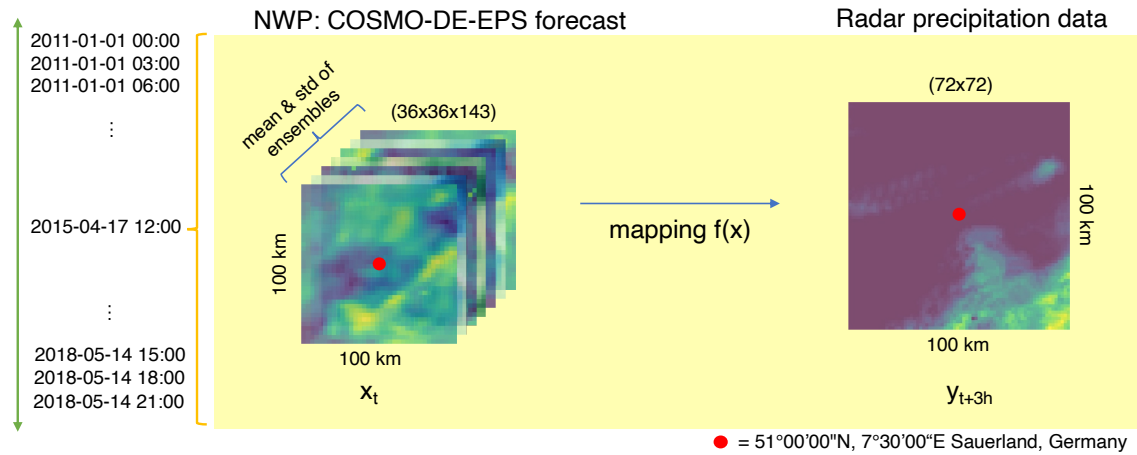



Figure 1. Task description: our research goal is to perform a direct mapping between the low-resolution atmospheric forecast and the high-resolution radar observations using the complete information of the NWP as input

2 Data

The complete data set was composed of all eight initializations of the COSMO-DE-EPS forecast with 3 hours lead time, and their target precipitation radar observations, over the period between the beginning of 2011 until the end of May 2018, in a

70 selected area of about $100 \times 100 \text{ km}^2$ in Western Germany region.

The ~~chosen region covers parts of the Sauerland low mountain range and the eastern part of the Rhine valley. Due to predominant western flow~~ , precipitation patterns are frequently subject to lee- and luv effects which lead to high spatial variability of~~ selected area is located in the central parts of Western Germany where interaction between low mountain ranges and the dominant flow patterns induces strong spatial variability in the yearly precipitation amount ~~in the target region (see Fig.

75 2). The~~ (see also, e.g., Kreklow et al., 2020). Due to prevailing westerly and south-westerly winds, the south-western part of the target area is characterized by low precipitation amounts between 500 and 700 mm/yr due to lee effects of the Eifel mountain range. ~~In~~ By contrast, yearly precipitation exceeds 1200 mm/yr between Wuppertal and Luedenscheid due to luv effects at the Sauerland mountain range. The high spatial variability in precipitation ~~,~~ together with a ~~suitable~~ good data coverage due to

overlapping observation ~~areas~~ area of different radar stations (see Fig. 1 in Pejcic et al., 2020) ~~,~~ makes the region suitable to try
80 and test our ~~algorithm~~deep neural networks.

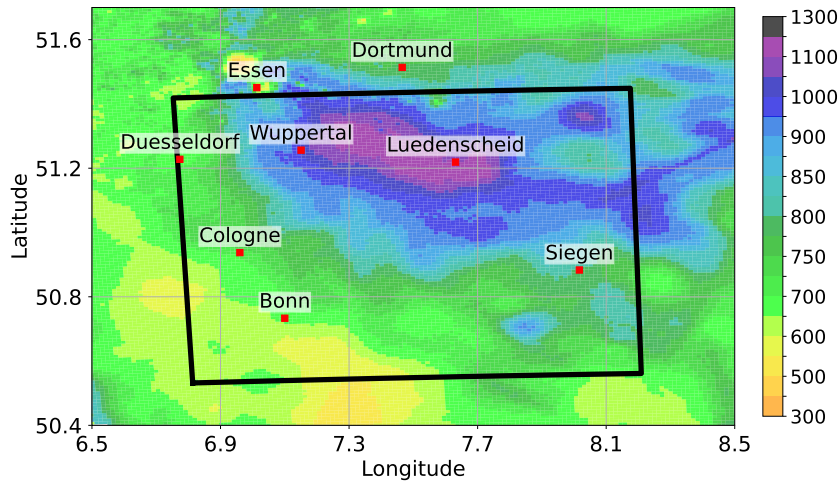



Figure 2. Average yearly precipitation (colours, mm/yr) from 2001-2021 ~~based on~~ compiled from the RW product (rain-gauge adjusted hourly precipitation) of the RADKLIM dataset (cf. Section 2.2). The lateral boundary target region is highlighted by the black box. Contour lines show the surface elevation retrieved from SRTM dataset (JPL)

2.1 NWP output

We use the output of the COSMO-DE-EPS forecast (Peralta et al., 2012) as input, which was the German Meteorological Service (DWD) operational ensemble predictions system until May 2018, which provided numerous data points for developing and testing deep learning applications. The COSMO-DE-EPS is the German adaptation of the COSMO model, a well documented
85 and reliable NWP model operating in several EU countries (Marsigli and Schumann, 2005; Valeria and Massimo, 2021).

We selected the forecast with 3 hours of lead time. ~~For each of the 143 forecast variables, we calculated the~~ We calculate the mean and the standard deviation (i.e., ensemble spread) from the 20 ensemble members~~.~~, for a total of 143 forecast. Each of the forecast variable provided information about meteorological states (wind speed, temperature, pressure, etc.) and soil and

surface states (water vapor on the surface, snow amount, etc.) providing a multidimentional description of the atmospherical state for the time predicted. A detailed description of the COSMO-DE-EPS output can be consulted in Schättler et al. (2019).

Given that each pixel of the COSMO-DE-EPS grid covered 7.8km^2 ($2.8 \times 2.8 \text{km}$), the correspondent 100 km^2 forecast around the coordinates (51°00'00", 7°30'00) has an extension of 36x36 pixels , for and we have 143 variables (channels), this give a final shape of 36x36x143 forecast for the area.

2.2 Radar precipitation observations

Our prediction target consist of constitutes quality-controlled precipitation observations of the German radar network provided by DWD. Here, we use version 2017.002 of the RADar KLIMatologie RADar KLIMatologie (engl.: Radar climatology) dataset (Winterrath et al., 2018). This dataset is based on the RADar OnLine AdjustemeNt (RADOLAN) procedure which converts the reflectivity measured by 17 ground-based C-Band radar stations to precipitation rates (Bartels et al., 2004). The procedure constitutes a synthesis between radar and rain gauge observations and includes several corrections to eliminate backscattered noise due to non-meteorological targets (e.g. insects or solid objects as wind power plants). Additional corrections are included for the RADKLIM dataset compared to the RADOLAN procedure (Winterrath et al., 2018).

The radar data is given on a polarstereographic grid which provides a quasi-equidistant grid coverage over Germany at a resolution of 1km. For consistency between the input and the target, the Climate Data Operators (CDO) software (Schulzweida, 2019) is used to remap the data onto the same rotated pole grid as the COSMO-DE-EPS COSMO data (see Section 2.1), but at a higher spatial resolution of 1.4km for downscaling purposes. The first-order conservative remapping method provided by the YAC's (Cet Another Coupler) interpolation stack (Hanke et al., 2016) ensures that the area-integrated area-intergrated precipitation amount is approximately conserved during the remapping process. Additionally, the YW product , which provides precipitation rates for every 5 minutes , is used to retrieve hourly precipitation. This is mandatory, since the original hourly RQ product is not valid at full hours but at minute 50 , which would result in which would in turn result into a mismatch with the time stamp of the NWP data. The resulting target data then comprises 72×72 pixels which are aligned with the coarser NWP input data.

2.3 Pre-processing

Once the forecasts were matched with their corresponding observation, all forecast-precipitation pairs containing missing values were discarded (3.93% of the complete set). Next, the original values of the forecast were standardized in order to avoid numerical problems due to inputs in different units of measurement.

The complete data set was split into train, test, and validation set by date. All forecasts with dates of 1st, 9th, 17th, and 25th, regardless of month and year, were selected as part of the test set (2671 forecasts and observations, 12.96% of the samples). In the same way, the validation set was built with the forecast of the 5th, 13th, 21st, and 28th of all months and years (2725 forecast and observations, 13.25% of the complete set). Finally, all remaining dates were chosen as part of the train set (15189 forecast and observations, 73.79% of the dataset). All datasets were stored and retrieved using TFRecords format from the TensorFlow library (Abadi et al., 2015).

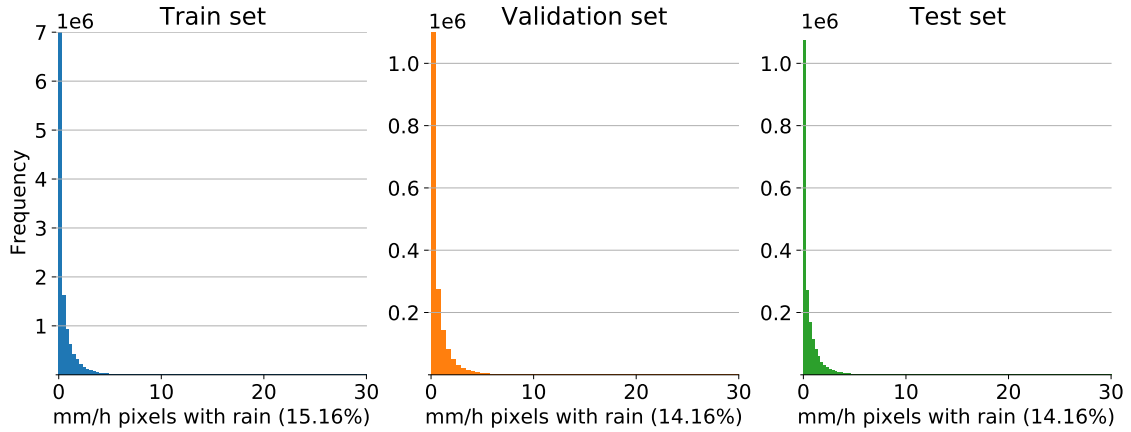
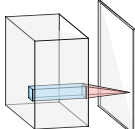
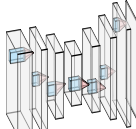
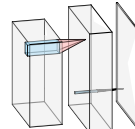
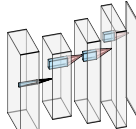
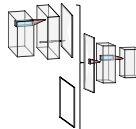



Figure 3. ~~Task description: our research goal is to perform a direct mapping between~~ Distribution of registered precipitations for pixels with precipitation > 0 mm/h in the ~~low-resolution atmospheric forecast and the high-resolution~~ radar observations~~using~~. In all datasets the ~~complete information~~ majority of the ~~NWP as input~~ pixels registered no rain. For the pixels with rain, we found the rain amounts characteristic gamma-shaped distribution.

3 Methodology

According to the recent literature, two main deep learning approaches can be considered helpful to solve this task: deconvolutional models and conditional generative adversarial models. Both of these approaches perform a non-linear mixing of the input variables to generate high resolution precipitation maps.

125 Deconvolutional models use deconvolutional layers to increase the resolution of the input. The deconvolution operation (also called transposed convolution or unsampled convolution) works inversely to convolution. Instead of creating an abstract representation of the input by reducing its spatial dimension, deconvolutions upsample the input to the desired feature map using learnable parameters by multiplying by a kernel (Dumoulin and Visin, 2018). Deconvolutional techniques have been widely

130 used in super-resolution (Long et al., 2015) and image segmentation tasks (Ronneberger et al., 2015). However, applications of deconvolutional networks to generate precipitation maps (Ayzel et al., 2020) have generated blurry outputs with high spatial smoothing, making the images look unrealistic and fail to predict high precipitation events.

On the other hand, generative adversarial networks (GANS) have been used to generate realistic precipitation maps that do not incur spatial smoothing or blurring (Ravuri et al., 2021). Generative Adversarial Networks (GANS) are a generative model

135 composed of two separate networks: a generator network that creates a fake image from a random input and a discriminator network, with the job of discriminating between the real and the fake images. While the generator is trained to try to deceive the discriminator, the discriminator is trained to separate the authentic images from the fake ones. Both networks are trained against each other, which forces the generator to create realistic outputs (Goodfellow et al., 2014).

GANS can also be used to solve image-to-image translation tasks by substituting the random input for a conditional input
140 (Isola et al., 2016) (now called conditional generative adversarial networks or CGANS). While the task of the discriminator remains unchanged, the generator is asked to not only fool the discriminator but also to be near the truth value, combining the loss with the distance between output and true values regulated by a hyperparameter Lambda. In this way, the generator produces an output that is not just accurate but also resembles the characteristics of a real image.

In this work, five different deep learning models were developed and tested: two baseline models for comparison with
145 previously published algorithms, two deconvolutional models, and ~~one CGANS approach. Given the random initialization of the kernel parameters, 20 independent initializations were conducted for each of the models to test. Each model was trained during 24 epochs using a mini-batch size of 20. Mean squared logarithmic error was used as loss measure due to the gamma-like distribution of the rain amount on the radar images, and Adam, with a learning rate of 0.001, was used as optimizer~~the CGAN. The development of the deconvolutions and the CGAN were guided by previously published literature about deep
150 learning applications for super-resolution tasks but adapted to our dataset characteristics and the result of hyperparameter experimentation. Validation of hyperparameters and parameters is performed by testing the model against unobserved data (test set). Each of the models is explained in detail next and summarized in Table 1.

Table 1. Summary of the deep learning models trained and tested in this study

Name	Baseline	U-Net	~~Deconv1L~~ Deconv1L4Rain v1	~~Deconv3L~~ Deconv3L4Rain v1	~~CGAN~~ CGAN4F
Layers	2dDeconv(1)	See	2dDeconv(32)	Max pooling(2)	~~Deconv3L~~ Deconv
		Appendix A	BatchNorm	2dDeconv(32)	discrimina
			Conv(1)	BatchNorm	
				2dDeconv(16)	
				Conv(1)	
Kernel size	7x7		7x7	5x5	5x5
Stride	2		2	2	2
Parameters	7,008	153,849	224,673	127,841	127,841

3.1 Deep learning models

3.1.1 Baseline model

155 As a comparison starting point, a single 7x7 deconvolutional kernel with relu activation function is used to generate a single precipitation map. The kernel combined the local input information to calculate a high-resolution rain map. The goal of this model was to offer a baseline comparison point of the simplest model to compare with more complex models.

3.1.2 U-Net

As it was mentioned in the introduction, the U-Net architecture (Ronneberger et al., 2015) was previously used to nowcast
160 and downcast meteorological data. The U-Net is composed of an initial contracting path followed by an expansive path that increases the resolution of the output, adding skip connections between the stages of the paths.

Due to its proven good performance in the above-mentioned tasks, it was considered essential to explore its performance to solve our present task and have it as a reference point to evaluate the performance of our models. In this work, we adapt the architecture used by Ayzel et al. (2020) to the dimensionality of our input and output data while conserving the essential
165 contracting and expanding paths. The detailed architecture can be found in appendix A.

3.1.3 One level deconvolution (~~Deconv1L~~Deconv1L4Rain v1)

In the ~~Deconv1L~~Deconv1L4Rain, a first 7x7 deconvolutional kernel is applied to the input that generates 32 high-resolution feature maps. Next, batch normalization is applied to avoid overfitting, followed by a convolution kernel that combines the 32 feature maps into a single precipitation image as output.

170 #### 3.1.4 Three levels deconvolution (~~Deconv3L~~Deconv3L4Rain v1)

The Deconv3L model first applies max-pooling to the input information to reduce its resolution to half (4.8 km^2). After that, it applies a first deconvolutional 5x5 kernel to generate 32 feature maps, followed by batch normalization. An additional deconvolutional kernel is applied over the generated feature maps to generate 16 combined feature maps. Finally, a single convolutional kernel estimates the output layer based on the previously mixed maps.

175 #### 3.1.5 Conditional Adversarial Generative Networks (~~CGANs~~CGAN4Rain v1)

For the ~~CGAN, the Deconv3L~~ CGAN4Rain, the Deconv3L4Rain was used as a generator to create rain maps based on the COSMO-DE-EPS input. Additionally, a discriminator model was implemented with the task of distinguishing between authentic images and generated images. The discriminator network was composed by 3 blocks of a convolutional kernel together with a dropout layer, followed by a flattening of the feature maps and a fully connected output. Differently from the previous
180 models, the difference between the generated images and the actual values was not directly back-propagated to the weights.

Instead, the generator loss is a combination of the distance between the predictions and outputs with an attempt to fool the discriminator, which leads the generator to create realistic outputs based on the input without blur.

3.2 Implementation

The ~~pre-processing, management and analysis of the data, as well as the training and testing of the algorithms~~ training of the models was performed on the JUWELS Supercomputer at the Jülich Supercomputer Center (JSC)~~. The~~, on standart compute nodes. Each node contained 2 Intel Xeon Platinum 8168 CPU with 24 cores each of 2.7 GHz, and 96 GB DDR4 of ram at 2666 MHz. The training took between 30 min. and 1 hour, depending on the complexity of the model. The implementation was based on Python 3.8.3 (Van Rossum and Drake Jr, 1995). Numpy 1.19.1 (Harris et al., 2020) was used to perform array operations. TensorFlow 2.3.1 (Abadi et al., 2015) is utilized ~~for the implementation of the~~ to implement the deep learning algorithms together with the Keras 2.4.0 library (Chollet et al., 2015). The mpi4py 3.0.3 library (Dalcin et al., 2019) was used to distribute ~~tasks on the HPC~~the initializations training on different Juwels JSC cores. Scikit-learn metrics 0.23.2 (Pedregosa et al., 2011) was used ~~to calculate the difference between predictions and real arrays, and Matplotlib (Hunter, 2007) was used to plot the results.~~ for verification and Matplotlib (Hunter, 2007) for plotting the results.

The complete implementation was subdivided into three sections. First, pre-processing was needed to prepare the data for training and to generate the different datasets from the complete set of forecasts and observations. Given that the performance of deep learning models can be affected by the random initialization of the kernel parameters, 20 independent random initializations were trained for each of the models of interest to make the study's results robust. Therefore, next, each initialization of the models was trained using the training set and monitored using the validation set. Last, the performance of each model initialization was evaluated using the test set. Each of the steps is explained in detail in the following subsections.

3.2.1 Pre-processing

The first step of the pre-processing was to match each of the COSMO-DE-EPS forecasts with their corresponding radar observation. All forecast-precipitation pairs containing any missing values were discarded (3.93% of the total). Next, the original values of the forecast were standardized in order to avoid numerical problems due to inputs in different units of measurement. The complete data set was split into train, test, and validation set by date. All forecasts with dates of 1st, 9th, 17th, and 25th, regardless of month and year, were selected as part of the test set (2671 forecasts and observations, 12.96% of the samples). In the same way, the validation set was built with the forecast of the 5th, 13th, 21st, and 28th of all months and years (2725 forecast and observations, 13.25% of the complete set). Finally, all remaining dates were chosen as part of the train set (15189 forecast and observations, 73.79% of the dataset). Finally the three datasets were stored and retrieved using TFRecords format from the TensorFlow library (Abadi et al., 2015) (the complete preprocessing routine was implemented and can be found in the notebook preprocessing.ipynb, part of the repository part of this paper).

3.2.2 Training

One single script was developed (baseline.py, unet.py, deconv1l.py, deconv3l.py, and cgan.py) for each model. The 20 initializations of each model were distributed between 20 independent cores of the JSC HPC sing the mpi4py library (Dalcin et al., 2019). Each core reads the training set stored as TFRecords, initializes a model with random initial weights, and fits the parameters to the train set. The training was performed during 24 epochs using a mini-batch size of 20. Mean squared logarithmic error was used as the loss measure due to the gamma-like distribution of the rain amount on the radar images, and Adam, with a learning rate of 0.001, was used as the optimizer. Once each core finished its respective training, the resulting model was stored in Tensorflows HDF5 format, and their loss during training was saved and plotted to check for convergence.

3.2.3 Verification and plotting

Once the trained models were stored, their performance was evaluated using the test set (evaluate.ipynb). Each of the 20 initializations was loaded and used to generate predictions for the input test set. Next, the predictions were evaluated using the verification methods (detailed in the following section) and the results were stored. Finally the evaluations of each model were plotted and illustrated (plot.ipynb)

3.3 Verification methods

Predictions generated by the trained models, as well as the predictions from COSMO-DE-EPS, were evaluated using two different types of metrics: continuous and dichotomous. The continuous verification metrics were: mean absolute error (MAE) and linear error in probability space (LEPS) based skill scores. The output of the models was also evaluated using three different dichotomous metrics based on the truth table (CSI, ETS, and frequency bias), for which five different thresholds were applied to the output: 0.2, 0.5, 1, 2, and 5 $\mathrm{mm/h}$. Each of the verification metrics is presented in detail in the following paragraphs.

3.3.1 Mean Absolute Error (MAE) based skill score

The mean absolute error measures the absolute difference between paired observations expressing the same phenomenon, in this case the difference between the forecast and the observed radar images. Considering $\boldsymbol{F}$ as the forecast vector and $\boldsymbol{O}$ as the observations, the MAE is defined as:

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\boldsymbol{F}_i - \boldsymbol{O}_i| \tag{1}$$

The term "skill score" indicates the degree of improvement of the new predictions compared to the original COSMO-DE-EPS forecast. The forecast is perfect when the skill score equals 1, while a skill score of zero and below indicates that there is no improvement or less skill in the new predictions compared to the reference. The MAE based skill score is calculated as follows:

$$\mathrm{skill\,score} = 1 - \frac{\mathrm{MAE}_{\mathrm{model\,predictions}}}{\mathrm{MAE}_{\mathrm{COSMO\text{-}DE\text{-}EPS\,predictions}}}. \tag{2}$$

3.3.2 Linear Error in Probability Space (LEPS) based skill score

LEPS is the mean absolute difference between the values that the forecast and observation take in the climatological cumulative distribution function (CDF) of the observations, i.e., LEPS is analogue to the MAE but in CDF space (Ward and Folland, 1991). Considering $\boldsymbol{F}$ as a distinct forecast, $\boldsymbol{O}$ as the respective observation and $\boldsymbol{CDF_o}$ as the CDF of the observations determined by an appropriate climatology, LEPS is defined as:

$$\text{LEPS} = \frac{1}{N}\sum_{i=1}^{N}|CDF_o(\boldsymbol{F}_i) - CDF_o(\boldsymbol{O}_i)| \tag{3}$$

Similar to other skill scores, a perfect forecast obtain a skill score equals to 1 and a skill score of zero and below would indicate a decrease in the performance compared to the reference models. The LEPS skill score is calculated as follows:

$$\text{skill score} = 1 - \frac{\text{LEPS}_{\text{model predictions}}}{\text{LEPS}_{\text{COSMO-DE-EPS predictions}}}. \tag{4}$$

3.3.3 Equitable ~~Thread~~ Threat Score (ETS)

The ETS measures the skill of the forecast relative to chance by measuring the fraction of the observed forecast events that were correctly predicted, adjusted for corrected predictions associated with random chance. It ranges between -1/3 and 1, with 0 and below indicating no skill and 1 representing a perfect forecast. Specifically, it is calculated in the following way:

$$\text{ETS} = \frac{\text{hits} - \text{hits}_{\text{random}}}{\text{hits} + \text{misses} + \text{false alarms} - \text{hits}_{\text{random}}} \tag{5}$$

With:

$$\text{hits}_{\text{random}} = \frac{(\text{hits} + \text{misses})(\text{hits} + \text{false alarms})}{\text{total}} \tag{6}$$

3.3.4 Critical Success Index (CSI)

The CSI measures the fraction of observed and/or forecast event that were correctly predicted. It ranges between 0 to 1, with 0 indicating no skill and a perfect score of 1.

$$\text{CSI} = \frac{\text{hits}}{\text{hits} + \text{false alarms} + \text{misses}} \tag{7}$$

3.3.5 Frequency Bias (BIAS)

The frequency bias measures the ratio of the frequency of forecast events to the frequency of observed events, it indicates if the forecast has a tendency to underforecast (BIAS < 1) or overforecast (BIAS > 1). It ranges from 0 to ∞ with a perfect score of 1, and does not measure how well the forecast corresponds to the observations, only relative frequencies.

$$\text{BIAS} = \frac{\text{hits} + \text{false alarms}}{\text{hits} + \text{misses}} \tag{8}$$

Table 2. Summary of the results: median and best performing model are presented for each of the metrics

metric	COSMO-DE-EPS score	Baseline median	Baseline best	U-Net median	U-Net best	Deconv1L4Rain v1 median	Deconv1L4Rain v1 best	Deconv3L4Rain v1 median	Deconv3L4Rain v1 best	CGAN4Rain v1 median	CGAN4Rain v1 best
MAE skill	0	.382	.441	.47	.497	.487	.499	.497	.516	.506	.513
LEPS skill	0	.683	.7	.651	.715	.726	.739	.725	.758	.755	.761
ETS 0.2	.309	.119	.255	.399	.406	.393	.398	.402	.409	.372	.386
ETS 0.5	.284	.128	.247	.315	.339	.335	.347	.315	.338	.301	.325
ETS 1	.235	.127	.216	.231	.257	.209	.265	.199	.255	.212	.251
ETS 2	.17	.109	.161	.117	.179	.044	.121	.042	.122	.074	.134
ETS 5	.067	.029	.05	0.0	.023	.002	.003	.001	.001	.001	.002
CSI 0.2	.355	.138	.282	.43	.437	.424	.431	.434	.441	.399	.414
CSI 0.5	.314	.14	.267	.332	.357	.354	.369	.332	.357	.318	.344
CSI 1	.252	.135	.229	.24	.267	.216	.275	.206	.265	.22	.261
CSI 2	.177	.112	.165	.119	.183	.045	.124	.043	.124	.076	.136
CSI 5	.068	.029	.051	.0	.023	.002	.003	.001	.001	.001	.002
FBias 0.2	2.057	.31	.272	.996	.002	.044	.994	.007	.001	.756	.875
FBias 0.5	2.007	.34	.778	.702	.801	.822	.997	.656	.841	.63	.774
FBias 1	1.987	.394	.974	.523	.664	.443	.767	.378	.638	.474	.637
FBias 2	1.877	.496	.92	.228	.487	.081	.3	.074	.27	.156	.326
FBias 5	2.078	.642	.041	.009	.072	.011	.056	.002	.013	.006	.029

Perfect score for all metrics = 1

4 Results

Our research goal was to explore applying deep learning models to generate high-resolution precipitation maps using low-resolution NWP simulations as input. We developed and tested five deep learning algorithms to achieve this goal. For each model, 20 independent runs with initial randomizations were performed, and their predictions were evaluated using the afore-mentioned metrics.

The results of this work demonstrate the general ability of deep learning algorithms to calculate high-resolution precipitation maps. Like in most machine learning applications, a significant influence of the architecture and the total number of parameters was found, indicating that the algorithm's ability to solve these tasks depends on finding the right level of complexity for the problem. The scores are summarized in table 2 and illustrated in figures 4 and 5. The models' predictions are shown in figure 6. A detailed analysis of the performance of each model is presented next.

First, we need to evaluate the skill of the original COSMO-DE-EPS to forecast precipitation. To do so, we compared the total precipitation variable against a low-resolution version of the radar observation using all metrics. To merge the RADKLIM

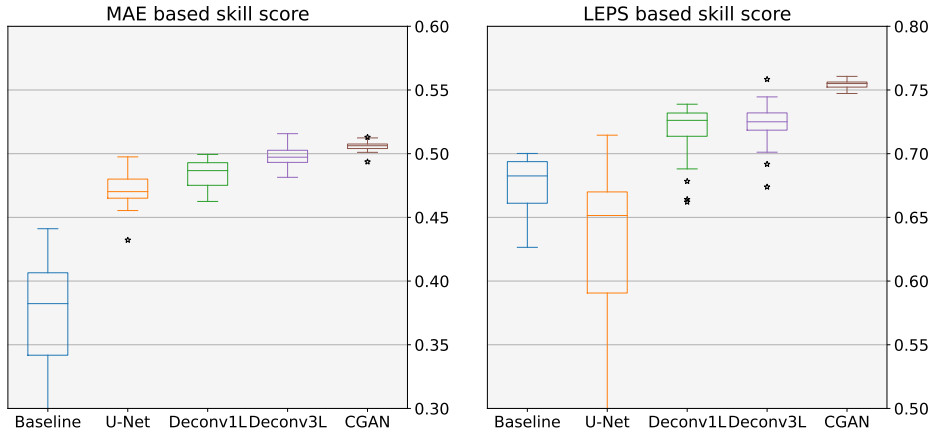



Figure 4. Quantitative metrics: each subplot compares the skill score of the different models. On the left side, each boxplot illustrates the MAE skill score, and at the right size, the LEPS skill score of 20 runs for each model with different initializations is depicted. The higher the skill score, the higher the improvement over the reference (in this case, COSMO-DE-EPS), a skill score of 0 represents no improvement, and a negative skill score indicates decay in the performance.

data onto the rotated pole grid of the COSMO-DE-EPS data, a conservative remapping step with CDO is performed, similar to the procedure described in the data section.

Given that MAE and LEPS skill scores are calculated in reference to the COSMO-DE-EPS performance, the respective
280 skill scores are equal to zero. Following the ETS and CSI scores, we observe an initial high performance of the model in low threshold events and a progressive decline with the threshold increase. This finding indicates a good general skill to predict low rain events and less skill to predict high precipitation events. However, COSMO-DE-EPS presents the highest scores in predicting high precipitation events (2 and 5 $\mathrm{mm/h}$). Analysing the frequency bias, a strong tendency to ~~overcast~~ overforecast rain regardless of the thresholds can be observed, a tendency that can be confirmed by examining the example predictions in
285 figure 6.

Starting with the baseline model, we can observe an increase in both skill scores concerning COSMO-DE-EPS. This improvement is limited compared to more complex models, which is expected due to the simplicity of the model and the reduced amount of parameters involved. Regarding ETS and CSI, a significant decrease in the median scores in all thresholds can be viewed, together with high variability, which signalizes a lack of consistency in the models' performance. Observing the
290 example precipitation maps, the problem comes to light: the precipitation maps show checkerboard artifacts, which leads to numerous rain points being missing. This artifact is caused by the superposition of the kernel operation, caused by the only deconvolutional layer included in the model. Despite the artifacts, the model generates a significant improvement in the frequency bias, obtaining values close to one in small thresholds. However, these results disappear at higher thresholds, indicating that the model does not produce stable results.

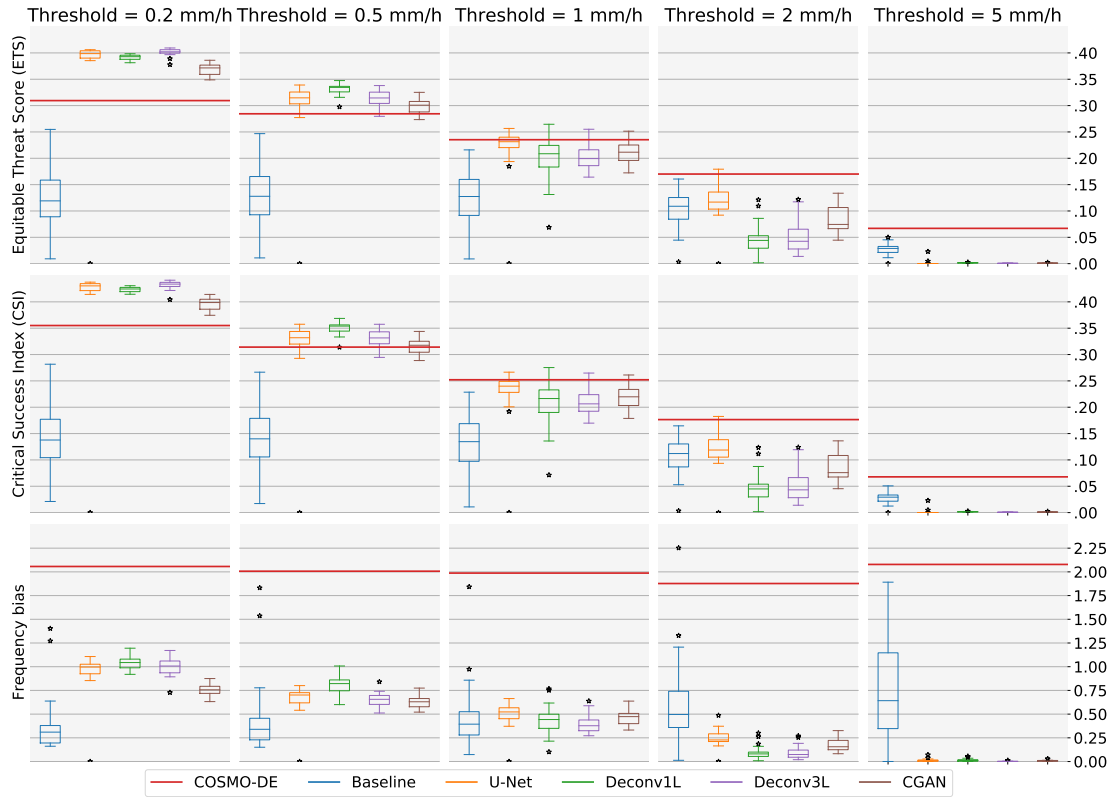



Figure 5. Dichotomous metrics: each subplot illustrates the metrics score of 20 independent models with different initial conditions after a certain threshold was applied. ETS and CSI vary between 0 and 1, with 0 indicating no skill and 1 a perfect forecast. Frequency bias ranges from 0 to +∞, with bias <1 indicating underforecast, bias >1 overforecast and a perfect score of 1.

The second comparison point was the U-Net-based model. The U-Net model presents a significant improvement in absolute error but a decrease with very high variability in the LEPS skill score. By observing the ETS, CSI, and frequency bias scores, a substantial improvement can be observed in its ability to predict low rain events. Regarding the performance for thresholds of 1 and 2 mm/h, this architecture obtains even better median scores than the more complex models, indicating a good ability to predict mid-range rain events. These scores suggest that the U-Net has a lower bias towards zero rain amounts than the baseline model and is prone to generating medium rain forecasts, which generates a higher average error that is reflected by the high variability of the LEPS skill score. This effect can be confirmed in the precipitation maps with more extensive and more intense precipitation regions.

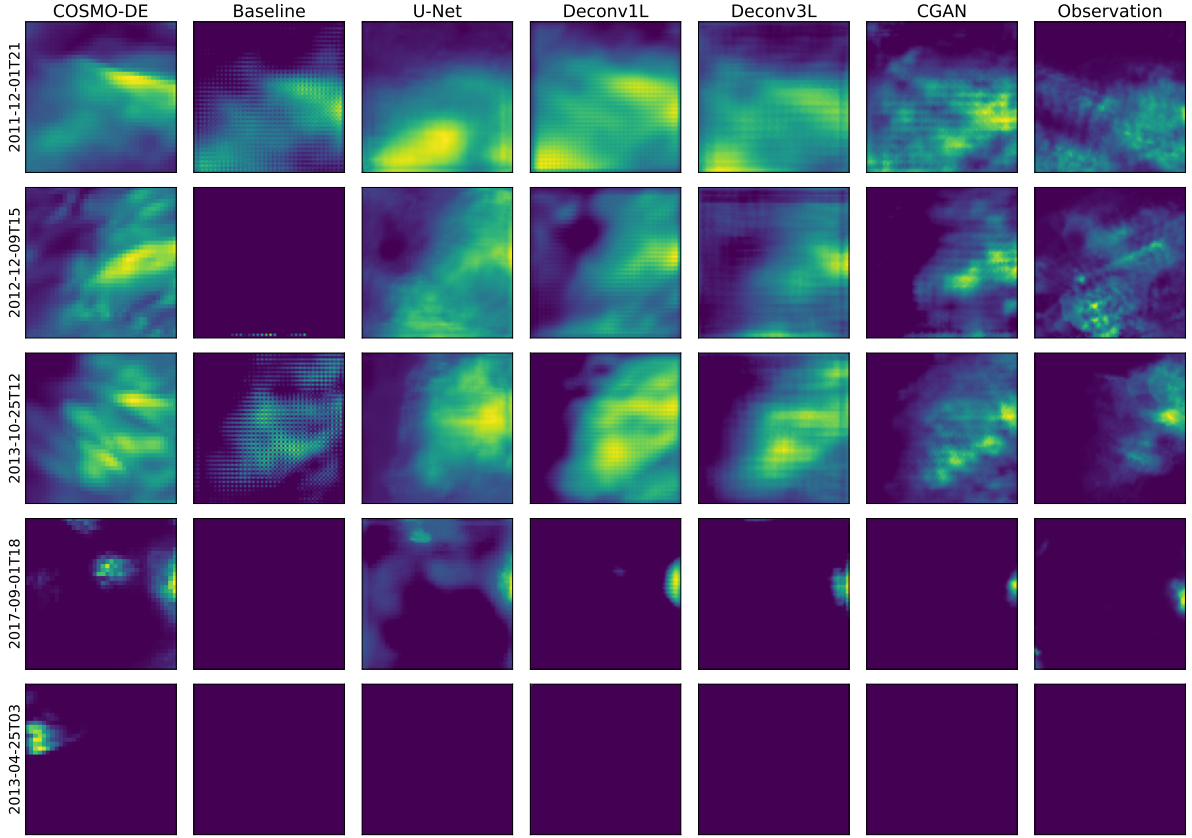



Figure 6. Example of predictions made by the different models versus the true values. For each of the models, the initialization with the highest LEPS skill score was selected to generate the predictions.

Moving to the main models of this work, the first deconvolutional model ~~Deconv1L~~ Deconv1L4Rain outperforms significantly the baseline and U-Net models in the MAE and LEPS skill scores. Observing the ETS and CSI scores, we can see a significant improvement in predicting low rain events, obtaining the best scores between all models for .5 mm/h and 1 mm/h thresholds. However, this ability starts to decrease significantly with the utilization of higher thresholds, which indicates the presence of a bias from the model to favor low precipitation events and difficulties in predicting high rain events. This tendency can be confirmed by observing the frequency bias, where the ~~Deconv1L~~ Deconv1L4Rain model obtains almost perfect scores for low thresholds but under forecasts for rain events of higher intensity. The example rain maps confirmed a solid tendency to predict low rain events in a generate spatial smoothing of the rain maps, which also explains the increase in the skill scores.

In the case of ~~Deconv3L~~ Deconv3L4Rain, we found a similar but slightly better performance compared to ~~Deconv1L~~ Deconv1L4Rain for the quantitative metrics. The median MAE skill score is above all previous models, and the median LEPS skill score is comparable to the ~~Deconv1L~~ Deconv1L4Rain, with the best initialization outperforming all previous models. Looking at the ETS and CSI, we found a similar performance to the ~~Deconv1L~~ Deconv1L4Rain model, with minor improve-

ments in the 0.2 mm/h threshold but a slight decrease in the performance of all other thresholds, which indicates a higher bias to perform spatial smoothing and to favor low rain events. This bias can be confirmed by observing the frequency bias scores, where the performance improves for the small thresholds but then decays compared to ~~Deconv1L~~ Deconv1L4Rain in all other thresholds. Example predictions confirm similar spatial smoothing found in other deconvolutional models. However, it is essential to consider that this model reaches comparable performance to ~~Deconv1L~~ Deconv14RainL using approximately half of the parameters, enabling more efficient use of computational resources.

Considering the ~~excellent~~ superior performance shown by the ~~Deconv3L~~ Deconv3L4Rain model, we used it as a generator model in a conditional generative adversarial model (~~CGANS~~CGAN4Rain). A significant improvement in performance and consistency can be found using Deconv3L as a generator in the MAE and LEPS skill scores. For lower thresholds (.2 and .5 mm/h), we observe a decay in the performance compared to the deconvolutional models but improved scores for higher thresholds. Additionally, the CGANS models have a lower frequency bias for these same thresholds. These metrics indicate a better model performance to predict high precipitation events, which indicates a correction of the spatial smoothing caused by the deconvolution operation. This correction is confirmed by observing the prediction examples, where we can find the more realistic high-resolution outputs between all models.

In summary, COSMO-DE-EPS tends to overpredict rain events, shown by a high-frequency bias and poor scores in ETS and CSI for lower thresholds. This tendency can be partially corrected by implementing deconvolutional models that present a superior performance to the reference models (Baseline, U-Net). However, these deconvolutional models induce a new type of bias towards low rain events, which provokes spatial smoothing in the examples and reduces the total level of error, and increases the accuracy for low rain events while deteriorating the skill to predict high precipitation events. This blurry effect can be countered by integrating the deconvolutinal model as a generator in a conditional generative model, which can generate accurate outputs resembling real rain maps. An additional finding was that the U-Net architecture did not achieve superior or stable scores for this problem. Possible reasons for this are discussed in the following section. Additionally, all models present a decrease in the scores when increasing the rain thresholds, indicating a general difficulty in predicting high precipitation events.

5   Conclusions

The results obtained in this work provide significant evidence for the application of deep learning models to perform bias correction of the NWP input and to generate high-resolution rain maps with improved accuracy. This improvement proves that using the complete information of the NWP variables and combining them in a non-linear fashion using feature maps helps to achieve higher quality and more precise precipitation forecasts.

The direct mapping between the physical simulations of the weather and the precipitation maps captured by radars can be achieved with the use of deep learning algorithms in a single step that combines increasing the resolution and correcting such inaccuracies of the original forecast. A significant amount of improvement performed by the deep learning models consisted

of correcting the tendency of the COSMO-DE-EPS to over-forecast, but the application of the developed models introduced new different types of bias.

The U-Net-based model presented an important bias to mid-range precipitation, which increased the average error of the predictions and made it less suitable to solve the task. A possible reason is that the contracting path of the architecture destroyed a significant part of the needed information to generate high-resolution predictions, causing the network to overestimate the amount of rain registered and, therefore, to produce inaccurate predictions.

This conclusion is sustained by the observation that the architecture of one of the best-performing models (~~Deconv3L~~Deconv3L4Rain) is inspired by the U-Net architecture, except for the contracting path that is substituted with a single max-pooling layer. The performance and consistency shown by the ~~Deconv3L~~ Deconv3L4Rain model are highly superior to the U-Net network. This contrasts with a series of works in the last years that used the U-Net architecture for super-resolution tasks with meteorological data (Ayzel et al., 2020; Serifi et al., 2021). In this particular case, where the input has an elevated number of channels (in this case, 143), the expanding path alone provided higher performance than the complete U-shaped structure.

On the other hand, the spatial smoothing caused by the deconvolution operation that has been reported in previous works (Ayzel et al., 2020; Ravuri et al., 2021) was also reproduced by our deconvolutional models. Consistent with the work of Ravuri et al. (2021), using the ~~Deconv3L~~ Deconv3L4Rain model as a generator in a conditional generative model helped to find the right trade-off between accuracy and realistic outputs and generated realistic high-resolution rain maps with the lowest error between all models. This gain in performance is due to back-propagating a combined loss function (chances to fool the discriminator and distance to real values) instead of only the difference between the real values and the model predictions, which avoids that the model optimizes just in terms of the logarithmic mean squared error, which causes the spatial smoothing.

An additional conclusion from this work is the relevance of using multiple metrics for the evaluation of precipitation predictions. Our results showed that the utilization of the LEPS could reveal inconsistencies first unattended by the MAE. Especially in the case of precipitation, the single use of MAE as a performance metric can be misleading due to the values' closeness to zero.

We provided important information about the characteristics of the application of models with different characteristics to the mapping between meteorological simulations and high-resolution rain maps that could guide the development of potential real-world implementations. The ~~main~~ first limitation of our results is the limited spatial and temporal resolution of our predictions. Our scope was limited to a forecast lead time of 3 hours and an area of $100 \times 100 \mathrm{km}^2$, which compared to real meteorological applications is a small domain. Also, nowcasting applications usually provide predictions for several time points in the future, while our work is limited to one. Additionally, modern practical meteorological applications offer probabilistic outputs, and our models generate deterministic predictions, which would be a limitation for a direct application of the algorithms. Most of these limitations are caused by our intention to generate an initial approximation of the capabilities of deep learning to solve this task instead of generating a fully functional and applicable model.

The key element for the improvement of the quality of the forecast is the inclusion of multiple variables (as channels), which provide enough information about the different states of the atmosphere and soil. The developed models combine and weights the input information (and its combinations), learning the relevant patterns in the input information to correct the

precipitation predictions and increase its resolution. Nonetheless, combining multiple variables entails two consequences: first, it requires higher computational resources to train the models than single or fewer variables. Second, it makes it very difficult to distinguish which variables are relevant to improve the forecast, given the non-linear mixing performed by the algorithm. Future research must explore the relevance of single variables and train models with subsets of variables that make more efficient use of the computational resources.

Another shortcoming of our approach is overlooking the different types of precipitation. Our models make no explicit differentiation between the different types of rain (stratiform/convective). However, it could be that the best-performing models have learned different patterns of input information to improve each type of phenomenon, but proving this requires a different analysis that is out of the scope of the present work. Future research could explore generating distinct models for the different types of precipitation and the role of the different input variables for each type of precipitation.

The accurate prediction of high rain events remains a challenge, which is partially due to the limited number of samples to train the algorithms, which makes this an extremely difficult event to predict from the deep learning perspective. The integration of deep learning in meteorological workflows seems to be a useful tool for improving meteorological models. In our case, the use of deconvolutional networks as a part of a CGAN provides promising results for the generation of accurate high-resolution precipitation outputs. Future research should extend the application of these models to bigger spatial and temporal domains, as well as more complex topographies, and at the same time search for the set of hyperparameters that allows for decreasing inaccuracies in the predictions.

Code and data availability. The code used in this study has been made public using the public repository https://github.com/DeepRainProject/ models_for_radar and can be downloaded under the direction https://doi.org/10.5281/zenodo.7535434. The datasets can be accessed under https://doi.org/10.5281/zenodo.7244319

A

Appendix A: U-Net architecture

Following the architecture proposed by Ronneberger et al. (2015) and Ayzel et al. (2020) the U-Net architecture was adapted
405   to the input and output size of the current task using the following architecture:

Layer (type)	Output shape	Param #	Connected to
input_1 (InputLayer)	(None, 36, 36, 143)	0	-
conv2d (Conv2D)	(None, 36, 36, 16)	20608	input_1[0][0]
conv2d_1 (Conv2D)	(None, 36, 36, 16)	2320	conv2d[0][0]
max_pooling2d (MaxPooling2D)	(None, 18, 18, 16)	0	conv2d_1[0][0]
conv2d_2 (Conv2D)	(None, 18, 18, 32)	4640	max_pooling2d[0][0]
conv2d_3 (Conv2D)	(None, 18, 18, 32)	9248	conv2d_2[0][0]
max_pooling2d_1 (MaxPooling2D)	(None, 9, 9, 32)	0	conv2d_3[0][0]
conv2d_4 (Conv2D)	(None, 9, 9, 64)	18496	max_pooling2d_1[0][0]
conv2d_5 (Conv2D)	(None, 9, 9, 64)	36928	conv2d_4[0][0]
up_sampling2d (UpSampling2D)	(None, 18, 18, 64)	0	conv2d_5[0][0]
concatenate (Concatenate)	(None, 18, 18, 80)	0	up_sampling2d[0][0], max_pooling2d[0][0]
conv2d_6 (Conv2D)	(None, 18, 18, 32)	23072	concatenate[0][0]
conv2d_7 (Conv2D)	(None, 18, 18, 32)	9248	conv2d_6[0][0]
up_sampling2d_1 (UpSampling2D)	(None, 36, 36, 32)	0	conv2d_7[0][0]
concatenate_1 (Concatenate)	(None, 36, 36, 175)	0	up_sampling2d_1[0][0], input_1[0][0]
conv2d_8 (Conv2D)	(None, 36, 36, 16)	25216	concatenate_1[0][0]
conv2d_9 (Conv2D)	(None, 36, 36, 16)	2320	conv2d_8[0][0]
up_sampling2d_2 (UpSampling2D)	(None, 72, 72, 16)	0	conv2d_9[0][0]
conv2d_10 (Conv2D)	(None, 72, 72, 8)	1160	up_sampling2d_2[0][0]
conv2d_11 (Conv2D)	(None, 72, 72, 8)	584	conv2d_10[0][0]
conv2d_12 (Conv2D)	(None, 72, 72, 1)	9	conv2d_11[0][0]

Total parameterss: 153 849, Trainable parameters: 153 849, Non-trainable parameters: 0

Author contributions. A. R. development and implementation all the deep learning models, methodology, visualization, software and writing original draft. M. L. selection of the domain region, data curation, review and editing. M. W. data curation, validation and editing. G. P. conceptualization, resources, supervision and editing.

Competing interests. The authors declare that they have no conflict of interest.

Acknowledgements. This work was possible thanks to the DeepRain project funded by the Bundesministerium für Bildung und Forschung (BMBF) under grant agreement 01 IS18047A. The authors gratefully acknowledge Jülich Supercomputing Centre (JSC) for the computing time and support to develop this project. The authors would like to thank Pascal Nieters for his initial guidance and Katharina Lüth for her attentive proofreading of this paper.

410

References

415   Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Good-
fellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S.,
Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F.,
Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous
Systems, https://www.tensorflow.org/, software available from tensorflow.org, 2015.

420   Agrawal, S., Barrington, L., Bromberg, C., Burge, J., Gazen, C., and Hickey, J.: Machine Learning for Precipitation Nowcasting from Radar
Images, 2019.

Ayzel, G., Heistermann, M., Sorokin, A., Nikitin, O., and Lukyanova, O.: All convolutional neural networks for radar-based precipitation
nowcasting, Procedia Computer Science, 150, 186–192, https://doi.org/https://doi.org/10.1016/j.procs.2019.02.036, proceedings of the
13th International Symposium "Intelligent Systems 2018" (INTELS'18), 22-24 October, 2018, St. Petersburg, Russia, 2019.

425   Ayzel, G., Scheffer, T., and Heistermann, M.: RainNet v1.0: a convolutional neural network for radar-based precipitation nowcasting, Geo-
scientific Model Development, 13, 2631–2644, https://doi.org/10.5194/gmd-13-2631-2020, 2020.

Bartels, H., Weigl, E., Reich, T., Lang, P., Wagner, A., Kohler, O., Gerlach, N., et al.: Projekt RADOLAN–Routineverfahren zur Online-
Aneichung der Radarniederschlagsdaten mit Hilfe von automatischen Bodenniederschlagsstationen (Ombrometer), Deutscher Wetterdi-
enst, Hydrometeorologie, 5, https://www.dwd.de/DE/leistungen/radolan/radolan_info/abschlussbericht_pdf, 2004.

430   Bauer, P., Thorpe, A., and Brunet, G.: The quite revolution of numerical weather prediction, Nature, pp. 47–55,
https://doi.org/10.1038/nature14956, 2015.

Chollet, F. et al.: Keras, https://github.com/fchollet/keras, 2015.

Dalcin, L., Mortensen, M., and Keyes, D.: Fast parallel multidimensional FFT using advanced MPI, Journal of Parallel and Distributed
Computing, https://doi.org/10.1016/j.jpdc.2019.02.006, 2019.

435   Dumoulin, V. and Visin, F.: A guide to convolution arithmetic for deep learning, 2018.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y.: Generative Adversarial
Networks, https://doi.org/10.48550/ARXIV.1406.2661, 2014.

Hanke, M., Redler, R., Holfeld, T., and Yastremsky, M.: YAC 1.2. 0: new aspects for coupling software in Earth system modelling, Geosci-
entific Model Development, 9, 2755–2769, https://doi.org/10.5194/gmd-9-2755-2016, 2016.

440   Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J.,
Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant,
P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E.: Array programming with NumPy, Nature, 585,
357–362, https://doi.org/10.1038/s41586-020-2649-2, 2020.

Hunter, J. D.: Matplotlib: A 2D graphics environment, Computing in Science & Engineering, 9, 90–95,
445   https://doi.org/10.1109/MCSE.2007.55, 2007.

Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A.: Image-to-Image Translation with Conditional Adversarial Networks,
https://doi.org/10.48550/ARXIV.1611.07004, 2016.

JPL, N.: NASA Shuttle Radar Topography Mission Global 1 arc second NetCDF, https://doi.org/10.5067/MEaSUREs/SRTM/SRTMGL1_NC.003.

Kimura, R.: Numerical weather prediction, Journal of Wind Engineering and Industrial Aerodynamics, 90, 1403–1414,
450   https://doi.org/https://doi.org/10.1016/S0167-6105(02)00261-1, fifth Asia-Pacific Conference on Wind Engineering, 2002.

Kreklow, J., Tetzlaff, B., Burkhard, B., and Kuhnt, G.: Radar-Based Precipitation Climatology in Germany—Developments, Uncertainties and Potentials, Atmosphere, 11, 217, https://doi.org/10.3390/atmos11020217, 2020.

Long, J., Shelhamer, E., and Darrell, T.: Fully Convolutional Networks for Semantic Segmentation, 2015.

Marsigli, C., M. A. P. T. S. D. W. A. A. M. and Schumann, T.: Evaluation of the Performance of the COSMO-LEPS System, Tech. Rep. Technical Report No. 8, Consortium for Small-Scale Modelling, Deutscher Wetterdienst, 63004 Offenbach, Germany, https://doi.org/https://doi.org/10.5676/DWD_pub/nwv/cosmo-tr_8, 2005.

Park, S. C., Park, M. K., and Kang, M. G.: Super-resolution image reconstruction: a technical overview, IEEE Signal Processing Magazine, 20, 21–36, https://doi.org/10.1109/MSP.2003.1203207, 2003.

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

Pejcic, V., Saavedra Garfias, P., Muehlbauer, K., Troemel, S., and Simmer, C.: Comparison between precipitation estimates of ground-based weather radar composites and GPM's DPR rainfall product over Germany, Meteorologische Zeitschrift, pp. 451–466, https://doi.org/10.1127/metz/2020/1039, 2020.

Peralta, C., Ben Bouallègue, Z., Theis, S. E., Gebhardt, C., and Buchhold, M.: Accounting for initial condition uncertainties in COSMO-DE-EPS, Journal of Geophysical Research: Atmospheres, 117, https://doi.org/https://doi.org/10.1029/2011JD016581, 2012.

Ravuri, S., Lenc, K., Willson, M., Kangin, D., Lam, R., Mirowski, P., Fitzsimons, M., Athanassiadou, M., Kashem, S., Madge, S., Prudden, R., Mandhane, A., Clark, A., Brock, A., Simonyan, K., Hadsell, R., Robinson, N., Clancy, E., Arribas, A., and Mohamed, S.: Skilful precipitation nowcasting using deep generative models of radar, Nature, 597, 672–677, https://doi.org/10.1038/s41586-021-03854-z, 2021.

Ronneberger, O., Fischer, P., and Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation, CoRR, abs/1505.04597, http://arxiv.org/abs/1505.04597, 2015.

Schultz, M. G., Betancourt, C., Gong, B., Kleinert, F., M, L., Leufen, L. H., Mozaffari, A., and Stadtler, S.: Can deep learning beat numerical weather prediction?, Phil. Trans. R. Soc. A., https://doi.org/10.1098/rsta.2020.0097, 2021.

Schulzweida, U.: CDO User Guide, https://doi.org/10.5281/zenodo.3539275, 2019.

Schättler, U., Doms, G., and Schraff, C.: A Description of the Nonhydrostatic Regional COSMO-Model: Model Output and Data Formats for I/O, Consortium for Small-Scale Modelling, http://www.cosmo-model.org/content/model/documentation/core/default.htm, 2019.

Serifi, A., Günther, T., and Ban, N.: Spatio-Temporal Downscaling of Climate Data Using Convolutional and Error-Predicting Neural Networks, Frontiers in Climate, 3, https://doi.org/10.3389/fclim.2021.656479, 2021.

Sha, Y., II, D. J. G., West, G., and Stull, R.: Deep-Learning-Based Gridded Downscaling of Surface Meteorological Variables in Complex Terrain. Part II: Daily Precipitation, Journal of Applied Meteorology and Climatology, 59, 2075 – 2092, https://doi.org/10.1175/JAMC-D-20-0058.1, 2020.

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., kin Wong, W., and chun Woo, W.: Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting, 2015.

Shi, X., Gao, Z., Lausen, L., Wang, H., Yeung, D., Wong, W., and Woo, W.: Deep Learning for Precipitation Nowcasting: A Benchmark and A New Model, CoRR, abs/1706.03458, http://arxiv.org/abs/1706.03458, 2017.

Stengel, K., Glaws, A., Hettinger, D., and King, R. N.: Adversarial super-resolution of climatological wind and solar data, Proceedings of the National Academy of Sciences, 117, 16 805–16 815, https://doi.org/10.1073/pnas.1918964117, 2020.

Valeria, G. and Massimo, M.: Reforecast of the November 1994 flood in Piedmont using ERA5 and COSMO model: an operational point of view, Bulletin of Atmospheric Science and Technology, 1, 339–354, https://doi.org/https://doi.org/10.1007/s42865-020-00027-0, 2021.

490   Van Rossum, G. and Drake Jr, F. L.: Python reference manual, Centrum voor Wiskunde en Informatica Amsterdam, 1995.

Ward, M. N. and Folland, C. K.: Prediction of seasonal rainfall in the north nordeste of Brazil using eigenvectors of sea-surface temperature, International Journal of Climatology, 11, 711 – 743, https://doi.org/10.1002/joc.3370110703, 1991.

Winterrath, T., Brendel, C., Hafer, M., Junghänel, T., Klameth, A., Lengfeld, K., Walawender, E., Weigl, E., and Becker, A.: RADKLIM Version 2017.002: Reprocessed gauge-adjusted radar data, one-hour precipitation sums (RW),
495   https://opendata.dwd.de/climate_environment/CDC/help/landing_pages/doi_landingpage_RADKLIM_RW_V2017.002-en.html, https://doi.org/10.5676/DWD/RADKLIM_RW_V2017.002, 2018.