Improving precipitation nowcasting for high-intensity events using deep generative models with balanced loss and temperature data: a case study in the Netherlands

Charlotte Cambier van Nooten, Koert Schreurs, Jasper S. Wijnands, Hidde Leijnse, Maurice Schmeits, Kirien Whan, Yuliya Shapovalova

a Institute for Computing and Information Sciences, Radboud University Nijmegen, Toernooiveld 212, Nijmegen, The Netherlands
b Royal Netherlands Meteorological Institute (KNMI), Utrechtseweg 297, De Bilt, The Netherlands
c Institute for Environmental Sciences, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands

Corresponding author: Charlotte Cambier van Nooten, charlotte.cambiervannooten@ru.nl

Early Online Release: This preliminary version has been accepted for publication in Artificial Intelligence for the Earth Systems, may be fully cited, and has been assigned DOI 10.1175/AIES-D-23-0017.1. The final typeset copyedited article will replace the EOR at the above DOI when it is published.

© 2023 American Meteorological Society. This is an Author Accepted Manuscript distributed under the terms of the default AMS reuse license. For information regarding reuse and general copyright information, consult the AMS Copyright Policy (www.ametsoc.org/PUBSReuseLicenses).
ABSTRACT: Precipitation nowcasting is essential for weather-dependent decision-making, but it remains a challenging problem despite active research. The combination of radar data and deep learning methods have opened a new avenue for research. Radar data is well-suited for precipitation nowcasting due to the high space-time resolution of the precipitation field. On the other hand, deep learning methods allow the exploitation of possible nonlinearities in the precipitation process. Thus far, deep learning approaches have demonstrated equal or better performance than optical flow methods for low-intensity precipitation, but nowcasting high-intensity events remains a challenge. In this study, we have built a deep generative model with various extensions to improve nowcasting of heavy precipitation intensities. Specifically, we consider different loss functions and how the incorporation of temperature data as an additional feature affects the model’s performance. Using radar data from KNMI and 5-90 minutes lead times, we demonstrate that the deep generative model with the proposed loss function and temperature feature outperforms other state-of-the-art models and benchmarks. Our model, with both loss function and feature extensions, is skilful at nowcasting precipitation the high rainfall intensities, up to 60 minutes lead time.
1. Introduction

Reliable precipitation forecasts are essential for decision-making in agriculture (Ingram et al. 2002), road transportation (Changnon 1996), aviation (Kulesa 2003) and construction (Senouci et al. 2018). In particular, short-term precipitation forecasts of up to two hours (i.e., nowcasting) are important to enable water-related risk management. Short-term forecasting systems rely on nowcasts to provide decision-making support for taking preventative actions (Thirugnanam et al. 2020; De Luca 2013). Given that the frequency and intensity of heavy rainfall events are expected to increase due to climate change, the need for accurate precipitation nowcasting becomes ever more important (Attema et al. 2014).

Traditionally, nowcasts are made using Numerical Weather Prediction (NWP) models or radar extrapolation-based models (Bauer et al. 2015). NWP models use physics-based simulation to forecast the weather. These models suffer from spin-up problems and they are therefore less skilful in precipitation nowcasting than radar extrapolation-based models such as Spectral Prognosis (S-PROG) (Seed 2003a). These models forecast precipitation by computing a motion field from recent radar data and advecting real-time observations along this trajectory. Optical flow methods are designed with the assumptions of Lagrangian persistence in mind, assuming that both the total rainfall intensity and the motion field remain constant (Germann and Zawadzki 2002). However, Lagrangian persistence does not always hold as motion, and intensity can vary with time, and thus these assumptions can limit model performance.

More recently, various studies have successfully applied machine learning for weather nowcasting (e.g., Shi et al. 2017; van der Kooij 2021). Machine learning methods, such as deep learning (DL), take advantage of the large-scale availability of historical weather data. These models do not require the assumption of Lagrangian persistence. For example, Trebing et al. (2021) proposed the SmaAt-UNet model for precipitation nowcasting, which is a lightweight version of the UNet architecture. Further, RainNet (Ayzel et al. 2020) is based on the UNet and SegNet neural network architectures. Several proposed deep learning models for nowcasting are based on the sequence-to-sequence Long Short-Term Memory (LSTM) framework (Sutskever et al. 2014). For example, Shi et al. (2015) proposed a convolutional LSTM (ConvLSTM) approach for precipitation nowcasting using radar images. This was further improved in the trajectory gated recurrent unit (TrajGRU) model (Shi et al. 2017). ConvLSTM and TrajGRU can capture spatiotemporal correlation better than a
fully connected LSTM and optical flow-based methods such as S-PROG (Shi et al. 2017). MetNet (Sønderby et al. 2020) used spatial downsampling, temporal encoding and axial attention-based aggregation for radar-based forecasting. MetNet outperformed NWP and optical flow models in terms of the F1 score, based on US radar data.

In addition to novel deep learning models for precipitation nowcasting, other machine learning techniques have shown their usefulness in this field. Data augmentation and transfer learning, in particular, deserve mention as they help enhance deep learning models’ performance and generalisation capabilities. Data augmentation plays a role in mitigating the challenges posed by limited training data and improving the robustness of DL classifiers. For example, UNet-based models, widely used for precipitation nowcasting, have demonstrated improved results when combined with data augmentation strategies (Ronneberger et al. 2015a; Falk et al. 2019; Shorten and Khoshgoftaar 2019). The training dataset can be expanded by applying rotations, flips, and scaling techniques, enabling the model to learn more diverse and representative features. Furthermore, transfer learning has emerged as a powerful approach to leverage pre-trained models and adapt them to new domains or tasks. Transfer learning techniques in weather nowcasting models can facilitate generalisation to different locations and enhance performance (Khorrami et al. 2021; Herruzo et al. 2021; Han et al. 2021). By incorporating data augmentation, models such as UNet-based architectures can benefit from an expanded training dataset and learn more representative features. Transfer learning, on the other hand, enables models like ConvLSTM, TrajGRU, and MetNet to leverage pre-existing knowledge and adapt it to new locations.

Although these deep learning approaches typically perform better than traditional approaches, they can also lead to forecasts that are smooth, blurry or look unrealistic (Tran and Song 2019). Generative Adversarial Networks (GANs) have been shown to address this issue (Hayatbini et al. 2019; Jing et al. 2019; Tian et al. 2019). In the GAN framework, two DL models are pitted against each other in an adversarial way (Goodfellow et al. 2014). The generator creates synthetic samples and the discriminator has to differentiate between real and synthetic samples. The generative model does not solely optimise the distance to a target image, but also aims to minimise the penalty that occurs when the discriminator assesses the image as unrealistic. Eventually, this leads to more detailed and realistic forecasts. For example, Jing et al. (2019) proposed the Adversarial Extrapolation Neural Network (AENN) GAN model that outperformed various other nowcasting...
models. Further, Ravuri et al. (2021) proposed a deep generative model of radar (DGMR), based on a conditional generative model with one generator and two discriminators. One discriminator focused on spatial consistency, and the other on temporal consistency. The model was evaluated quantitatively using several statistical metrics, and qualitatively by a group of expert meteorologists in the UK. The model showed more fine-grained details and more realistic results than other traditional and DL methods.

Despite recent advancements, several gaps remain that require further research. First, while the issue of decreased performance (in terms of accuracy) for high-intensity rainfall and long lead times is not unique to deep learning methods, various studies have shown that deep learning approaches are also susceptible to decreased performance in these cases (Ayzel et al. 2020; Shi et al. 2015, 2017). Further, deep learning models typically forecast precipitation using limited or no physical constraints. However, weather-based features can contain much information and dependencies for precipitation (Bouget et al. 2020). Our study has investigated several potential extensions to the state-of-the-art DGMR model. First, the influence of different loss functions on forecast accuracy was explored, including an extended balanced loss function that assigns more weight to severe events. Second, the fusion of radar data and weather-based features has been explored to incorporate additional meteorological information into the modelling. Through a comparison with various benchmark methods, we provide qualitative and quantitative insights on how these experiments can further improve precipitation nowcasting performance.

2. Methods

a. Data

As a case study, all models have been trained and evaluated on precipitation radar data from the Royal Netherlands Meteorological Institute (KNMI). Data from the period of 2006-2021 was obtained from operational radar stations located in Herwijnen and Den Helder, which provided full coverage of the Netherlands and surrounding areas in Belgium, Germany, and the North Sea. This research has used the real-time radar product with a five-minute temporal and 1×1 km spatial resolution. To explore the potential of other meteorological variables for precipitation nowcasting, air temperature data was obtained from KNMI. In total, more than 45 automatic weather stations in the Netherlands provided the 1.5-meter air temperature in degrees Celsius at 10-minute intervals.
The automatic weather stations are located on sea and land. The stations on sea are mostly on platforms with a higher altitude and therefore the measured temperature has been corrected for this higher altitude, by applying a moist-adiabatic lapse rate of 0.006 K/m.

**b. Pre-processing steps**

The radars record the amount of backscattered radiation, defined as radar reflectivity $z$, which is measured on a logarithmic scale in dBZ. Precipitation rates can be retrieved from $z$ through the $z - r$ relationship

$$z = 200r^{1.6},$$

(1)

with rainfall rate $r$ in mm/h (Marshall and Palmer 1948).

As this study has used a real-time radar product, various additional pre-processing steps have been implemented to enhance data quality. First, a mean-field bias correction was applied to the data using automatic rain gauge stations. Further, extreme rainfall intensity values (e.g., caused by hail or clutter) were clipped at 100 mm/h. The real-time product also contains various instances of clutter. Clutter consists of unwanted echoes (i.e., objects not part of the radar’s target), such as airplanes, birds, ships, or wind turbines. Although machine learning models could learn to handle clutter, it is important to remove it during pre-processing. Firstly, machine learning models would have an unfair advantage compared to extrapolation-based methods. Secondly, clutter in the reference data set will have a negative influence on both the model and the verification scores. Rainfall tends to gradually increase from low to high precipitation values and visa versa, while clutter sudden increases and decreases in radar reflectivity, resulting in abnormally large gradients. The amount of clutter in the entire data set was reduced by discarding single rain images that contained many pixels with a high gradient (Schreurs et al. 2021). Finally, in order to make the models faster, radar images with very low precipitation intensities were discarded to avoid images that did not contain any rainfall at all. Images were labelled as 'containing precipitation', if the mean intensity per pixel was above 0.01 mm/h. This led to the removal of 58.22% of samples (see Table 1). Note that only 4.18% of the samples contain heavy precipitation with a peak intensity higher than 10 mm/h. All rainfall rates have been normalized using min-max normalization for further processing, resulting in values between 0 and 1.
Table 1: Statistics of precipitation intensity using radar data (years 2006-2021). 5-minute frames/sequences containing a specific amount of precipitation (mm/h is defined by peak value over the entire frame), expressed in percentages. Precipitation sequences of 30 minutes, contain 6 consecutive images of 5-min frames.

<table>
<thead>
<tr>
<th>Precipitation (5 min)</th>
<th>41.78 %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precipitation sequence (30 min)</td>
<td>39.21 %</td>
</tr>
<tr>
<td>Low intensity (5 min, (0.01 – 1.0] mm/h)</td>
<td>5.94 %</td>
</tr>
<tr>
<td>Moderate intensity (5 min, (1.0 – 5.0] mm/h)</td>
<td>19.65 %</td>
</tr>
<tr>
<td>Moderate-high intensity (5 min, (5.0 – 10.0] mm/h)</td>
<td>12.01 %</td>
</tr>
<tr>
<td>High intensity (5 min, (10.0 – 20.0] mm/h)</td>
<td>3.20 %</td>
</tr>
<tr>
<td>Very high intensity (5 min, &gt; 20.0 mm/h)</td>
<td>1.04 %</td>
</tr>
</tbody>
</table>

The radar images in the data set have an original size of 765×700 pixels. Zero padding was used to increase the dimensions to 768×768 pixels. Bilinear interpolation reduced the image size by a factor of three, obtaining a final precipitation image with a size of 256×256 pixels. A sequence of radar images is used to capture the evolution of precipitation. To obtain 30-minute episodes of rain events, only sequences containing six consecutive five-minute frames, with precipitation, were retained (see Table 1). The 30-minute sequences were used for precipitation nowcasting for the next 5, 10, 15, 30, 60 and 90 minutes. Forecasts were evaluated using the original radar images. Therefore, the model outputs were upsampled using bilinear interpolation with a factor of three and cropped to a size of 765×700 pixels.

When dealing with time series data, splitting the data randomly into training and test sets is not recommended, as data leakage may occur (Fu 2011). In order to prevent this, the data set was split using consecutive years. Years from 2008 to 2020 were used as a training set, while 2006 was used as a validation set to optimize hyperparameters. The most recent year available at the time of this study (i.e., 2021) was used as a test set.

c. Deep generative model of radar

As in this paper we extend the DGMR model (Ravuri et al. 2021), a brief description of this model is provided first. The DGMR consists of three components, a generator and two discriminators, which are trained simultaneously. The generator is a convolutional neural network consisting of a conditioning stack and a sampler. The conditioning stack enables the generative model to use radar data from multiple resolutions (Choi and Kim 2021; Ravuri et al. 2021) and is based on
residual blocks and spectrally normalised convolutional 2D layers (SNConv2D) (Miyato et al. 2018). Subsequently, the sampler generates several predictions (i.e., future radar images). The sampler consists of multiple modules, including a latent conditioning stack and convolutional gated recurrent unit (ConvGRU) layers. Two discriminator models check whether generated images are realistic in terms of spatial and temporal features. The spatial discriminator ($S$) assesses a generated radar image at a single time step using 2D convolutions (i.e., SNConv2D). In contrast, the temporal discriminator ($T$) uses 3D convolutions to consider the temporal consistency of a generated sequence. The spatial and temporal discriminators are trained by minimizing loss functions $L_S$ and $L_T$ correspondingly:

$$L_S(\varphi) = \mathbb{E}_{X_{1:M+N};Z}[ReLU(1 - S_{\varphi}(X_{M+1:M+N}) + ReLU(1 + S_{\varphi}(G(Z;X_{1:M})))],$$  

$$L_T(\psi) = \mathbb{E}_{X_{1:M+N};Z}[ReLU(1 - T_{\psi}(X_{1:M+N})) + ReLU(1 + T_{\psi}((X_{1:M};G(Z;X_{1:M}))))],$$  

where $X_{1:M}$ is the radar observation input sequence ($M = 6$ frames), $Z$ the latent estimates using Monte Carlo (Seoh 2020), $G$ the generator function and ReLU the rectified linear unit activation function. $\{X;G\}$ denotes the concatenation of the two different fields. The generator incorporates the losses of both discriminators, maximising the following objective:

$$L_G(\theta) = \mathbb{E}_{X_{1:M+N}}[\mathbb{E}_{Z}[S(G_\theta(Z;X_{1:M}))+T((X_{1:M};G_\theta(Z;X_{1:M})))]] - \lambda L_R(\theta),$$  

with $L_R(\theta)$ a regularisation term based on the $\ell_1$ distance (i.e., Manhattan distance). Loss functions in Equations (2)-(4) are minimised using neural network weights $\varphi$ for the spatial discriminator $L_S$, $\psi$ for the temporal discriminator $L_T$, and $\theta$ for the generator $L_G$. For further details about the DGMR neural network architecture and modelling assumptions, see Ravuri et al. (2021).

d. Model extensions

First, the impact of different loss functions on the ability to accurately forecast higher rainfall intensities was investigated. Importantly, the proportion of rainfall events at different intensity thresholds is imbalanced, with only a minority of the data capturing severe events (see Table 1). Standard loss functions, therefore, optimise the model mainly for moderate intensity events since samples are treated as equally important, and most samples fall in the moderate and moderate-
heavy categories. However, high-impact events are typically of most importance for meteorological applications. Different loss functions could encourage the model to predict higher rainfall intensities more accurately. Note that this could also lead to (marginally) lower performance for lower rainfall intensities.

In a conditional GAN, the reconstruction loss $L_R$ captures the distance between the observed value $X$ and the forecast $G(\cdot)$. The root mean squared error (RMSE) loss function was used as a baseline for $L_R$. In DGMR, the regularisation term only corresponds to the $\ell_1$ distance, also referred to as the Manhattan distance. The $\ell_1$ distance looks at the sum of the absolute values of a vector. This was compared with the Hinge loss (e.g., see Ravuri et al. 2021), a balanced loss function using only the Manhattan distance, and an extended balanced loss function (e.g., see Shi et al. 2017).

Specifically, as the reconstruction loss, we used the sum of the balanced mean absolute error (B-MAE) and balanced mean squared error (B-MSE). This extended balanced loss function is defined for the 256 by 256 pixels grid as

$$L_R = \text{B-MAE} + \text{B-MSE},$$  

(5)

where

$$\text{B-MAE} = \frac{1}{N} \| \mathbb{E}_{Z}(G_{\theta}(Z; X_{1:M}) - X_{M+1:M+N}) \odot w(X_{M+1:M+N}) \|_1,$$  

(6)

$$\text{B-MSE} = \frac{1}{N} (\| \mathbb{E}_{Z}(G_{\theta}(Z; X_{1:M}) - X_{M+1:M+N}) \odot w(X_{M+1:M+N}) \|_2)^2,$$  

(7)

with $w$ the weight corresponding to the sample. These weights were assumed to be equal to the observed intensity of rainfall in mm/h ($R$) at the corresponding location,

$$w(y) = R(y) = \min(\max(1, y), 30),$$  

(8)

with the value of the weight being clipped to a minimum of 1 mm/h and a maximum of 30 mm/h. The proposed maximum threshold of 24 mm/h by Ravuri et al. (2021), resulted in being too low for the Dutch events. The assignment of more weight to severe events should, in theory, lead to more consistent nowcasting performance across moderate and high-intensity precipitation. The extended balanced loss incorporates the Manhattan distance ($\ell_1$) and the Euclidean distance ($\ell_2$),
resulting in the sum of precipitation values for the weighted mean average error and the mean squared error.

Besides different loss functions, the addition of weather-related features to the model was also investigated. Temperature data from KNMI was transformed to a 765×700 grid to align with the dimensions of the precipitation data. Temperature data was used from automatic weather stations on land and sea. For each automatic weather station, the temperature measurement was placed onto the grid, based on location. Empty grid cells were filled using inverse distance-weighted interpolation (IDW) (Lu and Wong 2008). The IDW method interpolated the pixel cell values by averaging the station data points in the neighbourhood of each processing cell, using a 5 km block. Zero padding and bilinear interpolation reduced the size to 256×256 pixels and values were normalized. Finally, the precipitation and temperature images were stacked along the temporal axis, resulting in a generated input tensor with dimensions of 6×256×256×2 (i.e., two channels capturing six precipitation maps and six temperature maps). This way, the convolution filters take into account that rainfall and temperature are provided at the same location and time.

e. Implementation and calibration

The DGMR model with the extensions described in Section 2d was implemented using Python and the TensorFlow Keras library (Abadi et al. 2016). Fig. 1 presents a schematic overview of the implemented modelling pipeline. The code of our implementation, using KNMI data, is available through GitHub1.

Models were trained and evaluated on the CPU and GPU cluster nodes of the Data Science department at Radboud University (i.e., 2x Intel Xeon Silver 4214 2.2 GHz, 128 GB, 7x GPU). Hyperparameters were iteratively changed to optimize forecast accuracy on the validation set and improve the stability of the models. The scaling parameter for the regularisation term in DGMR was set to 21. A batch size of 16 was used, and the model was trained for 125 epochs using the Adam optimizer with a learning rate of 0.0005 for the generator and 0.00015 for the discriminators. The discriminators were trained two times for every update step of the generator. All hyperparameter values for the DGMR model are reported in Table B2 of Appendix B.

1https://github.com/charlottecvn/precipitationnowcasting-generativemodels-highevents
f. Benchmark models

Model performance in the different experiments was compared to various benchmark models. The selected benchmark models were the optical flow model S-PROG and the deep learning models UNet, MetNet and AENN GAN. A description of these models is provided below. As the focus of this study is on precipitation nowcasting in the Netherlands, all benchmark models were calibrated using the data described in Section 2a. While data augmentation can be beneficial for specific scenarios and models (Shorten and Khoshgoftaar 2019), additional data augmentation is not included to compare with the DGMR (Ravuri et al. 2021) directly.

1) S-PROG

S-PROG (Seed 2003b) is an optical flow-based extrapolation method. First, S-PROG converts radar rainfall rates (mm/h) to radar reflectivity values (from R to dBR to dBZ). The motion field is constructed using the Lucas-Kanade method (Lucas et al. 1981), and semi-Lagrangian extrapolation is applied to the reflectivity field. An auto-regressive AR(2) model was used to estimate the Lagrangian auto-correlation coefficients. The radar field is then decomposed into eight cascade levels, after which the forecasts are constructed using the sum of all the Lagrangian nowcasts at different cascade levels. Finally, values are converted back to rainfall rates (mm/h)
(Imhoff et al. 2020). As the Lagrangian persistence assumes that the rainfall intensity and motion field are stationary for predicting the evolution of the motion field, S-PROG does not need any training. In this study, the open-source implementation (i.e., Pysteps deterministic) and set-up from Schreurs et al. (2021) have been used.

2) UNet

UNet (Ronneberger et al. 2015b) is an encoder-decoder model for image segmentation that can also be used for precipitation nowcasting (e.g., Agrawal et al. 2019; Ayzel et al. 2020; Nie et al. 2021). The standard UNet model uses convolutional and transposed convolutional layers for the encoder and decoder, respectively. Skipping connections between corresponding layers in the encoder and decoder improve details in the output image. Ravuri et al. (2021) implemented an adapted version of the UNet architecture by switching the convolutional layers for residual blocks, leading to a slight improvement for all rain intensities. Therefore, this adapted version of the UNet has been used as a benchmark model. A batch size of 8 was used, and the model was trained for 300 epochs using the Adam optimizer with a learning rate of 0.0002, and default exponential decay rates.

3) MetNet

MetNet is a deep learning framework for precipitation nowcasting proposed by Sønderby et al. (2020). The model architecture is based on spatial downsampling, temporal encoding (using the ConvLSTM framework as encoder), and axial attention-based aggregation. Further, the network uses a separate loss term for each grid cell and produces a probabilistic precipitation map. MetNet uses different layers of input data for the network, including (i) radar precipitation data, (ii) Geostationary Operational Environmental Satellite 16 (GOES-16) data (providing data over the Americas), (iii) per-pixel elevation and position embeddings and (iv) per-frame time embeddings. Not all of those inputs were easily accessible. Therefore, the model implementation in this study only used the radar data and no additional data (as in Ravuri et al. 2021). Although MetNet was evaluated by Sønderby et al. (2020) on a prediction lead time of up to 8 hours, it was trained in our study with lead times of 5 to 90 minutes. A batch size of 8 was used, and the model was trained for 500 epochs using the Adam optimizer with a learning rate of 0.00002, and a default decay rate of 0.
4) AENN GAN

The AENN GAN (Jing et al. 2019) uses a generator and two discriminators to address the issue of blurry forecasts. The generator uses an encoder-decoder architecture for sequence-to-sequence prediction (Sutskever et al. 2014). First, the conditional generator extracts features from the input images. The sequence in feature space is modelled using ConvLSTM layers to capture the temporal dynamics. Similar to the DGMR model, there is a spatial discriminator for generated single frames and a temporal discriminator for frame sequences. Previously, the AENN GAN was implemented using radar data from KNMI for precipitation nowcasting (Schreurs et al. 2021). This implementation replaced the ConvLSTM layers with convolutional gated recurrent unit (ConvGRU) layers to further improve accuracy. Here, this adapted AENN GAN was used as a benchmark model. A batch size of 8 was used, and the model was trained for 100 epochs using the Adam optimizer with a learning rate of 0.0001 for the generator and discriminator.

g. Evaluation criteria

In order to evaluate the experiments and benchmark methods, forecasts were generated for lead times of 5, 10, 15, 30, 60, and 90 minutes. Model forecasts were resized and cropped to 765×700 pixels for comparison with the ground truth radar images. Our study assesses the models from both a qualitative and a quantitative perspective. The qualitative assessment focuses on whether the forecast broadly captures the meteorological characteristics visible in the ground truth image, the level of details visible in the forecast and whether the generated radar image and sequence looks realistic. For the quantitative assessment, both continuous (i.e., MSE and MAE) and categorical metrics are used. The categorical metrics are used to analyse the performance of the models at different rainfall intensity levels (i.e., thresholds). Therefore, the forecasted and observed images are converted into binary maps based on a chosen precipitation threshold, indicating at each pixel location whether precipitation rates are higher than the threshold or not. Three rainfall intensity thresholds are selected for evaluation: 1, 10 and 20 mm/h. The latter two thresholds capture high and very high rain-intensity events (see also Table 1). The highest precipitation threshold that we consider is 20 mm/h, which is the 98th percentile of the data set. The amount of cases with a threshold higher than 20 mm/h is not enough to verify the results in a meaningful way. The binary maps are used to compute various statistical metrics (Imhoff et al. 2022). Statistical metrics use
a contingency table, with four possible outcomes: (i) true positives (TP), (ii) false positives (FP), (iii) true negatives (TN) and (iv) false negatives (FN). The Critical Success Index (CSI, also known as threat score), $CSI = \frac{TP}{TP + FP + FN}$, measures if the correctly forecasted positives correspond well to the total number of cases minus the correctly forecasted negatives. The F1-score, does not take true negatives into account, and can be defined as, $F1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$. The F1-score indicates the amount of correctly predicted nowcastings. The Probability of Detection (POD), $POD = \frac{TP}{TP + FN}$, measures the ratio of the correctly predicted positives to all observed positives. The False Alarm Rate (FAR), $FAR = \frac{FP}{TP + FP}$, expressed the ratio of false positives to all forecasted positives. Model bias was computed using the POD and the success ratio (i.e., $\frac{TP}{TP + FP}$). For the CSI and POD, higher values indicate better model performance. For the MSE, MAE, and FAR, better performance is achieved with lower values.
3. Results & Discussion

In this section, we present and discuss the results of the proposed methods. Models are evaluated using a shorter maximum lead time than former related studies (Ravuri et al. 2021; Schreurs et al. 2021). As previously discussed, the data from year 2021 is used as test set, see Appendix A for the behaviour of the model towards the amount of high-intensity events in the test set.

a. Benchmark Comparison

Since the original DGMR model (Ravuri et al. 2021) has not been applied to radar data from the Netherlands (KNMI), we compare the DGMR against four benchmarks (AENN GAN, S-PROG, UNet, and MetNet) using those data. The nowcasting performances are evaluated using the MSE, MAE and the categorical score CSI. Finally, the runtime performance on CPU and GPU is evaluated.

Figures 2a and 2b show that the DGMR, without extensions, performs better than the S-PROG in terms of MAE and MSE. Compared to the deep learning models, the S-PROG model performs better than the UNet model for longer lead times, but worse than the other DL models. The MetNet model performs worse than the DGMR, and similarly (in terms of MSE) or worse (in terms of MAE) than AENN GAN. Especially for the shorter lead times, the DGMR performs best.

![Figure 2a](image1.png) ![Figure 2b](image2.png)

Fig. 2: a) Performance of the different models, measured with mean absolute error in mm/h b) Performance of the different models, measured with mean squared error in (mm/h)$^2$.

Table 2 shows the CSI for the generative models without extensions and the benchmark models (aggregation over all lead times). Comparing the models without extensions against the benchmark
models, we see that the DGMR performs best regarding the CSI for the 1 and 20 mm/h rainfall intensity threshold. For 10 mm/h rainfall intensity, the MetNet model performs best.

<table>
<thead>
<tr>
<th>Model</th>
<th>1 mm/h</th>
<th>10 mm/h</th>
<th>20 mm/h</th>
</tr>
</thead>
<tbody>
<tr>
<td>AENN GAN</td>
<td>0.381</td>
<td>0.115</td>
<td>0.031</td>
</tr>
<tr>
<td>DGMR</td>
<td>0.487</td>
<td>0.217</td>
<td>0.135</td>
</tr>
<tr>
<td>S-PROG</td>
<td>0.485</td>
<td>0.210</td>
<td>0.130</td>
</tr>
<tr>
<td>UNet</td>
<td>0.379</td>
<td>0.208</td>
<td>0.116</td>
</tr>
<tr>
<td>MetNet</td>
<td>0.481</td>
<td>0.219</td>
<td>0.134</td>
</tr>
</tbody>
</table>

**Table 2:** Critical Success Index (CSI) for the baseline models and AENN GAN and DGMR using different losses and features; the best result (highest value) for each threshold is indicated in bold. Aggregation over all lead times is used.

When using the default loss functions, the results show that the DGMR scores higher on nowcasting for smaller lead times (i.e., below 30 minutes), in terms of MSE and MAE. Although the DGMR also scores better on higher rainfall intensities (i.e., above 10 mm/h) in terms of the CSI. There are more wrongly predicted nowcasting samples when using the DGMR for higher rainfall intensities (i.e., FAR score of 0.219). This is consistent with the work of Ravuri et al. (2021) that the prediction of high rainfall intensities at longer lead times remains a problem. Using the default loss function, neither of the models do well for high rainfall intensities at longer lead times, but the DGMR model performs the best.

The inference speed is evaluated on the CPU and GPU. The reported time in Table 3 is the median execution speed time for the selected models across ten samples. For training times, see Table B1. The execution speed of UNet using the CPU/GPU is significantly shorter than for the other models, followed by the AENN GAN and DGMR. Except for MetNet, all the deep learning models are faster than the S-PROG model.

<table>
<thead>
<tr>
<th>Model</th>
<th>S-PROG (seconds)</th>
<th>UNet (seconds)</th>
<th>MetNet (seconds)</th>
<th>AENN GAN (seconds)</th>
<th>DGMR (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU (sec)</td>
<td>72.91</td>
<td>4.83</td>
<td>106.82</td>
<td>24.75</td>
<td>37.81</td>
</tr>
<tr>
<td>GPU (sec)</td>
<td>5.33</td>
<td>0.21</td>
<td>8.94</td>
<td>1.78</td>
<td>2.94</td>
</tr>
</tbody>
</table>

**Table 3:** Median execution speed time for the selected models across 10 samples.

Besides the quantitative analyses above, radar nowcast images from the different models have also been assessed from a qualitative perspective. Figure 3 illustrates the model behaviour and the difference of nowcasts by the different models for a rainy event, initialised at $t_0 = 2021-05-$
01 11:05:00, with a lead time of 30 minutes. Figure 3 shows that the DGMR outperforms the benchmark models. See Figure 9 for the model behaviour of the DGMR, with lead times of 5, 10, 15, 30, 60 and 90 minutes. All four models capture more or less the same rain pattern, but the AENN GAN model suffers from checkerboard artefacts. Further, MetNet and S-PROG produce more blurry results for higher lead times. The DGMR consists of more (trainable) parameters compared to the AENN GAN. This might be a reason for the increase in performance at several intensity thresholds. However, it also comes with a slight increase in computational requirements. Furthermore, additional examples of the prediction frames are provided in Appendix D.

![Fig. 3: Nowcast and target value for different models for a rainy event, initialised at \( t_0 = 2021-05-01 11:05:00 \) and lead time +30 minutes, in (mm/h).](image)

**b. Balanced Loss Function**

In this subsection, the model performances of the DGMR and AENN GAN, with four loss versions are compared: the RMSE loss, the Hinge loss (Equation 4), the balanced loss function (Equation 6), and the extended balanced loss function (Equation 5). All four losses are reconstruction/regularisation losses. We propose the extended balanced loss function with the goal of improving model performance for higher precipitation intensities. The extended balanced loss uses the Manhattan distance (\( \ell_1 \)) and the Euclidean distance (\( \ell_2 \)), incorporating Euclidean distance for the minimisation of large outliers. The nowcasting performances are evaluated using the MSE, MAE and the categorical scores CSI, F1, POD, and FAR. Comparing the models in this manner enables us to distinguish the effect of the extended balanced loss we propose. Due to the aim of improving the precipitation nowcastings using the DGMR, performances are only compared against the other generative network (AENN GAN).
As shown in Figures 4a and 4b, the DGMR-BalancedExtended is somewhat better than the DGMR using the other losses in terms of MAE and MSE for the shorter lead times. Overall, the DGMR-BalancedExtended has the lowest errors in MAE and MSE. The DGMR-Balanced loss performs best for the lead time of 90 minutes. Thus, based on the MSE and MAE, the proposed extended balanced loss is, in particular, promising for short-term nowcasting, while the balanced loss from Ravuri et al. (2021) is preferable for lead times over 90 minutes.

![Graph showing performance of DGMR using different loss functions](image)

**Fig. 4:** Performance of DGMR using different loss functions, measured with (a) mean absolute error and (b) mean squared error in (mm/h)^2.

Table 4 and Figure 5 show the CSI scores of the two models for different losses and thresholds of precipitation. Higher CSI scores indicate better performance, while the number of false negatives and false positives should be as low as possible. The DGMR-BalancedExtended performs best, followed by DGMR-Balanced; this is the case for all rainfall intensities. The AENN GAN-RMSE performs better than the AENN GAN-Hinge, but it should be noticed that the differences between the RMSE loss and the Hinge loss are often minimal in terms of the CSI and F1 (Tables 4 and 5).

High rainfall events are detected more often, as confirmed by the higher value of the POD (Table 6), when using the extended balanced loss. Moreover, in the case of the extended balanced loss, there are also less false alarms for the highest rainfall intensity events, looking at the FAR (Table 7).

Overall, the models using the extended balanced loss perform better in terms of the CSI, F1 and POD. Nevertheless, these models generally show a higher FAR, while a lower score would be preferred. Using the DGMR with the balanced loss shows the second-best results for the CSI, F1 and POD and the best result for the FAR differs per threshold.
Fig. 5: Performance diagram for AENN GAN and DGMR, using different extensions and rainfall intensity thresholds. The diagonal represents the bias ratio of 1, better CSI values lay close to the top right of the figure. Aggregation over all lead times is used.

A point to notice is the performance of the AENN GAN with the Hinge loss; the CSI and POD results show that this combination performs the worst. The Hinge loss together with the AENN GAN, is making use of the \( \ell_1 \) distance instead of the RMSE. It is suspected that the AENN GAN with the Hinge loss is not confident enough nowcasting larger lead times. Optimising with the Hinge formulation instead of the \( \text{minmax} \) problem could lead to more confident results (Ravuri et al. 2021; Choi and Kim 2021).

c. Temperature as Additional Feature

This subsection evaluates the model performances of the AENN GAN and DGMR when a feature is added (using the default loss functions). The temperature maps are used as additional input features. The nowcasting performances are evaluated using the MSE, MAE and the categorical scores CSI, F1, POD, and FAR.

The overall evaluation results for adding the temperature maps to the input, are summarised in Figures 6a, 6b, Table 4 and Figure 5. The best performance is achieved by the DGMR, using the
stacked vector with the temperature grid as input for the model. For all lead times, adding the temperature feature substantially improves the performance in terms of MAE and MSE.

![Graphs showing performance of DGMR with and without temperature data as additional input](image)

**Fig. 6:** a) Performance of DGMR with(out) temperature data as additional input, measured with (a) mean absolute error and (b) mean squared error in (mm/h)^2.

See Table 4 and Figure 5 for the CSI scores, the scores for a threshold of 1.0 mm/h, 10.0 mm/h and 20 mm/h increase when using the temperature as an additional feature. This increment in the score is the case for both models. The F1 scores for the AENN GAN and DGMR are provided in Table 5, averaged over the leadtimes. Similarly to the CSI score, including temperature as an additional feature improves F1 score over all thresholds.

High rainfall events are detected more often when using the temperature maps as additional feature; this is confirmed by the higher value of the POD (Table 6) when using both features. A significant increase in the POD is seen when using the DGMR with the additional temperature data. The largest increase is seen when using the DGMR with the additional temperature data, for a threshold of 1.0 and 20.0 mm/h. The DGMR with additional temperature data consistently outperforms the models the different loss functions. Looking at the FAR (Table 7), there are less false alarms for the threshold of 20.0 mm/h, when using the temperature maps as additional features.

Overall, the models using the temperature data as additional feature perform generally better regarding the CSI, F1, POD and FAR. Using the AENN GAN with temperature data shows the best result regarding the FAR for high-intensity rainfall (Table 7); in this case, the amount of falsely predicted events is lowest. The addition of temperature data (DGMR) leads to a skill increase.
for all selected rainfall thresholds, with superior performance for lower lead times but an inferior performance for higher lead times. At a threshold of 20.0 mm/h, the DGMR with temperature data shows a CSI that is almost 1.5 times the CSI of the corresponding AENN GAN model.

Finally, we evaluate the impact of the temperature on the prediction. Shapley values are an explanation method for deep learning models, using the Kernel SHAP method from Molnar (2022). Shapley values (Lundberg and Lee 2017) are a mathematical concept derived from cooperative game theory. In the context of machine learning, Shapley values quantify the impact of each feature on the prediction outcome of a model. They provide a fair and interpretable way to attribute importance to features by considering their marginal contributions within all possible feature combinations. In our study, we employ Shapley values to assess the individual contribution of the temperature feature in the precipitation nowcasting process. Positive Shapley values indicate a feature’s positive impact on precipitation prediction, while negative values indicate the opposite. The magnitude of the value signifies the strength of the feature’s influence. In Figure 7, we can see the dependency plot of the rainfall intensities from the DGMR using the temperature as additional feature. The figure shows two distinct regimes: one with a smaller slope between the SHAP values and the precipitation intensities for lower temperatures and one with a larger slope for higher temperatures. This can be explained by the mostly (non-) convective nature of precipitation in lower temperatures.

Fig. 7: Dependency plot between SHAP values and rainfall intensities, showing the interaction with the (air) temperature (colour scale). Blue colours indicate lower temperature values and orange colours indicate higher temperature values.
summer (winter) (Xu 2013). Using this information, adding new weather-related features to the models can give more insight into the underlying relations and an improved precipitation nowcast.

d. Combination of extensions

We analyse the radar nowcast images from the DGMR with extensions to illustrate differences in nowcasting. Figure 8 presents the model behaviour and the differences of nowcasts using the different extensions for a rainy event at $t_0 = 2021-05-01 11:05:00$, with a lead time of 30 minutes. All variations in the models capture similar motion characteristics of the rain. See Figure 9 for the model behaviour of the DGMR and the DGMR using both extensions, with lead times of 5, 10, 15, 30, 60 and 90 minutes. The DGMR is able to learn more from the temperature data for larger lead times. Looking at areas with high rainfall intensities, we can see that the extended balanced loss increases the intensities in rain fields. Using the temperature data improves the high rainfall intensities and seems to improve the motion and evolution of precipitation.

The bias measures the ratio of the frequency of forecast events to the frequency of observed events. It indicates whether the forecast system has a tendency to underestimate or overestimate events. In Figure 5, according to the diagonal, the AENN GAN - without extensions for 1 mm/h - has the best tendency to estimate the nowcasting (ideal bias of 1). The figure also shows that for all 3 intensities, the DGMR with temperature data (i.e., triangle left) has higher CSI scores (i.e., blue contours) than the other models. For the highest threshold the second best is the DGMR with extended balanced loss. The extensions to the DGMR (extended balanced loss and temperature)

Fig. 8: Nowcast and target value for different models, for a rainy event, initialized at $t_0 = 2021-05-01 11:05:00$, in mm/h.
Fig. 9: Nowcast and target value for different models, for a rainy event, initialized at $t_0 = 2021-05-01\ 11:05:00$, in mm/h.

gradually improve the CSI by increasing the probability of detection (y-axis), while generally preserving the performance for the success ratio (x-axis).

Next, results for running the generative networks with both extensions in the same experiment are described. Results of both models are compared with the results of the generative networks without any extensions, the generative networks with different loss functions, and the generative networks using the temperature data as an additional feature.

Incorporating the extended balanced loss helps to put more importance on high rainfall events. Including the physical features, and temperature data, helps to improve the nowcasting by using the underlying relations between temperature and precipitation. The CSI, F1 and POD of the DGMR with both extensions show improvements compared to the DGMR and the DGMR with single extensions (Tables 4-7). For the shorter lead times, the improvement is the most significant.

A point to notice is the increase in CSI, F1 and POD when using the combination of extensions for the AENN GAN model. When using the combination, the AENN GAN performs (slightly) better than the standard DGMR. Looking at the AENN GAN for the threshold of 20 mm/h, CSI shows a more than four-fold increase when using extended balanced loss compared to the AENN GAN without any extensions. Using the combination of extensions, shows an additional increase of
CSI (AENN GAN) | CSI(DGMR)
---|---
Radar data | 1 mm/h | 10 mm/h | 20 mm/h | 1 mm/h | 10 mm/h | 20 mm/h
+ RMSE Loss | 0.381 | 0.115 | 0.031 | 0.487 | 0.217 | 0.135
+ Hinge Loss ($\ell_1$) | 0.374 | 0.114 | 0.028 | 0.492 | 0.228 | 0.142
+ Balanced Loss ($\ell_1$) | 0.469 | 0.200 | 0.099 | 0.497 | 0.230 | 0.178
+ BalancedExtended Loss ($\ell_1 + \ell_2$) | 0.485 | 0.221 | 0.136 | 0.499 | 0.231 | 0.191
+ Temperature data | 0.460 | 0.195 | 0.135 | 0.544 | 0.301 | 0.197
+ BalancedExtended + Temperature data | 0.506 | 0.233 | 0.161 | **0.599** | **0.346** | **0.207**

Table 4: Critical Success Index (CSI) for AENN GAN and DGMR using different extensions. The best results (highest values) are indicated in bold.

F1 (AENN GAN) | F1 (DGMR)
---|---
Radar data | 1 mm/h | 10 mm/h | 20 mm/h | 1 mm/h | 10 mm/h | 20 mm/h
+ RMSE Loss | 0.464 | 0.140 | 0.038 | 0.592 | 0.264 | 0.164
+ Hinge Loss ($\ell_1$) | 0.455 | 0.139 | 0.034 | 0.598 | 0.277 | 0.173
+ Balanced Loss ($\ell_1$) | 0.571 | 0.244 | 0.121 | 0.604 | 0.280 | 0.217
+ BalancedExtended Loss ($\ell_1 + \ell_2$) | 0.590 | 0.269 | 0.166 | 0.606 | 0.281 | 0.232
+ Temperature data | 0.560 | 0.238 | 0.164 | 0.661 | 0.366 | 0.240
+ BalancedExtended + Temperature data | 0.616 | 0.284 | 0.196 | **0.728** | **0.421** | **0.252**

Table 5: F1 score for AENN GAN and DGMR using different extensions. The best results (highest values) are indicated in bold.

POD (AENN GAN) | POD(DGMR)
---|---
Radar data | 1 mm/h | 10 mm/h | 20 mm/h | 1 mm/h | 10 mm/h | 20 mm/h
+ RMSE Loss | 0.626 | 0.472 | 0.258 | 0.653 | 0.411 | 0.201
+ Hinge Loss ($\ell_1$) | 0.605 | 0.369 | 0.149 | 0.792 | 0.543 | 0.289
+ Balanced Loss ($\ell_1$) | 0.691 | 0.507 | 0.239 | 0.798 | 0.582 | 0.405
+ BalancedExtended Loss ($\ell_1 + \ell_2$) | 0.731 | 0.559 | 0.290 | 0.829 | 0.619 | 0.478
+ Temperature data | 0.795 | 0.618 | 0.370 | 0.904 | 0.620 | 0.488
+ BalancedExtended + Temperature data | 0.830 | 0.622 | 0.418 | **0.920** | **0.673** | **0.526**

Table 6: Probability of Detection (POD) for AENN GAN and DGMR using different extensions. The best results (highest values) are indicated in bold. Aggregation over all lead times is used.

around 18%. For the AENN GAN there is an increase in the F1 score when using the combination of extensions, for all rainfall rates. From the POD scores, it appears that for higher rainfall (i.e., 20 mm/h) there is a substantial increase in the probability of detection. The FAR score indicates the lowest number of high rainfall events falsely predicted when using the AENN GAN with the additional temperature data. However, the mixture of extensions is responsible for a slight increase in the FAR, but still lower compared to only using the extended balanced loss.
Looking at the DGMR for the threshold of 20 mm/h, CSI shows an increase of around 35% when using extended balanced loss compared to the DGMR without any extensions. Using the combination of extensions, shows an additional increase of around 8%. The DGMR with temperature data leads to a decrease of around 60% in terms of FAR for the threshold of 20 mm/h, and a decrease of around 55% when using the combination of extensions. The decrease in the FAR is related to less falsely predicted high rainfall events.

4. Conclusion

In this study we have compared several methods for nowcasting precipitation with a primary focus on higher rainfall intensities. In addition, we have proposed extensions to improve the performance of the deep learning approaches.

Using radar data from KNMI, deep learning models have been implemented for precipitation nowcasting in the Netherlands. The DGMR previously proposed in Ravuri et al. (2021), was compared to the AENN GAN and other benchmark models (i.e., S-PROG, UNet and MetNet). Precipitation has been forecast for 5, 10, 15, 30, 60, and 90 minutes lead times. S-PROG was chosen as the benchmark of an extrapolation-based method. AENN GAN and DGMR were both able to outperform S-PROG. Although AENN GAN beats S-PROG, AENN GAN is not skilful at predicting precipitation at thresholds 10 and 20 mm/h, for lead times of 15 minutes and longer. During the benchmark comparison, we have found that DGMR performs best for shorter lead times (i.e., below 30 minutes), while MetNet and AENN GAN perform best for longer lead times (i.e., above 30 minutes). Neither of the models, with the default loss function, do well for high rainfall intensities at longer lead times, but DGMR performs the best. We further proposed two extensions

| Far Alarm Ratio (FAR) for AENN GAN and DGMR using different extensions. The best results (lowest values) are indicated in bold. Aggregation over all lead times is used. |
|-------------------------------------------------|---------------------------------|---------------------------------|---------------------------------|
| Radar data                                      | 1 mm/h                         | 10 mm/h                        | 20 mm/h                        |
| + RMSE Loss                                     | 0.378                          | 0.402                          | 0.128                          |
| + Hinge Loss ($\ell_1$)                        | 0.394                          | 0.391                          | 0.319                          |
| + Balanced Loss ($\ell_1$)                      | 0.392                          | 0.390                          | 0.289                          |
| + Balanced Exended Loss ($\ell_1 + \ell_2$)    | 0.431                          | 0.589                          | 0.299                          |
| + Temperature data                              | 0.287                          | 0.340                          | 0.125                          |
| + Balanced Exended + Temperature data           | 0.306                          | 0.359                          | 0.191                          |

25

Accepted for publication in Artificial Intelligence for the Earth Systems. DOI 10.1175/AIES-D-23-0017.1.
to improve the performance of the generative models: the extended balanced loss function and
the addition of temperature data as a feature. Incorporating the extended balanced loss helps to
put more importance on high rainfall events. Including the physical features, and temperature
data, helps to improve the nowcasting by using the underlying relations between temperature and
precipitation.

Firstly, the model performance of DGMR with the proposed extended balanced loss function
has been evaluated against RMSE loss, Hinge loss and balanced loss. The extended balanced
loss substantially impacts the performance of both AENN GAN and DGMR. We found that the
performance of DGMR with the extended balanced loss function improves, particularly for shorter
lead times (i.e., below 30 minutes), which can be relevant for short-term forecasting systems. The
extended balanced loss increased the CSI, F1, POD, and the skilful lead times. However, there was
an increase in false alarms (i.e., FAR). AENN GAN with the extended balanced loss, leads to an
increase in performance for all selected rainfall thresholds.

Secondly, temperature data was included as an additional feature in the DGMR and AENN GAN
models. Including this additional feature further improves the DGMR performance, especially for
the CSI, F1 and POD. Additionally, the temperature feature leads to a skill increase for all thresholds.
Especially for the highest threshold (i.e., 20 mm/h), the AENN GAN improves substantially when
including the additional feature.

Using the combination of extensions leads to an overall improvement in performance for both
the AENN GAN and the DGMR models. The DGMR using the combination of extensions leads to
the best-achieved results, the prominent advantage is achieved for all rainfall intensity thresholds.

In all cases, with and without extensions, DGMR outperforms the other models. The DGMR,
especially with the extensions, is skilful at nowcasting the precipitation for high rainfall intensities,
up until 60 minutes. Neither the deep learning models nor the S-PROG model were skilful at
predicting high rainfall intensities for larger lead times (i.e., above 60 minutes). The Shapley
values confirm the distinct relationship between temperature and precipitation in summer versus
winter due to (deep) convection in summer. The DGMR could, in the future, be extended with
more weather features, for instance, the wind speed (Bouget et al. 2020).

Future work for (probabilistic) precipitation nowcasting of high-intensity events could explore:
1) using GANs for the generation of high-intensity events; 2) incorporating more weather-related
features; 3) the uncertainty of the loss functions. Furthermore, transfer learning could be considered to study the generalisation of the proposed method to regions with different rainfall characteristics.
Data availability statement. Real-time precipitation radar data was used from 2008 till 2021. The 1.5-meter air temperature data is retrieved from 45 automatic weather stations. The data from 2018 onwards is publicly available on the KNMI data platform (https://dataplatform.knmi.nl). This data set was used to train and validate the model.

APPENDIX

APPENDIX A

Robustness of the results to the choice of the test set

This paper used data from the year 2021 as test data for verifying the different models. The year 2021 contained more rainfall compared to some previous years. Especially, the months of July, August, and November captured most of the high-intensity events of that year. While the number of rainfall days was similar for 2021 compared to other years, the number of days with high-intensity precipitation was higher. Thus, using the extended balanced loss and 2021 as a test set can lead to biasing the results towards the high-intensity events. For that reason, the DGMR experiments are performed with two different test sets, one for the part of 2021 with more high-intensity events (July, August, October, and November; ”Test set - Heavy”) and the other set containing the remaining months (”Test set - Light”).

<table>
<thead>
<tr>
<th></th>
<th>CSI (&quot;Test set - Heavy&quot;)</th>
<th>CSI (&quot;Test set - Light&quot;)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 mm/h</td>
<td>10 mm/h</td>
</tr>
<tr>
<td>Radar data</td>
<td>0.489</td>
<td>0.227</td>
</tr>
<tr>
<td>+ Balanced</td>
<td>0.489</td>
<td>0.230</td>
</tr>
</tbody>
</table>

Table A1: Critical Success Index (CSI) for DGMR using different extensions and test sets. Aggregation over all lead times is used.

<table>
<thead>
<tr>
<th></th>
<th>POD (&quot;Test set - Heavy&quot;)</th>
<th>POD (&quot;Test set - Light&quot;)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 mm/h</td>
<td>10 mm/h</td>
</tr>
<tr>
<td>Radar data</td>
<td>0.791</td>
<td>0.543</td>
</tr>
<tr>
<td>+ Balanced</td>
<td>0.821</td>
<td>0.616</td>
</tr>
</tbody>
</table>

Table A2: Probability of Detection (POD) for DGMR using different extensions and test sets. Aggregation over all lead times is used.
Table A3: False Alarm Ratio (FAR) for DGMR using different extensions and test sets. Aggregation over all lead times is used.

Increasing the number of high intensity events in the test data, increases the number of high rainfall intensities that are correctly predicted (see Table A1, A2, and A3). Overall, the DGMR using "Test set - Heavy" performs better in terms of CSI and POD, but the models show also a higher FAR (which should be lower for better performance), especially for the threshold of 20 mm/h and using the extended balanced loss.

APPENDIX B

Training

Models were trained and evaluated on the 7 GPU nodes. See Table B1 for an overview of the training times for the deep learning models (without subset extractions).

<table>
<thead>
<tr>
<th></th>
<th>AENN GAN</th>
<th>GAN</th>
<th>DGMR</th>
<th>UNet</th>
<th>MetNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPU (hours)</td>
<td>67.2 (~ 2.8 days)</td>
<td>124.8 (~ 5.2 days)</td>
<td>163.5 (~ 6.8 days)</td>
<td>212.2 (~ 8.8 days)</td>
<td></td>
</tr>
</tbody>
</table>

Table B1: Training time of AENN GAN and DGMR, using 7 GPUs (time in hours).

Hyperparameters settings are optimised using grid search. Table B2 presents the hyperparameter setting of the DGMR used in Ravuri et al. (2021) and the setting used in this paper after performing a grid search. The optimal chosen hyperparameters set is the set that could gain the highest performance on the validation set.
Table B2: Hyperparameter settings for DGMR with both extensions, for the default case (DGMR, Ravuri et al. (2021)) and settings used after applying grid search.

**APPENDIX C**

**Comparison of loss functions**

Figure C1 illustrates the generator, discriminator and reconstruction losses of the DGMR (with both extensions). We observe that the training stabilises around 125 epochs.

![Graph showing generator, discriminator, and reconstruction losses over epochs](image)

**Fig. C1**: Performance of DGMR with both extensions, measured with three different loss functions.
APPENDIX D

Prediction frames

Figures D1 and D2 present additional examples of prediction frames.

Fig. D1: Nowcast and target value for different models, for a rainy event, initialized at $t_0 = 2021-02-16$ 19:00:00, in mm/h.

Fig. D2: Nowcast and target value for different models, for a rainy event, initialized at $t_0 = 2021-02-25$ 10:00:00, in mm/h.
References


32

Accepted for publication in Artificial Intelligence for the Earth Systems. DOI 10.1175/AIES-D-23-0017.1.


Ingram, K., M. Roncoli, and P. Kirshen, 2002: Opportunities and constraints for farmers of West Africa to use seasonal precipitation forecasts with Burkina Faso as a case study. *Agricultural systems*, 74 (3), 331–349.


Kulesa, G., 2003: Weather and aviation: How does weather affect the safety and operations of airports and aviation, and how does FAA work to manage weather-related effects? *The Potential Impacts of Climate Change on Transportation*.


