MSc Artificial Intelligence
Master Thesis

Challenging the Convention: Predicting Rainfall in Three Dimensions

by
Jochem Soons
11327030

April 28, 2023

48 EC
November 2021 - April 2023

Supervisor:
Dr. Pascal Mettes

External supervisors:
Daan Odijk, PhD.
Vincent Koops, PhD.

Examiner:
Rob Romijnders



Universiteit van Amsterdam

Abstract

Precipitation nowcasting, which involves the complex spatio-temporal predictive task of forecasting rainfall over a short period of time, is essential for weather-dependent decision-making. Recently, deep learning-based approaches trained on historical radar data have emerged as a promising alternative to traditional nowcasting methods. However, most of these approaches only predict radar data mapped to a 2D grid at a fixed altitude and disregard vertical dynamics of precipitation. In contrast, applying nowcasting techniques to volumetric radar data has been less explored.

This thesis aims to investigate whether conventional 2D nowcasting can effectively be extended to 3D radar space. To achieve this objective, we construct a dataset of multi-altitude radar data in the Netherlands and propose a novel 3D nowcasting architecture. Our approach integrates recent advancements in the field and is designed to explicitly model 3D radar data across multiple spatial scales while retaining all three spatial dimensions. Our experimental results show the effectiveness of the proposed approach in utilizing volumetric radar data for 3D nowcasting, reflected by our model outperforming various competitive baselines. Moreover, our results indicate that exploiting 3D radar data can help to improve the skill of predicting single altitude radar images, compared to the current standard of using only 2D radar data.

Overall, our findings strongly support the hypothesis that nowcasting techniques can benefit from considering the volumetric aspect of radar data, and highlight the potential of deep learning-based approaches for the 3D nowcasting task.

Acknowledgements

This thesis project was performed during an internship at the data science team of RTL in collaboration with Buienradar. I would like to thank Daan Odijk and Vincent Koops for offering me this opportunity and providing me with invaluable guidance throughout my research, as well as the rest of the data scientists at RTL for welcoming me into the team and creating a supportive atmosphere. Additionally, I would like to extend my gratitude to Sam Koch and the Buienradar team for offering me many valuable insights into the meteorological perspective of my research. It surprised me that discussing the weather can quickly turn into a lengthy and engaging conversation rather than just superficial chit-chat.

I am also immensely grateful to my internal supervisor at the UvA, Pascal Mettes, for always making time to help me navigate the various challenges I encountered during my research. Finally, I would also like to take the opportunity to thank my family for their continuous support throughout my studies, and the friends I made along the way for making it an enjoyable chapter of my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Precipitation nowcasting, which involves high-resolution forecasting of rainfall and hydrometeors in a local region over a relatively short period of time (i.e. 0-2 hours into the future), is crucial for weather-dependent decision-making [34, 30]. Moreover, such short-term weather predictions are not only vital to a vast majority of the human population in planning daily activities [35], but they also influence operations including agriculture, energy optimization, flood early-warning systems, and aviation management [34, 30]. In addition, due to climate change, there is an increased likelihood that severe weather events, such as convective storms and floods, will occur more frequently, underscoring the urgent requirement for reliable short-term weather forecasting [15, 3].

Recently, applications of supervised deep learning (DL) techniques to radar data have emerged as a promising alternative to traditional nowcasting approaches that suffer from numerous limitations: they are computationally expensive, unable to exploit big data, reliant on expert knowledge and most importantly, they are unable to capture the complex non-linear spatio-temporal nature of precipitation [6, 29]. Unlike traditional forecasting methods, DL-based approaches are able to recognize complex patterns and uncover non-linear relations in a data-driven way [9]. They scale well with data: by training on bigger, higher quality, and more recent radar data, the skill of the forecasts can improve [18].

Significant progress has been made in developing DL-based models for nowcasting. However, most of those works aim to predict radar data mapped to a two-dimensional grid at a fixed altitude, while radar data is three-dimensional in nature. Moreover, this volumetric nature of radar data can provide a wealth of information beyond what can be obtained from 2D radar images captured at a single altitude. By utilizing 3D radar data, it is possible to better capture the spatial and temporal variability of precipitation, including the vertical structure of the atmosphere and the distribution of hydrometeors within a cloud [22, 28]. Therefore, exploring the use of 3D radar data in nowcasting has the potential to improve the accuracy, informativeness and reliability of forecasts [39].

To the best of our knowledge, there has been limited research on applying DL-based nowcasting models to 3D radar data. In this thesis, we focus on bridging this gap in the literature by combining recent advancements of DL-based nowcasting for the volumetric nowcasting task. We define this volumetric nowcasting task as a spatio-temporal sequence-to-sequence prediction problem in which both the input and the prediction target are temporal sequences of three-dimensional radar data, i.e. radar volumes. To perform this task, we propose a novel 3D nowcasting framework that predicts a sequence of future radar volumes given a sequence of past atmospheric radar volumes.

Concretely, we seek to answer two main research questions in this thesis:

RQ1: Can the conventional 2D nowcasting task effectively be extended to 3D radar data space by deploying a deep learning-based model that is tasked to create 3D predictions given 3D input?

RQ1.1: Are performance benefits noticeable when the vertical dimension of data is explicitly treated as a third spatial dimension within our model?

RQ1.2: Are performance benefits noticeable when our model is designed to capture spatio-temporal precipitation dynamics on multiple spatial scales?

RQ2: Can conventional 2D nowcasting predictions be improved when our model considers 3D radar data?

By designing and implementing a methodology for answering these questions and subquestions, we state the primary contributions of this thesis as the following:

1. We extend the conventional 2D nowcasting problem to 3D space by setting up a data processing pipeline to create a new volumetric radar dataset of precipitation events in the Netherlands.
2. We propose a novel end-to-end trainable deep learning-based model for the 3D nowcasting task, which adopts and integrates recent advancements in the research field. Moreover, our approach is designed to explicitly model 3D radar data in three dimensions over multiple spatial scales.
3. Quantitative and qualitative experiments were performed to extensively analyze the performance of our model in the volumetric nowcasting task using a variety of evaluation metrics. Furthermore, results are analyzed for different optimization objectives and against multiple competitive baselines. The experimental results strongly support the effectiveness of our proposed approach in utilizing 3D radar data for nowcasting.

The rest of the thesis is subdivided into five chapters. In chapter 2, background information regarding radar data and related work in nowcasting is provided. In chapter 3, the methodology of our proposed approach is outlined. In chapter 4, technical details concerning our experimental setup are discussed. In chapter 5, the results of our experiments are presented. Finally, in chapter 6, the most important findings of this thesis are summarized. Here, we also discuss the limitations of our work and provide some recommendations for future research.

Chapter 2

Background and related work

In this chapter, essential background information regarding radar data and the commonly used products resulting therefrom is provided in section 2.1. Subsequently, previous work in nowcasting research is discussed in sections 2.2 and 2.3. Finally, a few existing approaches in volumetric nowcasting are discussed in section 2.4.

2.1 Radar data

The Royal Netherlands Meteorological Institute (KNMI) operates two active radar stations in the Netherlands, located in Den Helder and Herwijnen. These radar systems emit high-frequency electromagnetic pulses into the atmosphere, interacting with hydrometeors such as raindrops, snowflakes, and hailstones. This interaction causes the radar signal to be scattered, with some of the energy reflected back to the radar. By measuring the time delay and intensity of the returned signal compared to the emitted signal, the radar station can construct a map of location-specific precipitation measurements [41]. An important measurement in precipitation analysis through radar maps is the reflectivity factor (Z), which indicates the strength of the electromagnetic signal reflected back to the radar. The unit Z is proportional to the number of raindrops per m^2 atmosphere and is defined as follows:

$$Z = \sum_i n_i \cdot D_i^6 \tag{2.1}$$

Where n_i is the number of particles per unit volume with diameter D_i, meaning that a higher reflectivity factor Z indicates more raindrops per unit volume. A commonly used derivative measurement of Z is dBZ, which stands for decibels relative to Z. Dividing Z by the equivalent return of a 1 mm drop in a volume of a meter cube (the reference standard Z_0), and taking the logarithm of the result (because the values vary significantly across different precipitation types), yields the logarithmic reflectivity L_Z, in dBZ:

$$L_z = 10 \log_{10} \frac{Z}{Z_0} \text{dBZ} \tag{2.2}$$

At a fixed altitude, dBZ values can be converted to rainfall rates (R) in millimetres per hour using the Marshall-Palmer formula [20]:

$$\text{R} = \left(\frac{10^{(dBZ/10)}}{a}\right)^b \text{mm/h} \tag{2.3}$$

L_Z (dBZ)	R (mm/h)	Intensity
7	0.01	Light mist
15	0.3	Mist
20	0.6	Very light rain
25	1.3	Light rain
30	2.7	Light to moderate rain
35	5.6	Moderate rain
40	11.5	Moderate to heavy rain
45	23.7	heavy rain
50	48.6	Very heavy rain
55	100	Extreme rain / hail

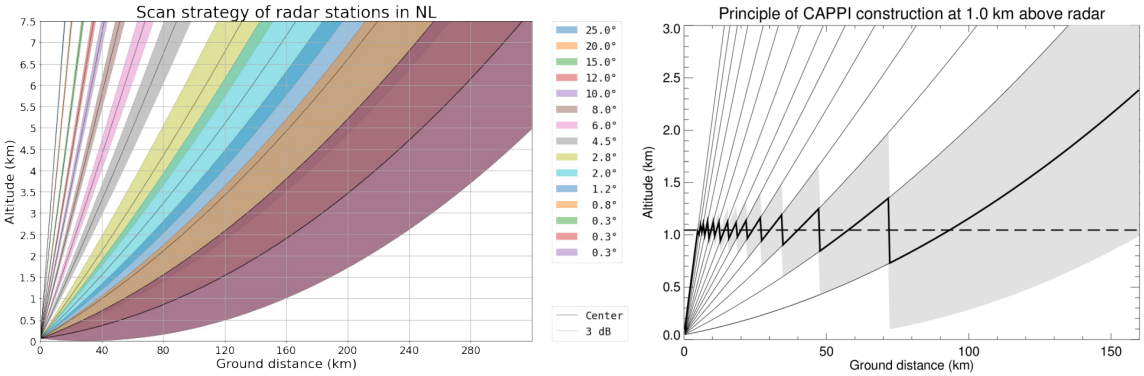
Table 2.1: Reflectivity in dBZ versus rain rate R, conversed using the Marshall Palmer formula described in equation 2.3.

Where typically, $a = 200$ and $b = 1.6$ [41]. Table 2.1 provides an overview of dBZ values with their corresponding rainfall intensities.

2.1.1 CAPPI, multiCAPPI and pseudoCAPPI

The two radar stations in the Netherlands emit pulses at 360 degrees at 15 discrete elevation angles, ranging from 0.3 to 25 degrees to the earth's surface. Figure 2.1a shows the corresponding atmospheric radar scan strategy. A full set of scans takes 5 minutes to complete, and the maximum range of both radars is 320 km. Scanning the atmosphere at different surface elevation angles produces a point cloud of reflectivity intensities. Figure 2.2 provides an example of such raw radar data. Typically, this volumetric radar data are mapped to a Cartesian grid and converted to an image at a single altitude by linearly interpolating reflectivity values in altitude, resulting in a 2D reflectivity image called a Constant Altitude Planning Position Indicator (CAPPI). Constructing a CAPPI image is thus essentially done by drawing a horizontal cross-section of fixed altitude through the scan data and interpolating values from the nearest available scan. Figure 2.1b provides a schematic overview of this principle. If multiple CAPPI images at different altitudes are stacked on each other, the resulting radar volume is called a multiCAPPI [25].

Within the global meteorology community, it is standard practice to create a two-dimensional image of reflectivity for precipitation nowcasting and visualization purposes at a relatively low altitude (e.g. 1500 meters above the ground). Choosing an even lower altitude is inconvenient because more clutter and noise disturb the signal (e.g. because of terrain interacting with the radar beams), while it can be challenging to translate hydrometeors at higher altitudes to surface rainfall (e.g. because of horizontal displacement or evaporation). However, at longer distances from the radar (exceeding 100km), even the lowest elevation angle exceeds the representative height of the CAPPI, which leads to blind spots. To combat this, meteorologists often use a pseudoCAPPI, which is a two-dimensional reflectivity image that follows the principle of constructing a CAPPI. However, when reaching blind spots, missing data is completed with data from the lowest available elevation scan above the CAPPI height [13]. A pseudoCAPPI can visualize parts within the radar range that cannot be measured directly, but not all data in the final product comes from the fixed CAPPI height and may be quite higher in actual height [13]



(a) Elevation angles of radar scans. (b) CAPPI construction process [41].

Figure 2.1: Scan strategy and CAPPI construction principle.

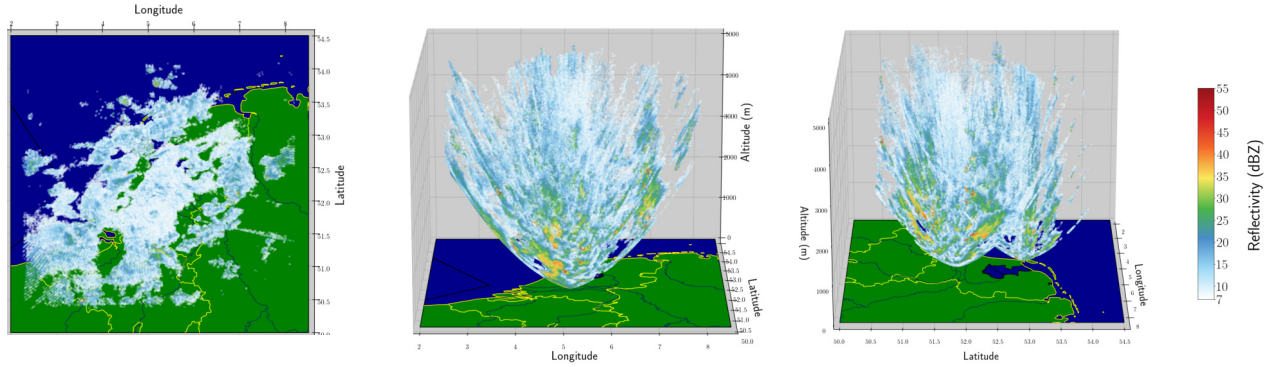


Figure 2.2: Example of raw radar data measured by KNMI-operated radar stations in Den Helder and Herwijnen. The time of the event is 28/10/2020, 16:10. Viewed from three different angles: top view (left), sidewards view facing north (middle), and sidewards view facing west (right). A curve-like shape in the data can be observed, corresponding to the discrete elevation angles in figure 2.1a.

2.2 Nowcasting: conventional approaches

The nowcasting task is similar to other spatio-temporal sequence-to-sequence prediction tasks, such as video prediction. However, notable characteristics set it apart from related problems; it requires dense pixel-wise regression, the ability to handle underlying structures across spatial and temporal scales, and the ability to manage imbalanced data and extreme events [26]. Additionally, the task involves dealing with regime changes both rapid, such as convective storm initialization, and more generally, such as differences between the seasonal distribution of precipitation, as an underlying necessity [26, 14].

Existing nowcasting techniques can roughly be divided into two categories, namely, Numerical Weather Prediction (NWP) models and Radar Echo Extrapolation (REE) methods [34]. NWP models are the traditional method for weather forecasting and involve physical models that simulate the dynamics of the atmosphere and precipitation processes [4]. However, in the context of nowcasting, these NWP models are computationally expensive, too sensitive to noises, unable to exploit big data, and highly reliant on expert knowledge and initial conditions [11, 38, 51, 33].

Therefore, current state-of-the-art operating nowcasting systems often adopt the faster and more reliable REE method that relies on extrapolating the movement and evolution of radar

data sequences observed in the atmosphere [49, 30, 29, 6]. These optical flow-based models estimate convective cloud movements from pseudoCAPPI radar images and predict future radar images using semi-Lagrangian advection. Popular examples of such methods include Tracking Radar Echoes by Correlation (TREC) [31], Real-time Optical flow by Variational methods for Echoes of Radar (ROVER) [50] and the open-source python framework pySTEPS [27]. Although these models are widely used in nowcasting due to their simplicity and computational efficiency, they suffer from significant limitations, such as the assumption of coherent motion and the inability to capture the complex non-linear spatio-temporal variability of precipitation [6, 29].

2.3 Nowcasting: deep learning approaches

After the successes of deep learning models in various tasks within the computer vision domain, such as image recognition, image segmentation, and video prediction, more research on applying DL-based techniques to the field of radar-based nowcasting followed with the hope of solving limitations of traditional approaches [26]. Prior works can broadly be divided into two categories: (1) Convolutional Neural Network (CNN)-based approaches and (2) Recurrent Neural Network (RNN)-based approaches. Both approaches use convolutions, which are kernel-based operations that slide over images to learn spatial features in data. Moreover, convolutions can be implemented in 2D space and 3D space. We refer readers interested in more details regarding the mathematical definitions of convolutional operations and differences in 2D and 3D space to [2, 37].

2.3.1 CNN-based approach: U-Net

CNN-based approaches for nowcasting typically employ the U-Net structure, which is an encoder-decoder CNN originally proposed for biomedical segmentation tasks [32]. The encoder part of the network performs downsampling pooling operations, while the decoder part performs upsampling convolution operations to reconstruct a dense output map. The network is designed to preserve spatial information using skip connections that connect the encoder and decoder at multiple spatial levels.

Moreover, [1] adapted the U-Net architecture for precipitation nowcasting by representing each radar image in the sequence as a different image channel in the network. By modelling the temporal aspect of the data in the channel dimension, the U-Net can capture both spatial and temporal features using two-dimensional convolutions [1]. However, instead of treating multiple time steps as different channels, time can also be treated as a depth dimension. This leads to the possibility of applying 3D-CNN architectures to the spatio-temporal sequence prediction task of nowcasting [42, 35, 43, 7].

2.3.2 RNN-based approach: Convolutional LSTM

Recurrent neural networks (RNNs) are a commonly used approach for temporal modelling as they are explicitly designed to model an evolving state over time [26]. This is done by iteratively using the current network state as input for the next timestep, which ensures that future states are predicted based on past information. To further enhance the capabilities of RNNs in learning long-range temporal dependencies, RNN-based variants such as long-short term memory (LSTM) [12] and gated recurrent unit (GRU) [8] have been developed that incorporate gating structures, so that information can be preserved over a longer duration [26].

An essential step in the field of nowcasting research was the work of [34], who extended the fully-connected LSTM framework to a Convolutional LSTM (ConvLSTM) by replacing internal matrix multiplications with 2D convolutions in both the input-to-state and state-to-state transitions. Through convolutional operations, the ConvLSTM cell retains spatial structure in the latent representation, i.e. the hidden state $\mathcal{H}$ and cell state $\mathcal{C}$ are spatial features of the same dimensionality as the input instead of flattened feature vectors in a regular LSTM cell. This allows the retention of spatial information while still modelling temporal dependencies using the underlying LSTM framework [26]. Typically, multiple ConvLSTM cells are stacked in layers so that a general predictive framework is formed that can iteratively produce output samples based on a sequence of input samples, as shown in the left side of figure 2.3.

2.3.3 TrajGRU and PredRNN

After the introduction of the ConvLSTM, numerous ConvRNN-based approaches have emerged and led multiple spatio-temporal modelling tasks in the deep learning domain, such as video frame prediction, medical image analytics, and nowcasting [39]. [35] introduced the Trajectory GRU (TrajGRU) model, which extends the ConvLSTM model by having spatial trajectories of data directly encoded into the network, ensuring that the model can actively learn location-variant motion patterns. Additionally, the authors proposed to solve the data imbalance problem in nowcasting by using weighted pixel-wise loss functions that assign heavier weights to higher precipitation values. The authors note that using weighted loss functions can help to create less blurry forecasts, which is a typical problem that ConvLSTM models suffer from when they are trained on standard pixel-wise regression loss functions [35, 38, 30]. Moreover, the work of [38] demonstrates that the blurry image issue can also be combatted by optimizing ConvLSTM-based models not only for pixel-wise loss functions but also for image quality assessment techniques drawn from the computer vision domain, such as structural similarity.

To jointly model spatial and temporal correlations, [45] proposed a new general memory prediction framework called the Predictive RNN (PredRNN). In the PredRNN framework, a spatio-temporal memory state $\mathcal{M}$ is added to the inner ConvLSTM cell that traverses the whole RNN framework via a zigzag path. The PredRNN framework with spatio-temporal memory flow is displayed on the right side of figure 2.3. The inner PredRNN cell that has increased memory capacity is shown in figure 2.4. We refer readers interested in more in-depth information regarding the inner workings of the PredRNN compared to regular ConvLSTM, including mathematical definitions, to [45]. The main takeaway is that by adding spatial-temporal memory flow to the ConvLSTM architecture in a zigzag pattern, PredRNN is able to learn correlations between visual dynamics at different levels of feature abstraction and achieves highly competitive results on various spatio-temporal prediction tasks, including precipitation nowcasting.

More works expand upon the basic ConvLSTM model as introduced by [34]. The authors that introduced PredRNN later proposed improved versions PredRNN-V2 [46] and PredRNN++ [44], while in [47], they extend the PredRNN framework by replacing forget gates in the PredRNN cell with two embedded LSTMs, thereby introducing the Memory in Memory (MIM) framework.

2.3.4 Multi-Scale Convolutional LSTM

Previous works have shown that widening or deepening a network, or adding memory states can increase prediction performance compared to the standard ConvLSTM model [35, 45, 46, 47]. However, this improvement typically comes at the cost of increased memory overhead, which may limit the development and real-time application of these ConvLSTM models because

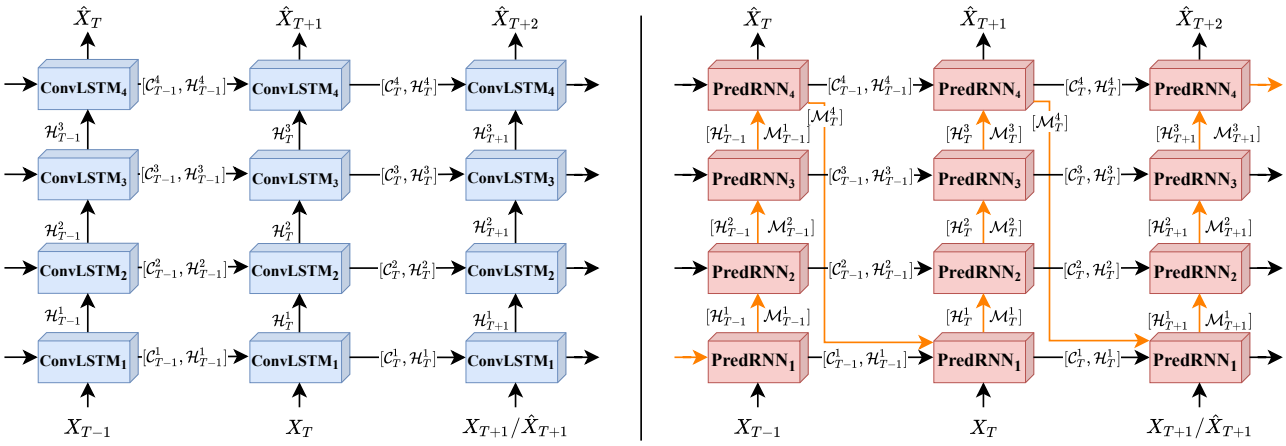


Figure 2.3: **Left:** the original ConvLSTM architecture with hidden state $\mathcal{H}$ and cell state $\mathcal{C}$, originally proposed by [34]. **Right:** the PredRNN architecture proposed by [45]. The orange arrows following the zigzag pattern indicate the transition paths of the newly added spatio-temporal memory state $\mathcal{M}$. At every time step, the PredRNN vertically transmits low-level information from input to output, while at the top layer, memory is passed back to the bottom layer at the next time step.

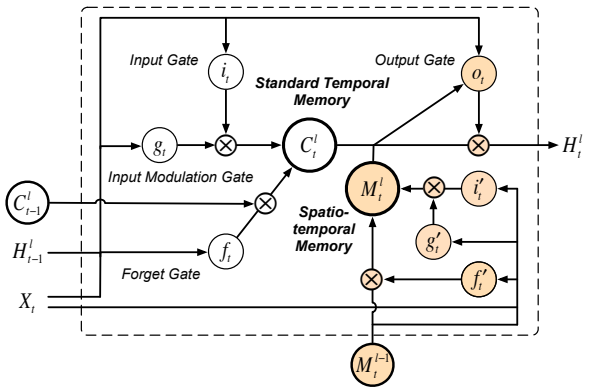


Figure 2.4: The inner PredRNN cell handling the added spatio-temporal memory state $\mathcal{M}$. The $\times$ symbols denote convolutional operations. The orange elements within the cell indicate the additions compared with the conventional ConvLSTM cell. Figure adapted from [46].

memory dedicated to training neural networks is usually limited and precious. For this reason, [19] introduced a flexible Multi-Scale RNN (MS-RNN) framework for ConvLSTM-based models. Moreover, the authors take inspiration from the U-Net architecture by adding downsampling (max pooling) and upsampling (interpolation) operations to construct a symmetric pyramid structure. They also add skip connections that combine low-level and high-level features of the same scale. The general MS-RNN architecture is visualized in figure 2.5.

The authors implement their multi-scale framework with various popular RNN-based models, including ConvLSTM, TrajGRU, and PredRNN. Results on different benchmark datasets indicate that incorporating their framework can improve model performance and reduce memory usage. Furthermore, they notice that by applying convolutions in conjunction with pooling operations, the receptive field of the model is effectively extended, which means that a longer-range spatio-temporal context can be captured compared to a regular ConvLSTM architecture. Lastly, the authors also note that using a multi-scale architecture can be especially useful in radar-based nowcasting, as radar images are typically relatively large in size (e.g. 500x500 pixels), and predictions need to be dense and high-resolution. Therefore, another strategy of simply down-sampling data before feeding it to a ConvLSTM network to reduce memory consumption inevitably leads to a loss of detail in predicted frames, which hurts the model's applicability.

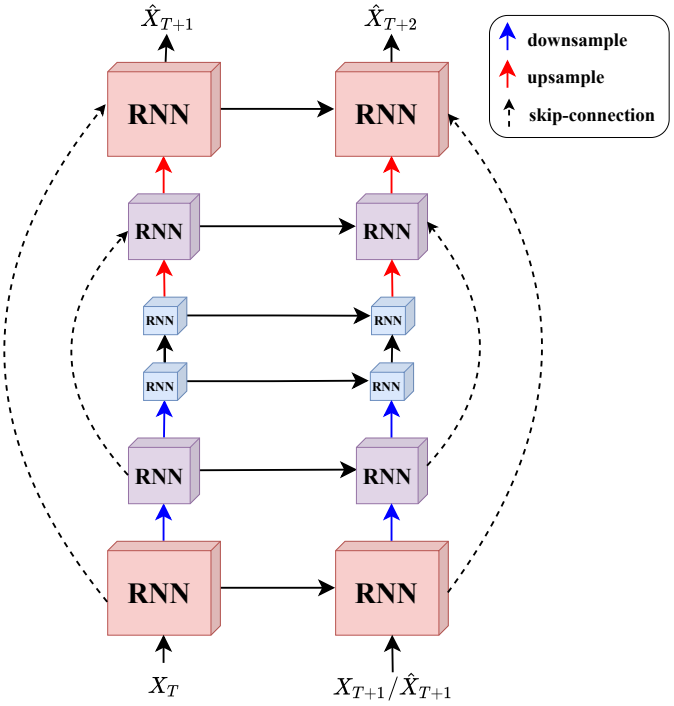


Figure 2.5: The general multi-scale RNN framework proposed by [19], using downsampling, upsampling and skip connections. Note that the architecture can be extended to the PredRNN framework by including the spatio-temporal memory flow.

2.4 Volumetric nowcasting

The different deep learning-based approaches we discussed were all applied to two-dimensional pseudoCAPPI sequences. However, we extend this task to 3D space by considering volumetric radar data. Previous works in which different optical-flow-based methods were applied to 3D radar data showed that they were more accurate in predicting fast-moving patterns, such as storm initialization, when compared to using only 2D data [22, 28]. Although these approaches still needed hand-crafted features and specific domain knowledge, they strongly indicate that including the vertical dynamics of precipitation in radar data can benefit predictions. Surprisingly, given the apparent advantages of using 3D radar data, few works in the literature apply deep learning-based techniques to the volumetric nowcasting task. We will discuss previous and concurrent approaches in the following sections.

2.4.1 Multi-channel approach

[39] implemented RNN-based models on multiCAPPI sequences consisting of 4 stacked CAPPI images at altitudes ranging between 0.5 to 3.5 km. The authors treated the depth dimension of the radar images as an image channel and thus considered the volumetric radar maps as multi-channel images, similar to how coloured images are represented in three image channels (RGB). Moreover, they implemented various popular RNN-based methods, including ConvLSTM, TrajGRU, and PredRNN. Their results indicated that the multi-channel PredRNN performed best in predicting volume sequences 48 minutes into the future. Furthermore, the authors noted that when treating the data as multi-channel, noise occurring at one altitude channel (e.g. radar clutter) affects the prediction quality of other altitude channels. The authors called this the cross-talk effect, and to mitigate this problem, a new output generation block is proposed that generates each channel independently by decomposing convolutional layers to have separate kernels after the first shared convolutional layer.

2.4.2 Towards a 3D approach

Because the cross-talk effect arises when treating volumetric radar data as multi-channel images, treating the vertical aspect of multi-CAPPI data as a third spatial dimension instead of a feature dimension might be beneficial. A logical approach would then be to design models that are able to explicitly extract and combine features represented in this extra spatial dimension, i.e. through the usage of 3D convolutions. To the best of our knowledge, only two related works use a 3D approach in predicting 3D radar data.

First, [25] implement a 3D U-Net model to train on volumetric radar data and compare results when training on two-dimensional radar data. To analyze the impact of adding a vertical dimension, the authors adopt the basic U-Net architecture proposed by [40] and extend the model to volumetric radar data by replacing 2D convolutional layers with 3D convolutions. The authors create two datasets with identical radar timestamps: 1) a 2D dataset consisting of CAPPI reflectivity maps at 2 km altitude, and 2) a 3D dataset consisting of vertically stacked CAPPI reflectivity maps (a multiCAPPI) at eight altitude levels, ranging from 0.5 km altitude to 4 km altitude. Subsequently, they train a 2D U-Net on the 2D dataset to predict a CAPPI image at 2 km and the newly proposed 3D U-Net on the 3D dataset to predict a multiCAPPI, with both models predicting only one future time step, 30 minutes into the future. From their results, the authors conclude that using 3D radar data results in a small but significant reduction in prediction error.

Second, an RNN-based approach can be extended to 3D space by utilizing 3D convolutions and 3D cell and memory states within the ConvLSTM unit. [11] designed such a 3D-ConvLSTM. Although this model could predict 3D radar images, there was no comprehensive analysis of its performance, and it was not compared with other extrapolation models or comparative baseline models that use 2D radar data. However, conducted in parallel with the work in this thesis, [36] implemented a 3D-ConvLSTM model combined with a 3D CNN encoder and 3D CNN decoder on multiCAPPI data. Their data consists of 16 stacked radar maps ranging from 1 to 10 km altitude. Their hybrid CNN-ConvLSTM architecture uses the 3D encoder to extract spatial features and downsample the data dimensionality before it is fed into the 3D-ConvLSTM network. After passing through the 3D-ConvLSTM, the decoder CNN reconstructs the output volume by increasing data dimensionality. Their results show that the 3D-ConvLSTM is able to predict convective storms more accurately than multiple 3D and 2D baselines, up to a lead time of 60 minutes.

Chapter 3

Methodology

In this chapter, the methodology of our approach is outlined. First, we formally define the volumetric nowcasting task in section 3.1. Subsequently, we present our proposed framework in section 3.2. We conclude the chapter by discussing the loss modules used in section 3.3.

3.1 Formal problem definition

Suppose that for our 3D nowcasting task, we have a sequence of radar volumes, where each volume is a multiCAPPI that can be represented as a tensor $V \in \mathbb{R}^{D \times H \times W}$. Now, say that we can divide this sequence into T past observations (the context), that we can define as $V_{in} = \{V_1, \ldots, V_T\}$ and K future sequences, defined as $V_{out} = \{V_{T+1}, \ldots, V_{T+K}\}$. Then, given V_{in}, the spatio-temporal predictive learning task is to predict the most probable sequence of future radar volumes V_{out}. In this thesis, deep learning-based models parameterized by θ are trained upon data, where the goal is to find a parameter set θ^* that maximizes the log-likelihood of a model predicting the true output sequence V_{out} given the input sequence V_{in}, for all N training pairs $\langle V_{in}, V_{out} \rangle$:

$$\theta^* = \arg\max_{\theta} \sum_{\langle V_{in}^n, V_{out}^n \rangle}^{N} \log p(V_{out}^n | V_{in}^n; \theta) \tag{3.1}$$

Concretely, this optimal set of parameters can be found by applying gradient descent over a defined loss function, which we will discuss in section 3.3. Moreover, for the volumetric nowcasting task in this thesis, we set the context length to $T = 6$ and the future sequence length to $K = 12$. For the spatial dimensions, we set $D = 8$ and $H = W = 240$ such that $V \in \mathbb{R}^{8 \times 240 \times 240}$. Note that our data processing pipeline for obtaining these multiCAPPI volumes is described in detail in chapter 4. Because the temporal interval between two radar scans in the Netherlands is 5 minutes, our model is tasked to predict one hour into the future, given the past half hour as input. This decision can be motivated as research has indicated that having a longer context sequence does not necessarily improve performance [30]. At the same time, a lead time of 60 minutes is similar to related approaches and also sufficiently far enough into the future to ensure that the task requires modelling challenging long-term spatio-temporal dynamics [34, 35, 39, 36]. Now that the problem definition is formally defined, we will describe the technical aspects of our approach in the next section.

3.2 Proposed approach: Multi-Scale 3D PredRNN

The model we implement takes inspiration from three works we discussed in the previous chapter: the PredRNN architecture [46], the 3D-ConvLSTM model [36], and the multi-scale RNN framework [19]. We combine elements from these three approaches to propose a novel end-to-end trainable RNN-based model architecture that we call the Multi-Scale 3D Predictive RNN (MS-3D PredRNN). Figure 3.1 shows a detailed overview of the model architecture. Our model consists of six PredRNN layers of different spatial scales with a hidden feature dimension of 16. Convolutions are performed with a $3 \times 3 \times 3$ kernel. Regarding the downsample and upsample operations, we argue that decreasing our relatively small depth dimension in the multi-scale network would harm the model's ability to retain vertical information through the layers accurately. Therefore, we apply these operations only to our height and width dimensions and keep our depth dimension fixed to $D = 8$ throughout the network.

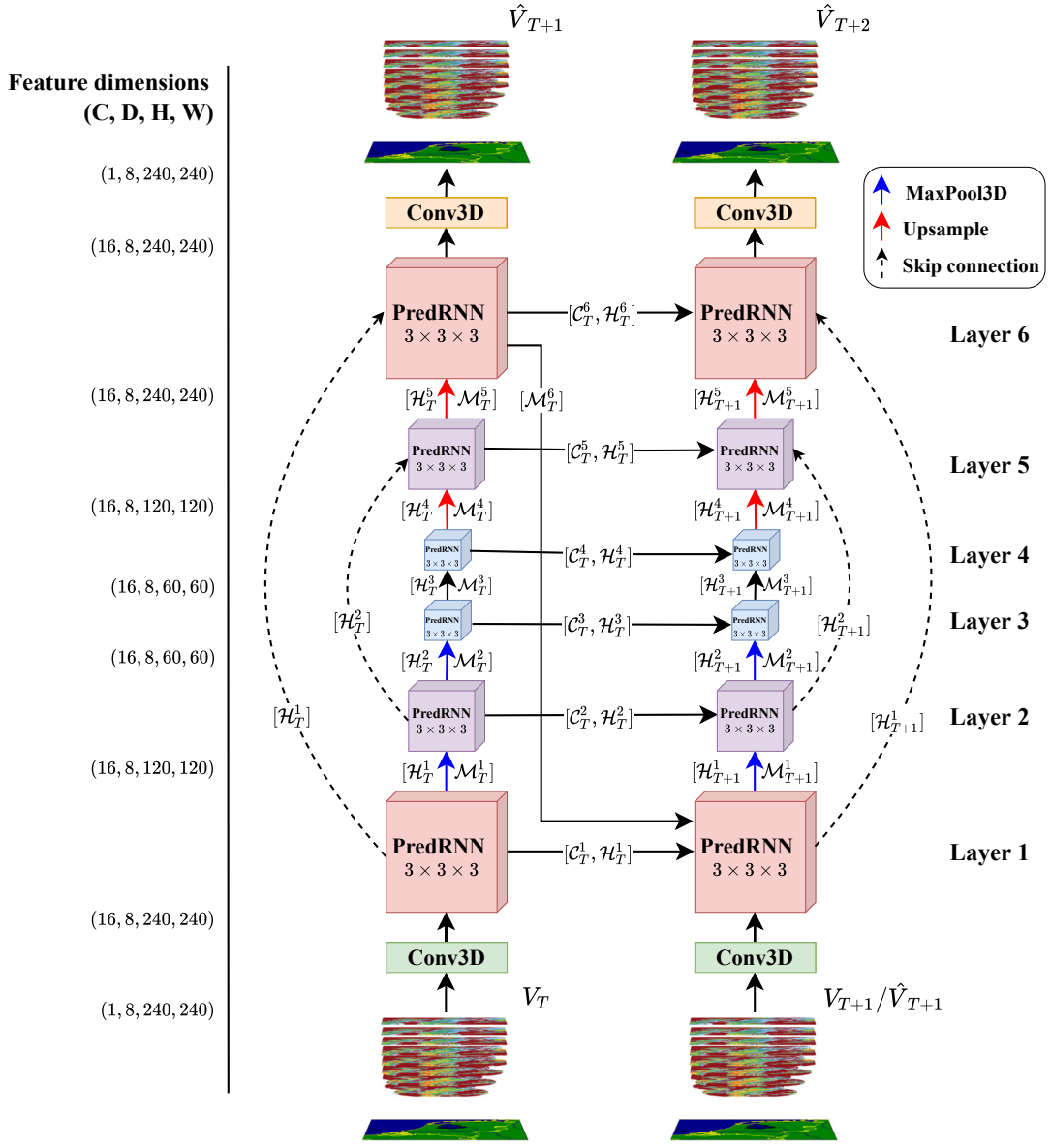


Figure 3.1: The proposed multi-scale 3D PredRNN architecture for predicting radar volumes. The model integrates the multi-scale framework with the PredRNN model and uses 3D convolutions and 3D cell states. Feature dimensionality throughout the network is provided on the left.

3.2.1 Dissection and motivation

We will further dissect and motivate the three core elements of the model. Firstly, we adopt the PredRNN model with the spatio-temporal memory state and zigzag memory flow in stacked layers to iteratively produce future predictions. We chose to implement the PredRNN framework as it seems to consistently outperform CNN-based models (i.e. U-Net variants) [36], but also the standard ConvLSTM model [34] and its adapted TrajGRU variant [35] in nowcasting research [46, 39, 19].

Secondly, we design our PredRNN cells to explicitly model 3D data in three dimensions by replacing 2D convolutions with 3D convolutions. Similarly, we extend the three cell states $\mathcal{C}$, $\mathcal{H}$ and $\mathcal{M}$ from 2D to 3D features so that they are able to retain the vertical dimension of our data. This extension is motivated by the hypothesis that treating the depth of radar volumes as a third spatial dimension could help models to better learn vertical precipitation patterns and avoid the cross-talk problem reported by [39] when radar volumes are treated as multi-channel images. Additionally, the concurrent work of [36] underscores the potential of this approach, as their results show that using a CNN-ConvLSTM hybrid model with 3D convolutions boosts performance in convective storm nowcasting compared with multi-channel baselines.

Thirdly, we implement our predictive framework using the multi-scale architecture with downsampling, upsampling and skip connections. Because the MS-RNN framework was designed to handle 2D image sequences, we replace MaxPool2D layers with MaxPool3D in downsampling transitions, while for upsampling, we replace bilinear interpolation with trilinear interpolation. Moreover, to embed our input volumes, which are single-channel, to our hidden feature size, we use a 3D convolutional layer with a kernel size of 1. After the input is passed through all layers, we reduce the hidden feature size back to a single output channel, also using a 3D convolutional layer with a kernel size of 1. These embedding and flattening layers are similar to those introduced in [19]. Still, identical to the state-to-state transitions within the PredRNN cells, 2D convolutions are replaced with 3D convolutions to retain the vertical information fully.

Our motivation for incorporating the multi-scale framework is that the original authors' results showed that the framework could enhance model performance while decreasing memory usage. The latter effect is particularly interesting in the context of volumetric nowcasting, as radar volumes are relatively large data objects. Therefore training models on this data can quickly reach the memory limits of modern GPUs. Additionally, we implement a 3D PredRNN model with 3D cell states, and because we have to keep track of these states over our whole sequence length to learn (to perform backpropagation through time), GPU memory accumulates even faster.

3.2.2 Novelty of approach

We implement an RNN-based approach, extended to 3D space, to the volumetric nowcasting task. Although the concurrent work of [36] is closely related to the work in this thesis, there are multiple essential differences to note: firstly, their work focuses on convective storm nowcasting, i.e. the task of predicting high reflectivity values instead of general precipitation nowcasting; secondly, we extend the concept of a 3D ConvLSTM to a more complex RNN-based architecture, namely PredRNN; thirdly, we analyze the performance of our model against different optimization objectives and test on a more diverse suite of operational metrics, inspired by the work of [38]; lastly, to limit memory footprint, we do not use a hybrid approach of combining an encoder and decoder CNN with a 3D ConvLSTM but instead use the multi-scale RNN framework with its U-Net adopted features, which has not been deployed in the context of volumetric nowcasting before.

3.3 Loss module

For our proposed model to learn how to predict radar volumes, we need to define an optimization objective that the model is tasked to minimize via gradient descent. Our loss function consists of different modules and closely follows the work of [38]. We will discuss the modules and the strategy for combining them in the following sections.

3.3.1 Pixelwise loss: MAE and MSE

We employ two commonly used pixel-wise loss functions in our experiments: Mean Average Error (MAE) and Mean Square Error (MSE). MAE, also named L1 loss, measures the average absolute difference between the predicted and ground-truth voxel values. In contrast, MSE measures the average squared difference between the predicted and ground-truth values. MAE is a robust error function that is less sensitive to outliers but fails to punish larger prediction errors. In contrast, MSE is sensitive to outliers and is useful when larger errors should be penalized more heavily. The equations for calculating MAE and MSE scores for a prediction and target sequence of radar volumes of length 12 are:

$$\text{MAE} = \frac{1}{12 \times DHW} \sum_{n=1}^{12} \sum_{d=1}^{D} \sum_{i=1}^{H} \sum_{i=1}^{W} \left| \hat{V}_{t+n,d,i,j} - V_{t+n,d,i,j} \right| \tag{3.2}$$

$$\text{MSE} = \frac{1}{12 \times DHW} \sum_{n=1}^{12} \sum_{d=1}^{D} \sum_{i=1}^{H} \sum_{i=1}^{W} \left(\hat{V}_{t+n,d,i,j} - V_{t+n,d,i,j} \right)^2 \tag{3.3}$$

Where D, W, H are the depth, width, and height dimensions, $\hat{V}$ is the reflectivity value of the predicted image, and V is the corresponding ground-truth target value.

However, since the proportions of reflectivity intensities in radar data are imbalanced, models trained on only MAE or MSE are likely to create blurry nowcasts that tend to underpredict especially higher rainfall intensities that occur less often but have a higher real-world impact [36]. Therefore, we follow other recent approaches [35, 25] and propose balanced MAE (B-MAE) and balanced MSE (B-MSE) functions that incorporate a weighting scheme. The basic idea is that higher reflectivity values get assigned a heavier weight w such that we emphasize models to predict high intensities correctly. Specifically, we define our weighted pixel-wise loss functions as follows:

$$\text{B-MAE} = \frac{1}{12 \times DHW} \sum_{n=1}^{12} \sum_{d=1}^{D} \sum_{i=1}^{H} \sum_{i=1}^{W} w_{t+n,d,i,j} \left| \hat{V}_{t+n,d,i,j} - V_{t+n,d,i,j} \right| \tag{3.4}$$

$$\text{B-MSE} = \frac{1}{12 \times DHW} \sum_{n=1}^{12} \sum_{d=1}^{D} \sum_{i=1}^{H} \sum_{i=1}^{W} w_{t+n,d,i,j} \left(\hat{V}_{t+n,d,i,j} - V_{t+n,d,i,j} \right)^2 \tag{3.5}$$

where:

$$w_{t+n,d,i,j} = \begin{cases} 1, & V_{t+n,d,i,j} < 7 \text{ dBZ} \\ 2, & 7 \text{ dBZ} \leq V_{t+n,d,i,j} < 20 \text{ dBZ} \\ 5, & 20 \text{ dBZ} \leq V_{t+n,d,i,j} < 35 \text{ dBZ} \\ 10, & V_{t+n,d,i,j} \geq 35 \text{ dBZ} \end{cases} \tag{3.6}$$

3.3.2 Structural similarity

Besides using MAE and MSE as optimization objectives, we follow the work of [38] in deploying structural similarity (SSIM) as an additional loss function. The SSIM method is a popular image quality assessment measure within the computer vision domain that computes the similarity between two images and was originally proposed for assessing the degradation of the visual quality of images, e.g. in image compression [48]. For measuring a similarity score between images x and y, SSIM is defined as follows:

$$\text{SSIM}(x, y) = l(x, y) \cdot c(x, y) \cdot s(x, y) = \Big(\frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}\Big) \cdot \Big(\frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}\Big) \cdot \Big(\frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}\Big) \tag{3.7}$$

In which $l(x, y)$ denotes the luminance similarity, $c(x, y)$ the contrast similarity, and $s(x, y)$ the similarity of structure. Furthermore, μ_x and μ_y are the pixel mean of images x and y, σ_x and σ_y the standard deviations of x and y, and σ_{xy} the covariance between x and y. Last, C_1, C_2, and C_3 are small positive constants to avoid zero division. The above formula is not applied over entire images but over small patches within the images, and the final SSIM score is the average score of these local patch similarities. In practice, this is done using a sliding window approach (i.e. through convolutional operations) over the two images. The final SSIM score ranges between -1 and 1, where -1 denotes perfect anti-correlation, 0 denotes no similarity, and 1 denotes perfect similarity.

Because SSIM combines structural information with perceptual phenomena (luminance and contrast), it has been suggested that SSIM is a quality measure that more closely represents human quality perception [48]. For the nowcasting task, it is important to predict visually realistic radar images that are similar to target images. Therefore, SSIM is regularly used as an evaluation metric to measure the predictive performance of nowcasting models [45, 44, 47, 46, 19, 38]. However, we incorporate SSIM as a loss function and not as an evaluation metric. By doing this, our model is directly being optimized towards producing images that are perceptually similar to the ground truth, which can lead to less blurry predictions of improved image quality [38]. The resulting SSIM loss function for volumes $\hat{V}$ and V is defined as follows:

$$\mathcal{L}_{\text{SSIM}}(\hat{V}, V) = 1 - \text{SSIM}(\hat{V}, V) \tag{3.8}$$

The standard SSIM function computes a similarity score over two-dimensional images, while our data is three-dimensional. Therefore, we implement 3D SSIM, which is an extension of SSIM to a 3D space by computing SSIM scores in local 3D blocks using 3D convolutions [52]. Unless specifically mentioned, from here onward, we refer to this 3D implementation when using the term SSIM.

3.3.3 Combining modules

In our experiments, we combine the different loss functions into one general function via a simple combination strategy [38]:

$$\mathcal{L}_{\text{combined}}(\hat{V}, V) = \sum_l \lambda_l \mathcal{L}_l(\hat{V}, V) \tag{3.9}$$

Where l is the loss function chosen from the set {MSE, MAE, SSIM, B-MSE, B-MAE} and λ_l is the scaling factor for each module. Specifically, we set $\lambda_l = 1$ if we include a module l in our combined loss function and $\lambda_l = 0$ when we exclude a module l from our loss function.

Chapter 4

Experimental setup

In this chapter, we provide details regarding the experimental setup of our experiments. First, we discuss the process of our dataset construction in section 4.1. Second, we outline the different baselines implemented in section 4.2. Third, we present our evaluation metrics in section 4.3. Last, we provide implementation details in section 4.4.

4.1 Dataset

In this section, we discuss our data processing pipeline step-by-step and explain our reasoning behind implementation choices when necessary.

4.1.1 Timestamp filtering

The first step of creating our dataset was to define a grid of 240x240 km as the area in which we would train our models to predict precipitation. The grid is shown in figure 4.1 and was chosen because it covers the larger part of the Netherlands, and thus poses a realistic operational nowcasting task, while also having good CAPPI coverage, even at lower altitudes at which CAPPI images suffer from blind spots. Subsequently, it is standard practice in nowcasting research to oversample rainy events in a dataset because the resulting dataset will then have a more balanced distribution of informative rainy and less informative non-rainy events, which allows models to better learn the characteristics of precipitation patterns and not be biased towards clear sky prediction [34, 35, 39, 14, 30, 25]. Therefore, the next step was to filter out rainy sequences to include in our dataset. The filtering process was conducted in the following manner, similarly to the strategy of [25]:

1. Readily available pseudoCAPPI reflectivity maps at 1500m altitude above the Netherlands were downloaded for all timestamps in 2020 and 2021[1].
2. For all timestamps, pixels with a minimum value of 7 dBZ were considered rainy, and pixels below 7 dBZ were considered non-rainy, as this is the threshold the KNMI typically uses.
3. Non-overlapping target sequences of length 12 were filtered in which at least 20% of the defined 240x240 grid was covered in rainy pixels and for which the previous 6 timestamps were available.

[1] `https://dataplatform.knmi.nl/dataset/radar-reflectivity-composites-2-0`.

4.1.2 Data processing

After filtering, we were left with 505 and 515 non-overlapping timestamp sequences in 2020 and 2021, respectively. The next step was to create our actual datasets. To do this, we first downloaded raw volumetric radar reflectivity data in polar coordinates from the radar stations in Den Helder and Herwijnen for all timestamps[2]. Next, Using all 15 available scans with different elevation angles from both radar stations, we created 1) a pseudoCAPPI at 1500 m altitude on the defined 240x240 grid and 2) a multiCAPPI consisting of 8 stacked CAPPI images on the 240x240 grid, ranging from 1000 meters to 2750 meters with a 250-meter vertical interval. Figure 4.1 shows an example of the obtained pseudoCAPPI and multiCAPPI.

In this process, we projected the raw radar data of both stations to a 3D volumetric Cartesian grid through interpolation. Subsequently, scan data from both radar stations was merged using pulse volume weighting [23]. This means that the two radar grids are combined into a single grid by taking a weighted average of radar reflectivities, where weights are a function of the distance of a given point to the radar. We also employed a popular attenuation correction method proposed by [17], in which radar data is adjusted to account for the loss of radar signal strength due to absorption and scattering by precipitation [41]. In addition, similar to how the KNMI processes radar data, we set reflectivity values below 7 dBZ to zero (no rain) to prevent noise accumulation, and we capped reflectivity values at 55 dBZ to suppress clutter-induced reflection [14].

Furthermore, for the multi-CAPPI images, we chose our specific vertical range, as this roughly follows the same range in [25, 39], but it is twice as dense. We hypothesize that having dense volumes could help the vertical motion of precipitation to be more easily modelled into our data. Additionally, we argue that radar data at a relatively low but dense altitude (below 3000 meters) could be more informative than sparse and more spread-out data when creating translations to surface precipitation, which could become an important direction for future research. Lastly, our data range around the typical industry standard pseudoCAPPI altitude of 1500-2000 m and contain a CAPPI slice at 1500 meters. Therefore, we could compare prediction performance at the 1500 m altitude when training a model on our 2D dataset.

4.1.3 Dataset details

By implementing our data processing recipe, we obtain two radar reflectivity datasets with a voxel range of (0,55), the unit being logarithmic reflectivity in dBZ: 1) a 2D dataset, consisting of pseudoCAPPI images $P \in \mathbb{R}^{240\times240}$ and 2) a 3D dataset consisting of multiCAPPI volumes $V \in \mathbb{R}^{8\times240\times240}$, where the first dimension indicates the vertical dimension. Moreover, for both the pseudoCAPPI and multiCAPPI, horizontal pixel resolution equals 1km, meaning that moving 1 pixel horizontally equals a distance of 1km within the 240x240 km grid. This resolution is typically used in nowcasting research, as it is important to have spatially dense radar images to create high-resolution forecasts [14].

It is important to emphasize that both datasets have matching timestamp sequences of length 18 and a temporal resolution of 5 minutes. As stated in our problem definition, the first 6 data instances (the past half hour) are provided to our models as context, and the subsequent 12 future timestamps are the prediction target (an hour of lead time into the future). Moreover, we use the 505 data sequences in 2020 for training and validation, while we use the 515 sequences in 2021 only for testing. This out-of-sample testing characteristic is typical in weather prediction, and it poses a more difficult obstacle for the nowcasting models than a cross-validation

[2] `https://dataplatform.knmi.nl/dataset/radar-volume-denhelder-2-0` and `https://dataplatform.knmi.nl/dataset/radar-volume-full-herwijnen-1-0`.

mechanism commonly used in machine learning, as data distributions of the train and test set are different [34, 39]. We highlight the dataset statistics and distribution differences in table 4.1 and figure 4.2.

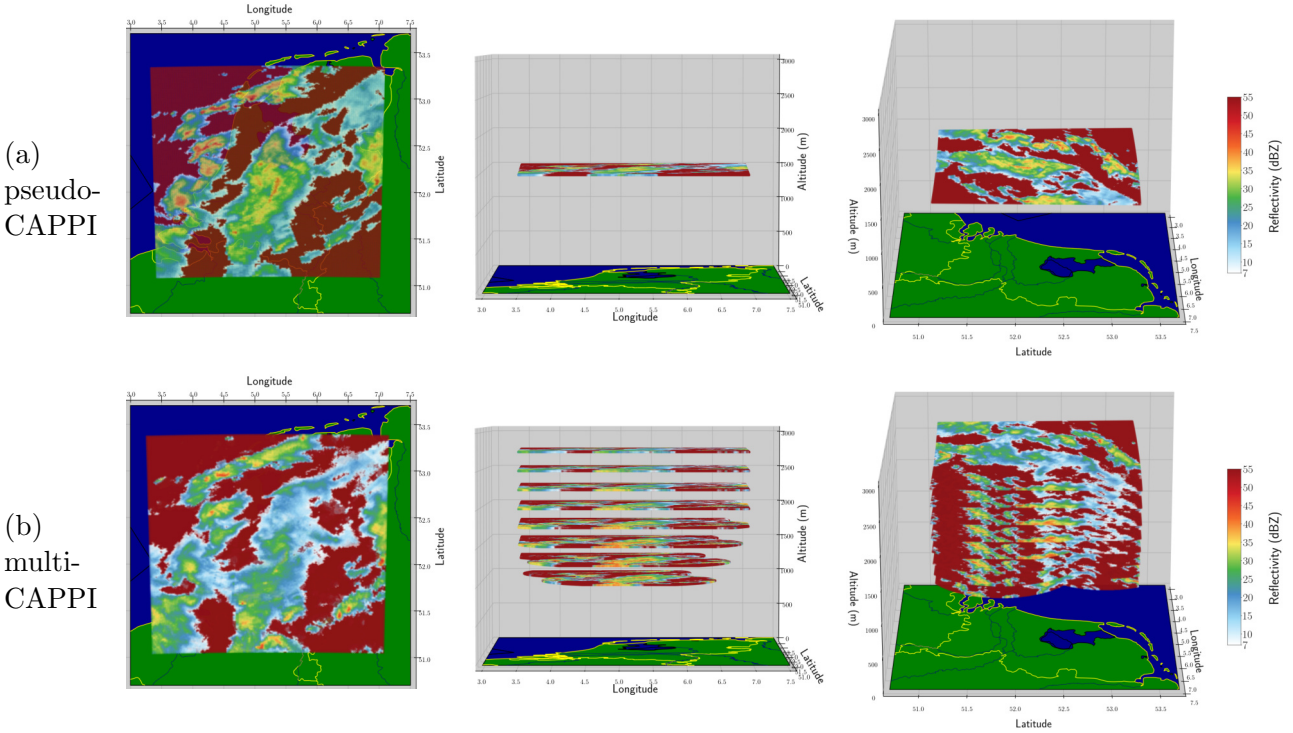


Figure 4.1: Example of processed pseudoCAPPI and multiCAPPI data in our dataset. The time of the event is 28/10/2020, 16:10. Viewed from three different angles: top view (left), sidewards view facing north (middle), and sidewards view facing west (right). For visualization purposes, non-rainy voxels within the 240x240 grid are displayed in red.

			Proportion in dataset (%)			
Year	**Subsets**	***N***	$x < 7$	$7 \leq x < 20$	$20 \leq x < 35$	$35 \leq x \leq 55$
2020	train + validate	505	56.93	22.62	18.99	1.46
2021	test	515	59.43	23.14	16.28	1.15

Table 4.1: Dataset statistics. N denotes the number of sequences in the dataset, while x indicates an arbitrary voxel value within all multiCAPPI volumes.

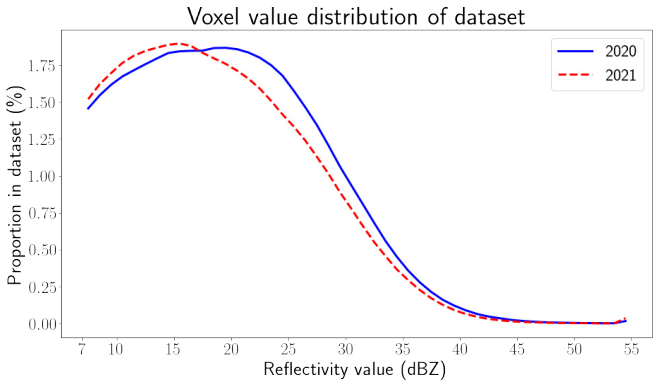


Figure 4.2: Positive voxel value distribution for all filtered sequences in 2020 and 2021.

4.2 Baselines

Multiple baselines were implemented to conduct a comparative performance analysis of the proposed MS-3D PredRNN. First, we have the Eulerian persistence model, a common baseline in precipitation nowcasting. The persistence model uses the final time step of the context sequence as the prediction for all future timestamps and is thus based on the naive assumption that weather conditions remain the same in the future. However, it is not trivial to be outperformed because time differences in nowcasting between images are minor (e.g. 5 minutes). Therefore, it provides a reasonable lower baseline to verify if our model produces meaningful predictions [40].

Secondly, we implement different baseline models that are more competitive and related to the proposed model. Specifically, we implement a regular PredRNN model without the multi-scale architecture, using 2D convolutions and cell states, thereby reproducing the multi-channel approach of [39] in which depth dimension is treated as the image channel. Moreover, we also implement a regular PredRNN model using 3D convolutions and cell states. We name these two models 2D PredRNN and 3D PredRNN, respectively. By including these two baseline models, we can examine the added value of having the multi-scale architecture in our proposed approach. Lastly, we deploy a multi-scale PredRNN framework that uses 2D convolutions and cell states, again treating depth information as a channel, to further test the hypothesized advantages of explicitly modelling depth as a third spatial dimension in our proposed approach. We name this model the Multi-Scale 2D PredRNN (MS-2D PredRNN).

4.3 Evaluation metrics

In nowcasting research, it is essential to design operational metrics that accurately reflect how skilful a model is in making predictions. Because different metrics represent a distinct aspect of a model's ability to predict, and no single verification score can capture all desired properties of a nowcast, it is advisable to implement a suite of metrics to evaluate performance [30]. Concretely, we implement three commonly used categorical operational metrics in nowcasting: Critical Success Index (CSI), False Alarm Ratio (FAR), and Probability of Detection (POD). To compute these metrics, the voxels within a predicted radar volume $\hat{V}$ and its corresponding target value V are binary classified based on a reflectivity threshold. Values above the threshold are set to 1 (rainy voxel), and values below the threshold are set to 0 (non-rainy voxel). Subsequently, we obtain a 2x2 binary confusion matrix, as shown in table 4.2. Using the values in this table, we can compute the CSI, FAR, and POD score according to equations 4.1, 4.2, and 4.3.

		Target	
		1	**0**
Prediction	**1**	True Positive (TP)	False Positive (FP)
	0	False Negative (FN)	True Negative (TN)

Table 4.2: 2x2 confusion matrix obtained after using a binary threshold to classify voxels as 1 (rainy) or 0 (non-rainy).

$$\text{CSI} = \frac{\text{TP}}{\text{TP} + \text{FN} + \text{FP}} \tag{4.1}$$

$$\text{FAR} = \frac{\text{FP}}{\text{FP} + \text{TP}} \tag{4.2}$$

$$\text{POD} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.3}$$

$$\tag{4.4}$$

CSI, FAR, and POD scores all range between 0 and 1. First, CSI measures the proportion of correct precipitation warning events out of all the warnings issued and missed events, with a value of 1 indicating a perfect forecast. Second, the FAR score measures what ratio of forecast positives are false alarms. Here, a value of 0 indicates a perfect score, meaning that none of the warnings are false alarms. Third, POD measures the proportion of correctly warned events out of the total number of precipitation events, with a value of 1 indicating a perfect score in which all precipitation events are forecast. Because CSI considers both false alarms and missed events, it can be seen as a more balanced score than FAR and POD. However, by jointly analyzing scores of all three metrics, we can obtain a more comprehensive understanding of how good a model's predictions are than by looking only at a single one of them.

In addition to the categorical operational metrics described above, we employed the frequency bias metric, which measures the ratio between predicted events and observed events. It is defined as:

$$\text{Bias} = \frac{\text{TP} + \text{FP}}{\text{TP} + \text{FN}} \tag{4.5}$$

It ranges from 0 to ∞, where bias < 1 indicates that precipitation events are less frequently forecast than that they occur, and bias > 1 indicates that precipitation events are predicted more frequently than they occur. Note that this metric does not measure how accurate predictions are, but it is helpful when analyzing how a model copes with imbalanced data and the task of predicting rare events.

Finally, we use the Pearson Correlation Coefficient (PCC) as an evaluation measure. The PCC score is commonly adopted in nowcasting research and measures the strength of linear association between the forecasts and observations [38]. The PCC score for two images or volumes x and y is calculated as follows:

$$\text{PCC}(x, y) = \frac{\sum(x - \mu_x)(y - \mu_y)}{\sqrt{\sum(x - \mu_x)^2 \cdot \sum(y - \mu_y)^2}} = \frac{\sigma_{xy}}{\sqrt{\sigma_x^2 \cdot \sigma_y^2}} \tag{4.6}$$

In which μ_x and mu_y are the mean values of images x and y, σ_{xy} is the covariance between the two images, and σ_x^2, σ_y^2 are the variances of both images. Instead of being a categorical metric for which we have to apply binary classification using a threshold, PCC is a distributional similarity metric that provides insights into how well predictions globally align with our target values. Therefore, PCC scores are helpful for analyzing the overall distributional agreement between predicted radar volumes and their ground truth counterparts.

4.4 Implementation details

For the final part of describing our experimental setup, we provide technical details regarding our implementation. This information can be subdivided into details about the training and technical setup.

4.4.1 Training setup

All models were trained using Stochastic Gradient Descent (SGD), using the commonly used Adam optimizer [16]. Moreover, we use a batch size of 1, as using a larger batch size would breach the GPU memory limit for the proposed model. We randomly assign 80% of the sequences within 2020 to our train set ($N = 404$) and add the remaining 20% to our validation set ($N = 101$). As stated in section 4.1.3, we add all 515 sequences in 2021 to our test set. Unless specifically mentioned, we train our models for 25 epochs in total.

Moreover, we use scheduled sampling, a learning strategy introduced by [5] that can improve sequence prediction performance. Recall that in the stacked ConvLSTM training protocol, generated outputs of the model are iteratively fed as inputs for the next time step to predict a future sequence. However, when using scheduled sampling, the training process gradually changes from a fully guided scheme, in which ground truth data instances are given to the next time step, towards a non-guided scheme that only uses the generated predictions in future time steps. During inference, only model-generated outputs are being used as inputs for the next time step. In our training regime, we apply scheduled sampling to gradually change from the fully-guided to the non-guided strategy in the first 20 epochs with a learning rate of 1e-3, after which we lower the learning rate to 3e-4 and train the model for 5 more epochs in the fully non-guided training scheme.

Furthermore, before feeding it into our model, we normalize our data from a (0,55) dBZ reflectivity range to a (0,1) range. After predicting a sequence, we obtain our predictions again in dBZ by multiplying with a factor of 55. When computing our loss and operational metrics, we mask and ignore the voxels in the multiCAPPI grid that are located in the blind spots of the radar. Regarding SSIM, we use a window size of 5 for computing the local patch scores. For the categorical metrics CSI, FAR, POD, and bias, we use thresholds of 7 dBZ (the minimum positive voxel value, indicating light mist), 20 dBZ (indicating rainfall) and 35 dBZ (indicating moderate to severe rain).

4.4.2 Technical setup

All models were trained on a Tesla T4 GPU with 16GB of working memory. Code for processing radar data used the open-source Wradlib library for weather data processing [10]. Code for creating and training the models was implemented using the popular PyTorch library with the support of the CUDA toolkit for GPU operations [24, 21]. Additionally, we implemented mixed precision training, causing some GPU operations to be performed in half-precision (i.e. using float16 tensors) to limit runtime and memory footprint. Using this setup, training the proposed MS-3D PredRNN model for 25 epochs took approximately 16 hours. Other approaches to limit memory usage and speed-up training, including truncated backpropagation through time and using larger down-sample and up-sampling operations to decrease network size, were explored but did not seem to impact performance positively.

Chapter 5

Experiments and results

In this chapter, we present results obtained from four experiments. First, we analyze the effects of using different loss modules in section 5.1. Second, we perform a quantitative comparative analysis of our proposed model versus the baselines in section 5.2. Third, we qualitatively study the predictive performance of our approach in section 5.3. Finally, we investigate if using 3D radar data can help to obtain better 2D predictions in section 5.4.

5.1 Experiment 1: loss module analysis

In our first experiment, we test the effect of applying different combinations of loss functions when training our proposed MS-3D PredRNN model. The metric scores evaluated on the validation set are reported in table 5.1. We grouped results into unbalanced experiments (using no weighting scheme) and balanced experiments (using the weighting scheme described in section 3.3). Note that we did not include the results of training the model only on the SSIM function, as this function alone did not provide enough information to our network to produce meaningful predictions. We analyze and compare the results of the unbalanced and balanced experiments in the following sections.

5.1.1 Unbalanced experiments

Several interesting conclusions can be drawn from the results of the unbalanced experiments. First, we see a similar trend in all loss combinations. CSI, POD, and PCC scores decrease significantly when reflectivity intensity increases, while FAR scores increase. This substantial decline in performance was expected, as high but rare intensities are more difficult for a model to predict, especially further ahead into the future.

Second, we notice that the MSE and SSIM combination performs poorly. Interestingly, the MAE and SSIM combination offers the best performance, outperforming the loss function combining all three modules, which was considered the best combination in [38]. Considering all unbalanced experiments, using MAE and SSIM yields the highest CSI scores for all three intensity thresholds and the lowest FAR scores except for the 35 dBZ threshold. Moreover, it offers the highest PCC score and produces competitive POD scores, especially for moderate and severe intensities.

Third, we observe that bias scores are below 1 and decrease even more for higher intensity thresholds, indicating that the model under-forecasts rainy voxels, especially for the higher but rarer intensities. Because of the data imbalance, this is expected behaviour, as the model tends

	CSI ↑			FAR ↓			POD ↑			PCC ↑	Bias		
dBZ thresh.	**7**	**20**	**35**	**7**	**20**	**35**	**7**	**20**	**35**	**-**	**7**	**20**	**35**
Unbalanced experiments													
MAE	.7135	.5064	.0674	.1379	.3268	.5758	.7933	**.6327**	.0827	.7678	0.92	0.92	0.15
MSE	.7236	.5038	.0671	.1525	.3113	.5791	.8218	.6143	.0832	.7907	0.98	0.86	0.15
MAE + MSE	.7264	.4953	.0604	.1513	.2887	.4592	.8238	.5862	.0697	.7905	0.98	0.78	0.11
MAE + SSIM	**.7291**	**.5232**	**.0946**	**.1191**	**.2825**	.4720	.7974	.6232	**.1129**	**.7936**	0.90	0.84	0.20
MSE + SSIM	.7014	.3855	.0371	.1768	.2887	.7232	.8126	.4384	.0503	.7447	0.99	0.57	0.13
MAE + MSE + SSIM	.7228	.4867	.0456	.1585	.3159	**.4441**	**.8250**	.5890	.0528	.7749	0.99	0.83	0.08
Balanced experiments													
B-MAE	.7170	.5266	.1019	.1768	.4034	.5694	.8342	**.7747**	.1336	.7596	1.02	1.38	0.28
B-MSE	.7111	.5122	.0951	.2193	.4014	.6187	.8814	.7356	.1256	.7646	1.16	1.27	0.32
B-MAE + B-MSE	.7195	.5269	.1024	.2270	.3986	.5800	**.9039**	.7629	.1326	.7732	1.20	1.32	0.27
B-MAE + SSIM	**.7358**	**.5442**	**.1213**	**.1569**	**.3570**	**.5266**	.8412	.7404	**.1585**	**.7835**	1.00	1.19	0.30
B-MSE + SSIM	.7284	.5363	.0844	.2045	.3716	.5847	.8908	.7475	.1034	.7790	1.15	1.24	0.21
B-MAE + B-MSE + SSIM	.7312	.5354	.1163	.2080	.3856	.6110	.8964	.7634	.1531	.7548	1.16	1.16	0.37

Table 5.1: Averaged validation metric scores of the MS-3D PredRNN model with different loss modules, grouped into balanced and unbalanced experiments. Highest score per experiment group (**bold**) and the overall highest score (**underlined**) are highlighted.

to "play it safe" by minimizing its predicted intensities, to avoid incurring severe loss penalties when predictions are wrong. The following section discusses how the balanced loss functions can overcome this problematic behaviour.

5.1.2 Balanced experiments

From the results of the balanced experiments, several noteworthy observations can be made. First, we notice from the bias scores of all balanced loss combinations that when being more severely punished for missing higher voxel intensities, the model learns to take more risks and even over-forecasts events with thresholds of 7 and 20 dBZ. The model still under-forecasts for the highest intensity threshold, but there is a significant increase in bias compared to the unbalanced experiments.

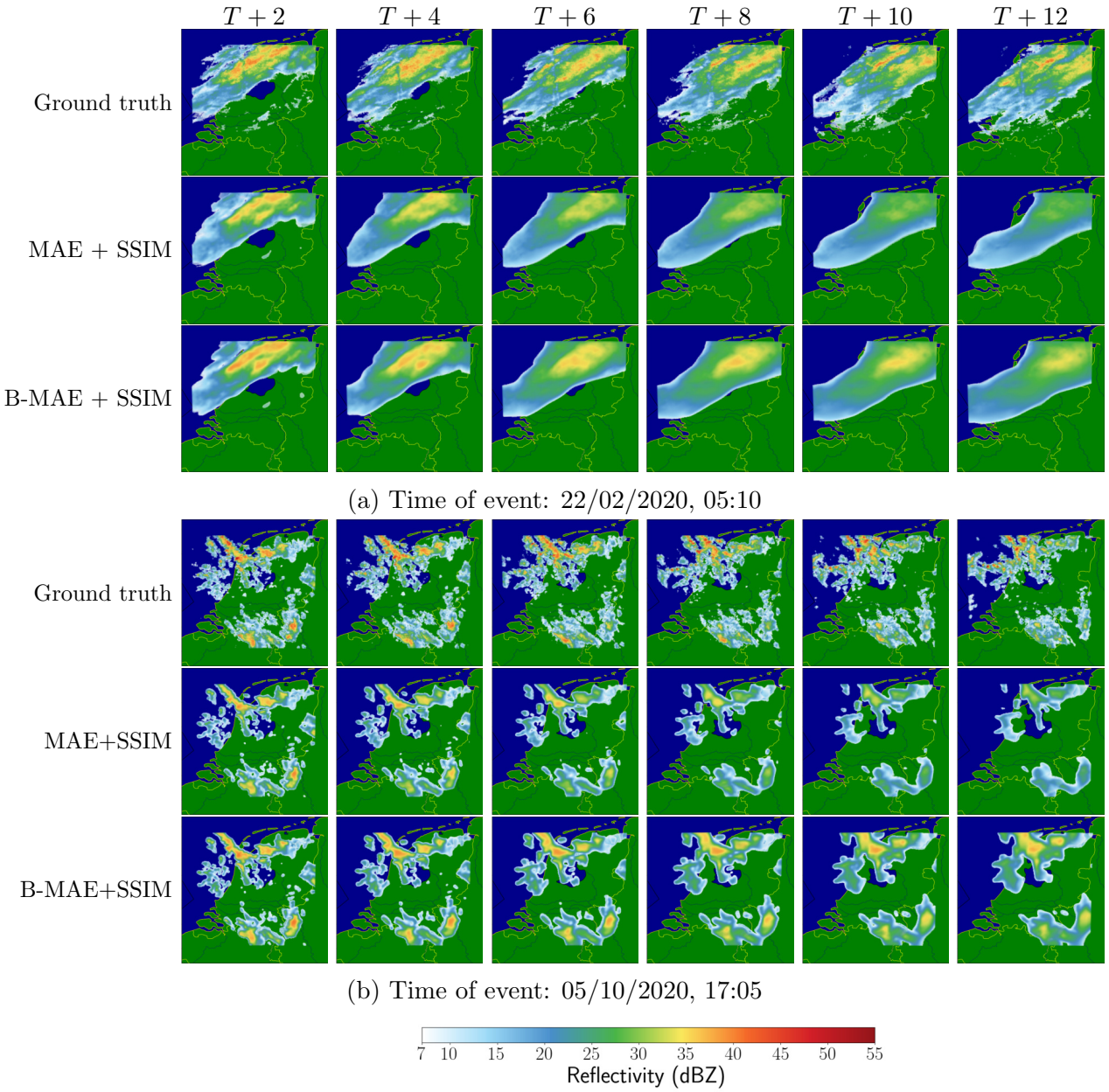
Secondly, again comparing with the unbalanced experiments, we note that the riskier nature of the models generally leads to higher CSI and POD scores, with especially noticeable improvements for the 20 and 35 dBZ threshold. However, this increase in prediction performance comes at the expense of higher FAR scores. This is not surprising behaviour, as we bias our model to place more emphasis on predicting higher reflectivity values. This ensures that the model predicts rainy voxels more often and is less severely punished for causing false alarms. Moreover, we also notice a slight decrease in PCC scores, which is explainable, as the weighted loss functions drive the model to learn a biased representation of the target distribution.

Thirdly, including the B-MSE module seems to impact performance negatively. Combinations using this module seem to particularly struggle with predicting high-intensity values. This behaviour could be explained by the sensitivity of MSE to rare but important events, where strong (squared) penalties are imposed when mispredicting high intensities.

Finally, similar to before, using B-MAE and SSIM performs best. This combination achieves the highest overall CSI scores for all three intensity thresholds while achieving significantly lower FAR scores than other balanced combinations. Furthermore, it reaches the highest PCC score within the group of balanced experiments and also produces competitive POD scores, including the highest overall POD score for the 35 dBZ threshold.

5.1.3 Conclusion

From analyzing the results of this first experiment, we can conclude that there is a trade-off between choosing a model with a balanced function with higher CSI and POD scores or choosing a model with an unbalanced loss function that achieves lower FAR scores and a higher PCC score. We can specify this trade-off in choosing between MAE and SSIM or B-MAE and SSIM, as these combinations performed best within their respective experiment groups. We argue that it is better to implement a model that tends to overestimate predictions slightly, with higher CSI and POD scores, than a model that underestimates its forecasts, with lower FAR scores, which causes it to miss important high-intensity events more often. Therefore, we conducted further experiments using the B-MAE and SSIM combination. Additionally, including a balanced loss function contributes to creating sharper predictions, which can be an important requirement of visualized nowcasts. To illustrate this, two visual examples of predictions made using both loss combinations are provided in figure 5.1.



(a) Time of event: 22/02/2020, 05:10

(b) Time of event: 05/10/2020, 17:05

Figure 5.1: Visual comparison of predictions of the MS-3D PredRNN model, using unbalanced and balanced loss functions. For visualization purposes, the context of both sequences has been left out, and only the top-down view is shown.

	Layers	Hidden size	Kernel	Params	Mem.
Persistence	-	-	-	-	-
2D PredRNN [39]	4	(32, 32, 32, 32)	3×3	1,200,936	5.7 GB
3D PredRNN	4	(16, 16, 16, 16)	3×3×3	891,057	14.1 GB
MS-2D PredRNN [19]	6	(32, 32, 32, 32, 32, 32)	3×3	1,801,128	4.6 GB
MS-3D PredRNN (ours)	6	(16, 16, 16, 16, 16, 16)	3×3×3	1,336,561	10.8 GB

Table 5.2: Architecture details and memory usage for our proposed model and baselines. References are provided to the original approaches that were adapted for our experiments.

	MAE ↓	SSIM ↑	MSE ↓
Persistence	.2590	.5790	.0275
2D PredRNN [39]	.1605	.6735	.0154
3D PredRNN	.1611	.6806	.0168
MS-2D PredRNN [19]	.1481	.7015	.0126
MS-3D PredRNN (ours)	**.1413**	**.7169**	**.0119**

Table 5.3: Averaged test loss scores for our proposed model and baselines. Best overall loss scores are highlighted in **bold**. Note that models were not optimized for MSE. References are provided to the original approaches that were adapted for our experiments.

5.2 Experiment 2: baseline comparison

In our second experiment, we want to obtain a more comprehensive understanding of the predictive performance of the MS-3D PredRNN model. To achieve this, we fix our loss function to the combination of B-MAE and SSIM and train our model and all baselines on a training set comprised of both our training and validation sequences ($N = 404 + 101 = 505$). After training for 25 epochs, all models are evaluated on the test set ($N = 515$). Moreover, the 3D models are implemented with a hidden size of 16 and the multi-channel models with a hidden size of 32 to keep the total number of model parameters roughly in the same order of magnitude. Last, to avoid "unlucky" random model initialization, we train each model three times using different random seeds and pick the model with the lowest test loss to be included in the results.

Model details and test loss scores are provided in tables 5.2 and 5.3, and corresponding test metric scores are reported in table 5.4. To compare the MS-3D PredRNN model with the various baselines more concretely, we first analyze the effects of two main features of our model architecture: 1) the multi-scale architecture and 2) treating depth dimension as a third spatial dimension, using 3D convolutions and 3D cell states.

5.2.1 Effects of Multi-Scale architecture

To analyze the effects of our model adopting the multi-scale architecture, we distinguish between two aspects: memory versus performance. First, the memory-reducing effect of the multi-scale architecture was a fundamental reason for implementing it, as applying models to volumetric radar data can cause memory usage to accumulate quickly. On the right side of table 5.2, details of model architecture, number of parameters, and GPU memory usage are provided for the proposed model and all baselines. From this table, it becomes evident that when adopting the multi-scale framework, we can increase the number of layers and the number of parameters of a model while still reducing memory usage compared to regular PredRNN architectures.

	CSI ↑			FAR ↓			POD ↑			PCC ↑
dBZ thresh.	**7**	**20**	**35**	**7**	**20**	**35**	**7**	**20**	**35**	**-**
Persistence	.5824	.3519	.0760	.2775	.5018	.8653	.7127	.4873	.1321	.5640
2D PredRNN [39]	.7034	.4881	.0803	.2113	.4137	.6675	.8480	.6987	.1049	.7525
3D PredRNN	.7061	.4915	.1255	.2151	.4426	.7220	**.8556**	**.7596**	**.2021**	.7545
MS-2D PredRNN [19]	.7219	.5210	.1105	.1754	.3694	.6165	.8369	.7103	.1473	.7873
MS-3D PredRNN (ours)	**.7340**	**.5351**	**.1342**	**.1702**	**.3521**	**.5765**	.8492	.7208	.1813	**.8012**

Table 5.4: Averaged operational test metrics of our proposed model and baselines. Best overall metric scores are highlighted in **bold**. References are provided to the original approaches that were adapted for our experiments.

Secondly, when analyzing model performance, several interesting findings can be derived from the results presented in table 5.3 and 5.4. Firstly, it is noticeable that all models significantly outperform the persistence baseline, which was expected given that it is a naive baseline. However, given that it is not trivial to beat a persistence baseline in nowcasting, these results strongly suggest that applying deep learning-based models to volumetric nowcasting makes sense. Secondly, we observe that the multi-scale implementations outperform their regular PredRNN counterparts. Table 5.3 shows that both the 2D multi-scale model and the proposed 3D multi-scale model have higher SSIM scores, and lower B-MAE and MSE scores. Moreover, both models score better regarding operational metrics (higher CSI and PCC, lower FAR) than the regular predRNN models. Although the 3D PredRNN model achieves the highest POD scores, it seems to strongly overpredict precipitation in wrong locations, reflected by lower CSI and PCC scores and significantly higher FAR scores.

5.2.2 Effects of 3D convolutions and 3D cell states

Similar to the previous section, we analyze the effects of treating the depth dimension as a third spatial dimension on memory usage and predictive performance. First, from the information in table 5.2, we notice that the two models explicitly handling volumetric radar data in three dimensions, i.e. 3D PredRNN and MS-3D PredRNN, have a significantly higher memory usage than their multi-channel (2D) counterparts, even though the feature dimension (hidden size) decreases from 32 to 16 and the models have fewer parameters. Specifically, for the regular PredRNN models, the memory usage increases from 5.7 to 14.1 GB (147% increase), and for the 3D models, the memory usage increases from 4.6 to 10.8 GB (134% increase). These numbers clearly illustrate that using 3D convolutions and 3D cell states in our model does come at the cost of additional computational overhead.

However, when inspecting model performance, using 3D models offers advantages. Considering the loss and metric scores in tables 5.3 and 5.4, respectively, there is no clear winner between the 2D PredRNN and the 3D PredRNN. The 3D PredRNN performs better for some operational metrics but has slightly worse loss scores than the 2D PredRNN. In contrast, our proposed MS-3D PredRNN model outperforms the MS-2D PredRNN baseline. Our proposed model achieves the highest SSIM and lowest B-MAE and MSE scores. Moreover, our proposed model also produces the highest CSI and PCC scores for all three intensity thresholds while maintaining the lowest overall FAR scores. It is only outperformed by the 3D PredRNN model in POD scores, but as mentioned before, this model also makes many prediction errors, reflected by poor CSI, FAR, and PCC scores.

5.2.3 Analysis of performance over lead time

So far, we have only analyzed averaged loss and metric scores. To investigate how our proposed model and all baselines perform when predicting further into the future, we plot the loss and metrics scores, previously summarized in tables 5.3 and 5.4, against the lead time in figure 5.2. A closer inspection of these plots offers some new insights.

Firstly, we notice that the performance of all models degrades over lead time. This is not unexpected because the nowcasting task becomes increasingly more difficult further ahead in the future. During inference, a model has to iteratively create predictions based on previous predictions, thereby expanding upon uncertainties and prediction errors accumulated in earlier time steps. Moreover, another general pattern is noticeable: performance degrades more quickly for higher intensities, reflected by the steeper lines within the plots. CSI scores for the 35 dBZ threshold start at 0.35 but are already declined below 0.10 at a lead time of $T + 30$ minutes, with a similar sharp degradation pattern visible for the FAR and POD plots. This shows that predicting rare but important severe precipitation voxels is more complicated than predicting low-intensity precipitation patterns, even when we apply a weighting scheme to our loss function.

In addition to these findings, the plots support our earlier conclusions. First, the large gap in performance difference between the persistence model and all other models is noticeable, even for a short lead time. Second, the performance gain of both multi-scale models over the regular PredRNN models can be observed. Again, only in the POD plots does the 3D PredRNN perform better than both multi-scale models, but the other plots show the tendency of this model to overpredict inaccurately, visible in high FAR and low CSI lines. Third, the MS-3D PredRNN model outperforms the MS-2D PredRNN model. Specifically, this performance advantage is consistently maintained throughout the entire hour of lead time, and we also notice the difference in performance relatively increasing for higher precipitation intensities.

5.2.4 Conclusion

Results have shown that all deep-learning-based easily outperform the naive persistence baseline, indicating that deep learning-based models are useful to deploy for the relatively unexplored volumetric nowcasting task. Additionally, we can conclude that adopting the multi-scale framework for this task boosts performance and reduces the memory footprint. Moreover, the proposed MS-3D PredRNN outperforming the MS-2D PredRNN baseline indicates that treating depth information as an additional spatial dimension instead of an image channel can help obtain nowcasts of higher quality. However, this advantage comes at the expense of a significant increase in computational memory and is only obtained when the models adopt the multi-scale architecture. Last, plotting the degradation of metric scores over lead time showed that predictive performance degrades heavily over time, especially for higher intensity thresholds.

Overall, we can conclude that our proposed model achieves top performance compared to the baselines. Specifically, this can be dedicated to the novel combination of adopting the multi-scale architecture and using 3D convolutions and 3D cell states. Our approach outperforms a competitive multi-channel baseline in predicting reflectivity with fewer false alarms and generates volumetric nowcasts with greater distributional similarity to our target volumes.

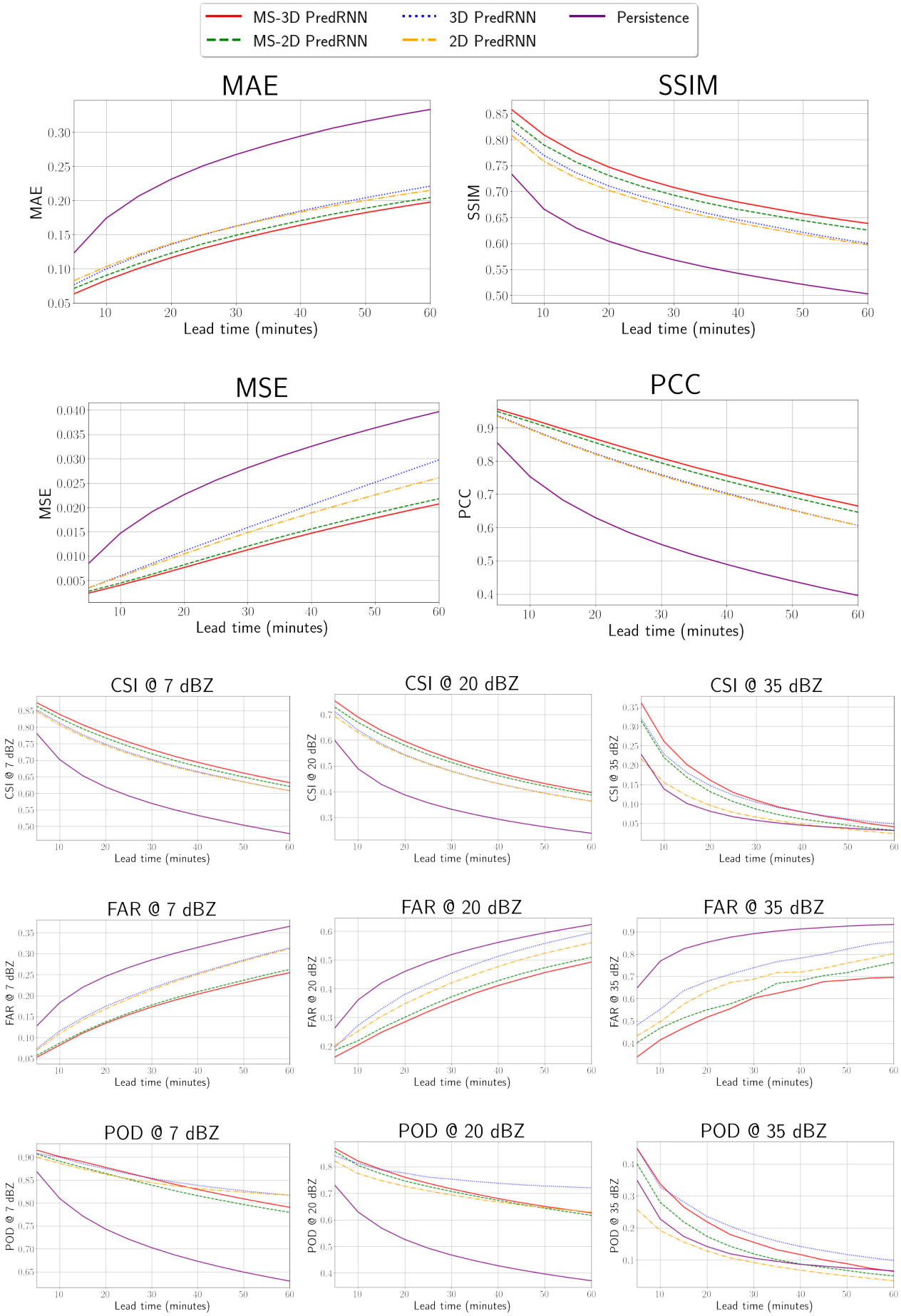


Figure 5.2: Test loss and metric scores over lead time.

5.3 Experiment 3: qualitative analysis

In our third experiment, a brief qualitative analysis of predictions is performed to obtain a visual understanding of the predictive performance of our model. For this purpose, we looked at forecasts of our proposed MS-3D PredRNN model in two scenarios: 1) when the model performed relatively well, i.e. when it strongly outperformed baselines in test loss scores, and 2) when the model performed relatively poor, i.e. when it did not outperform baselines as strongly. Two examples of the former scenario are shown in figure 5.3, and two examples of the latter scenario are shown in figure 5.4. Moreover, only predictions for the proposed MS-3D PredRNN model and the most competitive baseline, i.e. MS-2D PredRNN, are plotted here. More extensive visualizations, which include the other baselines, are shown in appendix A.

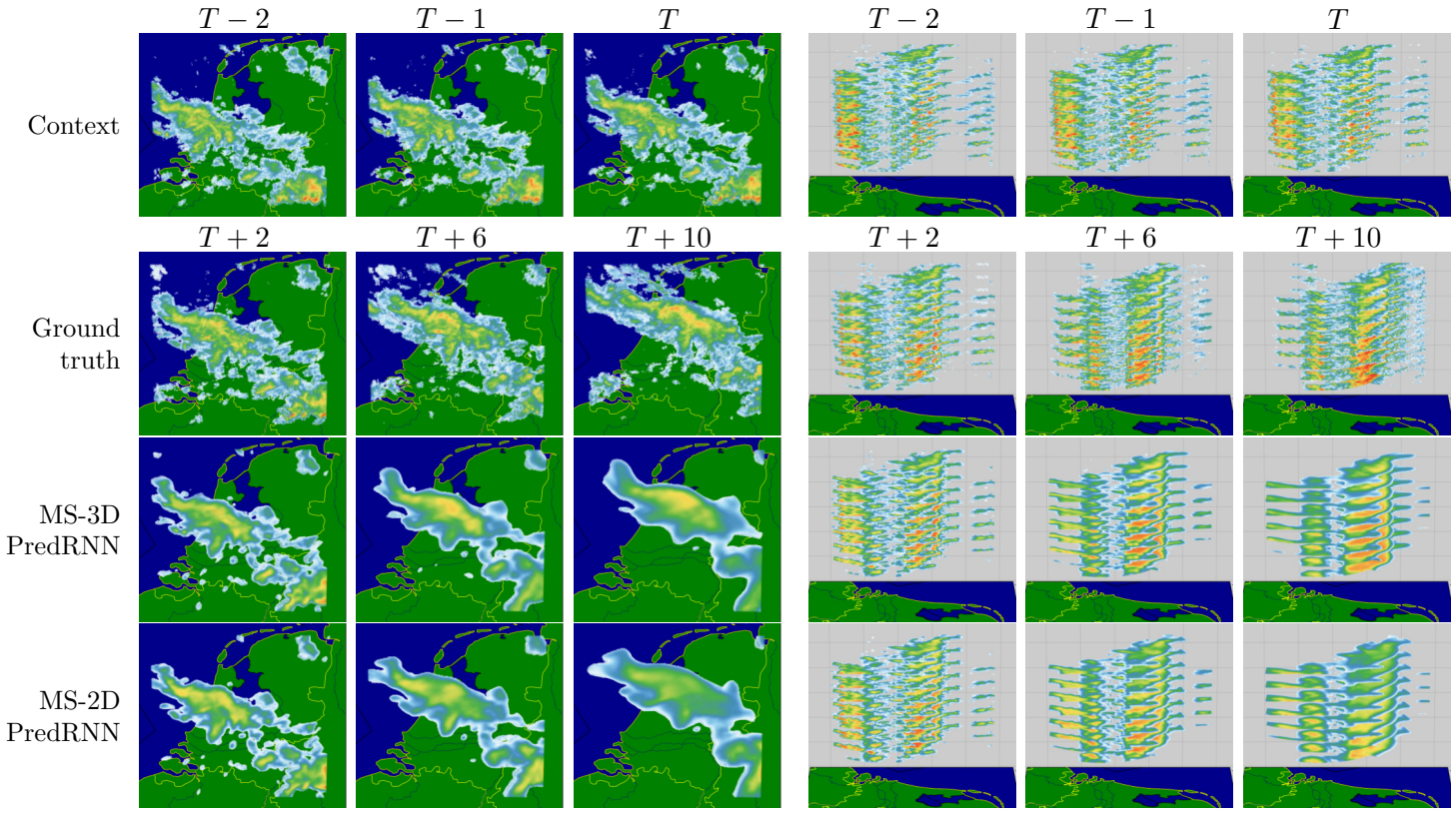
5.3.1 Results

First, we noticed that the loss scores of our model and the baselines were generally strongly correlated, indicating that when our model was relatively successful in creating accurate predictions, baselines were too. Therefore, we could not distinguish striking patterns when analyzing visualizations of sequences for which our model performed relatively well. The two examples plotted in figure 5.3 illustrate this: there is no clear difference in prediction quality visible, although our model seems to predict higher intensities slightly more accurately than the 2D baseline. Overall, we notice that predictions suffer from the problem of blurriness at further lead times, even though models were optimized using SSIM and balanced MAE.

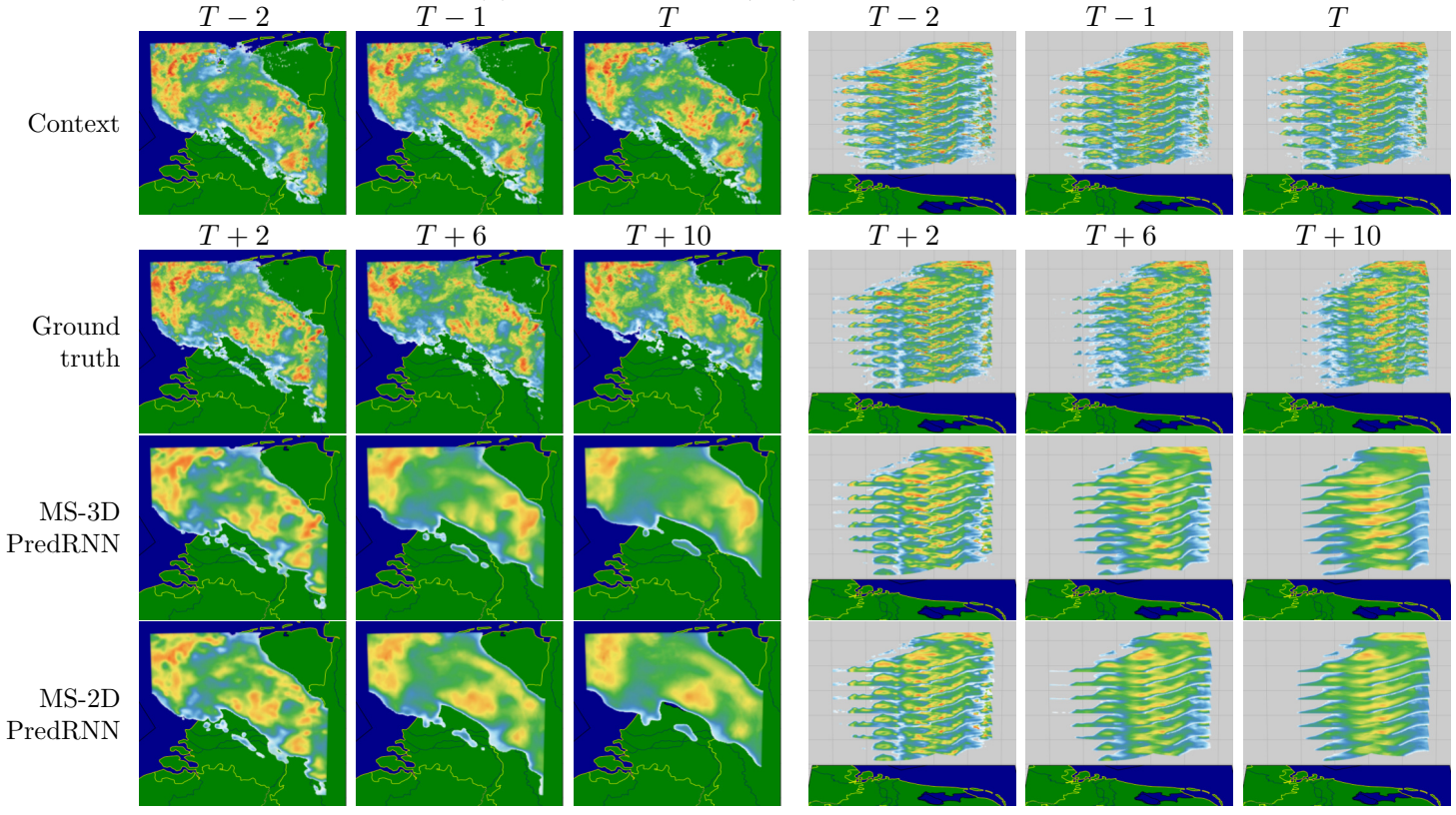
Second, considering scenarios in which our model performed relatively poorly, we noticed that our model had high loss scores when sequences contained a lot of heavy precipitation, as shown in figure 5.4a. For these instances, predictions become increasingly blurrier over time, once again indicating that our model has more difficulties in predicting severe precipitation patterns than low-intensity reflectivity. Additionally, it stood out that our model and all baseline models performed exceptionally poorly for a few outlier events, where even the persistence model outperformed them. When visualizing these sequences, it became clear that all these events contained noisy radar data, as illustrated in figure 5.4b. These events all had radar beams of extreme reflectivity intensity cluttering the radar volume with noise. Unsurprisingly, our model and all baseline models have difficulties predicting this noisy data.

5.3.2 Conclusion

Although we have concluded that our proposed model outperforms the MS-2D PredRNN model quantitatively in loss and metric scores, we could not observe a significant difference in prediction quality. Visualizations showed that our model still suffers from the problem of predicting blurry images further into the future. Moreover, visualizations showed that the model especially struggles to predict heavy precipitation events sharply. This confirms our earlier finding that the task of predicting rare but important high-intensity precipitation patterns proves more difficult than predicting low-intensity voxels.



(a) Time of event: 02/02/2021, 23:00



(b) Time of event: 21/08/2021, 23:25

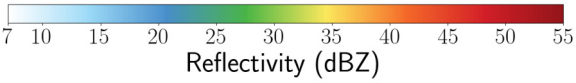
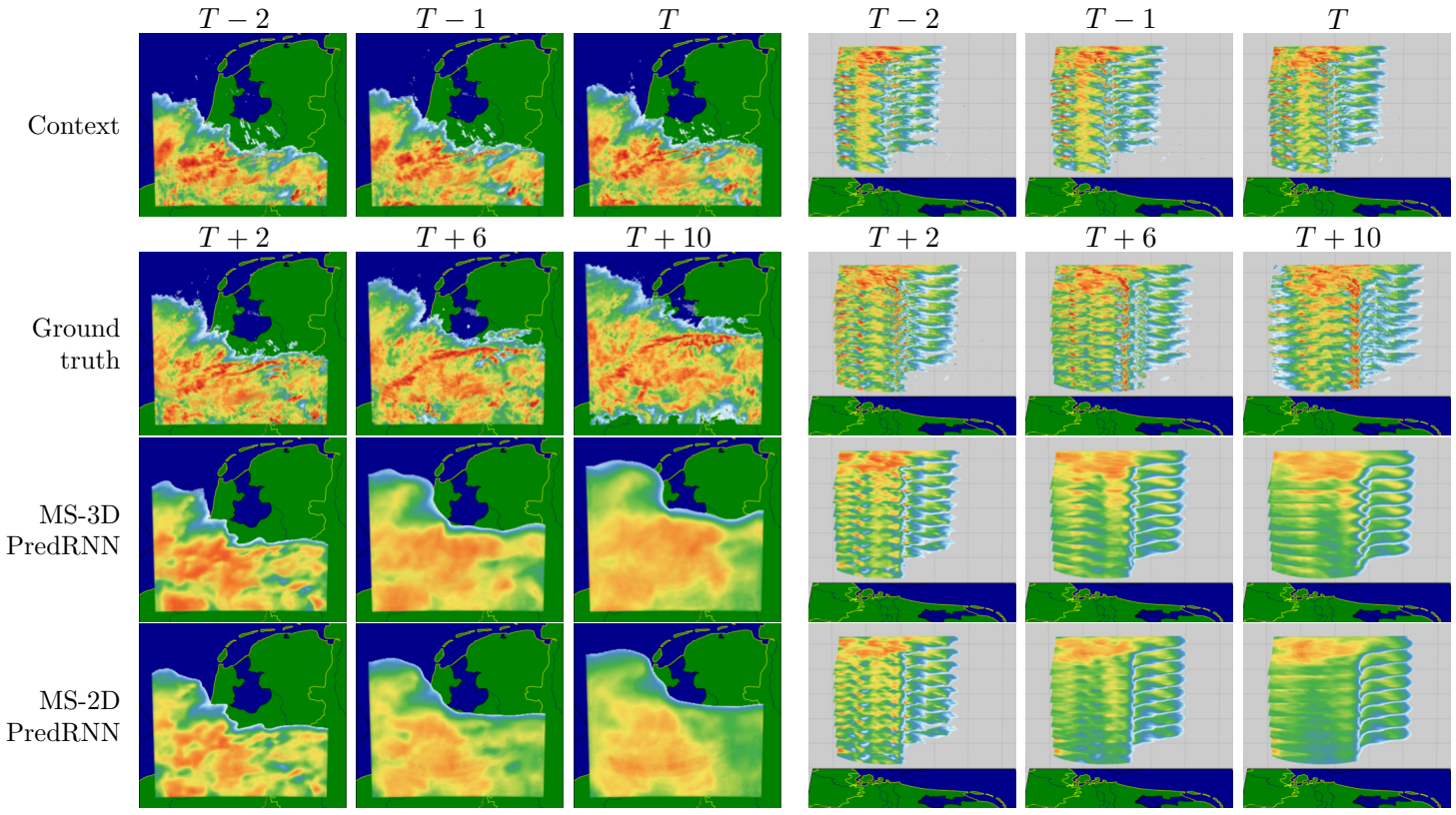
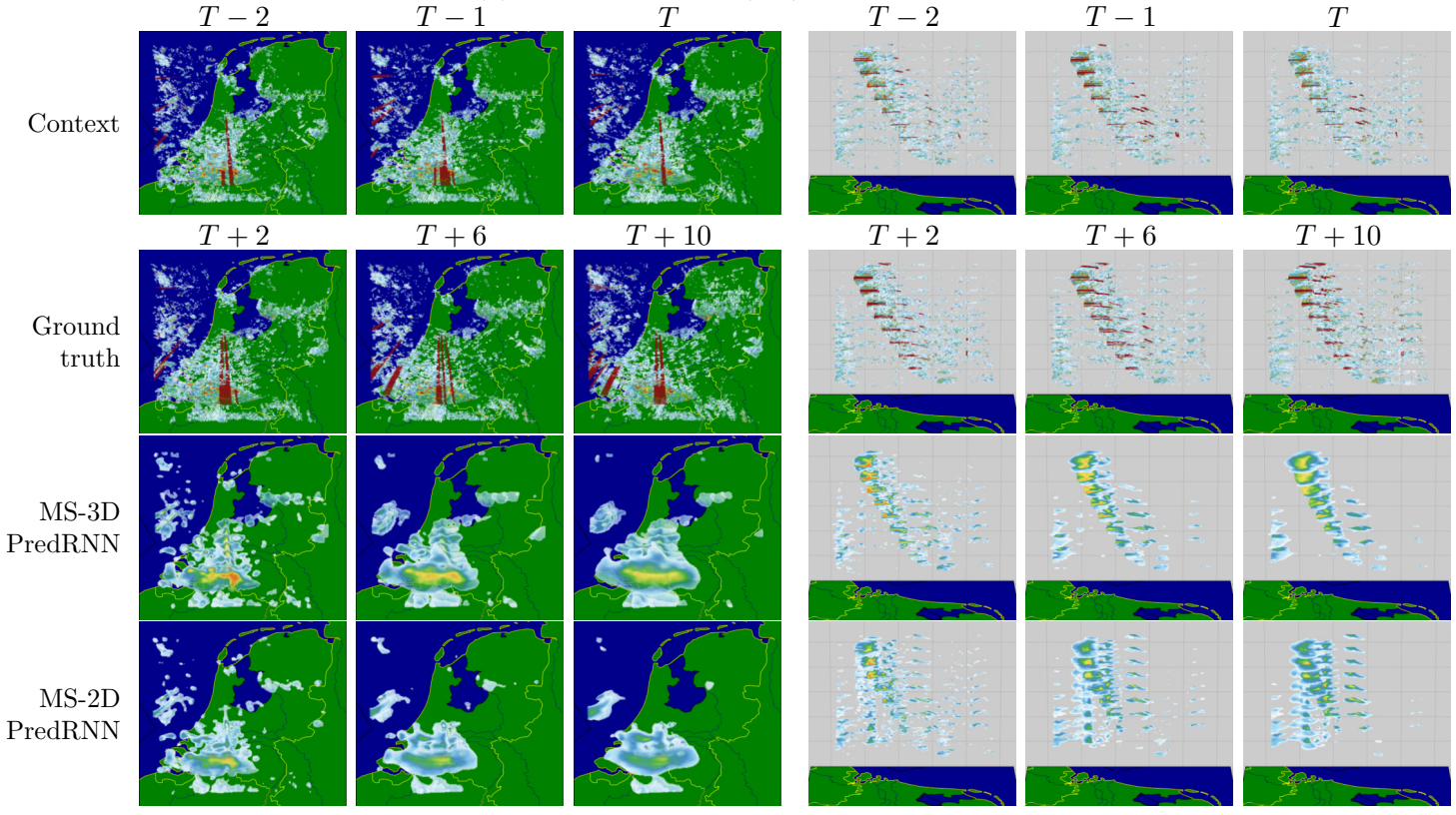


Figure 5.3: Visualization of sequences where the proposed MS-3D PredRNN model performed relatively well. The top view (left) and side view (right) are shown. For visualization purposes, only the last three timestamps of context and three future timesteps are displayed (with a lead time of 10, 30 and 50 minutes into the future).



(a) Time of event: 19/06/2021, 21:35



(b) Time of event: 09/09/2021, 01:10

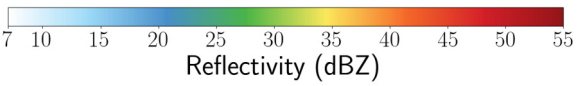


Figure 5.4: Visualization of sequences where the proposed MS-3D PredRNN model performed relatively poorly. The top view (left) and side view (right) are shown. For visualization purposes, only the last three timestamps of context and three future timesteps are displayed (with a lead time of 10, 30 and 50 minutes into the future).

	MAE↓	2D SSIM↑	MSE↓
MS-2D PredRNN 2D→2D	.1591	.6290	.0333
MS-2D PredRNN 3D→2D	.1545	.6274	.0313
MS-3D PredRNN 3D→2D	**.1510**	**.6357**	**.0306**

Table 5.5: Averaged test loss scores for the 2D→2D and 3D→2D experiments at 1500 m altitude. Note that models were not optimized for MSE. Best overall loss scores are highlighted in **bold**.

	CSI ↑			FAR ↓			POD ↑			PCC ↑
dBZ thresh.	**7**	**20**	**35**	**7**	**20**	**35**	**7**	**20**	**35**	**--**
MS-2D PredRNN 2D→2D	.7322	.5285	.0984	.1732	.3668	.6057	.8483	.7180	.1293	.7818
MS-2D PredRNN 3D→2D	.7359	.5366	.0972	**.1635**	.3489	.5951	.8433	.7154	.1262	.7944
MS-3D PredRNN 3D→2D	**.7425**	**.5437**	**.1165**	.1645	**.3479**	**.5909**	**.8544**	**.7317**	**.1585**	**.8011**

Table 5.6: Averaged test metrics scores for the 2D→2D and 3D→2D experiments at 1500 m altitude. Best overall metric scores are highlighted in **bold**.

5.4 Experiment 4: 3D to 2D predictions

In our final experiment, we regard the volumetric nowcasting task in a broader context by investigating if predicting in a three-dimensional space can benefit the industry's standard approach of creating predictions in a two-dimensional space. To make this comparison, we train a new multi-scale 2D PredRNN model on our 2D pseudoCAPPI dataset to predict pseudoCAPPI sequences. Furthermore, this model is the same as the previous MS-2D PredRNN model, but instead of taking multiCAPPIs as multi-channel images as input, this model takes pseudoCAPPI single-channel images as input at the KNMI's standard altitude of 1500 meters. For clarity, we call this model the MS-2D PredRNN 2D→2D model.

The goal is to compare how well this 2D→2D framework performs with our previously analyzed 3D→3D frameworks in predicting reflectivity values at a single altitude, which we denote as the 3D→2D setup. Moreover, because we know the altitude levels within our multiCAPPI volumes, we can compare the 2D→2D predictions with 3D→2D predictions by simply taking the corresponding 1500 meter 2D slice within predicted multiCAPPIs, similar to the concurrent works of [36, 25]. More sophisticated approaches of retraining or finetuning the 3D→2D models to optimize for the 1500 m altitude slice were explored, but surprisingly, this did not lead to any performance benefits.

To make a fair comparative analysis, we train the newly introduced 2D→2D model also for 25 epochs, using the same training regime as in experiment 5.2, except that we now have to include 2D SSIM instead of the previously used 3D SSIM in the loss function. In addition, because a pseudoCAPPI covers the entire 240x240 grid and a CAPPI slice at that altitude suffers from blindspots, we only compute loss and operational metrics over the area covered by the 1500-meter CAPPI slice. Loss and metric scores of this experiment are reported in tables 5.5 and 5.6. Note that we did not include 3D→2D predictions of the 2D and 3D PredRNN models, as we already concluded that both multi-scale variants outperform these models.

5.4.1 Results

The results obtained present some exciting findings; first, for the two 3D→2D models, B-MAE is slightly higher than reported previously in table 5.3, and SSIM scores seem to be significantly lower, indicating that the 1500 meter slice is relatively more difficult to predict than other slices within the multiCAPPI. However, it should be pointed out that we cannot fairly compare 2D SSIM scores with the previously reported 3D SSIM scores.

The second finding is more interesting. We note that the MS-2D PredRNN 3D→2D model does not outperform the 2D→2D counterpart. This indicates that when given a three-dimensional context, the multi-channel baseline does not predict the single altitude images better than when given only two-dimensional images as context. However, we do observe the proposed MS-3D PredRNN model benefitting from considering 3D data, reflected by the model outperforming the 2D→2D model, achieving top performance on nearly every evaluation metric.

5.4.2 Conclusion

The results of the fourth experiment illustrate that exploiting volumetric reflectivity data can improve the skill of predicting single altitude reflectivity images. This finding provides strong support for two of our hypotheses: 1) that nowcasting models can benefit from the additional information of vertical dynamics built into multiCAPPI sequences compared to strictly using two-dimensional pseudoCAPPI data, and 2) that models only demonstrate this performance benefit when they are explicitly designed to model vertical information as a third spatial dimension. By designing our proposed model this way, it can effectively model the 3D evolution of precipitation, resulting in its ability to predict reflectivity values at a 1500-meter altitude more reasonably than the baseline model considering only 2D data.

Chapter 6

Conclusion and discussion

In this thesis, we aimed to challenge the convention in the field of nowcasting of using two-dimensional radar data to create forecasts. Concretely, we examined two research questions: 1) whether we could effectively extend the nowcasting task to 3D radar data using a deep learning-based model and 2) whether we could improve predictions on a single altitude when vertical information of radar data was considered.

Considering the first research question, quantitative results have demonstrated that our proposed model outperformed all baselines in loss and metric scores. This indicates that a deep learning-based model can successfully perform the volumetric nowcasting task. Moreover, results have shown that explicitly treating the vertical dimension of radar data as a third spatial dimension in our proposed model can help obtain better predictions than a multi-channel approach. However, this boost in performance comes at the expense of a substantial increase in memory usage. Furthermore, we could only observe this advantage when precipitation dynamics were captured on multiple spatial scales using the multi-scale architecture of [19].

Regarding the second research question, a comparative analysis of predicting single altitude radar images using 3D or 2D context demonstrated that exploiting vertical precipitation information can help to improve predictions in the 2D space. Moreover, this result further supports the hypothesis that nowcasting techniques can benefit from considering the volumetric nature of radar data. Additionally, we concluded that this benefit was only noticeable when our model explicitly modelled the radar data in three spatial dimensions.

The experimental results strongly support the effectiveness of our proposed approach in utilizing 3D radar data for nowcasting. The success of our approach can be attributed to the innovative use of a multi-scale architecture combined with the application of 3D convolutions and 3D cell states in our framework. Overall, we can conclude that incorporating radar data from multiple altitude levels has the potential to improve nowcasts, in contrast to the current practice of relying solely on 2D precipitation maps. However, this thesis can only be considered explorative work in a relatively uncharted domain. Therefore, we also discuss some limitations of our approach and provide recommendations for future research.

First, the dataset created in this thesis is relatively small in the context of deep learning. Moreover, we made specific implementation choices in processing the volumetric nature of radar data. Because plenty of historical data is available, future research could focus on extending the 3D nowcasting task to a larger dataset covering more diverse precipitation events. Future work could also focus on modelling radar completely differently. For example, we hypothesize that an approach in which radar data is kept in its raw unstructured form could be interesting to investigate, i.e. by extending the predictive task from Cartesian grids to point clouds.

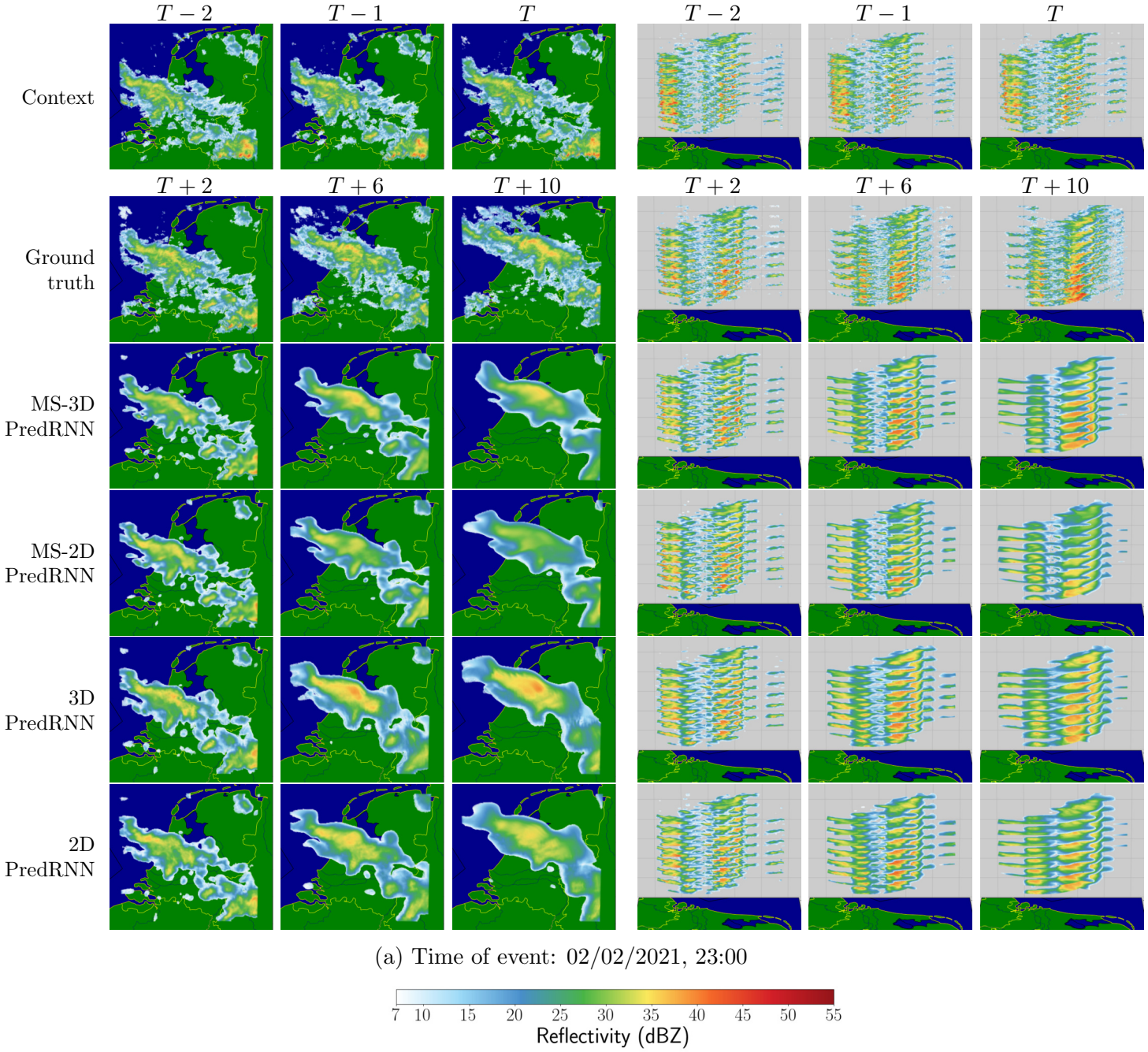
Second, our proposed approach still suffers from high memory complexity, which could be problematic if the nowcasting task is extended to a larger grid. Therefore, future research could focus on comparing the effectiveness of our proposed multi-scale approach for large-scale volumetric nowcasting with alternative approaches, such as combining CNN encoders and decoders with a ConvLSTM model as demonstrated in [36]

Third, quantitative and qualitative results have demonstrated that our model still suffers from the long-tail distribution of our data, meaning that our model has difficulties with correctly predicting rare but important high-intensity reflectivity. Moreover, performance degrades heavily over time, reflected by the model struggling to produce sharp and realistic nowcasts. Therefore, following the recent interesting work of [30], future work could explore if using Generative Adversarial Networks could be an effective approach in producing accurate and realistic volumetric nowcasts.

Lastly, our work is limited to predicting future radar sequences and does not focus on predicting actual surface precipitation. However, knowing how much rain will fall on the ground is arguably most important in precipitation nowcasting, as this directly affects most users relying on these nowcasts. Therefore, future work could focus on producing a framework for translating volumetric radar data to surface precipitation, e.g. using surface rainfall measurements.

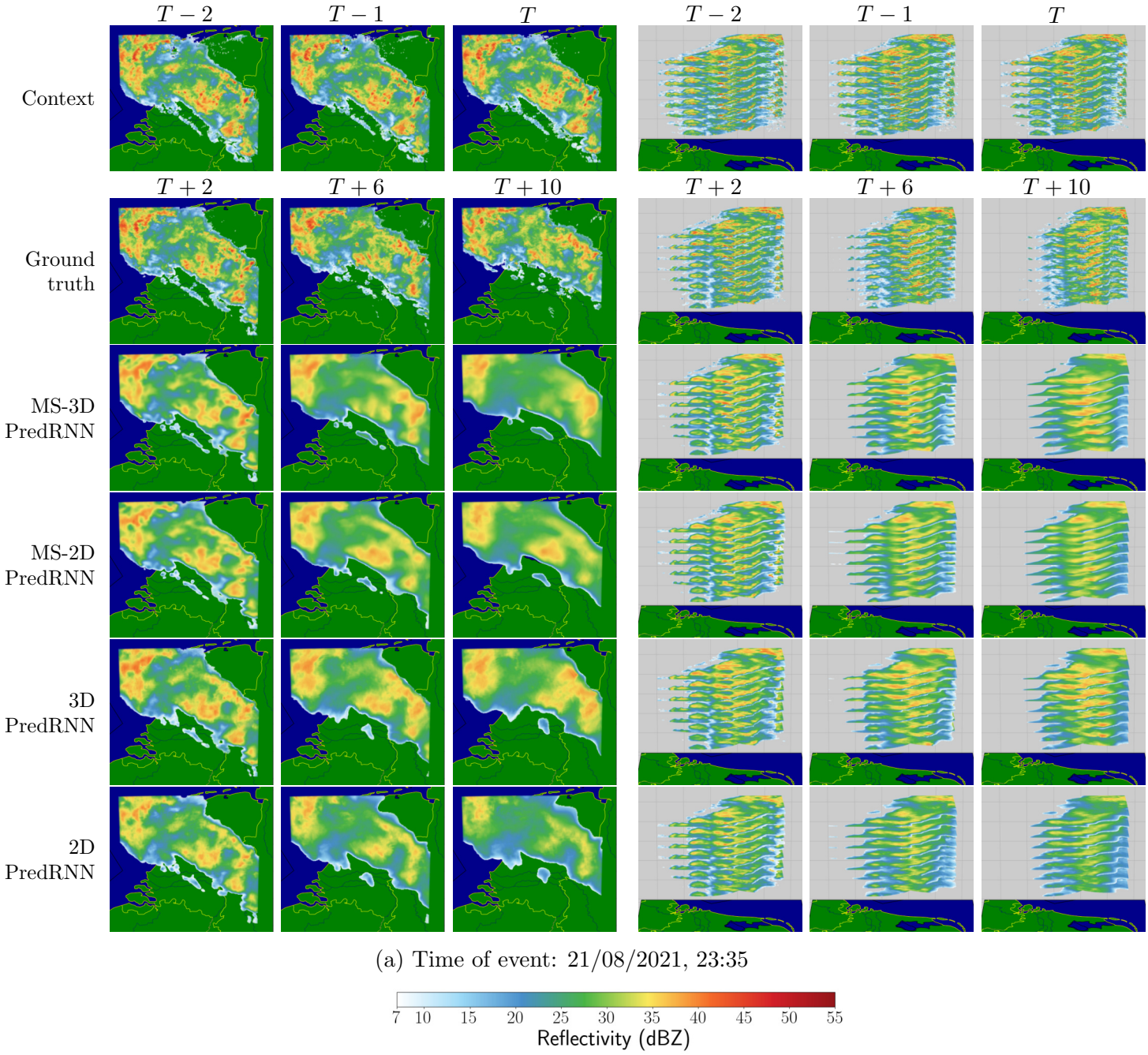
To conclude, the research in this thesis focused on incorporating the volumetric aspect of radar data into deep learning-based advancements within the nowcasting field. Our findings agree with the concurrent work of [25, 36] and strongly suggest that there are benefits to achieve from incorporating vertical dynamics of rainfall in a predictive model. However, it is up to future work to determine if using volumetric radar data could become the new convention in the rapidly evolving field of nowcasting research.

Appendix A Extended visualizations



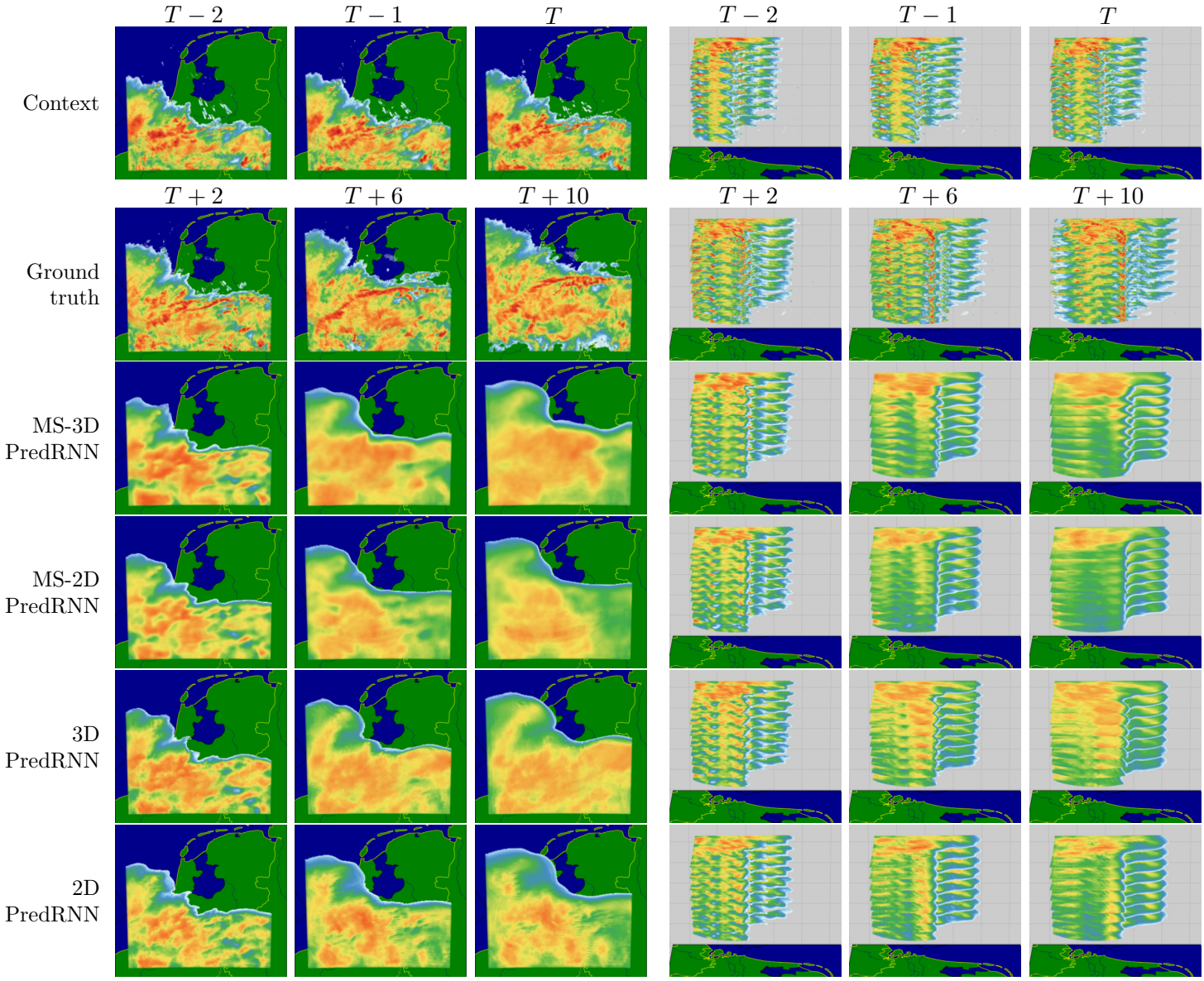
(a) Time of event: 02/02/2021, 23:00

Figure A.1: Extended visualization where the proposed MS-3D PredRNN model performed relatively well, including 3D PredRNN and 2D PredRNN baselines. For visualization purposes, only the last three timestamps of context and three future timesteps are displayed (with a lead time of 10, 30 and 50 minutes into the future).



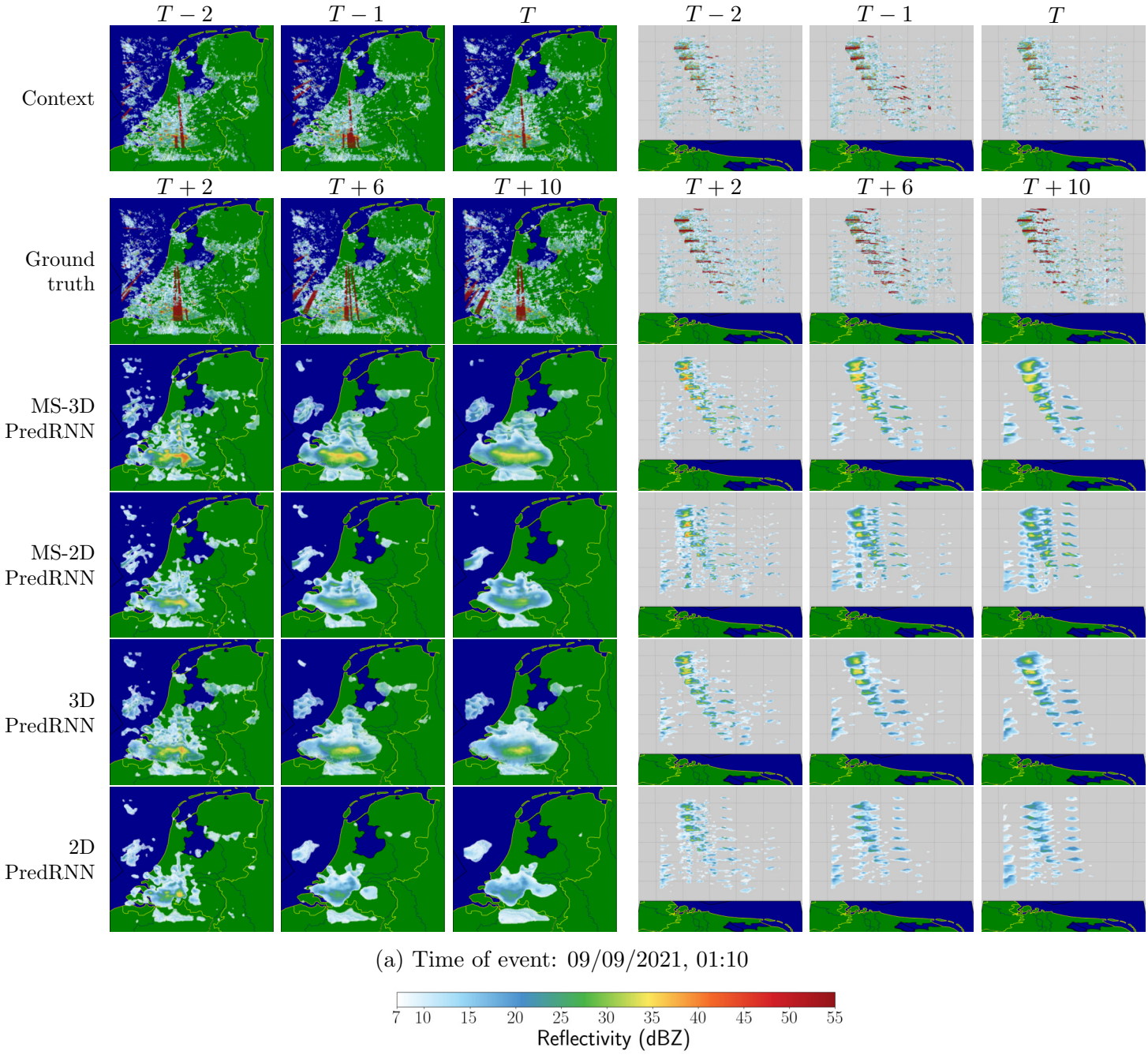
(a) Time of event: 21/08/2021, 23:35

Figure A.2: Extended visualization where the proposed MS-3D PredRNN model performed relatively well, including 3D PredRNN and 2D PredRNN baselines. For visualization purposes, only the last three timestamps of context and three future timesteps are displayed (with a lead time of 10, 30 and 50 minutes into the future).



(a) Time of event: 19/06/2021, 21:35

Figure A.3: Extended visualization where the proposed MS-3D PredRNN model performed relatively poorly, including 3D PredRNN and 2D PredRNN baselines. For visualization purposes, only the last three timestamps of context and three future timesteps are displayed (with a lead time of 10, 30 and 50 minutes into the future).



(a) Time of event: 09/09/2021, 01:10

Figure A.4: Extended visualization where the proposed MS-3D PredRNN model performed relatively poorly, including 3D PredRNN and 2D PredRNN baselines. For visualization purposes, only the last three timestamps of context and three future timesteps are displayed (with a lead time of 10, 30 and 50 minutes into the future).

Bibliography

[1] Shreya Agrawal, Luke Barrington, Carla Bromberg, John Burge, Cenk Gazen, and Jason Hickey. Machine learning for precipitation nowcasting from radar images. *arXiv preprint arXiv:1912.12132*, 2019.

[2] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)*, pages 1–6, 2017.

[3] Jisk Attema, Alexander Bakker, Jules Beersma, Janette Bessembinder, Reinout Boers, Theo Brandsma, Henk van den Brink, Sybren Drijfhout, Henk Eskes, Rein Haarsma, et al. Knmi'14: Climate change scenarios for the 21st century–a netherlands perspective. *KNMI: De Bilt, The Netherlands*, 2014.

[4] Peter Bauer, Alan Thorpe, and Gilbert Brunet. The quiet revolution of numerical weather prediction. *Nature*, 525(7567):47–55, 2015.

[5] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.

[6] Neill E Bowler, Clive E Pierce, and Alan W Seed. Steps: A probabilistic precipitation forecasting scheme which merges an extrapolation nowcast with downscaled nwp. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, 132(620):2127–2155, 2006.

[7] Hongshu Che, Dan Niu, Zengliang Zang, Yichao Cao, and Xisong Chen. Ed-drap: Encoder–decoder deep residual attention prediction network for radar echoes. *IEEE Geoscience and Remote Sensing Letters*, 19:1–5, 2022.

[8] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734. ACL, 2014.

[9] Bing Gong, Michael Langguth, Yan Ji, Amirpasha Mozaffari, Scarlet Stadtler, Karim Mache, and Martin G Schultz. Temperature forecasting by deep learning methods. *Geoscientific Model Development*, 15(23):8931–8956, 2022.

[10] Maik Heistermann, S Jacobi, and T Pfaff. An open source library for processing weather radar data (wradlib). *Hydrology and Earth System Sciences*, 17(2):863–871, 2013.

[11] Alexander Heye, Karthik Venkatesan, and Jericho Cain. Precipitation nowcasting: Leveraging deep recurrent convolutional neural networks. In *Proceedings of the Cray User Group (CUG)*, volume 2017. Cray User Group Washington, 2017.

[12] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[13] I Holleman, HRA Wessels, JRA Onvlee, and SJM Barlag. Development of a hail-detection-product: S10: Deep convection. *Physics and Chemistry of the Earth, Part B: Hydrology, Oceans and Atmosphere*, 25(10-12):1293–1297, 2000.

[14] RO Imhoff, CC Brauer, A Overeem, AH Weerts, and R Uijlenhoet. Spatial and temporal evaluation of radar rainfall nowcasting techniques on 1,533 events. *Water Resources Research*, 56(8):e2019WR026723, 2020.

[15] IPCC. *Climate Change 2014: Synthesis Report. Contribution of Working Groups I, II and III to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*. IPCC, Geneva, Switzerland, 2014.

[16] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[17] S Krämer. Quantitative radar data processing for rainfall forecasting and urban drainage. *dissertation*, 2008.

[18] Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Alexander Pritzel, Suman Ravuri, Timo Ewalds, Ferran Alet, Zach Eaton-Rosen, et al. Graphcast: Learning skillful medium-range global weather forecasting. *arXiv preprint arXiv:2212.12794*, 2022.

[19] Zhifeng Ma, Hao Zhang, and Jie Liu. Ms-rnn: A flexible multi-scale framework for spatiotemporal predictive learning. *arXiv preprint arXiv:2206.03010*, 2022.

[20] J. S. Marshall and W. Mc K. Palmer. The distribution of raindrops withs size. *Journal of Atmospheric Sciences*, 5(4):165 – 166, 1948.

[21] NVIDIA, Péter Vingelmann, and Frank H.P. Fitzek. Cuda, release: 10.2.89, 2020.

[22] Shigenori Otsuka, Gulanbaier Tuerhong, Ryota Kikuchi, Yoshikazu Kitano, Yusuke Taniguchi, Juan Jose Ruiz, Shinsuke Satoh, Tomoo Ushio, and Takemasa Miyoshi. Precipitation nowcasting with three-dimensional space–time extrapolation of dense and frequent phased-array weather radar observations. *Weather and Forecasting*, 31(1):329–340, 2016.

[23] Aart Overeem, TA Buishand, and Iwan Holleman. Extreme rainfall analysis and estimation of depth-duration-frequency curves using weather radar. *Water resources research*, 45(10), 2009.

[24] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[25] Peter Pavlík, Viera Rozinajová, and Anna Bou Ezzeddine. Radar-based volumetric precipitation nowcasting: A 3d convolutional neural network with u-net architecture. 2021.

[26] Rachel Prudden, Samantha Adams, Dmitry Kangin, Niall Robinson, Suman Ravuri, Shakir Mohamed, and Alberto Arribas. A review of radar-based nowcasting of precipitation and applicable machine learning techniques. *arXiv preprint arXiv:2005.04988*, 2020.

[27] S. Pulkkinen, D. Nerini, A. A. Pérez Hortal, C. Velasco-Forero, A. Seed, U. Germann, and L. Foresti. Pysteps: an open-source python library for probabilistic precipitation nowcasting (v1.0). *Geoscientific Model Development*, 12(10):4185–4219, 2019.

[28] Seppo Pulkkinen, V Chandrasekar, Annakaisa von Lerber, and Ari-Matti Harri. Nowcasting of convective rainfall using volumetric radar observations. *IEEE Transactions on Geoscience and Remote Sensing*, 58(11):7845–7859, 2020.

[29] Seppo Pulkkinen, Daniele Nerini, Andrés A Pérez Hortal, Carlos Velasco-Forero, Alan Seed, Urs Germann, and Loris Foresti. Pysteps: an open-source python library for probabilistic precipitation nowcasting (v1. 0). *Geoscientific Model Development*, 12(10):4185–4219, 2019.

[30] Suman Ravuri, Karel Lenc, Matthew Willson, Dmitry Kangin, Remi Lam, Piotr Mirowski, Megan Fitzsimons, Maria Athanassiadou, Sheleem Kashem, Sam Madge, et al. Skilful precipitation nowcasting using deep generative models of radar. *Nature*, 597(7878):672–677, 2021.

[31] RE Rinehart and ET Garvey. Three-dimensional storm motion detection by conventional weather radar. *Nature*, 273(5660):287–289, 1978.

[32] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[33] Martin G Schultz, Clara Betancourt, Bing Gong, Felix Kleinert, Michael Langguth, Lukas Hubert Leufen, Amirpasha Mozaffari, and Scarlet Stadtler. Can deep learning beat numerical weather prediction? *Philosophical Transactions of the Royal Society A*, 379(2194):20200097, 2021.

[34] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28, 2015.

[35] Xingjian Shi, Zhihan Gao, Leonard Lausen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Deep learning for precipitation nowcasting: A benchmark and a new model. *Advances in neural information processing systems*, 30, 2017.

[36] Nengli Sun, Zeming Zhou, Qian Li, and Jinrui Jing. Three-dimensional gridded radar echo extrapolation for convective storm nowcasting based on 3d-convlstm model. *Remote Sensing*, 14(17):4256, 2022.

[37] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[38] Quang-Khai Tran and Sa-kwang Song. Computer vision in precipitation nowcasting: Applying image quality assessment metrics for training deep neural networks. *Atmosphere*, 10(5):244, 2019.

[39] Quang-Khai Tran and Sa-kwang Song. Multi-channel weather radar echo extrapolation with convolutional recurrent neural networks. *Remote Sensing*, 11(19):2303, 2019.

[40] Kevin Trebing, Tomasz Stanczyk, and Siamak Mehrkanoon. Smaat-unet: Precipitation nowcasting using a small attention-unet architecture. *Pattern Recognition Letters*, 145:178–186, 2021.

[41] R Uijlenhoet, SH Van der Wielen, and A Berne. Uncertainties in rainfall retrievals from ground-based weather radar: overview, case study, and simulation experiment. *Hydrology and Earth System Sciences Discussions*, 3(4):2385–2436, 2006.

[42] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. *Advances in neural information processing systems*, 29, 2016.

[43] Cong Wang, Ping Wang, Pingping Wang, Bing Xue, and Di Wang. Using conditional generative adversarial 3-d convolutional neural network for precise radar extrapolation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14:5735–5749, 2021.

[44] Yunbo Wang, Zhifeng Gao, Mingsheng Long, Jianmin Wang, and S Yu Philip. Predrnn++: Towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning. In *International Conference on Machine Learning*, pages 5123–5132. PMLR, 2018.

[45] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S Yu. Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[46] Yunbo Wang, Haixu Wu, Jianjin Zhang, Zhifeng Gao, Jianmin Wang, S Yu Philip, and Mingsheng Long. Predrnn: A recurrent neural network for spatiotemporal predictive learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):2208–2225, 2022.

[47] Yunbo Wang, Jianjin Zhang, Hongyu Zhu, Mingsheng Long, Jianmin Wang, and Philip S Yu. Memory in memory: A predictive neural network for learning higher-order non-stationarity from spatiotemporal dynamics. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9154–9162, 2019.

[48] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[49] Wang-chun Woo and Wai-kin Wong. Operational application of optical flow techniques to radar-based rainfall nowcasting. *Atmosphere*, 8(3):48, 2017.

[50] Wang-chun Woo and Wai-kin Wong. Operational application of optical flow techniques to radar-based rainfall nowcasting. *Atmosphere*, 8(3), 2017.

[51] Günther Zängl, Daniel Reinert, Pilar Rípodas, and Michael Baldauf. The icon (icosahedral non-hydrostatic) modelling framework of dwd and mpi-m: Description of the non-hydrostatic dynamical core. *Quarterly Journal of the Royal Meteorological Society*, 141(687):563–579, 2015.

[52] Kai Zeng and Zhou Wang. 3d-ssim for video quality assessment. In *2012 19th IEEE international conference on image processing*, pages 621–624. IEEE, 2012.