DTU Compute
Department of Applied Mathematics and Computer Science

Rainfall forecasting from Radar Images using Deep Learning

Master Thesis

Rainfall forecasting from Radar Images using Deep Learning
Radar regnprognoser baseret på deep learning

Master Thesis
February,2022

By
Md Alamgir Kabir

Cover photo: Vibeke Hempler, 2012
Published by: DTU, Department of Applied Mathematics and Computer Science, Richard Petersens Plads, Building 324, 2800 Kgs. Lyngby Denmark
www.compute.dtu.dk
ISSN: [0000-0000] (electronic version)
ISBN: [000-00-0000-000-0] (electronic version)

ISSN: [0000-0000] (printed version)
ISBN: [000-00-0000-000-0] (printed version)

Approval

This thesis has been prepared over five months at the Department of Environmental Engineering, at the Technical University of Denmark, DTU, in partial fulfilment for the degree Master of Science in Engineering, MSc Eng.

It is assumed that the reader has a basic knowledge in the areas of machine learning and statistics.

Md Alamgir Kabir - s193074

.....

Signature

.....

Date

Abstract

Precipitation nowcasting is a technique for predicting rainfall intensity in a local region at high spatiotemporal resolutions and short lead times. In this study, the precipitation nowcasting task has been viewed from a machine learning perspective where forecasting has been treated as an image-to-image translation problem using the power of convolutional neural networks (CNN). The model architectures have been inspired by UNet for its successful application in image processing. The models were trained to predict precipitation nowcasts for a 1 hour lead time, using 14 months of radar observations and nowcasts provided by the Danish Meteorological Institute (DMI). Experiments on the test dataset demonstrate that the UNet-based precipitation nowcast model with custom loss function outperforms the traditional radar-based nowcasting algorithm, CO-TREC (Continuity of TREC vectors).

Acknowledgements

Md Alamgir Kabir, MSc Mathematical Modelling, DTU

Roland Löwe, Associate Professor, Department of Environmental Engineering, DTU
Supervisor

Karsten Arnbjerg Nielsen, Professor, Department of Environmental Engineering, DTU
Supervisor

Jonas Wied Pedersen, Industrial Postdoc, Danish Meteorological Institute and Department of Environmental Engineering, DTU
Supervisor

Abbreviations

Adam	Adaptive Moment Estimation
ANN	Artificial Neural Network
CNN	Convolution Neural Network
ConvLSTM	Convolution Long Short Term Memory
CO-TREC	Continuity of TREC vectors
CSI	Critical Success Index
DL	Deep Learning
DMI	Danish Meteorological Institute
DNN	Deep Neural Network
FAR	False Alarm Rate
GPU	Graphics Processing Unit
HPC	High Performance Computing
LSTM	Long Short Term Memory
MAE	Mean Absolute Error
MRMS	Multi-Radar Multi-Sensor
MSE	Mean Squared Error
NMSE	Normalised Mean Squared Error
NWP	Numerical Weather Prediction
ResNet	Residual Network
RNN	Recurent Neural Network
ROVER	Real-time Optical flow by Variational methods for Echoes of Radar
TrajGRU	Trajectory Gated Recurrent Unit
TREC	Tracking Radar Echoes by Correlation

Contents

Preface . . . ii
Abstract . . . iii
Acknowledgements . . . iv
Abbreviations . . . v

1 Introduction **1**
1.1 Importance of nowcasting . . . 1
1.2 Approaches for precipitation nowcasting . . . 1
1.3 Scope and Goal . . . 2

2 Theory **3**
2.1 What is Artificial Intelligence, Machine Learning and Deep Learning . . . 3
2.2 Artificial Neural Networks . . . 4
2.3 Training process of a Neural Network . . . 8
2.4 DNN architecture . . . 9
2.5 Nowcasting of radar rainfall . . . 10

3 Data **15**
3.1 Data pre-processing . . . 15
3.2 Datasets . . . 16

4 Methods **19**
4.1 Modelling . . . 19
4.2 Parameter Optimization . . . 24
4.3 Research Software . . . 25
4.4 Model Evaluation . . . 25

5 Results **29**
5.1 Qualitative Evaluation . . . 29
5.2 Evaluation scores . . . 34
5.3 Computational requirements and model losses . . . 39

6 Discussion **43**
6.1 Discussion on modelling and results . . . 43
6.2 Limitations . . . 43
6.3 Future works . . . 44

7 Conclusions **45**

Bibliography **47**

A Appendix **49**

1 Introduction

Precipitation nowcasting is a forecast that explains the precipitation intensity and shows the direction and speed of rain for a specific area. The task of precipitation nowcasting is to accurate prediction of rainfall for a short period typically from current time to a couple of hours ahead. Generally, precipitation nowcasting is performed based on some kind of observation such as radar echo maps, satellite maps or rain gauge, etc. The term nowcasting refers to the forecasts of precipitation in a local region for a very short term period of time and the widely accepted nowcasting range refers to the 0–6 hours(Shi et al. 2017; Sun et al. 2014).

1.1 Importance of nowcasting

Precipitation nowcasts have a great impact on our everyday lives. It helps the civil community in planning everyday activities, outdoor events and emergency planning. Natural extremes can not be avoided or prevented but the risk can be mitigated by providing early warning. Natural local phenomena like debris flows, avalanches or landslides, urban floods or flash floods occur on very shorter time scales within few minutes to couple of hours after severe rainfall (Nerini 2019). Short term precipitation nowcasts can plays a very important role in here. Precipitation nowcasts are important tools for not only the safety of individuals, but also for sectors like urban water management (i.e. in their work Jóhannesson et al. 2021, Thorndahl et al. 2013 and Achleitner et al. 2009 have been shown that precipitation nowcast can be used in modelling for flow forecasting in urban drainage systems), agriculture and aviation etc.

1.2 Approaches for precipitation nowcasting

Precipitation nowcasting is an old research topic in hydro-meteorological community and still, it is an active research topic. The techniques for precipitation nowcasts have been developed over time. A review article of present nowcasting techniques can be found in Sun et al. 2014. In this article they proposed that the traditional nowcasting systems can be classified broadly into two categories (a) the traditional *numerical weather prediction* (NWP) based method and (b) the radar echo extrapolation based method.

NWP method relies on complex mathematical models and it has been mainly used for synoptic and mesoscale weather prediction in the context of radar-based nowcasting (Prudden et al. 2020). It also performs long simulation and expensive computation which require high computing power. As a result it takes long time to produce the results. Therefore it is less suitable for nowcasting (Trebing et al. 2021). On the other hand in the radar echo extrapolation based method, the radar echoes are extrapolated in space and time to predict the future radar echoes. This techniques originates from the pattern recognition techniques known as tracking radar echoes by correlation (TREC)(Mecklenburg et al. 2000). Using TREC, future radar echoes are predicted by calculating the correlation of radar echoes with respect to several previous moments. Extrapolation based methods are not computational heavy they produce faster results than NWP. Moreover, the heuristic extrapolation based precipitation nowcasts outperforms NWP forecasts at short lead times (Sun et al. 2014).

In their review article Sun et al. 2014 also mentioned that *the current observational networks primarily designed for synoptic forecasting generally cannot provide the environmental conditions with the temporal and spatial resolution required by nowcasting*. Recent

advances in artificial neural networks (ANNs) and the availability of high quality radar data creates an oppurtuanity for machine learning to contribute in nowcasting. Deep learning (DL) have already been applied into different field including computer vision, medical image analysis, bioinformatics and other scientific areas where they have shown very promising results (Wikipedia contributors 2021a).

1.3 Scope and Goal

The overall goal of this project is to propose deep machine learning models that can accurately predict heavy rainfall in Denmark or specific areas of Denmark for a lead time of **1** hour as well as that outperforms the Danish Meteorological Institute's (DMI) traditional nowcasting system. Therefore, the research questions for this project are
(a) How can DL models be used for rainfall prediction and which type of models may be useful for this purpose?
(b) Is it possible to improve existing nowcast algorithms with deep learning?
(c) A model provides better prediction, does loss function weighting improve the results?
(d) Is it better to let deep learning track the rainfall movement or should it learn the residue?

The rest of this project has been organized as follows. In chapter 2 basic background about machine learning (ML), deep learning (DL) and artificial neural networks (ANN) as well as reviews of some previous DL-based method on precipitation nowcasts has been described. Chapter 3 represents the different datasets used in this work and how the data has been processed. Chapter 4 describes the proposed model architectures as well as model evaluation criteria. Chapter 5 and 6 presents results from the trained models and discussions about the results respectively. Lastly, some conclusive remarks has been given in chapter 7.

2 Theory

In this chapter, some background information about deep learning, machine learning, artificial neural networks, and their different components, as well as some information on a traditional radar-based extrapolation nowcast algorithm and review of DL-based method in precipitation nowcast has presented. In the following sections (section 2.1 to subsection 2.4.2) from the basics of artificial intelligence to a special type of convolution neural network called U-Net is described which is aimed to solve image-to-image translation problems. Then in subsection 2.5.1 a widely used radar-based extrapolation nowcast algorithm, CO-TREC (Continuity of TREC vectors) is described. Finally, in subsection 2.5.2 review of DL-based method in precipitation nowcast has been described.

2.1 What is Artificial Intelligence, Machine Learning and Deep Learning

One can easily think about artificial intelligence, machine learning, neural networks, and deep learning visually from the Russian nesting dolls in figure 2.1.

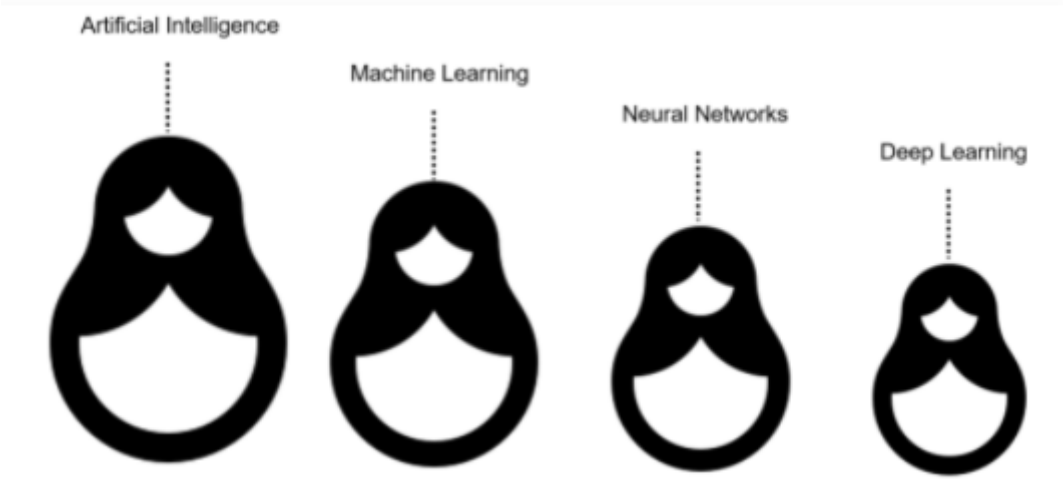


Figure 2.1: Visualizations of AI,ML,NN and DL. (IBM Cloud Education 2020)

Deep learning is a subset of machine learning, which is a subset of artificial intelligence (AI). Artificial intelligence is the science of intelligent machines or programs that allow machines to think and act like humans. Machine learning is a set of algorithms that have been trained on structured data to learn and act on that data over time. Deep learning is inspired by the structure of the human brain and the backbone of deep learning algorithms are comprised of multi-layered neural networks. Deep learning architectures have been applied in a variety of fields including computer vision, speech recognition, natural language processing, machine translation, bioinformatics, drug design, medical image analysis, material inspection, and many other areas where they produced comparable results also in some particular cases they perform better than human experts level of performance (Wikipedia contributors 2021a).

2.2 Artificial Neural Networks

Artificial neuronal network, often called ANN, is a computational model of the nervous system that is inspired by biological neural networks.The ANN consists of layers of nodes, each layer performing a different task. It is designed to process vast amounts of data through simulated neurons. It is popular for its ability to solve complex and non-linear problems and learning from both examples and feedback.

2.2.1 Nodes

It is the basic computation unit in a neural network which is also known as neuron or perceptron.An individual node may receives input from some other nodes and computes an output. Each input has an associated weight (w), which is assigned on the basis of its relative importance to other inputs. The node applies a function f to the weighted sum of its inputs to produce output as shown in Figure 2.2.

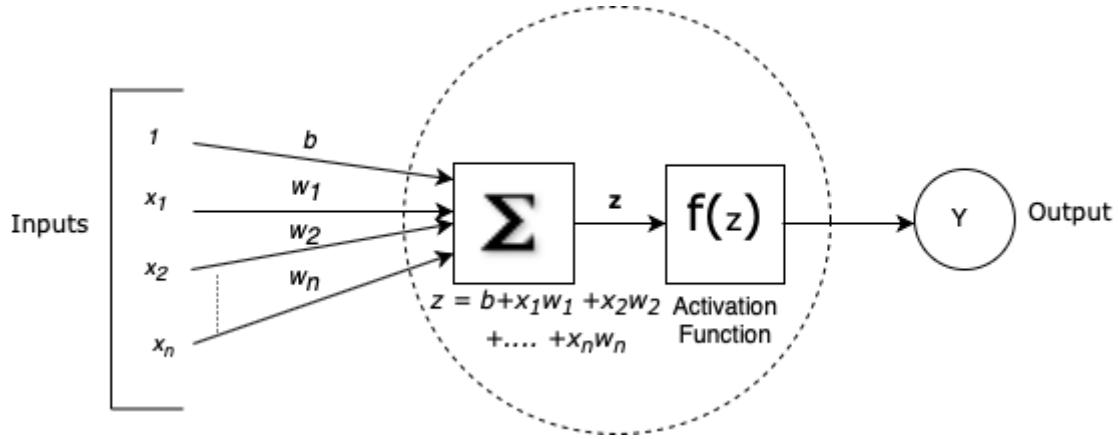


Figure 2.2: A single neuron (node) with **n** inputs.

The above node in figure 2.2 takes binary inputs x_1,x_2,...., x_n and w_1,w_2,...., w_n are the corresponding weights associated with those inputs. The other input 1 with weight b is called bias. The function f is a non-linear function and it is called the Activation Function.

2.2.2 Basic ANN Architecture

The simplest ANN architecture has been shown in figure 2.3, is the feed-forward neural network or multilayer perceptrons(MLPs). This type of network contains three layers: an input layer, a hidden layer, and an output layer. A node can receive data from either other nodes in its own layer or from previous layers. For example, if the node is located in the first layer of a feed-forward network, it will either receive data from other nodes in its own first layer or from nodes in any other preceding layers. Likewise, if it's located in the second layer, it will receive data only.

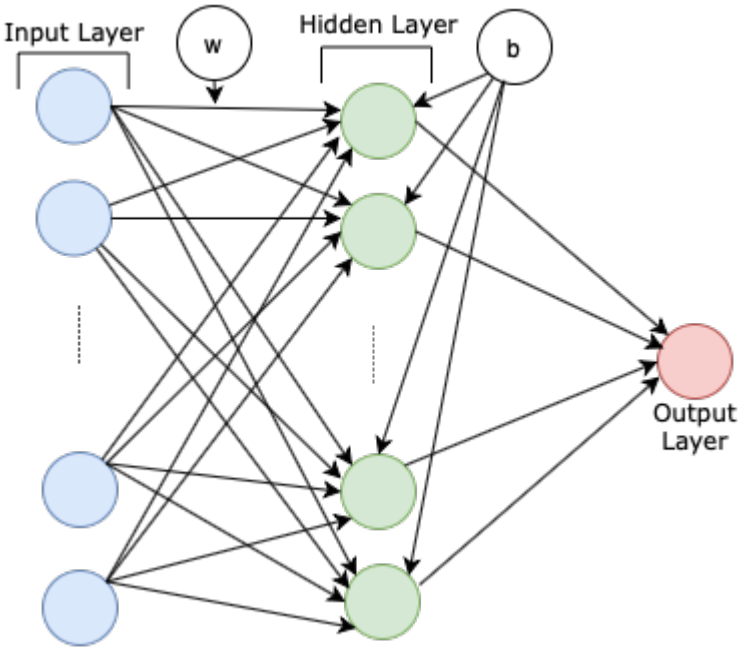


Figure 2.3: Three layers feed forward neural network

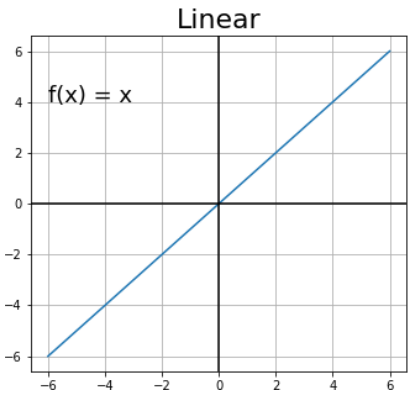
- Input Layer : The input data are fed to the nodes in the input layer. No computations are performed in any of the input nodes they just forward the input data to the next layer more specifically to the hidden layer.

- Hidden Layer: The hidden layer in a neural network resides between the input layer and output layer. The term hidden implies they are invisible to the external system and are private to the neural network. The neurons at hidden layer take a set of weighted inputs and produce an output through an activation function.

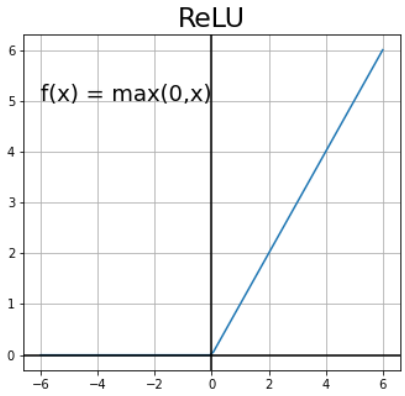
- Output Layer: The output layer is responsible for producing final results. The output layer receives the input data from the layer before it and performs the calculations via its neurons then produced final results.

2.2.3 Activation Functions

Every hidden layer in a neural network must have an activation function either it could be linear or non-linear. The purpose of the no-linear activation function is to introduce non-linearity to the output of a neuron. In general, most real world data are non-linear therefore it is important to have a non-linear activation in neurons to learn these non-linear representations. The most commonly used activation functions in neural network are *Linear*, *Sigmoid*, *ReLU* (Rectified Linear Unit), *LeakyReLU* and *tanh*. In this work activation functions, *Linear* and *ReLU* are used and the visual representation of these two is shown in Figure 2.4.



(a) Linear activation function.



(b) ReLU activation function.

Figure 2.4: Visual representation of the activation functions used in this project.

2.2.4 Convolution Neural Network

Convolution neural networks (CNNs) are special kind of neural network which are designed to work with grid-structured input data i.e image data. They are able to extract structural relations in the data such as spatial in images or temporal in time series. In a CNN, three common types of layers are present those are *convolution, pooling* and *ReLU*(Aggarwal 2018).

Convolution Layers

Convolution layers are the major building blocks of CNN where most of the computations are taken place. Convolution is a linear operation that involves the multiplication of the input data by a set of weights which is known as kernel or filter. The filter size is smaller than the input data and typically square in shape. The multiplication operation between a filter and filter sized image patch is a dot product. The convolution operation slides the filter at each possible location on the image so that the filter completely overlaps with the image and performs dot product. A convolution layer consists of a set of learnable filters. *The number of filters used in each layer controls the capacity of the model because it directly controls the number of parameters* (Aggarwal 2018). Let **F** is a $n \times n$ filter

$$\mathbf{F} = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n1} & w_{n2} & \dots & w_{nn} \end{bmatrix}$$

and $\mathbf{I}_{\mathbf{patch}}$ is a filter sized patch of an image **I**

$$\mathbf{I}_{\mathbf{patch}} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nn} \end{bmatrix}$$

Then,the convolution operation between the image patch $\mathbf{I}_{\mathbf{patch}}$ and filter **F** can be defined as

$$\mathbf{I}_{\mathbf{patch}} * \mathbf{F} = \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} w_{ij}$$

An example of convolution of an image **I** of size 7x7 by a filter **F** of size 3x3 is shown in Figure 2.5. It has been shown that the convolve or the output image size is 5x5 this

is because of stride = 1 and no zero padding is used in the convolution process. Stride defines how the filter will slide on the input image. When the stride is 1 then the filters move to 1 pixel at a time. When the stride is 2 then the filters move to 2 pixels at a time and so on. Padding or zero padding is used in convolution layers when the shape of the input data are need to preserved in the output data.

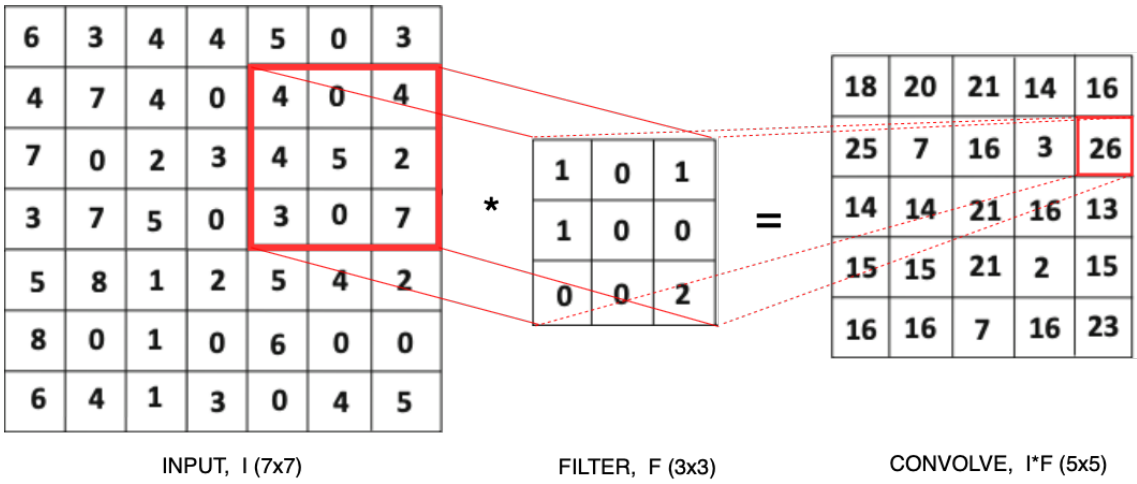


Figure 2.5: An example of a convolution operation in CNN. The image pixel values and the filter values are adopted from Aggarwal 2018.

In the example in Figure 2.5, the 3x3 filter **F** is

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

and the red marked 3x3 image patch $\mathbf{I}_{\mathbf{patch}}$ is

$$\mathbf{I}_{\mathbf{patch}} = \begin{bmatrix} 4 & 0 & 4 \\ 4 & 5 & 2 \\ 3 & 0 & 7 \end{bmatrix}$$

Hence, $\mathbf{I}_{\mathbf{patch}} * \mathbf{F} = 4*1+0*0+1*4+1*4+0*5+0*2+0*3+0*0+2*7 = 26$

Pooling Layer

A pooling layer performs almost similar to convolution layer, its function is to progressively reduce the spatial size of the convolved feature. By reducing the spatial dimensions it also helps to reduce the amount of parameters and computation in the network. Two types of pooling techniques are typically used in CNN: Max pooling and Average pooling. In the max pooling technique, the maximum value from the portion of the image covered by the filter is returned. On the other hand, average pooling returns the average of all the values from the portion of the image covered by the filter. The most common approach used in pooling is max pooling with filters of size 2x2 and strides of 2. An example of a max-pooling of one activation map of size 5x5 with 3x3 filter and strides of 2 is shown in Figure 2.6. A 3x3 filter and strides of 2 produces a 2×2 activation map.

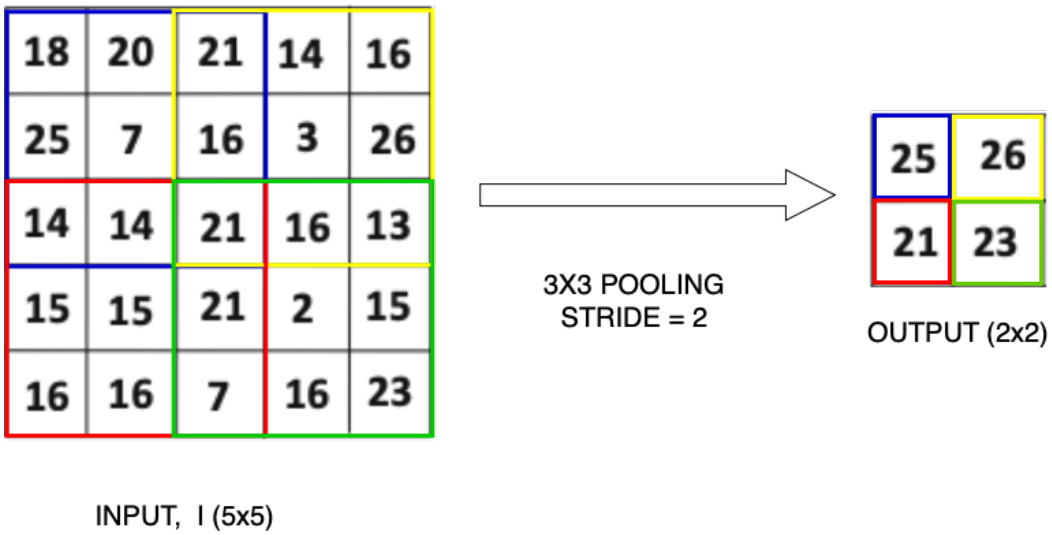


Figure 2.6: An example of a max-pooling of one activation map of size 5×5 with 3x3 filter and a stride of 2.

Transposed Convolution
In CNN, transposed convolution layer are used to upsample the output using some learnable parameters. More specifically it is used to generate an output feature map that has a spatial dimension greater than that of the input feature map. The parameters of transposed convolution layers are learnt through the training of the CNN. Transposed convolution also known as fractionally strided convolution.

2.3 Training process of a Neural Network

The parameters of a neural network are typically weights and biases. These parameters are learned during the training state of the network. In a feedforward neural network, backpropagation is the widely used algorithm to train the networks that involve two phases.

2.3.1 Forward Propagation

In forward propagation the parameters of the networks are fixed and the input data are propagated through the network layer by layer. The forward propagation ends with the computation of loss or error (E_i)

$$E_i = d_i - o_i \tag{2.1}$$

where d_i is the desired or target output and o_i is the output produced by the network in response to the input x_i.

2.3.2 Backward Propagation

In backward propagation the calculated error E_i is propagated through the network in backward direction (from output to input layer) by using chain rule. During backward propagation the network parameters are adjusted by minimizing the differential loss function using gradient descent. The basic idea of gradient descent is to find the derivatives of the loss function with respect to network parameters (weights and biases), then adjust them in the opposite direction of the slope.

2.3.3 Loss Function

Neural networks are trained using an optimization process that require a loss function to calculate model error during model training.A loss function $L(\mathbf{y}, \hat{\mathbf{y}})$ where **y** is the ground

truth and $\hat{\mathbf{y}}$ is the network output, measures the distance between ground truth and network output of a sample. Loss function is selected depending on the problem type. In case of regression problem most commonly used loss functions are mean squared error (MSE) as seen in Equation 2.2 and mean absolute error (MAE) as seen in Equation 2.3.

$$L(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{y}_i - \hat{\mathbf{y}}_i)^2 \tag{2.2}$$

$$L(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{N} \sum_{i=1}^{N} |\mathbf{y}_i - \hat{\mathbf{y}}_i| \tag{2.3}$$

2.3.4 Optimizers

In the training of neural network, the learning problem is formulated into an optimization problem where the objective is to minimize the loss function by training the parameters of the networks. It has already been mentioned in section 2.3.2 that backpropagation with gradient descent algorithm is widely used in neural network to minimize the loss function. Several optimization algorithm has been developed based on gradient descent such as stochastic gradient descent (SGD), Root Mean Square Propagation (RMSprop), Momentum, Adagrad, Adaptive Moment Estimation (Adam) etc. In this work optimizer Adam is used as an optimizer to train the model. Adam is a combination of momentum and RMSProp and among the adaptive optimizers it performs well enough in most cases. Moreover, Adam performs very well with the sparse data hence it is perfect for this work as the rain maps are also sparse in nature.

2.3.5 Epoch

In neural networks, the training dataset is split into several smaller parts called batches or mini-batches, which are fed to the network one at a time. One epoch is when the entire training dataset or batches are passed through the neural network only once. In other words, one epoch means that every sample in the training dataset has had an opportunity to update the network parameters (Jason Brownlee 2018).

2.4 DNN architecture

2.4.1 Generalized Residue Network

As the name implies, a generalized residual network (gResNet) consists of a deep neural network that models the generalized residue and generates the final prediction. The concept of gResNet proposed by Zhen Chen and Xiu 2021, it is an numerical approach for recovering unknown dynamical system using DNNs. The method is built upon a special kind of DNN called residual network (ResNet). In case of standard ResNet, residue is defined as the difference between the output data and input data of the model (Zhen Chen and Xiu 2021). In gResNet Zhen Chen and Xiu 2021 broadly define generalized residue is the discrepancy between the observation and the predictive output made by another model, which is also known as “prior prediction”.

2.4.2 UNet

UNet is a special type of DNN architecture for image segmentation purposes. Its architecture is developed based on fully convolutional networks and was first designed and applied in 2015 for the biomedical image segmentation (Wikipedia contributors 2021b). UNet is an encoder-decoder framework with same size inputs and outputs. The encoder path is also known as contraction path and the decoder path is known as expansion path. The job of encoder is to reduce the dimensions of images in every layer and increasing

the number of channels. On the other hand, the decoder increases the spatial dimensions while reducing the number of channels to restored back to original image size. Moreover, in UNet skip connections (the gray arrows in Figure 2.7) are used between the layers in the encoder and decoder part. Skip connections are the part of neural network that skips some of the layers in the neural network and feeds the output of one layer as the input to the next layer as well as some other layer (Sivaram T 2021). In UNet, skip connections are used to pass features from the encoder path to the decoder path to recover spatial information that were lost during downsampling. A basic UNet architecture for image to image translation problem can be seen in Figure 2.7.

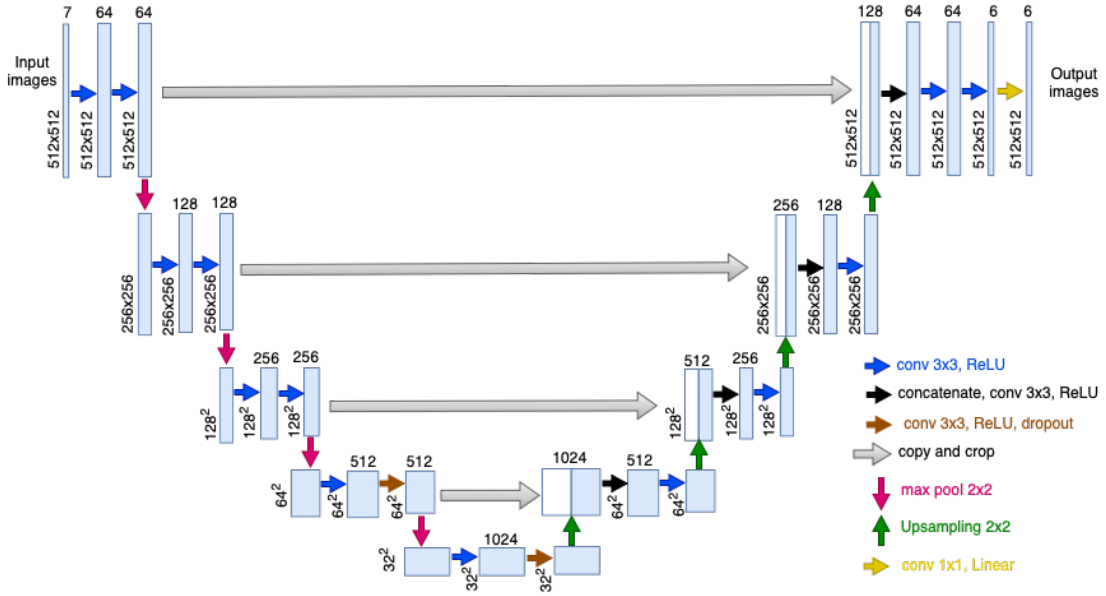


Figure 2.7: Basic UNet architecture for image to image translation

The model architecture shown in Figure 2.7 comprises with 20 convolutions (blue, brown, black and yellow arrows), 4 max-polling (red arrows), 4 upsampling (green arrows), 2 dropout layers and 4 copy and crop (gray arrows) connections. The encoder part applies two repeated convolution (blue arrows) and a max-pooling (red arrows) which doubled the number of features map and halves the image size respectively. Every convolution layer is followed by an activation function. All the convolution layers except the one denoted by yellow arrow use a kernel of size 3x3 and ReLU as activation function. The decoder part applies a upsampling to double the size of the feature map. Then the resulting feature maps is concatenated with previous encoder's output via skip connection (gray arrow). Lastly double convolution is applied on the concatenated feature maps to half the number of feature maps.

2.5 Nowcasting of radar rainfall

2.5.1 CO-TREC radar rainfall tracking algorithm

CO-TREC is a radar-based precipitation nowcasting system that works operationally around the world to provide precipitation nowcasts. In Denmark, DMI also uses the CO-TREC-based precipitation nowcasting system for precipitation nowcasting. Radar-based precipitation nowcasting algorithm consists of two major components, tracking and extrapolating. In the tracking step, the motion of the precipitation field is obtained from a series

of consecutive radar images. In the extrapolation step, the motion of the estimated precipitation field is used to predict the position and intensity of the future precipitation field. The existing tracking algorithm extrapolates radar images in space & time and it is derived from the pattern recognition technique TREC (Tracking Radar Echoes by Correlation) and COTREC (Mecklenburg et al. 2000). In TREC, future echoes are predicted by calculating the correlation of some previous consecutive radar echoes. A schematic graph of TREC is given in Figure 2.8 and it is adopted from Mecklenburg et al. 2000.

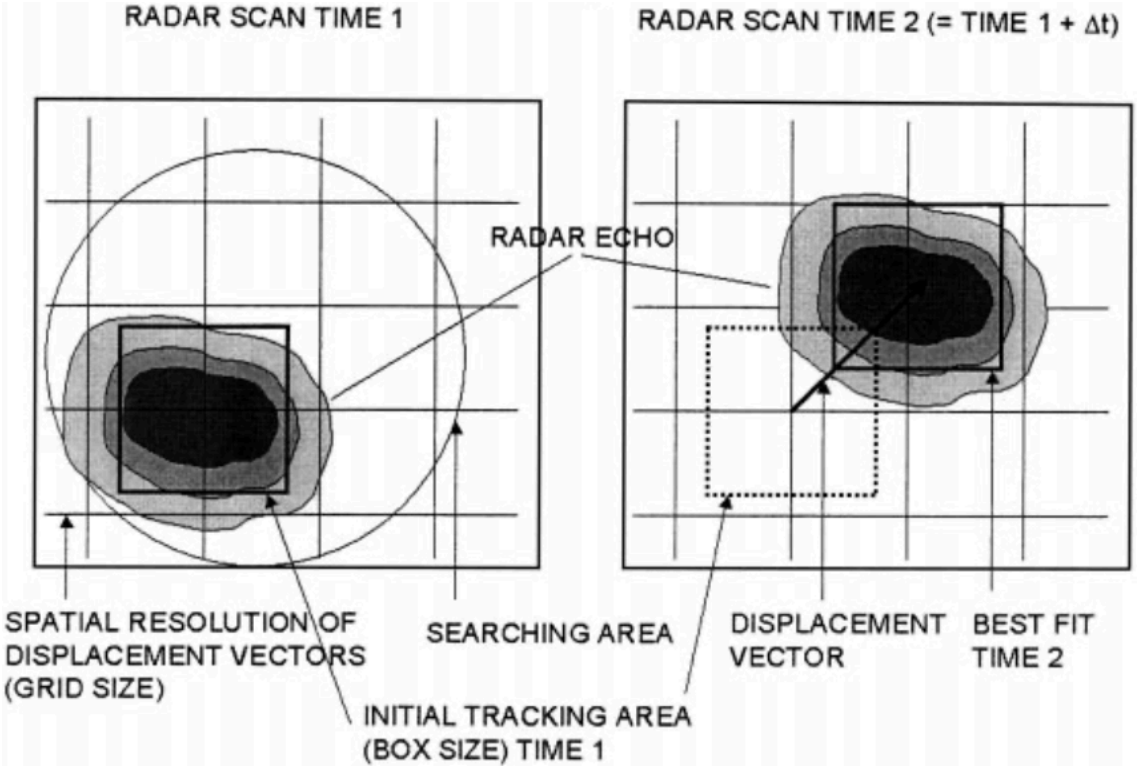


Figure 2.8: Schematic illustration of TREC method. The box at TIME 1 is compared to all boxes at TIME 2 that appear within the circular searching area. The matched box for which the correlation coefficient is maximum determines the best fit.

TREC vector field is often noisy and contains wrong displacement vectors with strong divergence due to shielding, ground clutter and the rapid changes in radar echo pattern (Li et al. 1995). To correct the noisy TREC vectors and improve the consistency of the resulting displacement vectors, Li et al. 1995 used variational technique where two-dimensional continuity equation was applied on TREC vectors. This method is well known as COTREC. The solution of variational analysis is obtained by minimizing a cost function:

$$J(u,v) = \int \left[\left(u - u^0\right)^2 + \left(v - v^0\right)^2 \right] dxdy \tag{2.4}$$

The constraint of two-dimensional continuity equation is considered as

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{2.5}$$

where u^0 and v^0 are the x and y components of the observed TREC vectors, whereas u and v are the solutions of the variational analysis by satisfying Equation 2.5 and minimizing Equation 2.4.

2.5.2 Review of DL based method in precipitation nowcast

Deep Learning is gaining popularity day by day because of it's high accuracy rate as it is trained with a large amount of data. In recent years, neural networks have been taken a lot of attention almost in every scientific computing areas including Meteorological Industry. In the past 5-10 years several deep learning strategy and approaches have been applied for precipitation nowcasting in Meteorological Industry. Table 2.1 below summarises some previous studies been done in this field in the last 5 years periods.

Title	Temporal Resolution	Spatial Resolution	Image Size	Input	Output	Network Type	Auxiliary Data
SmaAt-UNet: Precipitation nowcasting using a small attention-UNet architecture (Trebing et al. 2021)	5 minutes	1km x 1km	288x288	sequence of 12 precipitation maps	precipitation map 30 min later than the last input	U-Net	-
RainNet v1.0: a convolutional neural network for radar-based precipitation nowcasting (Ayzel et al. 2020)	5 minutes	1km x 1km	928x928	(t-15,t-10,t-5,t)	t+5	Unet + segnet	-
Convcast: An embedded convolutional LSTM based architecture for precipitation nowcasting using satellite data (Kumar et al. 2020)	30 minutes	10km x 10km	150x150	Observation at time point $t_1, t_2 ... t_{10}$	Prediction at time point $t_2, t_3 .... t_{11}$	ConvLSTM	-
Machine Learning for Precipitation Nowcasting from Radar Images (Agrawal et al. 2019)	6 minutes	1km x 1km	256x256	Sequence of N(10) images (time point t_1 to t_N)	1-hour nowcasting prediction	U-Net	-
Distributed Deep Learning for Precipitation Nowcasting (Samsi et al. 2019)	10 minutes	1km x 1km	256x256	Sequence of 7 observations t-60, t-50,... t	One hour (6 steps) predictions like t+10,t+20,... t+60	Fully CNN (like U-Net)	-
Deep Learning for Precipitation Nowcasting: A Benchmark and A New Model (Shi et al. 2017)	6 minutes	1.07km x 1.07km	480x480	Previous 5 frames	20 future frames	Trajectory GRU (TrajGRU)	-
Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting (Shi et al. 2015)	6 minutes	-	100x100	5 frames (out of 20)	15 frames(out of 20)	ConvLSTM	-

Table 2.1: Previous work done in this field

All the proposed DL-based precipitation nowcasting models shown in Table 2.1 are built on radar observations only in contrast with combining model prediction and observations except the one proposed by Agrawal et al. 2019. The models proposed by Shi et al. 2015; Shi et al. 2017; Kumar et al. 2020 learn both temporal and spatial correlation from data as they used 3D convolution with LSTM (Long Short-Term Memory) cell, exclusively; these models learn temporal correlation better. On the other hand, the models proposed by Samsi et al. 2019; Agrawal et al. 2019; Ayzel et al. 2020; Trebing et al. 2021 exclusively learn the low-level features and spatial correlation from data as they used UNet architecture. However, all the model architectures take a temporal sequence of images

as input to produce a sequence of future images. It is also worth mentioning that all of these frameworks are limited to generating predictions only for small areas; none of them predicts for large areas simply by using the same model for predicting different areas and subsequently stitching the images together.

In their work Agrawal et al. 2019 used multi-radar multi-sensor (MRMS) dataset where the *MRMS system combines radar with surface observations and numerical weather prediction methods to get a high-resolution map of current conditions* (Agrawal et al. 2019). In addition, since the spatial domain was too large to model all at once, they divided it into 256km x 256km tiles and made predictions independently for each. However, in their study, Samsi et al. 2019 trained their model using 256x256 pixels of precipitation maps randomly sampled from a 3520x5120 pixels grid.

3 Data

The data used in this project is the precipitation radar echo and radar nowcast data from Danish Meteorological Institute(DMI).The data is generated by 5 radars and they are situated in 5 five different locations (i.e.Stevns, Rømø, Sindal in Vendsyssel, Bornholm and last one in Virring in East Jutland.) in Denmark. All of the 5 radars are *dual-pole C-band* with Doppler effect. The radar images contains rain maps of whole Denmark and its neighbouring area for every 10 minutes intervals from December,2016 to January,2018. As each rain map contains 10 minutes rain information therefore, there are 144 rain maps for a day or 4320 rain maps for 30 days in the datasets.

3.1 Data pre-processing

The provided dataset is approximately 10 TB in size, and data processing was a major part of this project. It takes almost a month to process the data used in this project, even though the whole 10 TB has not been used. DMI provides the radar observation in HDF5-file format which has been extracted and pre-processed before feeding into the deep neural network (DNN) model. The process has been shown in figure 3.1 and described in the following section.

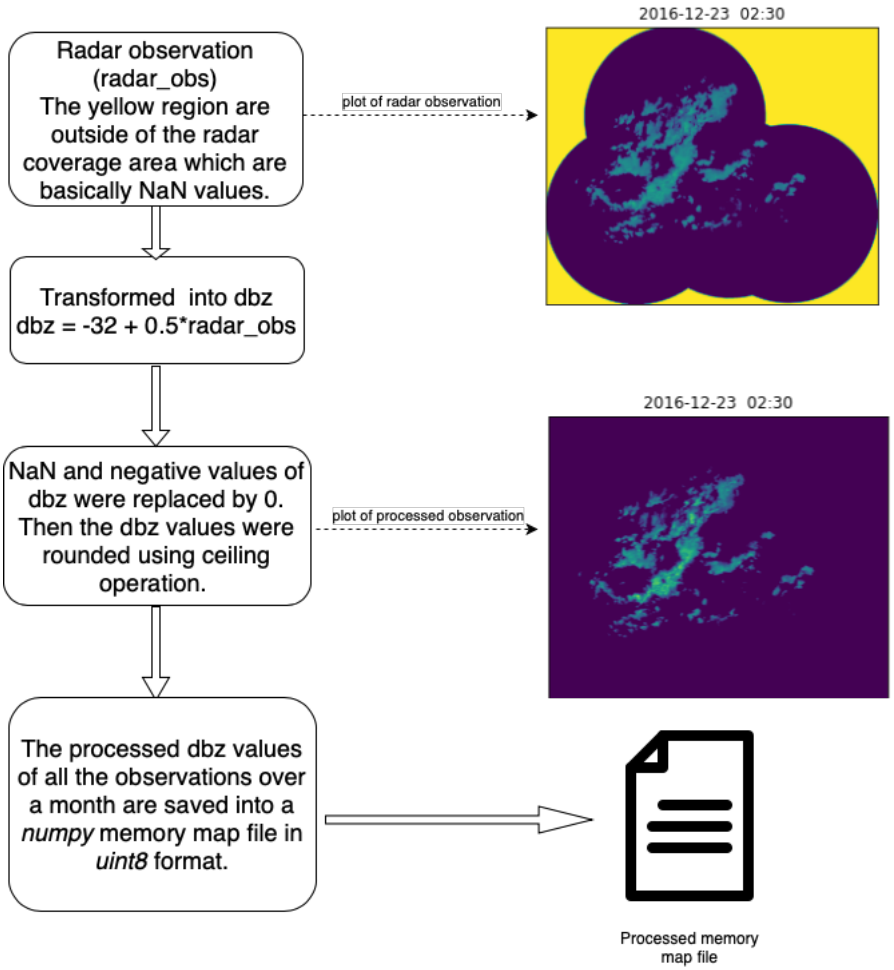


Figure 3.1: Data pre-processing flow chart

The processed data are in dbz unit and the models are trained on it. The model predictions are also considered in dbz unit. This completely makes sense because the rain intensities follow a very skewed distribution whereas the dbz follow the normal distribution. Another reason for choosing dbz unit is the file size of processed data. In processed data, the range of the dbz unit was between 0 and 95 which also helps to store the data in *uint8* format.Therefore, in the case of radar reflectivity images (dbz unit), the total file size of one month of processed data was approximately 10.5 GB on the other hand it was approximately 52 GB for rainfall intensity images. As 14 months (December,2016 to January,2018) of radar observations are used in this project so it could be very hard to handle (52x14) 728 GB data if rainfall intensity images are used for training.

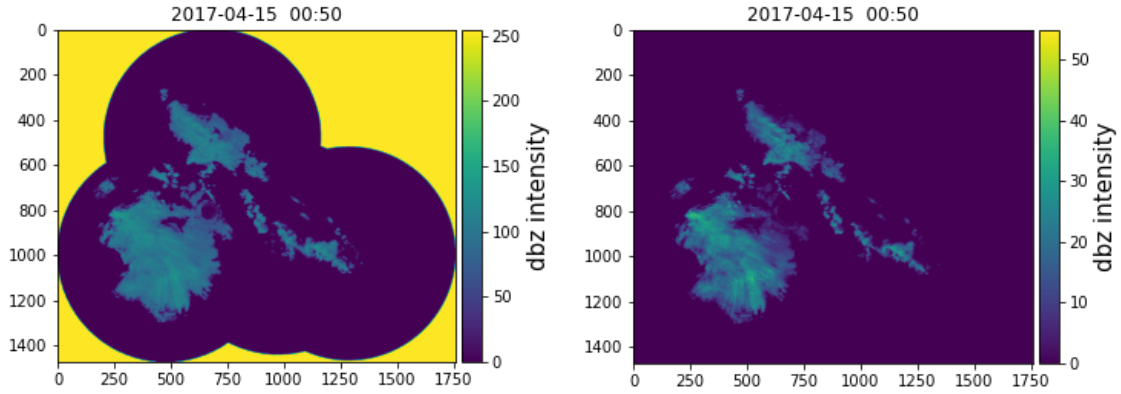
3.2 Datasets

Based on the model designing approaches described in section 4.1 two datasets (a) Dataset-1 and (b) Dataset-2 were considered as shown in Table 3.1.

Table 3.1: Data overview table

Dataset	Data	Data period	Spatial Domain	Temporal Resolution	Train/Val/Test
dataset 1	radar echo map	20/12 2016 11:40-30/04 2017 23:50	1472x1760	10 mins	70/20/10
dataset 2	radar echo map + radar nowcast	20/12 2016 11:40 - 31/01 2018 23:50	512x512	10 mins	70/20/10

Dataset-1
Dataset-1 contains only radar echo map for the period of December 2016 to April 2017. The size of images in dataset-1 is 1472x1760 which represents the entire area of Denmark and its neighboring area. The purpose of using dataset-1 in modelling is to identify the movement of the rainstorms. An example of a original radar observation and corresponding processed image for dataset-1 is shown in Figure 3.2. The yellow region in Figure 3.2a is the outside of the radars coverage area. The proportion of this region is 28.162%, which is the minimum in the whole working dataset, and therefore it has been chosen as a threshold value. Only the sequences with at least 12 consecutive radar echo maps that meet the threshold were included in dataset-1. The reason for thresholding is that, in the processed data, the yellow region is replaced with zero (0) to take this part outside of the convolution operation; in modeling, a mask layer is introduced based on this yellow region.



(a) Original radar observations at timestamp 17-04-2015 00:50

(b) Processed radar observation at timestamp 17-04-2015 00:50

Figure 3.2: An example of original and processed radar observation for dataset-1.

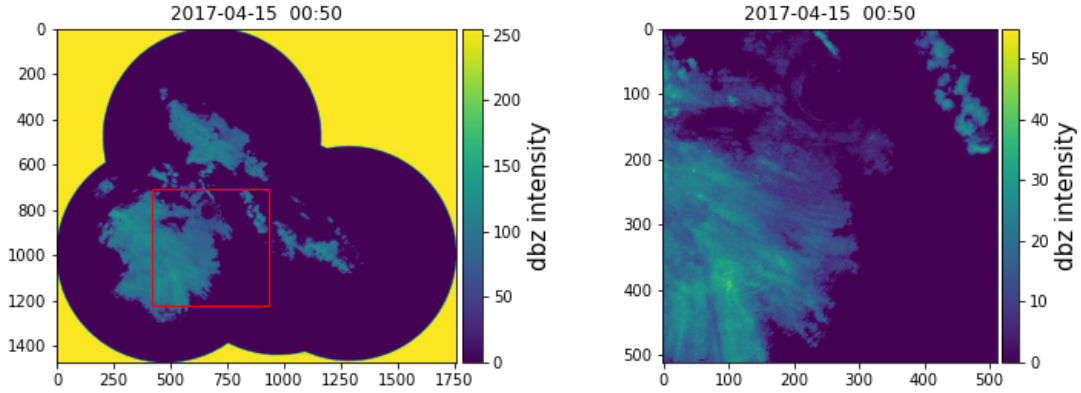
Dataset-1 contains 12050 valid sequences. Those are randomly shuffled, then the train, validation, and test sets are split. So in the case of dataset-1, there are 8435 train sequences, 2410 validation sequences, and 1205 test sequences.

Dataset-2

Dataset-2 contains both radar echo map and radar nowcast map for the period of December 2016 to January 2018. The size of images in dataset-2 is 512x512. The radar nowcast maps are the precipitation nowcast maps which are generated by COTREC algorithm, currently operational model of DMI. Hence, dataset-2 have the following features

- Only for cropped region
- Includes radar nowcast maps

An example of a original radar observation and corresponding processed image for dataset-2 is shown in Figure 3.3. The red marked squared region in Figure 3.3a is cropped from all of the radar observation and radar nowcast images for dataset-2. Moreover, only the sequences with at least 12 consecutive radar echo maps that contains at least 5% of rain data are included in dataset-2.



(a) Radar observations at timestamp 17-04-2015 00:50

(b) Processed radar observations at timestamp 17-04-2015 00:50

Figure 3.3: An example of original and processed radar observation for dataset-2.

Dataset-2 contains 20610 valid sequences. Those are also randomly shuffled first, then the train, validation, and test set are split. Therefore, in the case of dataset-2, there are 14450 train sequences, 4130 validation sequences, and 2030 test sequences.

4 Methods

This chapter describes the model architectures and setups that have been used to model the problem. The methodology flowchart is shown in Figure 4.1

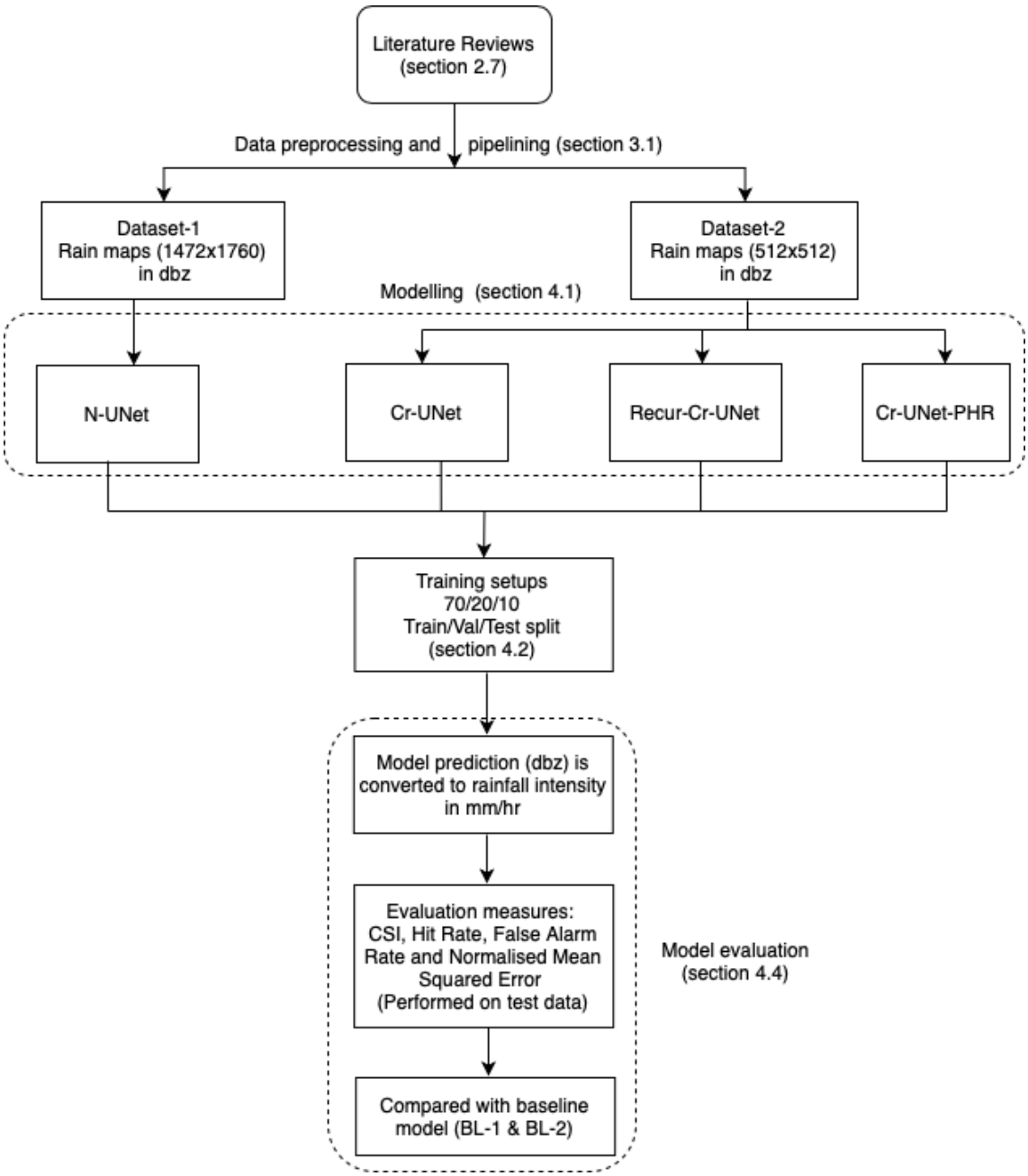


Figure 4.1: Methodology workflow diagram.

4.1 Modelling

All the models architecture designed and tested for this work have been inspired from UNet by its successful application in image to image translation. In subsection 2.4.2, introduction about a basic UNet architecture has been described. In computer vision,

UNet is typically used for classification or segmentation tasks where the network is trained to label each pixel of an image with a corresponding class of what is being represented. In this work the UNets are applied as time series regression tasks where the network is trained to approximate the exact value for each pixel.

Depending on the spatial resolution in this work four models are trained and two baseline models are formulated.

- **Baseline models (BL-1 & BL-2) -** are the radar nowcast (see section 3.2) with which performance of the trained models are evaluated. Spatial dimensions of an image for BL-1 model is 1472x1760 whereas for BL-2 is 512x512.
- **Naive UNet (N-UNet) -** model trained on Dataset-1 (see in section 3.2) to identify the development and movement of the rainstorms. The performance of N-UNet model basically compared with BL-1.
- **Model for cropped region (Cr-UNet) -** Dataset-2 is used to train this model, which contains radar observation and radar nowcast maps. The idea of this model is that the nowcast describes the anticipated movement of rainfall and that the NN only needs to learn the deviation between nowcast and observation.
- **Recursive model for cropped region (Rec-Cr-UNet) -** It is an extension of the Cr-UNet model. The main idea behind this model is that it will learn to simulate the evolution of rainfall, which can give more stable predictions.
- **Model for cropped region with penalty on high rainfall (Cr-UNet-PHR) -** It is also an extension of the Cr-UNet model. In this model, more weight is given to intense less frequent rainfall that is under-represented in the data; so that the model can capture intense rainfall.

An overview of all the models used in this project can be seen in Table 4.1.

Table 4.1: model overview table

Model	Abbreviated name	Number of Parameters	Input Dimensions	Output Dimensions	Dataset Used	Training Epoch
Base line model -1[a] (COTREC)	BL-1	-	-	1472x1760	-	-
Base line model -2[b] (COTREC)	BL-2	-	-	512x512	-	-
Naive model (UNet)	N-UNet	7,762,992	1472x1760	1472x1760	Dataset-1	>140
Model for cropped region	Cr-UNet	31,040,368	512x512	512x512	Dataset-2	>140
Recursive model for cropped region	Rec-Cr-UNet	46,560,432	512x512	512x512	Dataset-2	>140
Model for cropped region with penalty on high rainfall	Cr-UNet-PHR	31,040,368	512x512	512x512	Dataset-2	>140

[a] This is radar nowcast prediction where the spatial dimensions of nowcast map is 1472x1760
[b] This is also radar nowcast prediction where the spatial dimensions of nowcast map is 512x512

4.1.1 N-UNet architecture

The conceptual idea of this model is a sequence to sequence learning. Which means the model will take a sequence of data as input and will produce a sequence of data as output (Kumar et al. 2020; Agrawal et al. 2019; Samsi et al. 2019; Shi et al. 2017). This model simply maps the pixel values of a set of input rainfall map to the pixel values to some target rainfall map based on the training samples. The purpose of this modeling is to see how well the model can capture the movement and development of the rainfall. The model architecture is inspired from U-Net architecture presented in subsection 2.4.2 for its numerous success in image processing. The N-UNet model architecture is shown in Figure 4.2.

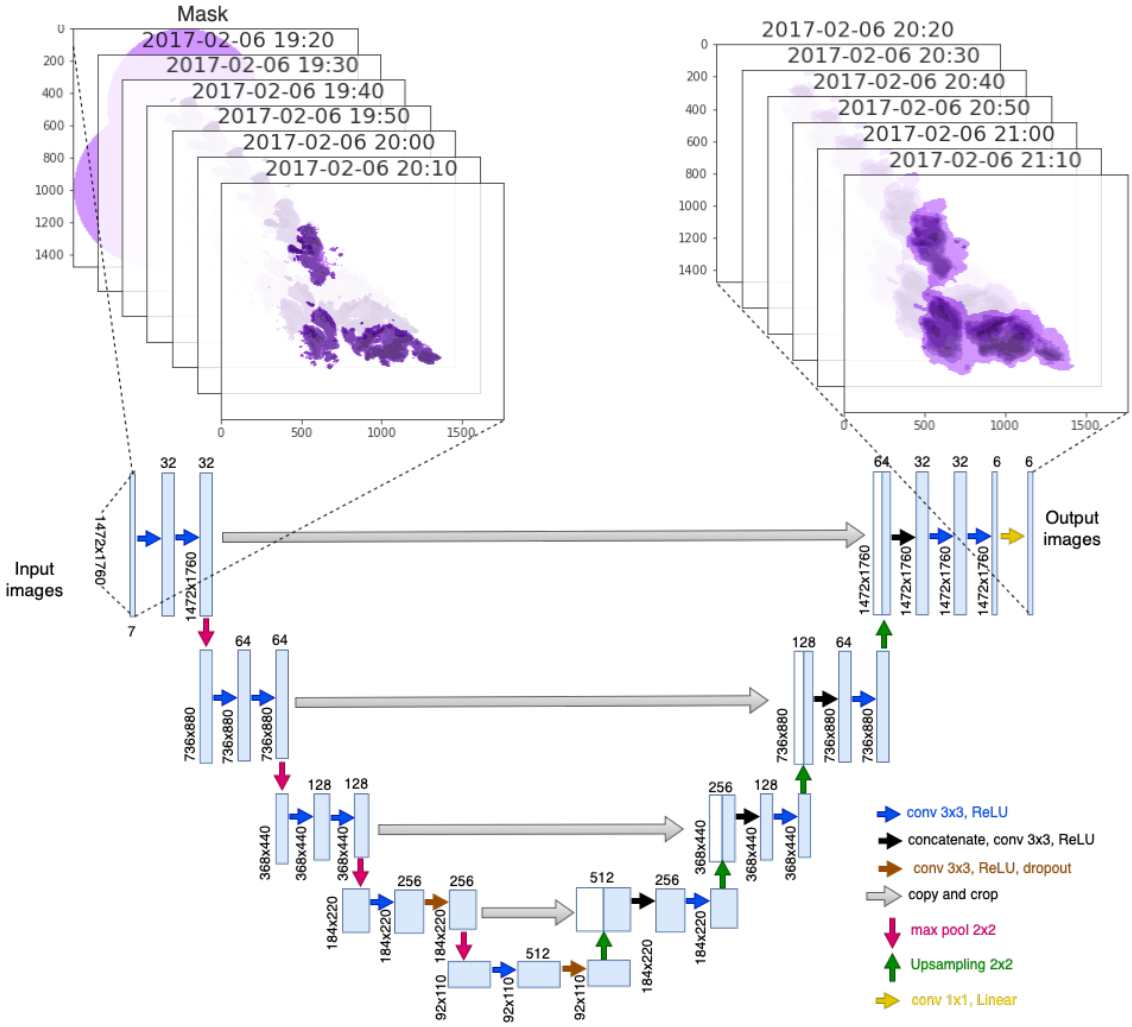


Figure 4.2: N-UNet

The model takes 6 radar reflectivity images at time-point t-50, t-40, t-30, t-20, t-10 min, t and a mask, each of dimensions 1472x1760 as input and produces predictions for 6 timestamps ahead e.g. t+10,t+20,t+30,t+40,t+50 and t+60 min. In radar reflectivity map, the outside of the radar coverage areas were set to zero (0) to take those part outside of the convolution operation a mask layer is used here. In the encoder side, input images are down scaled by a factor of 2^n where n is the number of max-polling layers.On the other hand, in decoder side the condensed high resolution feature map from encoder are expanded by every decoder block to get the original size of the input images. The last convolution (yellow arrow) layer with a kernel of size 1x1 and linear activation function produces the predicted radar reflectivity (in dbz) for timestamps t+10,t+20,t+30,t+40,t+50 and t+60 min.

4.1.2 Cr-UNet architecture

The core architecture is same as N-UNet model architecture but it differs by the number of inputs model receives and the dimensions of input and output. The dimension of the inputs and output images are 512x512. The architecture of Cr-UNet model is shown in Figure 4.3.

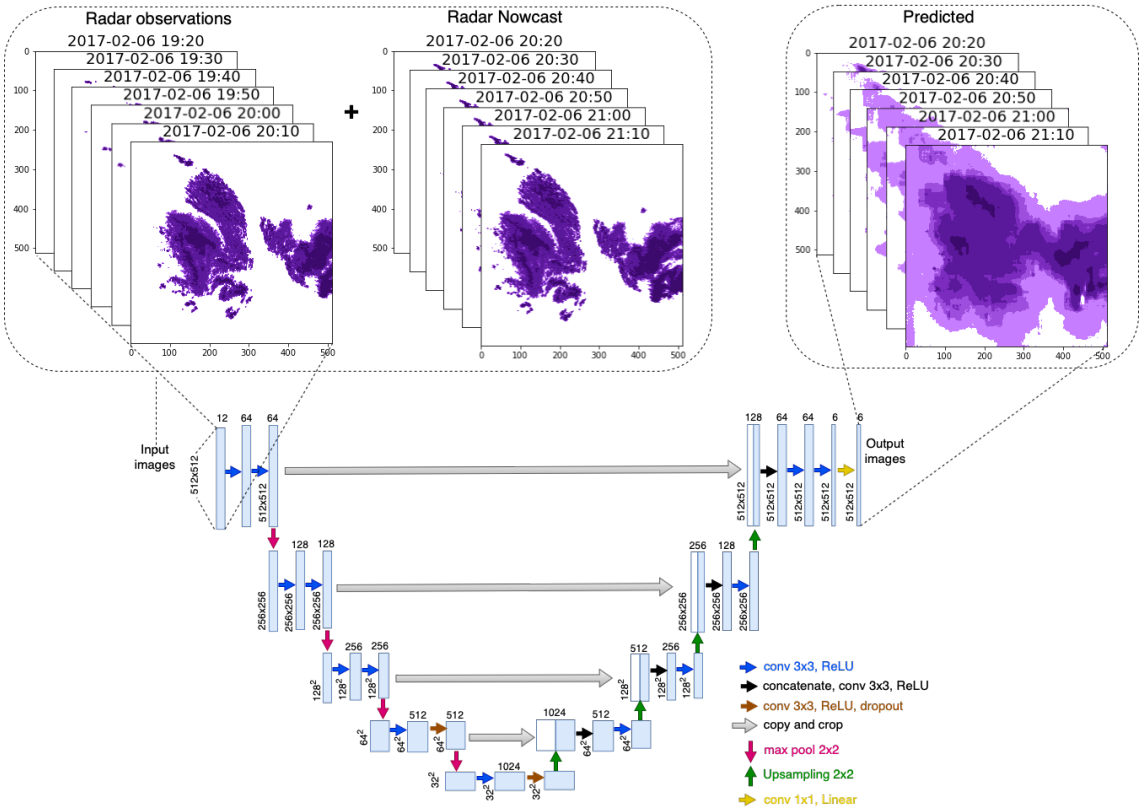


Figure 4.3: Cr-UNet

Unlike the N-UNet model, Cr-UNet model takes 6 radar reflectivity images at time-point t-50, t-40, t-30, t-20, t-10 min, t and 6 radar nowcast images at time-point t+10, t+20, t+30, t+40, t+50, t+60 min (12 radar images) as input and produces predictions for 6 timestamps ahead, e.g. t+10,t+20,t+30,t+40,t+50 and t+60 min. The radar nowcast describes the anticipated movement of the rainfall, so the purpose of this model is to learn the deviation between radar nowcasts and the observations. For example, the progression of rainfall in the context of change in size, intensity and shape.

4.1.3 Rec-Cr-UNet architecture

The schematic diagram of Rec-Cr-UNet model architecture is shown in Figure 4.4. The core backbone of this model is Cr-UNet; hence the dimensions of the input and output images are also 512x512. Moreover, in this case, the Cr-UNet unit takes only two images (1 radar observation /Cr-UNet output at timestamp t and 1 radar nowcast image at timestamp t) as input and produces output for timestamp t+10 min. The model as a whole takes 7 (1 radar observation at time t and 6 radar nowcast at timestamps t,t+10,t+20,t+30,t+40 and t+50 min) reflective images as input and produces predictions for 6 timestamps ahead, e.g. t+10,t+20,t+30,t+40,t+50 and t+60 min. Its principal objective is to simulate the progress of the precipitation, which might lead to more accurate forecasts (lower risk of overfitting).

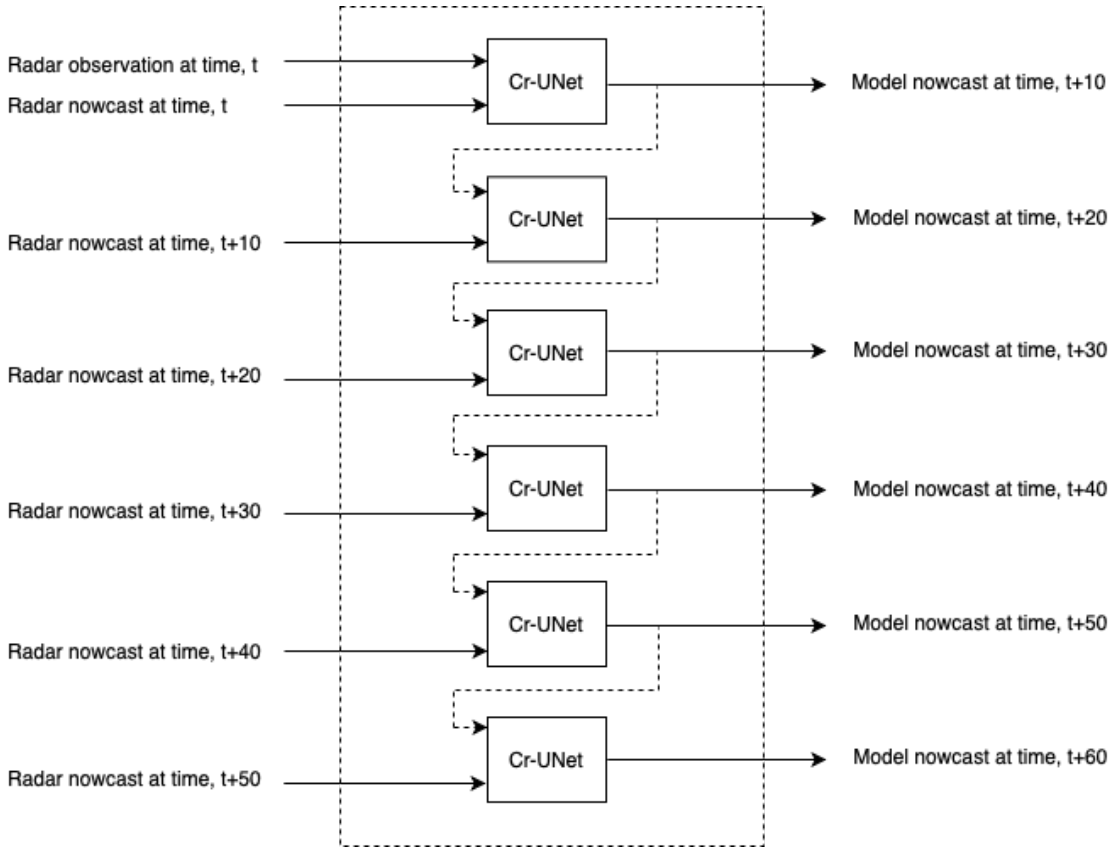


Figure 4.4: Schematic diagram of Recursive Model

4.1.4 Cr-UNet-PHR architecture

The architecture of the Cr-UNet-PHR model is exactly the same as the Cr-UNet model architecture. Hence the dimensions of the input and output images are 512x512. The difference between the Cr-UNet model and the Cr-UNet-PHR model is in parameters optimization. The parameters for the Cr-UNet model are optimized by using the MSE loss function, whereas in the case of the Cr-UNet-PHR model the model parameters are optimized by using a custom MSE loss function. The objective of this model is to assign weights to every individual pixel value according to their distribution so that the model can correctly capture all types of rainfall. Naturally, there are very few of high rainfall images in the datasets compared to low and very low rainfall images. Hence, the model is biased to capture low and very low rainfall better than high rainfall.

Custom MSE loss function

In rainfall prediction tasks, the majority of the output values are small and only a few are large values. Since the small values are much more common, and with a standard loss function the NN might get away with always predicting low values. To deal with this problem in this project a custom loss function is implemented that enables the NN to assign weights for every individual pixel value. The custom loss function built on top of the standard MSE loss function. Let **y** is the true observation and $\hat{\mathbf{y}}$ is the network output for a batch then the custom loss for the batch is calculated in the following steps.

- Based on the rainfall intensity pixel values of **y** are grouped into four groups G_1 (Rain intensity < 0.5) , G_2 (Rain intensity > 0.5 and < = 1), G_3 (Rain intensity > 1.0 and < = 5.0) and G_4 (Rain intensity > 5.0).

- Then weight for an individual group is calculated by dividing the number of pixels belonging to the group by the total number of pixels. Let N_1,N_2, N_3 and N_4 be the number of pixels in G_1, G_2,G_3 and G_4 respectively and N is the total number of pixels in **y**. Hence, the weights w_1, w_2,w_3 and w_4 for G_1, G_2,G_3 and G_4 are

$$w_1 = \frac{N_1}{N}, w_2 = \frac{N_2}{N}, w_3 = \frac{N_3}{N} \text{ and } w_4 = \frac{N_4}{N}.$$

- Squared error between **y** and $\hat{\mathbf{y}}$ has been calculated using Equation 4.1.

$$E(\mathbf{y}, \hat{\mathbf{y}}) = (\mathbf{y} - \hat{\mathbf{y}})^2 \tag{4.1}$$

Now every individual error term is divided by the group weight to which the error term belongs such as

$$E_1 = \frac{E(G_1)}{w_1}, E_2 = \frac{E(G_2)}{w_2}, E_3 = \frac{E(G_3)}{w_3} \text{ and } E_4 = \frac{E(G_4)}{w_4}.$$

- Finally, the custom mean squared loss is computed using Equation 4.2.

$$L(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{N} \sum_{i=1}^{N} E_i \tag{4.2}$$

4.2 Parameter Optimization

The parameters of all the models are optimized by using the MSE loss function (see subsection 2.3.3) except the Cr-UNet-PHR model. In case of Cr-UNet-PHR model custom MSE loss function (see subsection 4.1.4) is used. Moreover, optimizer Adam with *learning_rate* = $3 * e^{-4}$ are used for all the models. Furthermore, N-UNet uses a batch size of 5 whereas the other models use a batch size of 10. An example iteration of the parameters optimization procedure for a training sequence with batch size 1 for the Cr-UNet model is shown in Figure 4.5.

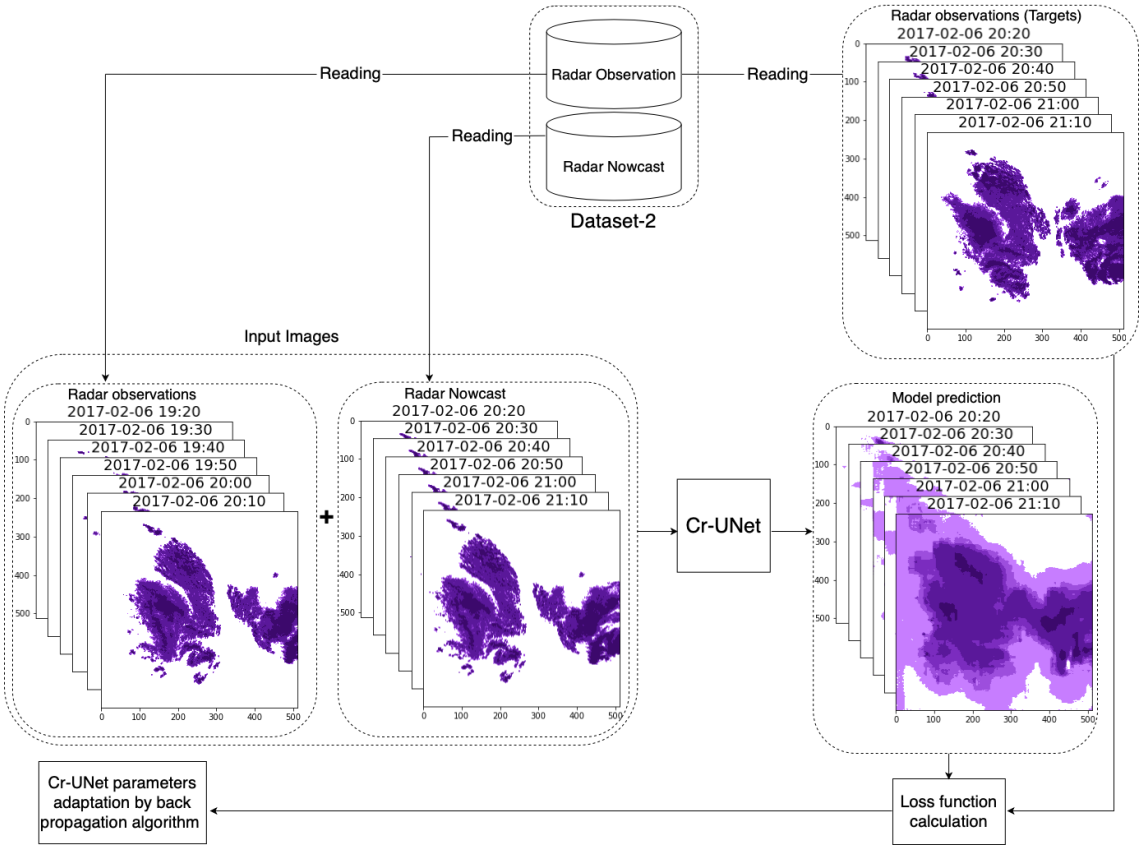


Figure 4.5: Visualization of one iteration of Cr-UNet parameters optimization procedure.

4.3 Research Software

In this project only open source and freely available software packages in Python programming language are used. Those are: h5py (h5py.org Collette 2013), Matplotlib (matplotlib.org Hunter 2007), TensorFlow (tensorflow.org Abadi et al. 2016), NumPy (numpy.org Harris et al. 2020), and Keras (keras.io Chollet et al. 2015).

4.4 Model Evaluation

To evaluate the performance of the different model several scores has been calculated such as critical success index (CSI),Hit Rate, false alarm rate (FAR) and normalised mean squared error (NMSE).

NMSE - It is a performance metric used to know how different model are performing. NMSE is calculated by dividing the MSE of an observation with the variance (σ^2_{ref}) of the observation. Let $\mathbf{y}$ is the true observation and $\hat{\mathbf{y}}$ is the model prediction then NMSE is calculated in the following steps.

- First, MSE between $\mathbf{y}$ and $\hat{\mathbf{y}}$ has been calculated using Equation 4.3.

$$\text{MSE}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{N} \sum_i (y_i - \hat{y_i})^2 \tag{4.3}$$

  where N is the number of pixels in $\mathbf{y}$.

- Then variance of **y** is calculated using Equation 4.4.

$$\sigma_{ref}^2 = \frac{\sum_i (y_i - \bar{y})^2}{N} \tag{4.4}$$

where $\bar{y} = \frac{1}{N}\sum_i y_i$

- Finally, NMSE is calculated using Equation 4.5.

$$\text{NMSE} = \frac{\text{MSE}(\mathbf{y}, \hat{\mathbf{y}})}{\sigma_{ref}^2} \tag{4.5}$$

4.4.1 Forecast verification score using binary contingency table

CSI, Hit Rate and FAR can easily be described with the help of 2x2 contingency table shown in Figure 4.6

2x2 Contingency table		Event Observed	
		Yes	No
Event Forecast	Yes	hits	false alarms
	No	misses	correct rejections

Figure 4.6: A 2x2 contingency table.

- **CSI -** also known as threat score (TS), is a verification measure of categorical forecast performance equal to the total number of correct event forecasts (hits) divided by the total number of rainstorm forecasts plus the number of misses (hits + false alarms + misses).

$$\text{CSI} = \frac{\text{hits}}{\text{hits + false alarms + misses}} \tag{4.6}$$

- **Hit Rate -** A verification measure of categorical forecast performance equal to the total number of correct event forecasts (hits) divided by the total number of correct event forecasts (hits) plus the number of misses (hits + misses).

$$\text{Hit Rate} = \frac{\text{hits}}{\text{hits + misses}} \tag{4.7}$$

- **FAR -** A verification measure of categorical forecast performance equal to the number of false alarms divided by the total number of event forecasts.

$$\text{FAR} = \frac{\text{false alarms}}{\text{hits + false alarms}} \tag{4.8}$$

The scores are calculated for rainfall bigger than 4 different threshold values 0.5mm/h, 1mm/h, 2.5mm/h and 5mm/h . As both the model prediction and observation are in dbz unit so all the predicted and their corresponding observation images are converted to rainfall rates in millimeters per hour. Then each pixel of the predicted output and observation images are converted to a boolean mask using a chosen threshold. From this, hit (prediction = 1, observation = 1), false alarm (prediction = 1, observation = 0) and misses (prediction = 0, observation = 1) are calculated. Finally, using Equation 4.6, Equation 4.7 and Equation 4.8 CSI, Hit Rate and FAR score are calculated.

4.4.2 Forecast verification using multi-class contingency table

In this section forecast verification score for multi-class contingency table has been studied. To do this the pixel values of a rainfall map are classified into four classes which are given below.

- Rain intensity < 0.5 in class 0 or very low rain.
- Rain intensity > 0.5 and < = 1 in class 1 or low rain.
- Rain intensity > 1 and < = 5 in class 2 or medium rain.
- Rain intensity > 5 in class 3 or high rainfall.

Then a multi-class contingency table were generated.From the multi-class contingency table number of *hits*, *false alarms*, *correct rejections* and *misses* for a particular rain class is calculated by the following way.

- hits: Each of the diagonal elements of the contingency table represents number of *hits* for a class.
- misses: The sum of values of corresponding rows except the *hits* value.
- false alarms: The sum of values of the corresponding column except the *hits* value.
- correct rejections: *correct rejections* can be calculated by subtracting the sum of number of *hits*, *false alarms* and *misses* from the total sum of the contingency table.

Then using Equation 4.6, Equation 4.7 and Equation 4.8 CSI, Hit Rate and FAR score are calculated. An example of a contingency table for an observation and prediction is shown in Figure 4.7.

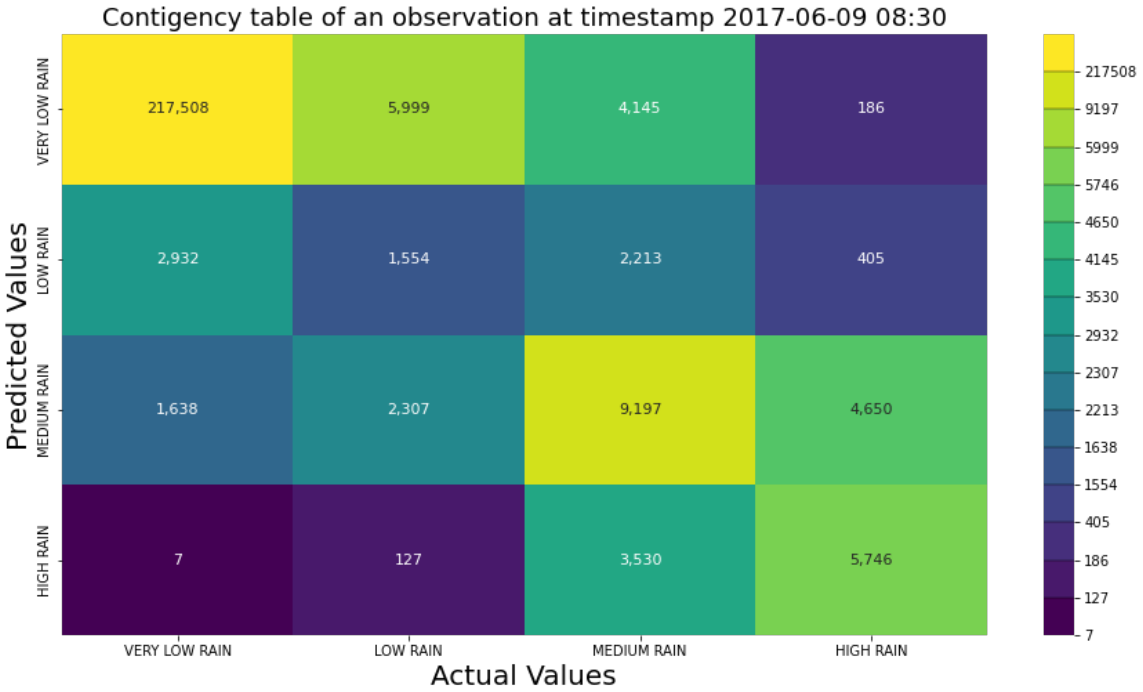


Figure 4.7: Contingency table between radar observation and Cr-Net model prediction at timestamps 2017-06-09 08:30

5 Results

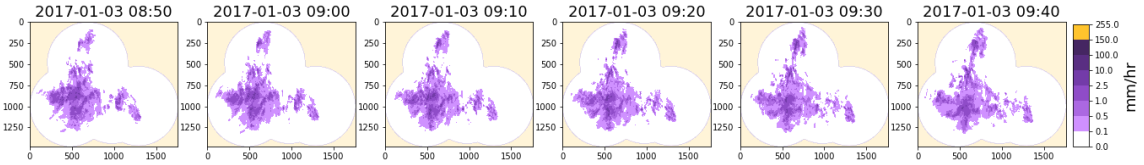
In this section, results from the different models have presented, and their performance has been evaluated. In the following section, predicted output along with true observation and model evaluation score (forecast verification score) for all models have been presented. Moreover, computational requirements such as training time, memory consummations, and training losses of all trained models also have been described here.

5.1 Qualitative Evaluation

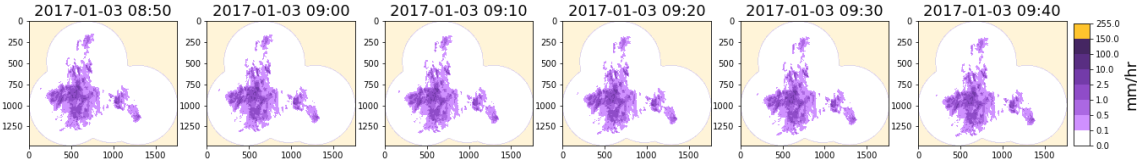
In this section, the precipitation nowcast for all models for two events from the test set is visualized. The models are trained on dataset-1 and dataset-2; therefore, for comparison, only the common events in both of these datasets are chosen for visualization. The precipitation nowcasts basically fall into two groups. The first group includes the observed precipitation and models (BL-1 and N-UNet) output for a precipitation nowcast in regards to the whole Denmark where the spatial resolution is 1472 x 1760 or (736 km x 880 km). On the other hand the second group includes the observed precipitation and models (BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR) output for a precipitation nowcast in regards to cropped region where the spatial resolution is 512 x 512 or (256 km x 256 km).

Example - 1

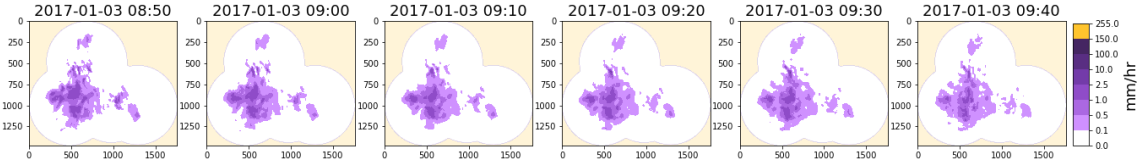
An example of a precipitation nowcast for all models at timestamp 2017-01-03 08:40 with a lead time of 1 hour is shown in Figure 5.1 and Figure 5.2. Figure 5.1 represents the observation and precipitation nowcasting of BL-1 and N-UNet models in regards to the whole Denmark. On the other hand, Figure 5.2 represents the observation and precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR models in regards to the cropped region (see Figure 3.3a) at timestamp 2017-01-03 08:40 with a lead time of 1 hour ahead.



(a) Radar observations for whole Denmark at timestamp 2017-01-03 08:50 to 2017-01-03 09:40.

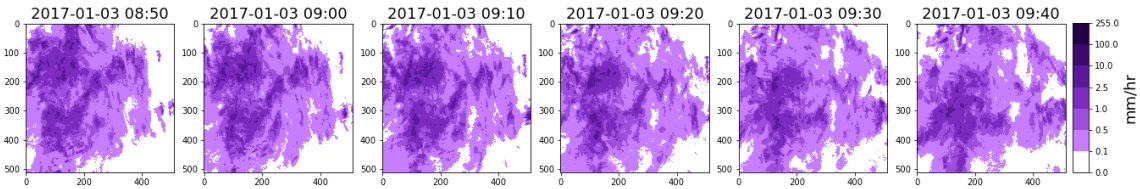


(b) BL-1 prediction for whole Denmark at timestamp 2017-01-03 08:50 to 2017-01-03 09:40.

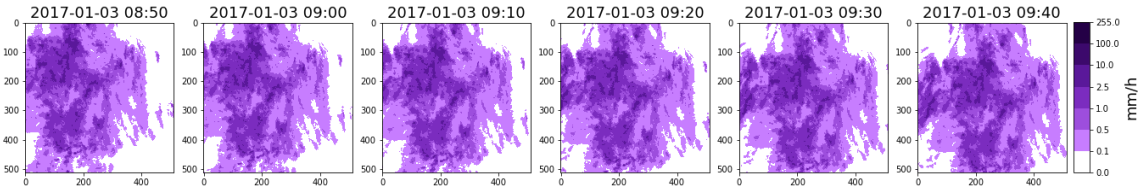


(c) N-UNet prediction for whole Denmark at timestamp 2017-01-03 08:50 to 2017-01-03 09:40.

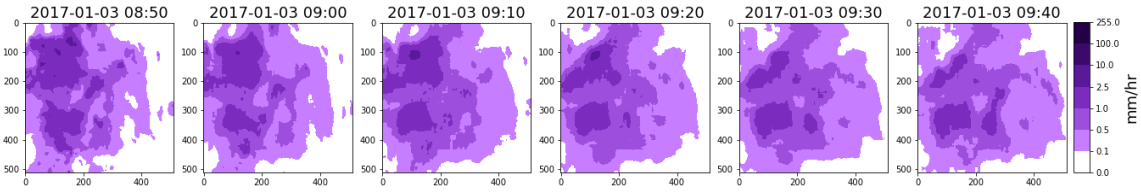
Figure 5.1: An example of precipitation nowcasting of BL-1 and N-UNet model with a lead time of 1 hour. The dimension of a nowcasting rain map is 1472 × 1760 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.



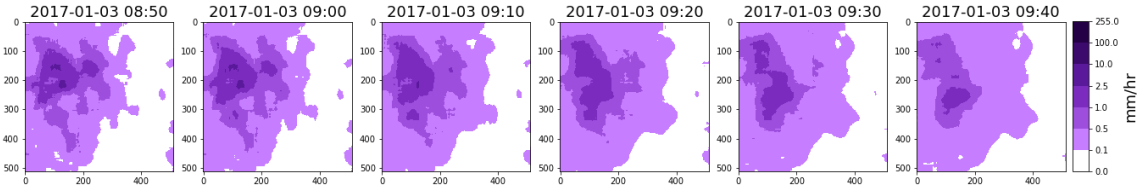
(a) Radar observations of cropped region (512x512) at timestamp 2017-01-03 08:50 to 2017-01-03 09:40.



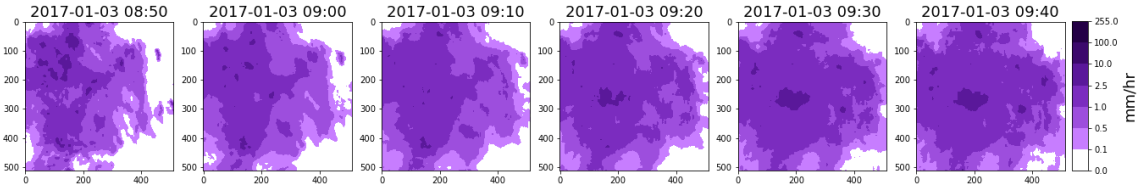
(b) BL-2 prediction at timestamp 2017-01-03 08:50 to 2017-01-03 09:40.



(c) Cr-UNet prediction at timestamp 2017-01-03 08:50 to 2017-01-03 09:40.



(d) Rec-Cr-UNet prediction at timestamp 2017-01-03 08:50 to 2017-01-03 09:40.

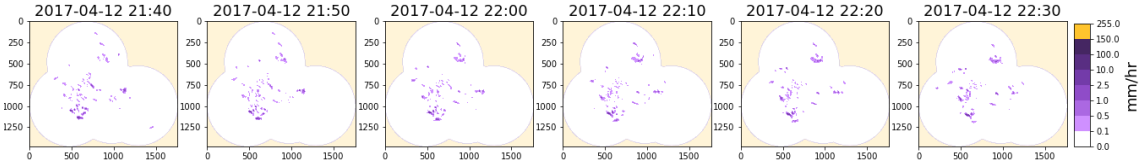


(e) Cr-UNet-PHR prediction at timestamp 2017-01-03 08:50 to 2017-01-03 09:40.

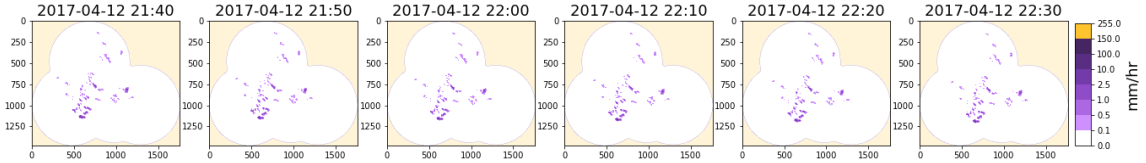
Figure 5.2: An example of precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model with a lead time of 1 hour. The dimension of a nowcasting rain map is 512 × 512 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.

Example - 2

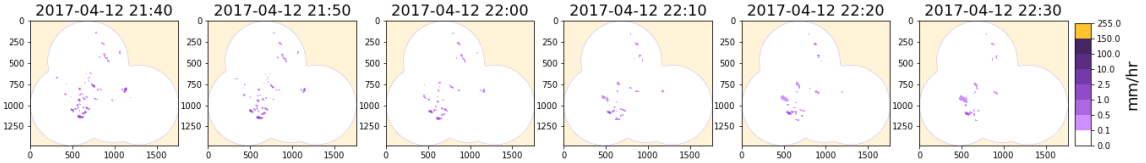
Another example of a precipitation nowcast for all models at timestamp 2017-04-12 21:30 with a lead time of 1 hour is shown in Figure 5.3 and Figure 5.4. Figure 5.3 represents the observation and precipitation nowcasting of BL-1 and N-UNet model in regards to the whole Denmark.On the other hand, Figure 5.4 represents the observation and precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model in regards to the cropped region (see Figure 3.3a) at timestamp 2017-01-03 08:40 with a lead time of 1 hour ahead.



(a) Radar observations for whole Denmark at timestamp 2017-04-12 21:40 to 2017-04-12 22:30.

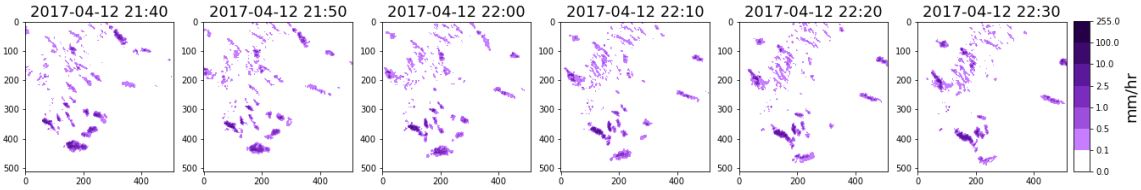


(b) BL-1 prediction for whole Denmark at timestamp 2017-04-12 21:40 to 2017-04-12 22:30.

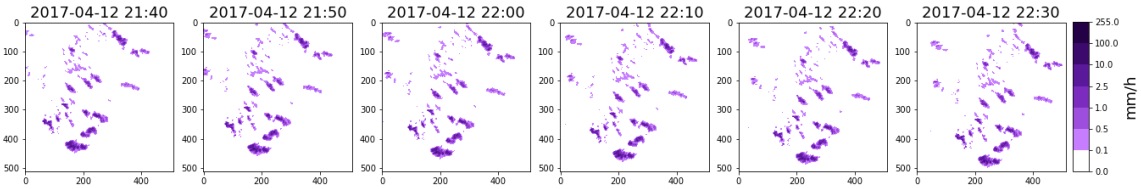


(c) N-UNet prediction for whole Denmark at timestamp 2017-04-12 21:40 to 2017-04-12 22:30.

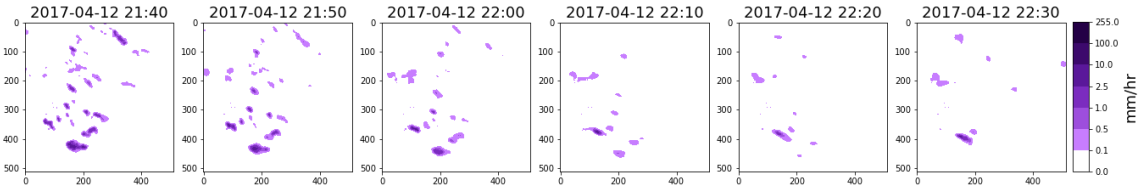
Figure 5.3: An example of precipitation nowcasting of BL-1 and N-UNet model with a lead time of 1 hour. The dimension of a nowcasting rain map is 1472 × 1760 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.



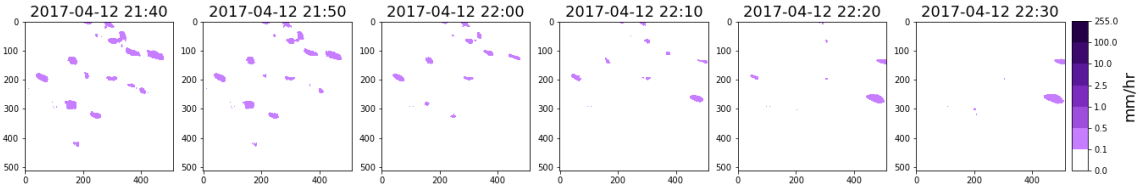
(a) Radar observations of cropped region (512x512) at timestamp 2017-04-12 21:40 to 2017-04-12 22:30.



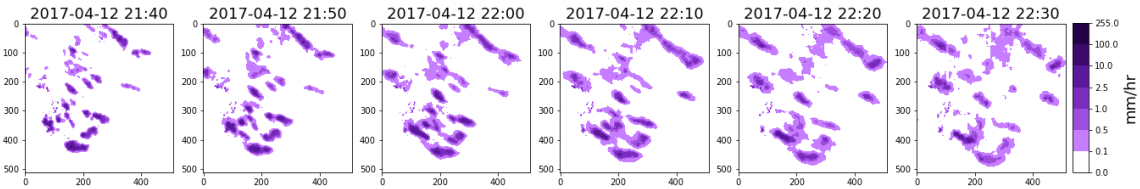
(b) BL-2 prediction at timestamp 2017-04-12 21:40 to 2017-04-12 22:30.



(c) Cr-UNet prediction at timestamp 2017-04-12 21:40 to 2017-04-12 22:30.



(d) Rec-Cr-UNet prediction at timestamp 2017-04-12 21:40 to 2017-04-12 22:30.



(e) Cr-UNet-PHR prediction at timestamp 2017-04-12 21:40 to 2017-04-12 22:30.

Figure 5.4: An example of precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model with a lead time of 1 hour. The dimension of a nowcasting rain map is 512 × 512 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.

The example-1 and example-2 demonstrated above represent the true observation along with precipitation nowcast generated by all the models at two different timestamps.

In example-1 from Figure 5.1 it can be seen that it is a stratiform precipitation event with little movement which means there is more horizontal development in comparison to the vertical development characteristic of convection. It is visually challenging to track the development and motion pattern of rainstorms at the scale of 736 km × 880 km by eye. With a very close look, one can see that the N-UNet model captures the development of low intense (up to 0.5 mm/hr) rainfall cluster better than BL-1 (see Figure 5.1b and Fig-

ure 5.1c) but failed to capture intense rainfall because the model is trained with dataset-1 where 50% of the rainfall map contains less than 1.5% of rain data which means 98.5% is no rain (0). As a result, the model tends to predict low intensities of rainfall.

From Figure 5.2 it can be seen that Cr-UNet is able to capture the development and motion of the rainstorm better than BL-2. This is because Cr-UNet uses both radar observation and radar nowcasts map as input, which helps Cr-UNet to learn the progression of rainfall. Rec-Cr-UNet does worse predictions compared to all others. It can be seen that, with the increasing of prediction time horizon Rec-Cr-UNet is unable to capture the development of both intense and low intense rainfall cluster. The reason could be that the Rec-Cr-UNet uses a recursive strategy where predictions are used instead of observations for multistep predictions; therefore the performance of the model degraded as the predicted time horizon increases even though the model uses same nowcast as input to compensate the error. Cr-UNet-PHR produces better output than all the models, but it predicts intense rainfall in spread that is pretty big compared to true observations and the other models predictions.

In the case of example-2, one can see that precipitation nowcasts by all neural networks models (N-UNet, Cr-UNet and Rec-Cr-UNet) except Cr-UNet-PHR for both first and second group (see Figure 5.3 and Figure 5.4) are quite worse than baseline (BL-1 and BL-2) predictions. One possible reason for this could be the sparseness of the observation events. However, the predicted output of Cr-UNet-PHR is better than the output from other models because of the customized loss function of it (see subsection 4.1.4), which provides equal priority to different rainfall classes.

Furthermore, in the case of example-1, it can be seen that in contrast to the ground truth images Figure 5.1a and Figure 5.2a the predicted precipitation maps of all trained models are quite blurry. One possible reason for this is the use of MSE as a loss function which is biased towards blurry images. An article on medium, Oleg Rybkin 2018 mentioned that models trained with MSE, tend to output the mean and blurry image. This is true, it can be seen in Figure 5.1c,Figure 5.2c, Figure 5.2d and Figure 5.2e that the predicted precipitation maps of all trained models look very smooth. It is also true in the case of example-2. Five additional examples have been presented in Appendix A through Figure A.1 - A.10.

5.2 Evaluation scores

5.2.1 Forecast verification scores and NMSE

The three different forecast verification scores (CSI, Hit rate and FAR) and NMSE for all the models are presented here. Figure 5.5 below represents the CSI scores for different rainfall intensity with a lead time of 1 hour. A higher score possessed by a model means better performance. BL-1 is the COTREC nowcast for Denmark as a whole, which is comparable to N-UNet. In contrast, BL-2 is COTREC's nowcast for the cropped region, which is comparable to Cr-UNet, Rec-Cr-UNet, and Cr-UNet-PHR.

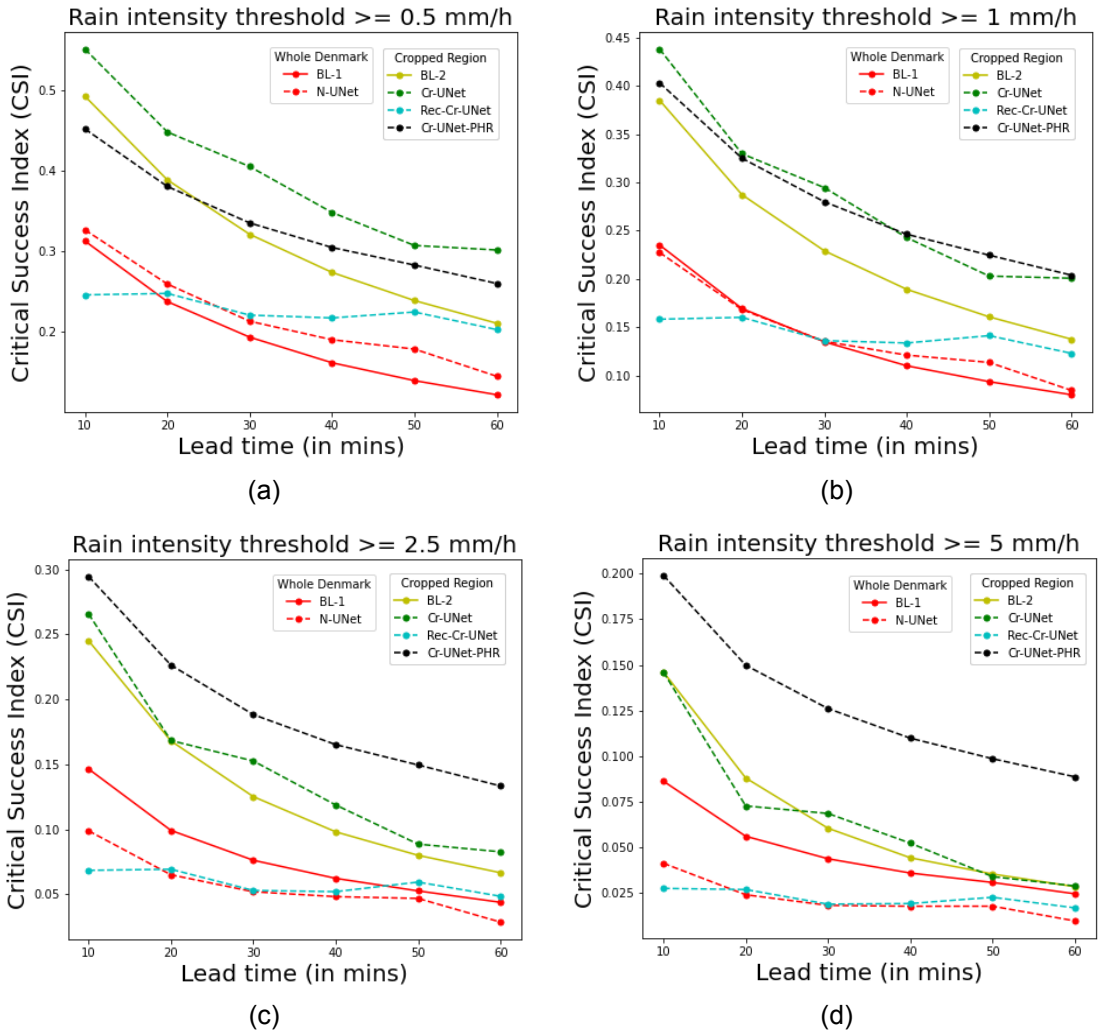


Figure 5.5: Comparison of models performance at four different rainfall intensity thresholds (>=0.5,>=1,>=2.5 and >=5 mm/h) and lead time by CSI score on test dataset. The metrics are shown as a function of lead time.

In Figure 5.6, hit rate scores for different rainfall intensity with a lead time of 1 hour for all the models have been shown. A higher score possessed by a model means better performance.

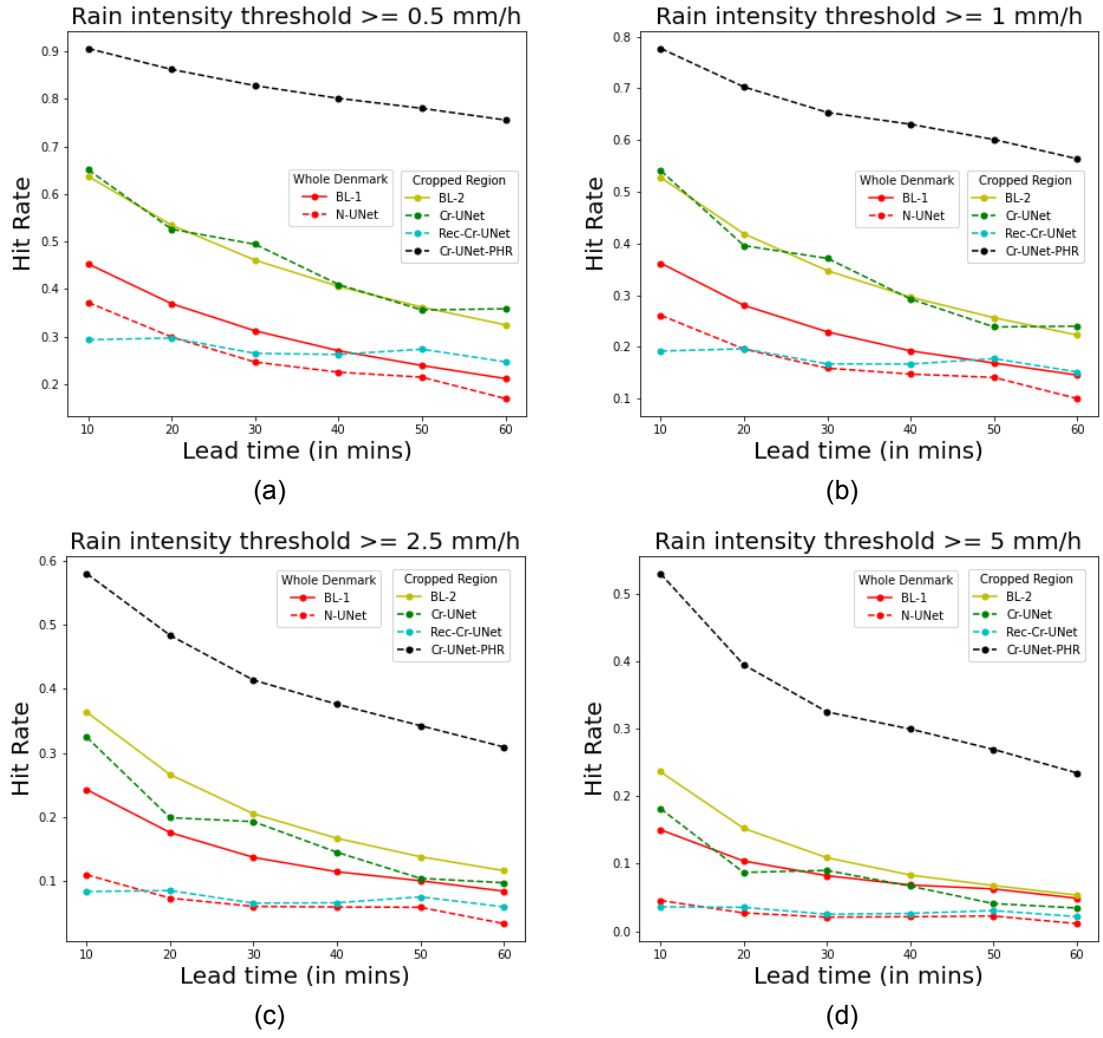


Figure 5.6: Comparison of models performance at four different rainfall intensity thresholds (>=0.5,>=1,>=2.5 and >=5 mm/h) and lead time by Hit rate on test dataset. The metrics are shown as a function of lead time.

In Figure 5.7, false alarm rate scores for different rainfall intensity with a lead time of 1 hour for all the models have been shown. A lower score possessed by a model means better performance.

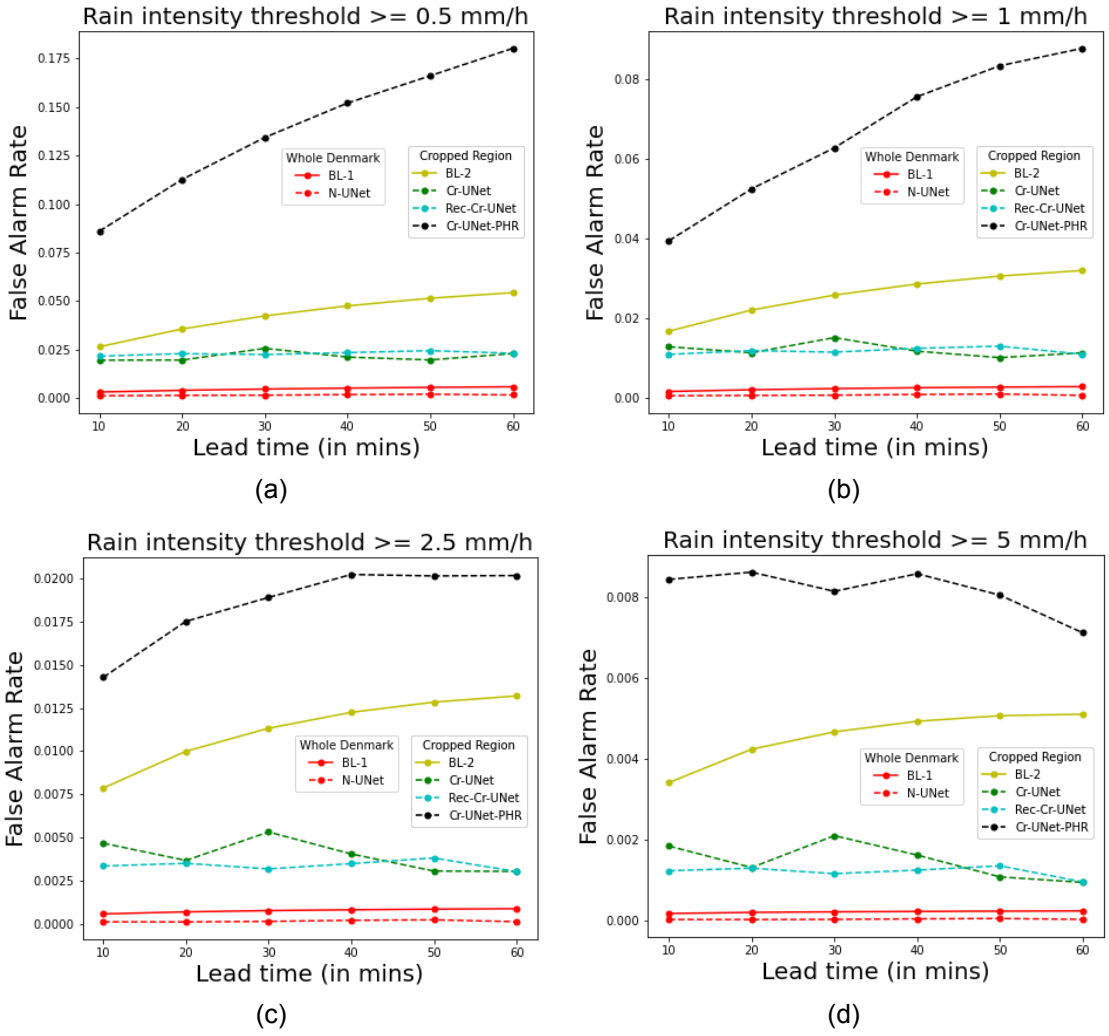


Figure 5.7: Comparison of models performance at four different rainfall intensity thresholds (>=0.5,>=1,>=2.5 and >=5 mm/h) and lead time by FAR score on test dataset. The metrics are shown as a function of lead time.

From Figure 5.5,Figure 5.6 and Figure 5.7 it can be seen that the overall performance of N-UNet model is worse than its corresponding baseline model BL-1. In the case of CSI metrics up to thresholds >= 1mm/h and for FAR metrics for all thresholds, it outperforms the baseline model BL-1 in all lead times. In contrast to N-UNet and other models, Cr-Unet model performs well in most scores in all lead times. However, from Figure 5.6 it can be seen that Cr-UNet model suffers a bit for higher intensity threshold (>=5 mm/h) in comparison with BL-2 and Cr-UNet-HPR. Cr-UNet-PHR is basically designed to serve this purpose. From Figure 5.5,Figure 5.6 and Figure 5.7 it is clear that Cr-UNet-PHR model perform very well for higher intensity thresholds. However, it can be seen on the Figure 5.7 that the FAR score of Cr-UNet-HPR model is a bit higher than other models.

In Figure 5.8 the normalized mean squared error (NMSE) between the true radar observations and predicted outputs by different models have been shown. It is noticeable that

Cr-UNet model possesses a very low NMSE than all other models for all lead times. It can also be seen on the figures (Figure 5.5,Figure 5.6 and Figure 5.7) that the different models are showing different results on different rainfall intensities and lead time. This is based on the same causes that happen for the different models, which is also explained in section 5.1. Above all, the performance of most of the models, in terms of CSI, Hit rate and NMSE metrics, decreases with increasing forecast lead time.

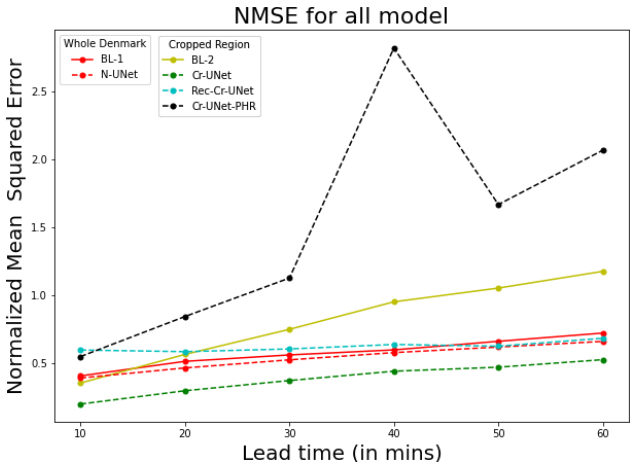


Figure 5.8: Normalized Mean squared Error.The metrics are shown as a function of lead time. All values represent the average of the corresponding metric over all test events.

5.2.2 Evaluation of multi-class contingency table

The multi-class contingency tables are generated for the models that are designed for cropped regions (BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR) because they used dataset-2, which contains data for the whole period (December 2016 to January 2018).The multi-class contingency table analysis has been performed on the forecasted rain map at timestamp t+60 min, where *t* is the forecast timestamp. Figure 5.9 below show the multi-class contingency table for BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model for all test events.

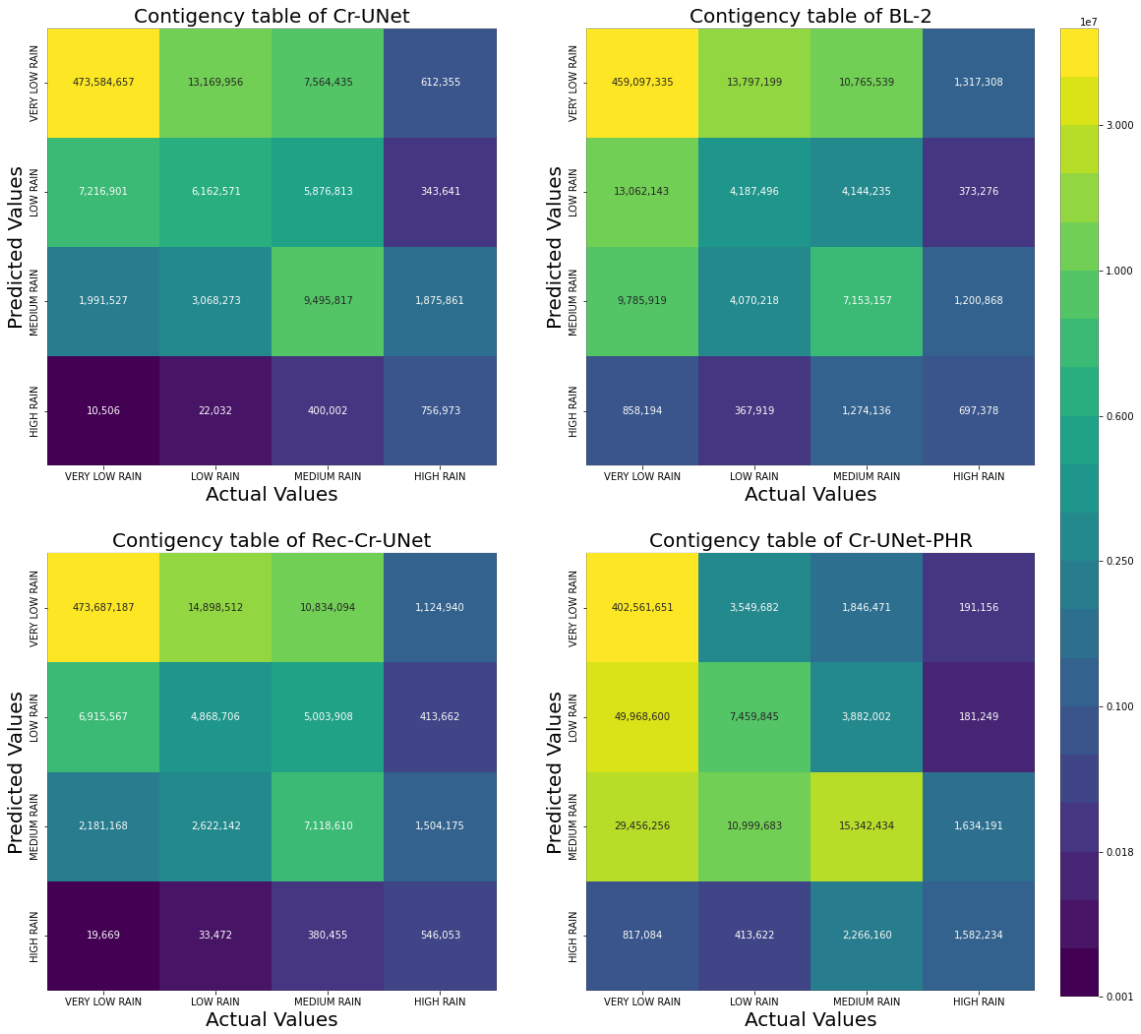


Figure 5.9: Comparison of multi-class contingency table among the models for cropped regions. The multi-class contingency table is generated with respect to true observation and model prediction at timestamp t+60 min for all test events based on the rainfall intensity threshold described at subsection 4.4.2.

From the multiclass contingency tables shown in Figure 5.9, it can be seen that overall the Cr-UNet model performs well here. It predicts all rain intensities better than the BL-2 and Rec-Cr-UNet models. It can also be seen from Figure 5.9 that on the one hand, the Rec-Cr-UNet model predicts low-intensity rainfall better than BL-2, but on the other hand, BL-2 predicts intense rainfall better than Rec-Cr-UNet. One can also see that, the Cr-UNet-PHR model outperforms in predicting intense rainfall than the BL-2, Cr-UNet and Rec-Cr-UNet models, which confirms the findings from the previous sections (section 5.1 and subsection 5.2.1).

5.3 Computational requirements and model losses

The models were trained on a single GPU (NVIDIA Tesla A100) at DTU HPC clusters. The average training time taken to complete 1 epoch, maximum memory consumption, average memory consumption, and batch size used by each of the models are tabulated in Table 5.1.

Table 5.1: Time and memory consumed by each models.

Model	Time taken for 1 epoch	Max Memory	Average Memory	Number of Parameters	Input Output Dimensions	Batch Size	Train Sequences	Validation Sequences
N-UNet	48 mins	14644 MB	9368 MB	7,762,992	1472x1760	5	8435	2410
Cr-UNet	20 mins	10324 MB	6580 MB	31,040,368	512x512	10	14450	4130
Rec-Cr-UNet	34 mins	7404 MB	5216 MB	46,560,432	512x512	10	14450	4130
Cr-UNet-PHR	23 mins	10556 MB	6740 MB	31,040,368	512x512	10	14450	4130

One can see from Table 5.1 that the N-UNet model has almost four times fewer parameters, is also trained using a few training samples, but takes more than twice the amount of time and consumes more memory than Cr-UNet and Cr-UNet-PHR models. Moreover, N-UNet could not be trained with a batch size greater than five (5) since the memory get exceeded. The reason is the number of input & output images and their dimensions. For example, in N-UNet, for a batch of size 6, *TensorFlow* needs to handle 78 images with dimensions 1472x1760 for an iteration that is somehow out of the scope of *TensorFlow*. Hence, the approach is not recommended from a computational perspective. Moreover, from Table 2.1 in subsection 2.5.2, one can also see that the largest image size used in the UNet-based precipitation study was 928x928 pixels.

The performance and convergence behavior of a model can be analyzed through the visualization of train and validation losses. The train and validation losses for all the trained models are shown in Figure 5.10. From Figure 5.10 it is clear that only the Cr-UNet model is converged for both train and validation losses approximately after 70 epochs. In the case of all other models, validation losses became converged after 50 or 60 epochs, but the train losses are still decreasing, which is the behavior of overfitting.

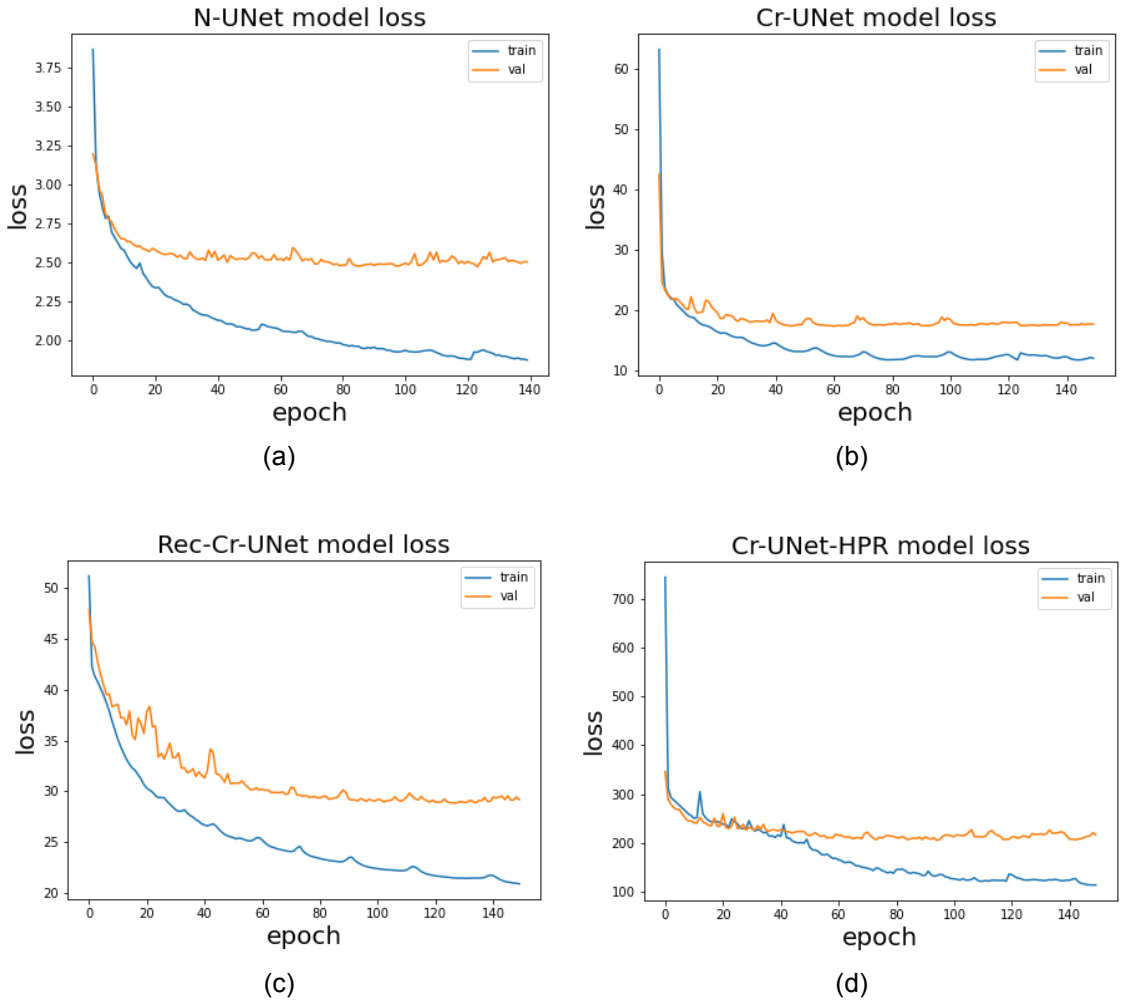


Figure 5.10: Train and validation losses of all trained model. All the model trained for 150 epochs except N-UNet. N-UNet trained for 140 epochs. The metrics are shown as a function of epoch.

6 Discussion

In the following section discussion on dastasets, models, results, and limitations & further works have been described.

6.1 Discussion on modelling and results

Throughout this project, U-Net based DL model has been described for precipitation nowcast at a lead time of 1 hour ahead. Dataset-1 and dataset-2 were tested on 4 models/variations (N-UNet, Cr-UNet, Rec-Cr-UNet and Cr-UNet-HPR) with different U-Net architectures.

The performance of N-UNet is limited. One can see it clearly from Figure 5.5 and Figure 5.6 that it performs worse compared to most of the models in all rainfall intensity thresholds because the model has several limitations due to the following reasons:

- The model trained on dataset-1, which contains only 8435 valid training events which is not enough in perspective to DNN.
- The dimensions of the input and output images are pretty big (1472X1760) which forces the model to be less complex in compare to other models to train at GPU.
- From Figure 5.10a it is clear that the model is prone to overfitting.

Nevertheless, the model outperforms the baseline model for low intensities, which is confirmed by the findings from the previous sections (section 5.1).

In contrast to N-UNet, BL-1 and BL-2, the results of the Cr-UNet model look consistent for all scores across all evaluation criteria. The model captures the development, movement and progression of rainfall better than the baseline models (BL-1 and BL-2) and N-UNet. This is because the Cr-UNet input includes additional data, CO-TREC nowcast maps, along with radar observations that describe rainfall movement. As well as in terms of computational complexity and training time requirements (see Table 5.1), the performance of Cr-UNet is better compared to all the trained model. However, it suffers for high intensity thresholds specially for intensity thresholds >=5 mm/h, because there are very less high rainfall samples in the dataset. To improve the performance of Cr-UNet models, it has been moved to Rec-Cr-UNet and Cr-UNet-PHR models. In the Rec-Cr-UNet model, multi-step forecasting has been handled recursively. Therefore, prediction errors are also accumulated; hence performance of the model deteriorates quickly. As the model is conservative, the UNet based precipitation nowcast model cannot be used in recursive autoregressive modelling for multistep prediction. The performance of this model is even worse than N-UNet model.

The result of Cr-UNet-PHR model shows that it outperforms all the models for higher intensity thresholds in all lead times in terms of CSI and Hit Rate. However, it performs badly in terms of FAR and NMSE score , for all lead times in comparison to all other models. One possible reason for this could be overfitting with the train data (see Figure 5.10d) as a result it predicts a lot of VERY LOW RAIN and LOW RAIN pixels to HIGH RAIN.

6.2 Limitations

The overall results from all trained models indicate that the models have some limitations which could be due to a variety of reasons (i.e., overfitting, use of inappropriate loss

functions, lack of hyper-parameter tuning, etc). From the loss plot in Figure 5.10 it can be observed that only the Cr-UNet model is converged for both train and validation losses; on the other hand, the others are prone to overfitting. To combat the overfitting 2 drop-out layers were used in the UNet architecture as well as datasets randomly shuffled, then the training, validation, and test set are split. Furthermore, some other techniques could be applied in model architecture such as adding normalized layers, few more drop-out layers, increasing or decreasing number of channels in convolution block as well as early stopping technique could be used during model training.

Another potential limitation is that predicted images are quite blurry. To get rid of this problem, a different loss functions such as MAE could be used. The model trained using MAE enhances the quality of the output image. Another solution for this problem could be using adversarial loss and by transforming the topological structure into Generative Adversarial Networks (GANs). GANs do not suffer from producing the mean image (Oleg Rybkin 2018).

Lastly, in the custom MSE loss functions weights for the different rain intensity groups have been calculated based on each and every training batch. However, the common practice for calculating weights in custom loss functions for different groups or classes is by considering the whole training dataset at once.

6.3 Future works

There are some clear opportunities for future work such as:

- Apply the Cr-UNet to predict other locations or train Cr-UNet model by taking samples from multiple locations.
- The topological structure of neural networks can be altered to exploit temporal correlation. Particularly, stacked ConvLSTM networks do a better job of capturing spatio-temporal relationships (Shi et al. 2015).
- It can also be applied to probability forecasting, where the Cr-UNet-PHR appears to be useful.
- Applying and combining additional data to the radar data, for extending the utility of the prediction. This could e.g. be data from other resources, such as wind data, three-dimensional radar volume data, dual-polarization radar moments, or the output fields of numerical weather prediction (NWP) models (Ayzel et al. 2020).

7 Conclusions

In this thesis, the convolution deep neural network model has been applied for the precipitation nowcasting task from a machine learning viewpoint. Throughout the study, the fully convolutional UNet architecture was chosen as the main model that could potentially provide the optimal solutions to fulfill the scope of the study. The outcome of the study, though, revealed that

- All of the DL models can be used for rainfall prediction. Model for dataset-1 can be used to predict rainfall for whole Denmark but its performance is low and hard to train. However, models for dataset-2 performs well except Rec-Cr-UNet and can be upscaled for predicting multiple locations.
- UNet-based model for cropped regions (Cr-UNet) can make better predictions than traditional radar-based precipitation nowcasting algorithm up to rainfall intensity thresholds >= 2.5mm/h.
- The study also shows that instead of standard MSE loss function custom MSE loss function significantly enhances model performance. The modified model, Cr-UNet-PHR outperforms the traditional radar-based precipitation nowcasting algorithm for all lead times (10 to 60 mins) and intensity thresholds.
- Learning the residue makes the Cr-UNet and Cr-UNet-PHR to track the rainfall movement better than N-UNet. However, learning residue will be a bit expensive as the model needs to process almost double the amount of data.

Bibliography

Abadi, Martín et al. (2016). “Tensorflow: A system for large-scale machine learning”. In: *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283.

Achleitner, Stefan, Stefan Fach, Thomas Einfalt, and Wolfgang Rauch (2009). “Nowcasting of rainfall and of combined sewage flow in urban drainage systems”. In: *Water Science & Technology*.

Aggarwal, Charu C. (2018). *Neural Networks and Deep Learning: A Textbook*. Springer Nature. Chap. 8, pp. 315–368. ISBN: 978-3-319-94463-0. DOI: https://doi.org/10.1007/978-3-319-94463-08.

Agrawal, Shreya, Luke Barrington, Carla Bromberg, John Burge, Cenk Gazen, and Jason Hickey (2019). *Machine Learning for Precipitation Nowcasting from Radar Images*. arXiv: 1912.12132 [cs.CV].

Ayzel, Georgy, Tobias Scheffer, and Maik Heistermann (2020). “RainNet v1.0: a convolutional neural network for radar-based precipitation nowcasting”. In: DOI: https://doi.org/10.5194/gmd-13-2631-2020.

Chen, Zhen and Dongbin Xiu (2021). “On generalized residual network for deep learning of unknown dynamical systems”. In: *Journal of Computational Physics* 438, p. 110362. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2021.110362. URL: https://www.sciencedirect.com/science/article/pii/S0021999121002576.

Chollet, Francois et al. (2015). *Keras*. URL: https://github.com/fchollet/keras.

Collette, Andrew (2013). *Python and HDF5*. O’Reilly.

Harris, Charles R. et al. (2020). “Array programming with NumPy”. In: *Nature* 585, pp. 357–362. DOI: 10.1038/s41586-020-2649-2.

Hunter, John D (2007). “Matplotlib: A 2D graphics environment”. In: *Computing in science & engineering* 9.3, pp. 90–95.

IBM Cloud Education (2020). *AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What’s the Difference?* https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks. [Online; accessed 20-October-2021].

Jason Brownlee (2018). *Difference Between a Batch and an Epoch in a Neural Network*. https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/. [Online; accessed 30-January-2022].

Jóhannesson, Ari, Luca Vezzaro, Peter Steen Mikkelsen, and Roland Löwe (2021). “Approaches for unsupervised identification of data-driven models for flow forecasting in urban drainage systems”. In: *Journal of Hydroinformatics*. DOI: 10.2166/hydro.2021.020.

Kumar, Ashutosh, Tanvir Islam, Yoshihide Sekimoto, Chris Mattmann, and Brian Wilson (2020). “Convcast: An embedded convolutional LSTM based architecture for precipitation nowcasting using satellite data”. In: DOI: https://doi.org/10.1371/journal.pone.0230114.

Li, L., J. Joss, and W. Schmid (1995). “Nowcasting of Motion and Growth of Precipitation with Radar over a Complex Orography”. In: *Journal of Applied Meteorology*.

Mecklenburg, S., J. Joss, and W. Schmid (2000). “Improving the nowcasting of precipitation in an Alpine region with an enhanced radar echo tracking algorithm”. In: *Journal of Hydrology*.

Nerini, Daniele (2019). “Ensemble precipitation nowcasting: limits to prediction, localization and seamless blending”. PhD thesis. ETH ZURICH.

Oleg Rybkin (2018). *The reasonable ineffectiveness of pixel metrics for future prediction (and what to do about it)*. https://medium.com/@olegrybkin_20684/the-reasonable-ineffectiveness-of-mse-pixel-loss-for-future-prediction-and-what-to-do-about-it-4dca8152355d. [Online; accessed 17-December-2021].

Prudden, Rachel, Samantha Adams, Dmitry Kangin, Niall Robinson†, Suman Ravuri, Shakir Mohamed, and Alberto Arribas (2020). *A REVIEW OF RADAR-BASED NOWCASTING OF PRECIPITATION AND APPLICABLE MACHINE LEARNING TECHNIQUES*. arXiv: 2005.04988 [physics.ao-ph].

Samsi, Siddharth, Christopher J. Mattioli, and Mark S. Veillette (2019). *Distributed Deep Learning for Precipitation Nowcasting*. arXiv: 1908.10964 [cs.LG].

Shi, Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo (2015). *Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting*. arXiv: 1506.04214 [cs.CV].

Shi, Xingjian, Zhihan Gao, Leonard Lausen, Hao Wang, and Dit-Yan Yeung (2017). *Deep Learning for Precipitation Nowcasting: A Benchmark and A New Model*. arXiv: 1706.03458 [cs.CV].

Sivaram T (2021). *All You Need to Know About Skip Connections*. https://www.analyticsvidhya.com/blog/2021/08/all-you-need-to-know-about-skip-connections/. [Online; accessed 08-January-2022].

Sun, Juanzhen et al. (2014). "USE OF NWP FOR NOWCASTING CONVECTIVE PRECIPITATION Recent Progress and Challenges". In: *Bulletin of the American Meteorological Society*. DOI: https://doi.org/10.1175/BAMS-D-11-00263.1.

Thorndahl, Søren et al. (2013). "Comparison of short-term rainfall forecasts for model based flow prediction in urban drainage systems". In: *Water Science & Technology*.

Trebing, Kevin, Tomasz Stańczyk, and Siamak Mehrkanoon (2021). "SmaAt-UNet: Precipitation Nowcasting using a Small Attention-UNet Architecture". In: *Pattern Recognition Letters*. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2021.01.036. URL: https://www.sciencedirect.com/science/article/pii/S0167865521000556.

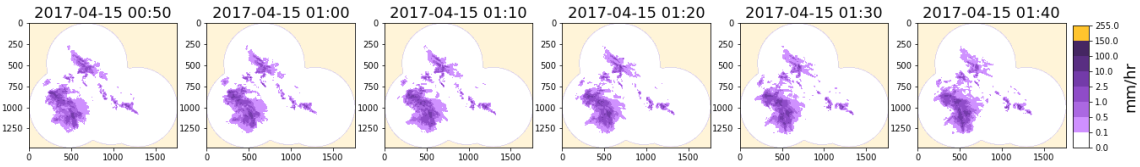
Wikipedia contributors (2021a). *Deep learning — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Deep_learning&oldid=1049101893. [Online; accessed 20-October-2021].

– (2021b). *U-Net — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=U-Net&oldid=1049752242. [Online; accessed 29-October-2021].

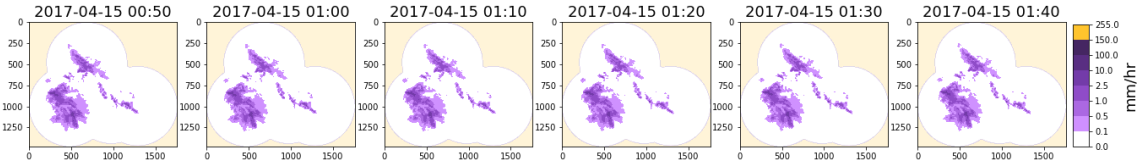
A Appendix

Example - 3

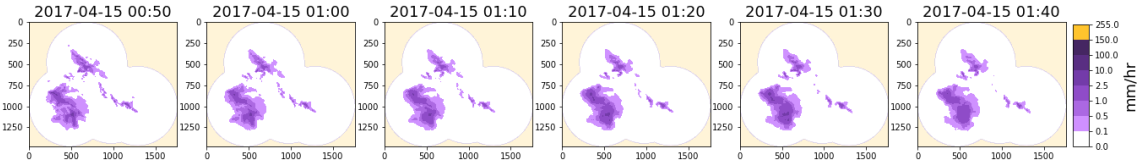
An example of a precipitation nowcast for all models at timestamp 2017-04-15 00:40 with a lead time of 1 hour is shown in Figure A.1 and Figure A.2. Figure A.1 represents the observation and precipitation nowcasting of BL-1 and N-UNet model in regards to the whole Denmark. On the other hand, Figure A.2 represents the observation and precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model in regards to the cropped region (see Figure 3.3a) at timestamp 2017-04-15 00:40 with a lead time of 1 hour ahead.



(a) Radar observations for whole Denmark at timestamp 2017-04-15 00:50 to 2017-04-15 01:40.

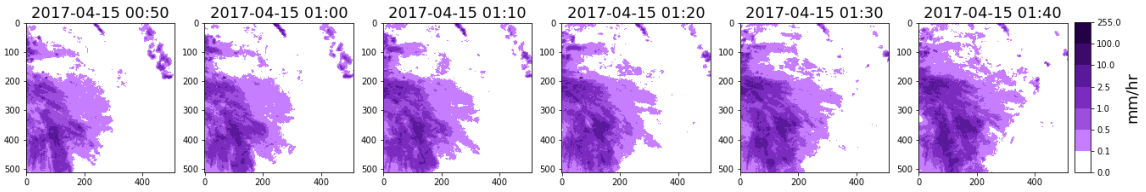


(b) BL-1 prediction for whole Denmark at timestamp 2017-04-15 00:50 to 2017-04-15 01:40.

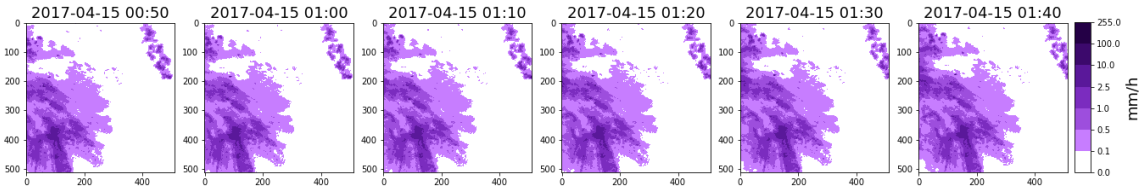


(c) N-UNet prediction for whole Denmark at timestamp 2017-04-15 00:50 to 2017-04-15 01:40.

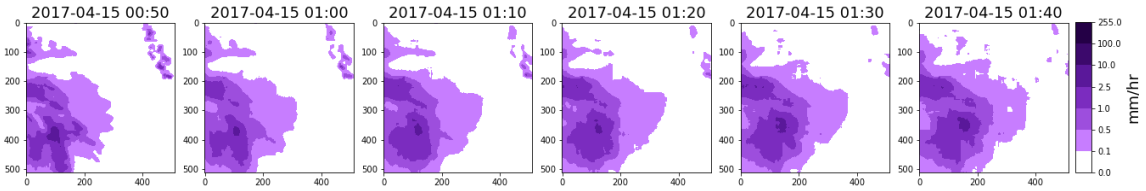
Figure A.1: An example of precipitation nowcasting of BL-1 and N-UNet model with a lead time of 1 hour. The dimension of a nowcasting rain map is 1472 × 1760 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.



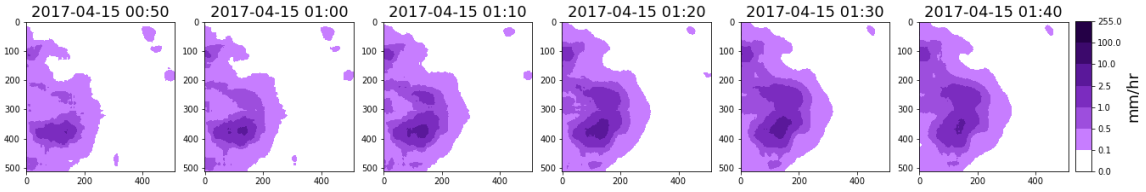
(a) Radar observations of cropped region (512x512) at timestamp 2017-04-15 00:50 to 2017-04-15 01:40.



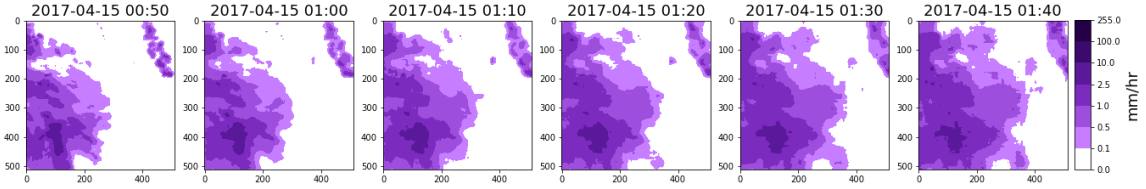
(b) BL-2 prediction at timestamp 2017-04-15 00:50 to 2017-04-15 01:40.



(c) Cr-UNet prediction at timestamp 2017-04-15 00:50 to 2017-04-15 01:40.



(d) Rec-Cr-UNet prediction at timestamp 2017-04-15 00:50 to 2017-04-15 01:40.

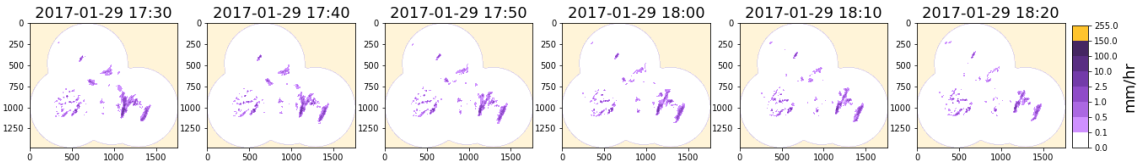


(e) Cr-UNet-PHR prediction at timestamp 2017-04-15 00:50 to 2017-04-15 01:40.

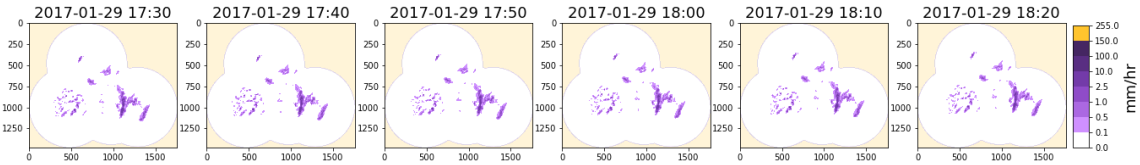
Figure A.2: An example of precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model with a lead time of 1 hour. The dimension of a nowcasting rain map is 512 × 512 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.

Example - 4

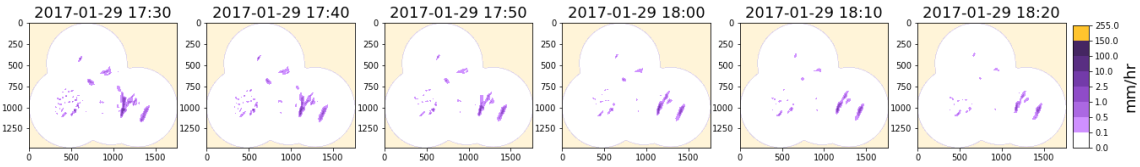
Another example of a precipitation nowcast for all models at timestamp 2017-01-29 17:20 with a lead time of 1 hour is shown in Figure A.3 and Figure A.4. Figure A.3 represents the observation and precipitation nowcasting of BL-1 and N-UNet model in regards to the whole Denmark. On the other hand, Figure A.4 represents the observation and precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model in regards to the cropped region (see Figure 3.3a) at timestamp 2017-01-29 17:20 with a lead time of 1 hour ahead.



(a) Radar observations for whole Denmark at timestamp 2017-01-29 17:30 to 2017-01-29 18:20.

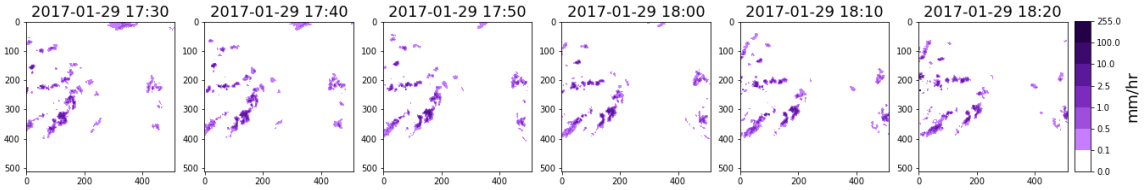


(b) BL-1 prediction for whole Denmark at timestamp 2017-01-29 17:30 to 2017-01-29 18:20.

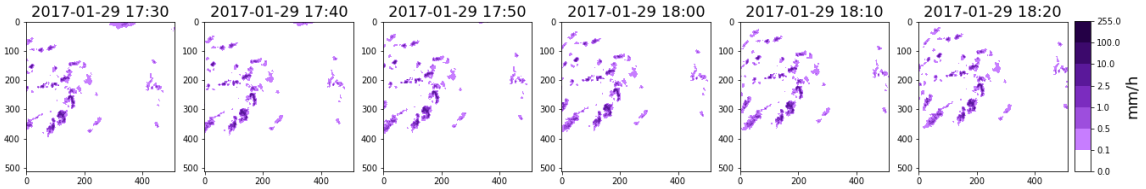


(c) N-UNet prediction for whole Denmark at timestamp 2017-01-29 17:30 to 2017-01-29 18:20.

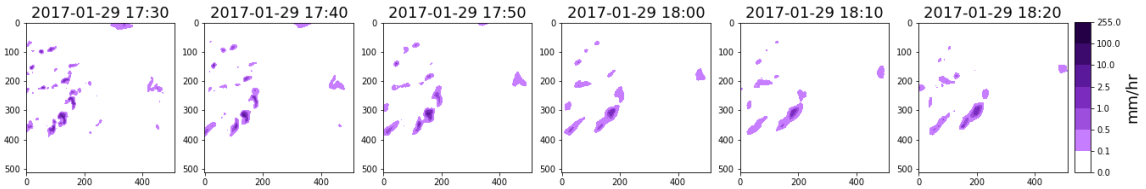
Figure A.3: An example of precipitation nowcasting of BL-1 and N-UNet model with a lead time of 1 hour. The dimension of a nowcasting rain map is 1472 × 1760 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.



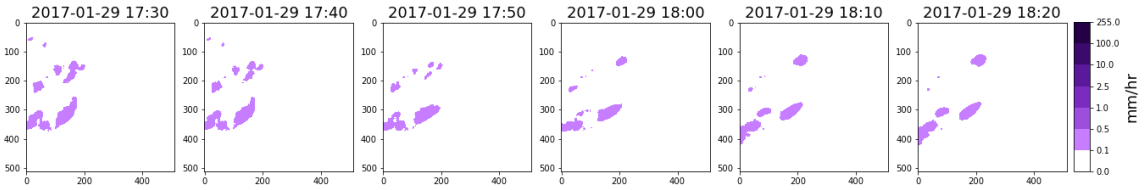
(a) Radar observations of cropped region (512x512) at timestamp 2017-01-29 17:30 to 2017-01-29 18:20.



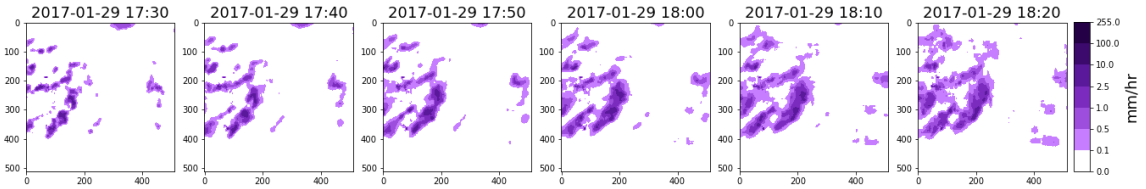
(b) BL-2 prediction at timestamp 2017-01-29 17:30 to 2017-01-29 18:20.



(c) Cr-UNet prediction at timestamp 2017-01-29 17:30 to 2017-01-29 18:20.



(d) Rec-Cr-UNet prediction at timestamp 2017-01-29 17:30 to 2017-01-29 18:20.

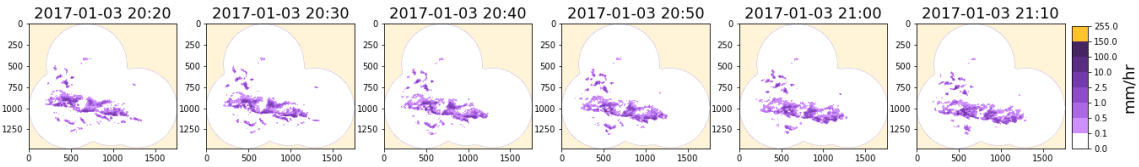


(e) Cr-UNet-PHR prediction at timestamp 2017-01-29 17:30 to 2017-01-29 18:20.

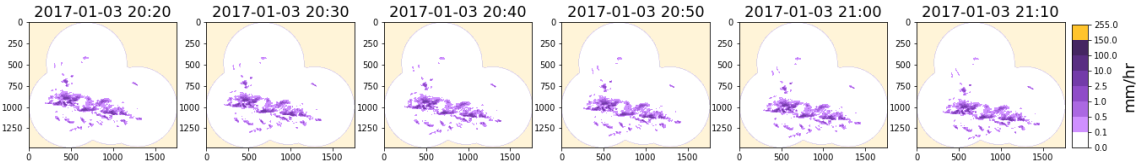
Figure A.4: An example of precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model with a lead time of 1 hour. The dimension of a nowcasting rain map is 512 × 512 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.

Example - 5

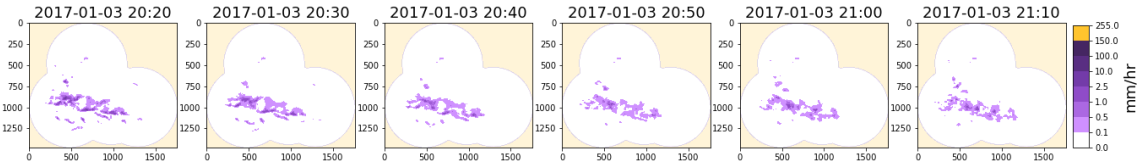
Another example of a precipitation nowcast for all models at timestamp 2017-01-03 20:10 with a lead time of 1 hour is shown in Figure A.5 and Figure A.6. Figure A.5 represents the observation and precipitation nowcasting of BL-1 and N-UNet model in regards to the whole Denmark. On the other hand, Figure A.6 represents the observation and precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model in regards to the cropped region (see Figure 3.3a) at timestamp 2017-01-03 20:10 with a lead time of 1 hour ahead.



(a) Radar observations for whole Denmark at timestamp 2017-01-03 20:20 to 2017-01-03 21:10.

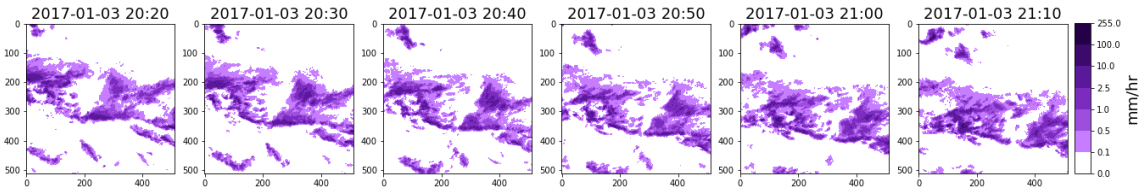


(b) BL-1 prediction for whole Denmark at timestamp 2017-01-03 20:20 to 2017-01-03 21:10.

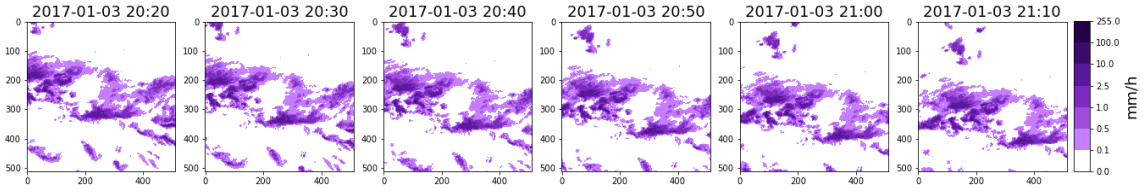


(c) N-UNet prediction for whole Denmark at timestamp 2017-01-03 20:20 to 2017-01-03 21:10.

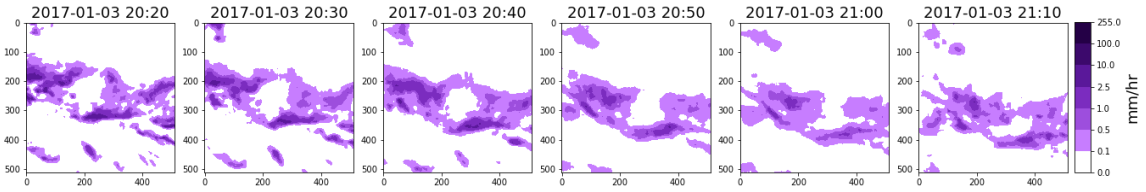
Figure A.5: An example of precipitation nowcasting of BL-1 and N-UNet model with a lead time of 1 hour. The dimension of a nowcasting rain map is 1472 × 1760 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.



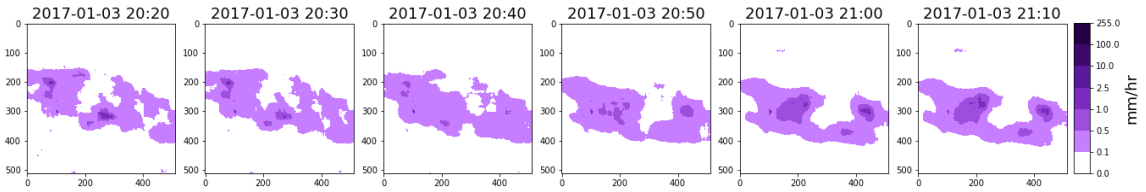
(a) Radar observations of cropped region (512x512) at timestamp 2017-01-03 20:20 to 2017-01-03 21:10.



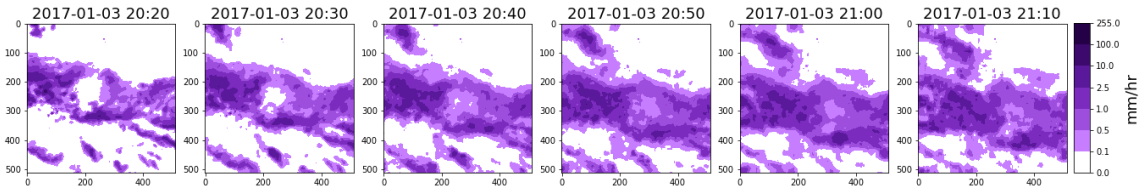
(b) BL-2 prediction at timestamp 2017-01-03 20:20 to 2017-01-03 21:10.



(c) Cr-UNet prediction at timestamp 2017-01-03 20:20 to 2017-01-03 21:10.



(d) Rec-Cr-UNet prediction at timestamp 2017-01-03 20:20 to 2017-01-03 21:10.

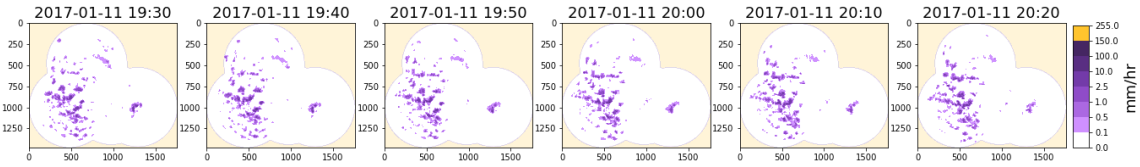


(e) Cr-UNet-PHR prediction at timestamp 2017-01-03 20:20 to 2017-01-03 21:10.

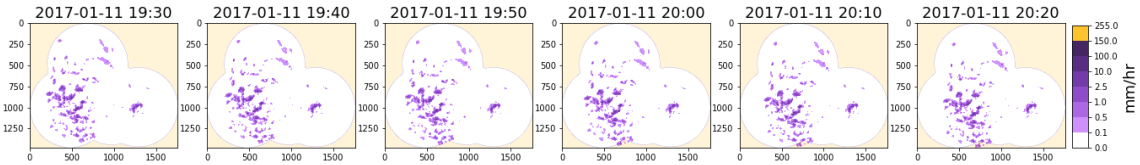
Figure A.6: An example of precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model with a lead time of 1 hour. The dimension of a nowcasting rain map is 512 × 512 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.

Example - 6

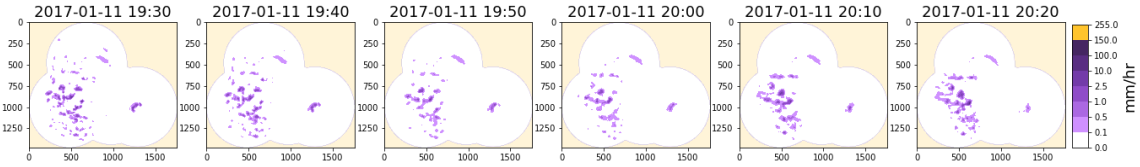
Another example of a precipitation nowcast for all models at timestamp 2017-01-11 19:20 with a lead time of 1 hour is shown in Figure A.7 and Figure A.8. Figure A.7 represents the observation and precipitation nowcasting of BL-1 and N-UNet model in regards to the whole Denmark. On the other hand, Figure A.8 represents the observation and precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model in regards to the cropped region (see Figure 3.3a) at timestamp 2017:01:11 19:20 with a lead time of 1 hour ahead.



(a) Radar observations for whole Denmark at timestamp 2017-01-11 19:30 to 2017-01-11 20:20.

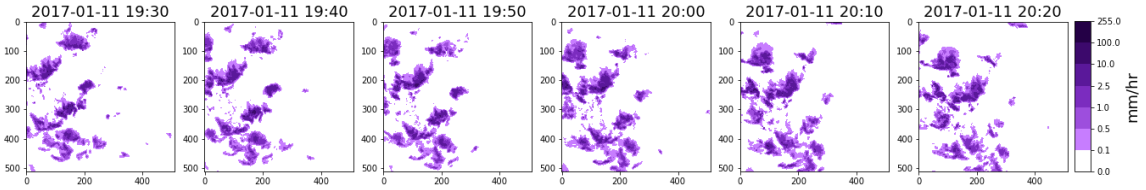


(b) BL-1 prediction for whole Denmark at timestamp 2017-01-11 19:30 to 2017-01-11 20:20.

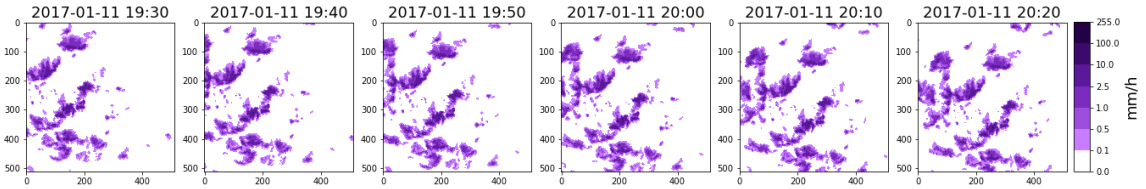


(c) N-UNet prediction for whole Denmark at timestamp 2017-01-11 19:30 to 2017-01-11 20:20.

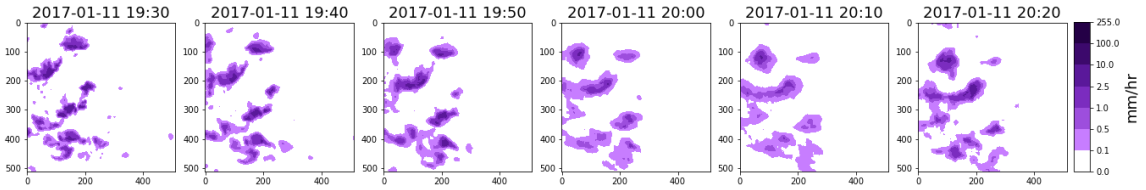
Figure A.7: An example of precipitation nowcasting of BL-1 and N-UNet model with a lead time of 1 hour. The dimension of a nowcasting rain map is 1472 × 1760 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.



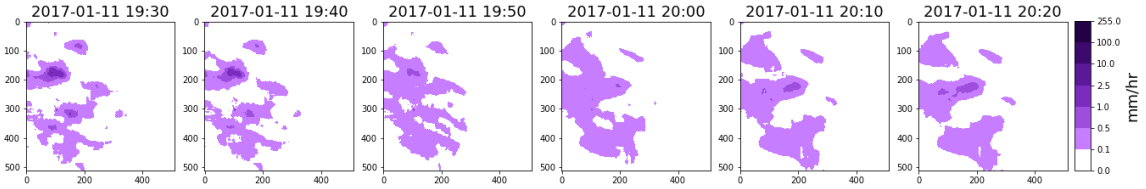
(a) Radar observations of cropped region (512x512) at timestamp 2017-01-11 19:30 to 2017-01-11 20:20.



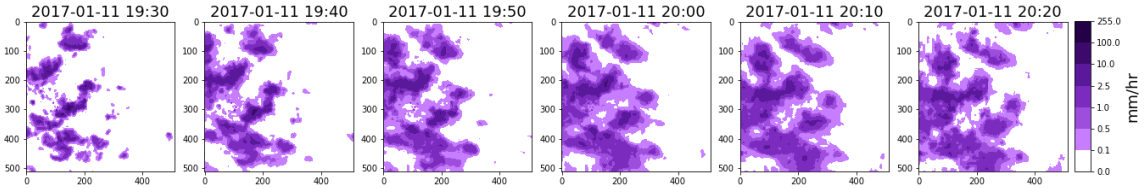
(b) BL-2 prediction at timestamp 2017-01-11 19:30 to 2017-01-11 20:20.



(c) Cr-UNet prediction at timestamp 2017-01-11 19:30 to 2017-01-11 20:20.



(d) Rec-Cr-UNet prediction at timestamp 2017-01-11 19:30 to 2017-01-11 20:20.

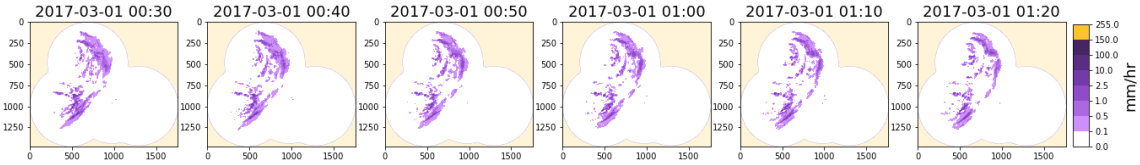


(e) Cr-UNet-PHR prediction at timestamp 2017-01-11 19:30 to 2017-01-11 20:20.

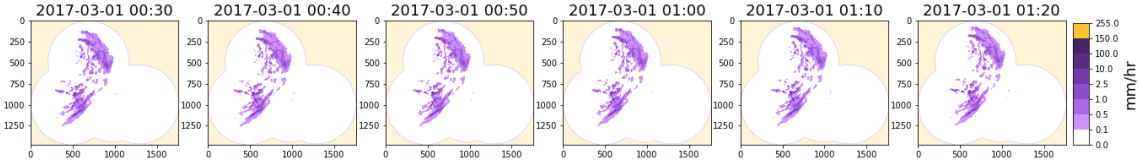
Figure A.8: An example of precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model with a lead time of 1 hour. The dimension of a nowcasting rain map is 512 × 512 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.

Example - 7

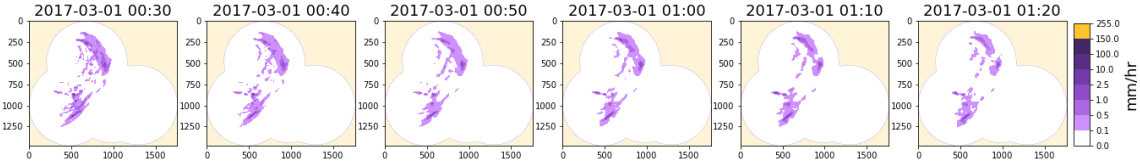
Another example of a precipitation nowcast for all models at timestamp 2017-03-01 00:20 with a lead time of 1 hour is shown in Figure A.9 and Figure A.10. Figure A.9 represents the observation and precipitation nowcasting of BL-1 and N-UNet model in regards to the whole Denmark. On the other hand, Figure A.10 represents the observation and precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model in regards to the cropped region (see Figure 3.3a) at timestamp 2017-03-01 00:20 with a lead time of 1 hour ahead.



(a) Radar observations for whole Denmark at timestamp 2017-03-01 00:30 to 2017-03-01 01:20.

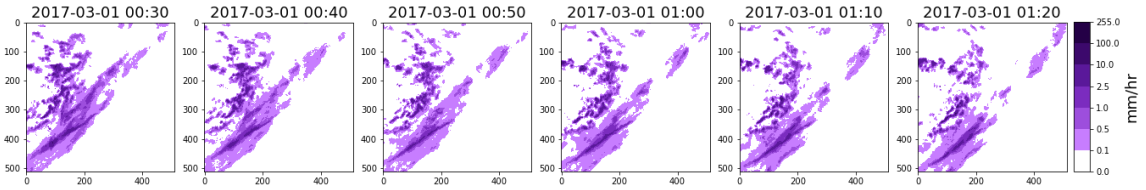


(b) BL-1 prediction for whole Denmark at timestamp 2017-03-01 00:30 to 2017-03-01 01:20.

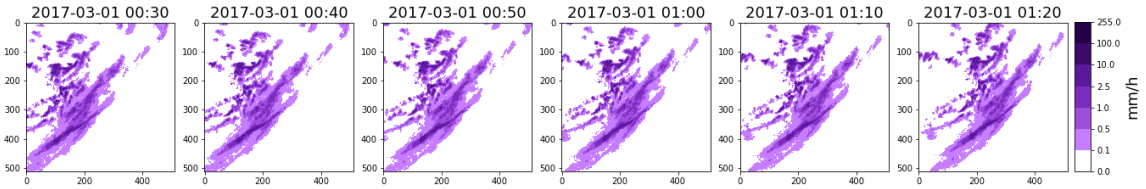


(c) N-UNet prediction for whole Denmark at timestamp 2017-03-01 00:30 to 2017-03-01 01:20.

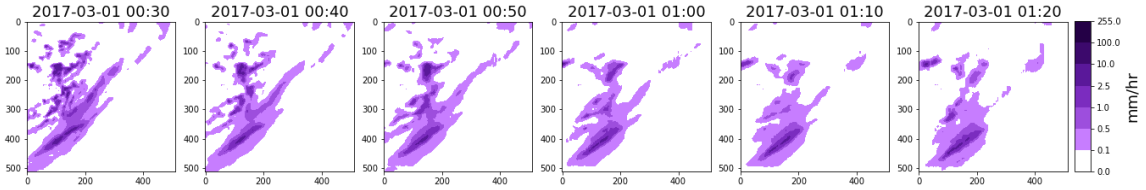
Figure A.9: An example of precipitation nowcasting of BL-1 and N-UNet model with a lead time of 1 hour. The dimension of a nowcasting rain map is 1472 × 1760 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.



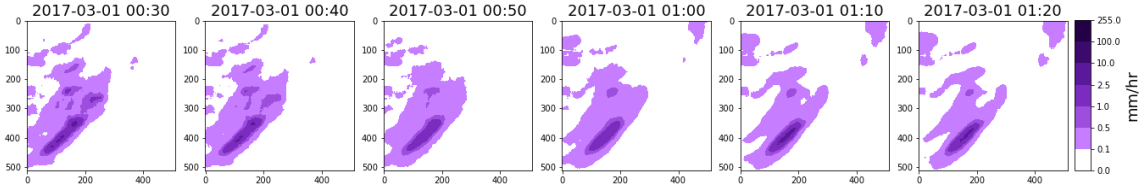
(a) Radar observations of cropped region (712x712) at timestamp 2017-03-01 00:30 to 2017-03-01 01:20.



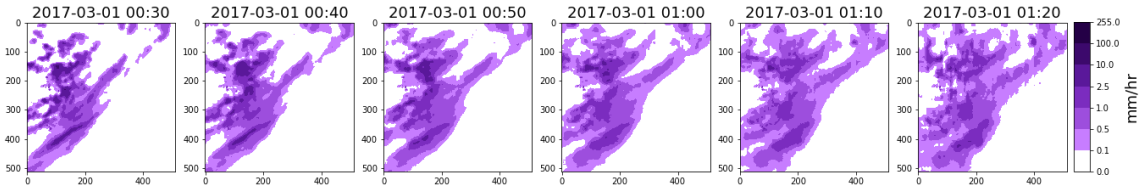
(b) BL-2 prediction at timestamp 2017-03-01 00:30 to 2017-03-01 01:20.



(c) Cr-UNet prediction at timestamp 2017-03-01 00:30 to 2017-03-01 01:20.



(d) Rec-Cr-UNet prediction at timestamp 2017-03-01 00:30 to 2017-03-01 01:20.



(e) Cr-UNet-PHR prediction at timestamp 2017-03-01 00:30 to 2017-03-01 01:20.

Figure A.10: An example of precipitation nowcasting of BL-2, Cr-UNet, Rec-Cr-UNet and Cr-UNet-PHR model with a lead time of 1 hour. The dimension of a nowcasting rain map is 512 × 512 and one pixel corresponds to the accumulated rainfall in the last 10 minutes on half-square kilometer.

Technical
University of
Denmark

Richard Petersens Plads, Building 324
2800 Kgs. Lyngby
Tlf. 4525 1700

www.compute.dtu.dk