Aalto University
School of Science
Master's Programme in Life Science Technologies

Bent Ivan Oliver Harnist

Probabilistic Precipitation Nowcasting using Bayesian Convolutional Neural Networks

Master's Thesis
Espoo, July 29th, 2022

Supervisor: Prof. Arno Solin
Advisors: D.Sc. Terhi Mäkinen,
D.Sc. Seppo Pulkkinen

Aalto University
School of Science
Master's Programme in Life Science Technologies

ABSTRACT OF
MASTER'S THESIS

Author: Bent Ivan Oliver Harnist

Title:
Probabilistic Precipitation Nowcasting using Bayesian Convolutional Neural Networks

Date: July 29th, 2022 **Pages:** 92

Major: Complex Systems **Code:** SCI3060

Supervisor: Prof. Arno Solin

Advisors: D.Sc. Terhi Mäkinen,
D.Sc. Seppo Pulkkinen

Precipitation nowcasting refers to forecasting precipitation at timescales of minutes to a few hours. This is usually approached by using models which are fed with real-time weather radar reflectivity images correlated with observed precipitation. Reliably nowcasting precipitation probabilities at fine spatial scales is a societally important challenge. This is due to the danger and economic losses potentially incurred by heavy rainfall and particularly flash floods provoked by them.

In this thesis a Convolutional Neural Network (CNN) previously used for nowcasting deterministic precipitation estimates is adapted into a Bayesian Neural Network (BNN) for producing an ensemble of possible prediction scenarios used to estimate precipitation probabilities. A BNN is based on the principle of placing probability distributions onto values of network parameters that are estimated using Bayesian inference. In this work these are modeled as Normal distributions, whose parameters are learned through backpropagation in a scheme called Stochastic Variational Inference (SVI). The prediction ensembles for a trained model can then be formed by Monte Carlo sampling each parameter from its distribution, thus inducing a predictive distribution on data.

The predictive skill of the developed model, abreviated BCNN, is evaluated against several established baseline models using a multitude of criteria covering various aspects of the precipitation nowcasts. On some aspects, BCNN is found to perform at a level close or equal to baseline models. However, it is found to be especially challenging to produce a well-calibrated and reliable probabilistic nowcast, owning likely to properties of the model and problems in the training procedure. Nevertheless, BCNN provided encouraging preliminary results, pointing out potential avenues of further study for probabilistic precipitation nowcasting using Bayesian Neural Networks.

Keywords: Precipitation nowcasting, Bayesian Neural Networks, Convolutional Neural Networks, probabilistic models

Language: English

Aalto-yliopisto
Perustieteiden korkeakoulu
Master's Programme in Life Science Technologies

DIPLOMITYÖN TIIVISTELMÄ

Tekijä:	Bent Ivan Oliver Harnist		
Työn nimi:	Probabilistinen Sateen Lähihetki-ennustaminen käyttäen Bayesilaisia Konvolutiivisia Neuroverkkoja		
Päiväys:	29. heinäkuuta 2022	**Sivumäärä:**	92
Pääaine:	Complex Systems	**Koodi:**	SCI3060
Valvoja:	Prof. Arno Solin		
Ohjaajat:	FT Terhi Mäkinen, FT Seppo Pulkkinen		

Sateen lähihetki-ennustaminen viittaa sen ennustamiseen ajanjaksolla joka ulottuu minuuteista muutamaan tuntiin. Tähän yleensä lähestytään käyttäen malleja, joille syötetään ajantasaisia säätutkien heijastuvuusmittauksia, jotka korreloivat havaitun sateen kanssa. Luotettavien lähihetken sateen todennäköisyys ennusteiden aikaansaaminen korkealla paikkaresoluutiolla on yhteiskunnallisesti tärkeä haaste. Tämä johtuu etenkin rankkasateiden ja niitä seuraavien äkkitulvien vaarasta ja taloudellisista vahingoista.

Tässä työssä aiemmin sademäärien piste-ennusteisiin käytetty Konvolutiivinen Neuroverkko (CNN) muunnetaan Bayesilaiseksi Neuroverkoksi (BNN) tuottaakseen parven eri ennuste-skenaarioista, jolla estimoidaan sadetodennäköisyyksiä. BNN:n toiminta perustuu verkon parametrien mallintamiseen todennäköisyysjakaumina, jotka estimoidaan käyttäen Bayesilaista päättelyä. Tässä työssä käytetään normaalijakaumia, jonka parametrit opitaan Stokastinen variationaalinen menetelmän (SVI) avulla. Ennuste parvet muodostetaan tällöin koulutetun mallin parametrien Monte Carlo otannalla, täten indusoiden ennustejakauman.

Kehitetyn mallin (lyhennettynä BCNN) suorituskykyä ennustus tehtävään arvioitiin ja verrattiin useisiin vakiintuneisiin malleihin, tämä käyttäen useita eri kriteerejä. Joissakin näistä BCNN ylsi samoihin tai lähes yhtä hyviin suorituskykyihin kuin vakiintuneet mallit. Kuitenkin, luotettavien ja hyvin-kalibroituneiden todennäköisyys-ennusteiden aikaansaaminen osoittautui haastavaksi kehitetylle mallille, luultavasti johtuen mallien ominaisuudesta tai koulutus-proseduurista. Näistä ongelmista huolimatta BCNN tarjoaa ensi-askeleet rakentaen pohjan tuleville ja kehittyneimille yrityksille rakentaa sateen probabilististisia lähihetki-ennusteita käyttäen Bayesilaisia Neuroverkkoja.

Asiasanat:	Sateen lähihetki-ennustaminen, Bayesilaiset Neuroverkot, Konvolutiiviset Neuroverkot, probabilistiset mallit
Kieli:	Englanti

Acknowledgements

First and foremost, I would like to thank my beloved wife Yukino for her continued support through the process of writing this thesis. As without her I wouldn't have been able to finish it in time. She always believed in me, even in my biggest moments of doubt and helped me get back on my feet, in order to continue the endeavor.

Next up, I owe enormous thanks to both my supervisor Arno and my advisors Terhi and Seppo for continuously offering precious advice and being willing to sacrifice much of their summer vacation to accommodate my graduation schedule through the month of July. During the recent months, they have also tremendously helped me improve my expertise both in precipitation nowcasting and Deep Learning.

Lastly, all of my colleagues at the Finnish Meteorological Institute deserve gratitude for the numerous fruitful discussions on subjects more or less related to the thesis and its topics, that have helped me shape my perspective further towards understanding.

Otaniemi, July 29th, 2022

Bent Ivan Oliver Harnist

Acronyms and Symbols

Acronyms

FMI	Finnish Meteorological Institute
NWP	Numerical Weather Prediction
Z	Reflectivity factor (mm^6/m^3)
dBZ	Decibel relative to Z
DL	Deep learning
NN	Neural Network
BNN	Bayesian Neural Network
ELBO	Evidence Lower Bound
CNN	Convolutional Neural Network
BCNN	Bayesian Convolutional Neural Network
SVI	Stochastic Variational Inference
BBB	Bayes-By-Backprop
NLL	Negative Log Likelihood
STEPS	Short-Term Ensemble Prediction System
LINDA	Lagrangian Integro-Difference equation model with Autoregression
CSI	Critical Success Index
ETS	Equitable Threat Score
FAR	False Alarm Rate
POD	Probability of Detection
MAE	Mean Absolute Error
ME	Mean Error
FSS	Fractions Skill Score
RAPSD	Radially-averaged Power Spectral Density
CRPS	Continuous Ranked Probability Score
ROC	Receiver Operating Characteristic
AUC	Area Under the Curve

Symbols

Possibly confusing symbols are listed here (non-exhaustively).

σ_q	Gaussian Posterior standard deviation
μ_q	Gaussian Posterior mean
σ_ℓ	Gaussian Negative Log Likelihood observational noise parameter
σ_1, σ_2	Gaussian Scale Mixture prior standard deviations one and two
σ_P	Gaussian Prior standard deviation
μ_P	Gaussian Prior mean

Contents

Chapter 1

Introduction

Nowcasting is defined as weather forecasting at a local scale up to six hours, according the World Meteorological Organization definition of 2010 [57]. The ability to provide an accurate precipitation nowcast has grown during this century into a meteorologically and societally significant challenge. This significance emanates from the fact that early warnings of severe precipitation enable authorities and other actors to make early decisions enabling for example disaster damage control, traffic safety, and in the case of heavy rainfall flash flood prevention, as well as the mitigation of economic loss incurred by them. High-density urbanization has exacerbated these issues, making it evermore vital to discover reliable and skillful methods for precipitation nowcasting.

Numerical weather prediction (NWP) solves the partial differential equations governing the physical processes of weather and climate to produce forecasts. NWP has seen great improvements over decades, up to the point where it can be used to produce accurate forecasts for up to a week in some cases [9]. However, such method is not applicable to predicting at timescales as short as 0 to 6 hours. This is mainly due to imperfect initialization making simulations unable to reach numerical stability in such short time timescales. Other problems with NWP include its computational cost and generally having insufficient spatiotemporal resolution to predict convective events and heavy localized rainfall [59].

To compensate for the shortcomings of NWP in the realm of precipitation nowcasting, many different dedicated models and algorithms have been developed [44]. Contrary to NWP, these models do not combine data from multiple sources in making predictions, but use only simple input images, and deterministically output their predicted evolution. In practice, many of those systems are based on the estimation of the precipitation advection field from *radar echo image sequences* and the further extrapolation of these

sequences along the advection field [44, 50]. Other methods have been developed that track and try to nowcast the evolution of individual rain cells instead of the whole grid [18, 44].

Dedicated precipitation nowcasting methods have had great success in forecasting the immediate future, but their performance typically degrades quickly, becoming unreliable often in the range of an hour. This is related to the fact that basic extrapolation-based methods have limitations such as failing to capture nonlinear patterns like growth and decay of precipitation, as well as convective cell initiation and their life-cycle.

A lot of work has been done in nowcasting in trying to incorporate modeling of higher-order and multi-scale phenomena into advection-extrapolation based models. Recently, Deep Learning (DL) based precipitation nowcasting approaches have started showing promising results delving into the issues of traditional methods thanks to the high volume of training data available, the increase in computational power, the expressivity of those models and their relative cheapness of inference [8, 62, 63]. The common denominator of all these models is that they are based on Convolutional Neural Networks (CNN), which is a class of DL model based on the inductive bias that in image tasks, it is often sufficient to learn patterns across a small neighborhood around a pixel rather than the entire image, which is often true in precipitation nowcasting.

1.1 Problem Statement

While the preceding introduction has cast the problem of nowcasting as finding a single optimal point estimate for future precipitation, it is useful to think about it in a probabilistic way. Because in nowcasting the most interesting phenomena are extreme events, which are those requiring preparation and early warnings, it makes intuitive sense that accurate and reliable probabilistic nowcasts are primordial for operational decision-making in meteorological crises. Also, since smaller spatial scales are less predictable, Adding a degree of uncertainty improves the model usefulness at those scales [24].

The interest in probabilistic precipitation nowcasts has grown lately and many probabilistic models have been introduced in the last decades [6, 21, 58, 61], notably the Short-Term Ensemble Prediction System (STEPS) [13] and more recently the Lagrangian Integro-Difference equation model with Autoregression (LINDA) [47]. These methods produce ensembles, that are used to estimate underlying distributions of data and precipitation probabilities. They perform reasonably well, but suffer from the same limitations as other extrapolation based methods, namely a limited ability to predict

nonlinear patterns of growth and decay.

So far, only little work has been done on using DL to produce probabilistic precipitation nowcasts, with the biggest breakthrough perhaps being Ravuri et al. [49] using adversarially trained deep generative models to produce ensemble nowcasts. There exists many possible ways of making probabilistic nowcasts using deep learning, with most of them focusing on directly modeling probability distributions of data. Another approach, that will be focused on from now on is instead to model the uncertainty of model parameters rather than that of the data, as would be done with STEPS or LINDA. This is done by using Bayesian Neural Networks (BNN) and has an advantage of providing implicit regularization of model parameters thus reducing overfitting, while outputting an ensemble of predictions whose variability in part reflects network parameter uncertainty given the training data. Bayesian neural networks have been proposed for use in other risk-averse application such as biomedical image segmentation [32] and autonomous vehicles [38]. This work is the first attempt to apply such a method to precipitation nowcasting.

In this work, the problem of making skillful probabilistic precipitation nowcasts is approached by building an uncertainty-aware Neural Network. The first goal is to formulate a Bayesian Convolutional Neural Network (BCNN) model for precipitation nowcasting by turning a baseline CNN into a BNN with optimization over parameters performed using stochastic variational inference (SVI). Secondly, the model is selected and trained using Finnish Meteorological Institute (FMI) radar reflectivity composite images. Finally, various metrics are calculated for BCNN nowcasts in order to assess different aspects of their probabilistic and deterministic predictive skill against baseline models.

1.2 Structure of the Thesis

The present work is organized as follows. Chapter 2 contains background and a literature review on precipitation nowcasting and Bayesian deep learning, aiming to familiarize the reader with essential concepts regarding the subject. Chapter 3 describes the experimental details of the work performed, including the datasets used, the models implemented, as well as verification methods and baselines. Chapter 4 presents the results of the experiments performed, including nowcasting examples of meteorologically interesting events, and calculated metric values compared to the baseline models. Chapter 5 discusses these results, their impact, and their validity in details. Finally, Chapter 6 closes the thesis by summarizing the most important findings and takeaways.

Chapter 2

Background

This background chapter aims to familiarize the reader with essential concepts necessary to understand the subsequent experiments and their results. First, in chapter 2.1, a brief introduction on where NWP stands in solving the problem at hand, as well as on weather radars and their use in nowcasting is given. Next classical deterministic and probabilistic methods for nowcasting are presented, as well as basic concepts regarding uncertainty qualification in the nowcasting context. In Section 2.2, an introduction to Neural Networks and their use in nowcasting is given. Finally, in Section 2.3 by a basic explanation is given on Bayesian inference and Bayesian Deep Learning with Stochastic variational inference.

2.1 Precipitation Nowcasting

Numerical weather prediction (NWP) is nowadays the main driver of operational weather prediction worldwide [9, 59]. On a very coarse level, NWP works by aggregating meteorological observations from sensors, which are used as initial state in solving atmospheric equations. These data can come from *in situ*, i.e. ground-based sensors such as rain gauges and thermometers, or upper atmosphere measurements such as those from radio sondes or aircraft sensors. Additionally and most importantly, remote-sensing sources such as satellites and weather radars provide lots of observational data, and their efficient use is one of the reasons for the improvements of NWP. Then, data assimilation is performed with that data, filling in gaps and agglomerating different sources to be consistent with each others, making the output of this process appropriate to be fed as input into a numerical simulation of the atmosphere. These simulations then consist of solving Navier-Stokes equations multiple timesteps into the future. With the model output in hand,

different post-processing can be applied to answer various questions about the future state of weather, such as making precipitation forecasts.

NWP can be performed on multiple spatio-temporal scales. Spatio-temporally coarse models cover a wide, sometimes global area and usually make simulations with longer *leadtimes*, meaning at further timesteps in the future. On the other hand, other models are specialized in providing finer spatiotemporal details with a smaller spatial extent. One example of this type of model is the High-Resolution Rapid Refresh (HRRR) model [5], developed by the United States National Oceanic and Atmospheric Administration (NOAA). HRRR covers the continental united states, and has 3 kilometer spatial and 1 hour temporal resolution. Although very useful for many purposes, even such fine-detailed NWP models have not quite enough details to be most useful in very short term and very localized precipitation nowcasting [48] as described in Chapter 1. As such, weather radars and nowcasting techniques based on them shall be now described.

Weather Radars and Rain Rate Estimation

Precipitation nowcasting is largely based on the data collected from weather radars. There are many reasons for this, the main one being that radars are capable of directly observing precipitation particles (called hydrometeors) over a significant range with a high update rate and resolution [57]. This is something that no single other ground, upper-atmosphere, or radar observation is capable of providing, especially on the mesocscale ranging from tens to hundreds of kilometers. Combining many radars into a network can further increase sensing capacity to areas covering countries such as NEXRAD covering the continental united states [2], or entire continents such as the OPERA network covering most of Europe [55].

Radars were invented as a method to perform effective military surveillance during the second world war. Their potential use as weather observation tools was discovered coincidentally when it was realized that some echoes of unknown origin in these surveillance radars were indeed precipitation. After the war, weather radars started to be routinely used in affluent parts of the world to provide precipitation observations and warnings.[19]

The functioning of weather radars is based on emitting a very strong, directed and polarized short electromagnetic pulse, which will interact with objects and particles in the atmosphere. This interaction is scattering, which reflects back a tiny fraction of the electromagnetic energy emitted at first. A receiver then listens to these returning signals, that are often called *radar echoes*. A radar typically scans the atmosphere radially at multiple azimuthal elevation angles. In addition to that angle, the vertical path of the beams,

and consequently the vertical position of objects detected, are affected by the curvature of the earth and the refractive index of the atmosphere decreasing with height, bending the beam downwards. [19]

The lowest elevation angle scan is the one giving the most accurate information about precipitation actually hitting the ground, with echoes further away being less reliable, as they come from targets upper in the sky. Higher elevation angle scans or products derived from a multitude of those can be used to further inform about the dynamics and development of precipitation. Combining scans and derived products from multiple radars and elevation angles into a single image makes what is called a *radar composite*.

One of the most important features of weather radars is the wavelength of the signal emitted by the radar. Longer wavelengths are less attenuated going through the atmosphere, but can only be used to detect larger particles. Equipment using longer wavelengths is known to be more expensive than that using shorter wavelengths. Oppositely, shorter wavelengths detect smaller particles but their operational range is limited because of their higher attenuation. Weather radars are classified according to their wavelength using a band denomination. The three most common classes are S-band, C-band, and X-band radars, in the order of longest to shortest wavelength.

In addition to meteorological echoes, various non-meteorological sources are picked up by weather radars, such as solid obstacles, birds migrating, and insects. These result in specific patterns visible on radar images, the nature of which depends on the source. It is consequently important to filter out or at least acknowledge those sources when preparing data for precipitation nowcasting. [19]

Reflectivity ($\mathrm{mm}^6/\mathrm{m}^3$) symbolized Z, is the quantity describing the amount of signal reflected from hydrometeors in the atmosphere back to the radar receiver. It is usually described in units of decibel relative to Z, denoted dBZ. Standard weather radar beams are horizontally polarized, which produces the signals of interest when looking at radar scans for precipitation. One recent development of weather radars is the introduction of dual-polarization, meaning that in addition to the horizontal one, there are scans performed with vertically polarized beams. This enables estimation of precipitation type, drop shape, and helps with separating non-meteorological echoes. Another important improvement to weather radars is the addition of Doppler capacity, enabling estimation of target velocities from radar scans. Variables using information on both polarizations are called polarimetric variables.

Converting the knowledge of (horizontal) reflectivity to that of rainrate is a highly non-trivial task and relies on approximations, because while water content is proportional to the third power of water droplet diameter, reflectivity of a single drop is proportional to its sixth power. Hence, total

reflectivity depends on the drop size distribution, which depends on climatology and precipitation types. Given appropriate knowledge, the relationship between reflectivity and rainrate, commonly known as the Z-R relationship, is defined as

$$Z = AR^b, \tag{2.1}$$

where Z is reflectivity (mm^6/m^3), R is rainrate (mm/h), and A as well as b are empirically determined parameters. The most famous and widely used A and b parameters come from a study on drop size distribution by Marshall and Palmer [36] in 1948, where $A = 200$ and $b = 1.6$.

2.1.1 Classical Deterministic Methods

When considering the atmosphere as a fluid, it is possible to apply tools and concepts of fluid dynamics to understanding processes happening in it, such as precipitation. In particular, one specific point of view is to consider precipitation as particles or matter moving through the forces exerted on it by the background atmospheric flow, here modeled as a liquid. This movement exerted by the surrounding medium is called *advection* and the medium dynamic itself the *advection field*. In practice, the flow consists of winds and other forces exerted on clouds and hydrometeors. Advection is described by the advection equation

$$\frac{\partial \psi}{\partial t} + \nabla \cdot (\psi \boldsymbol{v}) = 0, \tag{2.2}$$

where ψ denotes the precipitation field and $\boldsymbol{v}$ the advection field. With this in mind, the advection of precipitation can be observed from two distinct points of view, corresponding two different sets of coordinates. These are *Eulerian* and *Lagrangian* coordinates. The simpler one is Eulerian coordinates, which are those of the grid or space in which precipitation evolves. Lagrangian coordinates on the other hand center around particles or discrete unit being advected, so that each stays at the origin in its own set of Lagrangian coordinates, while the surroundings evolve relative to it.

A precipitation field where the only time evolution happening is the precipitation being advected has a common Lagrangian coordinate system for all precipitation units. This is never truly the case, but the idea is useful for developing nowcasting models. This concept can be formalized as that of *Lagrangian persistence*, meaning that in Lagrangian coordinates, the field remains constant. This is easy to understand by connecting the concept to its more natural counterpart of Eulerian persistence, where the precipitation

field just remains constant from a motionless observer's point of view. Lagrangian persistence allows separating the time evolution of the precipitation field into an advection term and a source term, representing the divergence of the field, that is creation and weakening of precipitation. Assuming no divergence in the advection field, and using Eq. (2.2) this is formalized as

$$\frac{\partial \psi}{\partial t} + \nabla \psi \cdot \boldsymbol{v} = S, \tag{2.3}$$

where S denotes the source/sink term. Basic precipitation field extrapolation based methods usually focus on modeling the advection part as accurately as possible, while methods willing to go a step further try and model the source, representing nonlinear evolution of the field.

Extrapolation Based Models

The most basic form of modern model of the first category is pure extrapolation along an estimated advection field. Very often nowadays the advection field is estimated using an optical flow method, such as Lucas-Kanade optical flow [34] on successive radar frames and the precipitation field is extrapolated along that advection field. This is usually done using a Semi-Lagrangian Extrapolation scheme [66], which is a cheap way of performing numerical integration for the advection problem by breaking it into multiple interpolation operations.

Other models of the first category attempt to model the time evolution of the advection field and add some sort of diffusion term to better represent the loss of predictability being faster at smaller scales [24]. One such model is developed by Ryu et al. [53] in 2020. This model models the evolution of the advection field using Burgers' equation, containing an advection and diffusion term, and additionally adds a facultative diffusion term to the nowcasted field. Another example of a similar modification is the 2013 work of Sakaino [54] where advection field temporal evolution was modeled using the Navier-Stokes with a continuity equation, and an anisotropic diffusion term, better conserving important details, was applied to the field.

Classical techniques such as Tracking Radar Echoes through Correlation (TREC) [50] still are in great use. TREC simply calculates correlations between neighboring areas in successive frames and extrapolates the precipitation field along directions of maximum correlation. This technique is simple but suffers from problems in maintaining the spatial structure of echoes over longer leadtimes. Over the years, there has been an accumulations of models proposing improvements to TREC. There exists also many storm cell based nowcasting techniques such as TITAN by Dixon and Wiener [18], introduced

in 1993. These techniques are often also based on extrapolation and many times enable tracking and forecast of cell life-cycle features, that are particularly useful in the context of convective storms.

Modeling Non-linear Evolution of Precipitation Fields

One of the early successful (in the sense that it presents improvement over advection-based extrapolation) attempts at modeling the time-evolution of precipitation fields is the Spectral Prognosis (S-PROG) model by Seed [60] from 2003. The model is based on the observation that precipitation feature lifetimes depend on their scale, with smaller features lasting shorter than bigger ones. This information is used by decomposing the precipitation field into a cascade of fields, each corresponding to features of a certain spatial scale. Each of those cascade level time evolution is then modeled separately using autoregressive (AR) models in Lagrangian coordinates.

A second recent model taking a similar approach is ANVIL by Pulkkinen et al. [46] from 2020. Instead of single reflectivity scans or reflectivity derived products, ANVIL utilizes Vertically Integrated Liquid (VIL), an estimate of the total precipitation mass in a cloud, as a proxy to precipitation hitting the ground. ANVIL decomposes the field into a scale cascade just like S-PROG, but uses an autoregressive integrated process (ARI) to model the time-evolution at each level. ARI are autoregressive processes applied to the time derivative of the fields. This helps in avoiding the loss of small-scale precipitation features in the process. ANVIL was shown to surpass S-PROG for the prediction of intense precipitation (> 5 mm/h and > 20 mm/h) using multiple verification metrics.

There is also examples of using phased-array weather radars to do three-dimensional extrapolation nowcasting, allowing to capture some physical processes such as linear patterns of growth and decay. One example of this is the 2016 work of Otsuka et al. [41].

2.1.2 Classical Probabilistic Methods

Probabilistic nowcasting models can be roughly divided into two categories: Quantile-based and ensemble-based methods, the latter of which being the focus of this work. In ensemble-based methods, multiple stochastic nowcasts are generated, the set of them being called an ensemble, and probabilistic features such as rainrate exceedance probabilities are estimated from the data distribution of that ensemble. There in addition exists neighborhood methods, in which new precipitation values are drawn from neighbors at

random to produce the probabilistic nowcast. The first of such methods having been developed is that of Andersson and Ivarsson [6] in 1991.

The most influential of modern probabilistic nowcast methods is certainly STEPS by Bowler et al. [13], dating from 2006. It is an ensemble method that is close to S-PROG in that it divides the precipitation field into a scale cascade and models their evolution independently through AR models. Uncertainty in STEPS is modeled through the injection of stochastic noise per ensemble member at different scales. One interesting feature of STEPS is that it allows blending an NWP forecast to radar images to create a more reliable nowcast at longer leadtimes. STEPS is shown to retain some degree of prediction skillfulness up until leadtimes of six hours. [13]

Another more recent ensemble nowcasting model is Lagrangian Integro-Difference equation model with Autoregression (LINDA) by Pulkkinen et al. [47] in 2021. LINDA is designed specifically to accurately nowcast heavy localized rainfall and comes in two variants: LINDA-D for deterministic nowcasts, and LINDA-P for probabilistic nowcasts. LINDA is composed of multiple steps that are the identification of rain cells, advection, AR modeling for the growth and decay of features, convolutions described loss of detail at small scale, and finally stochastic perturbations in the case of LINDA-P. LINDA-D is shown to surpass both S-PROG and ANVIL in terms of skill at > 5 mm/h and > 20 mm/h. LINDA-P on the other hand is shown to produce more reliable and discriminative, as well as better calibrated forecasts than STEPS at high thresholds such as those described above. The forecasts are also less blurry than those of S-PROG or STEPS. In essence, LINDA describes the state-of-the-art in terms of nowcasting high-intensity localized rain.

2.1.3 Predictive Uncertainty Estimation and Decomposition

Predictive uncertainty is defined as the total uncertainty arising in practice when making predictions with a probabilistic or ensemble based model. Broadly speaking, in statistical modeling and also Deep Learning, predictive uncertainty can be divided into two main categories. These are *aleatoric uncertainty*, which originates from the input data, and *epistemic uncertainty*, which originates in the inaccuracy of the model. Epistemic uncertainty can be reduced with more training or other changes, while aleatoric uncertainty can not be by definition. Aleatoric uncertainty is further classified into homoscedastic and heteroscedastic aleatoric uncertainties. The former assumes that all observation share the same underlying uncertainty, while the latter

allows the uncertainty of each observation to differ [64]. Characterization of uncertainty is important because not taking it into account might lead to over- or under-estimation of the failure probability of a model, as explained by Kiureghian and Ditlevsen [31].

Uncertainty in Radar-based Nowcasting

In radar-based nowcasting, the aleatoric uncertainty of models can be divided into that emanating from measurements and that coming from other factors induced by the processing of data inside of the nowcasting process. Measurement uncertainty can again be subdivided into sampling-based and system-based uncertainties. Sampling uncertainty refers to the randomness coming from which hydrometeor or obstacle the radar beam encountered in the scanning process, and system-related ones come from various factors in the radar itself. According to Cao et al. [15], sampling-based uncertainties dominate in well-configured radar systems. Other factors inducing aleatoric uncertainty include all stages where information is possibly lost, such as data compression operations. Additionally, for methods where model outputs are iteratively fed back into the algorithm to produce new predictions, the whole compound uncertainty of the previous step output is accounted as aleatoric uncertainty in the next step.

Prediction error and uncertainty are related to each others but not equivalent. Sources of errors can be used to partly infer sources of uncertainty, keeping in mind that model outputs may have high bias, i.e. systematic error, but low variance, i.e. uncertainty. Bowler et al. [13] enumerates sources of prediction errors in the context of advection based models, dividing them into three categories. These are errors in estimating the initial advection field, in modeling its time evolution and the Lagrangian evolution of features. Part of these errors are related to the model itself, and as epistemic uncertainty can by definition be reduced, these sources of error can be in decreased in practice. This can be achieved by e.g. improving the quantity and quality of data with regards to the task at hand, improving the functional model to better express dependencies between inputs and outputs, and improving the model optimization procedure and assumptions made to better fit the problem.

2.2 Deep Learning for Nowcasting

Artificial Neural Networks, abbreviated as Neural Network or NN in a machine learning context, are computational systems inspired by biological neu-

ral networks, such as those present in the human brain. In this work, the terms Deep Learning is used as meaning "regarding neural networks and their usage". Neural networks serve to accomplish many tasks, usually of the domain of artificial intelligence, such as regression, classification, or agent decision making. NN consist of discrete units called neurons linked to each others, usually arranged in functional units known as layers, in which case connections are formed between layers. Input data is fed into the NN to an input layer, transforming the input through learnable parameters, and this transformed signal is propagated through other layers further transforming the signal. The organization of layers and the way they are connected is known as the *network architecture*. Finally, signals converge to an output layer giving the product of the network, such as a prediction or class of input data. Between neurons and layers there are non-linear *activation functions*, which are in essence non-linear transformation on data, which are the basic mechanism that enables neural networks to learn complex nonlinear patterns.

Neural Network Training

In Neural Networks, parameters are usually not tuned by hand due to the sheer complexity of the task. Instead, finding optimal parameters for performing the task is framed as an optimization problem. This is accomplished using standard unconstrained optimization tools and the Backpropagation algorithm, allowing the learning of the parameters, despite of the problem being very high-dimensional and non-convex. This algorithm is based on producing an output by passing input data through the network, and then calculating a metric of how well the network succeeded at the task, known as a *loss function* or cost function. The gradient of the output related to network parameters are then calculated backwards starting from the loss function, flowing through the network using the chain rule of derivation. These gradients are then used to update model parameters using simple gradient descent or its variations.

Usually, the input data is divided input small subsets called *mini-batches*, here abbreviated as batches (although a batch may refer to the whole data in literature), and the gradient updates are calculated by going iteratively through them in a process called *training*. Gradient descent over these mini-batches is called Stochastic Gradient Descent. Going through all of the training data once is called an *epoch*, and training is usually continued for many epochs. In deep learning, available data is usually split into training, validation, and test data. Validation data is used for the validation process, in which performance of the NN is assessed on data unused for training. This information on performance can then be used to tune *hyperparameters*, that

is parameters that are constant and set before training, or perform other types decision-making related to the training process, such as early stopping of the training if performance is likely not to improve anymore [43]. The reason why training data is not used here is that the network usually performs better on data that it was trained on compared to other data, and continuing the training ad infinitum improves the training performance while worsening generalization ability (overfitting). Lastly, test data is used for independent assessment of the network performance. This is needed because even though validation data is not used for training, the network is still biased to perform better on it if decision that improve performance were made based on it. Validation data can thus not be trusted on as an independent evaluator of network performance.

Different Varieties of Neural Networks

There are multiple types of Neural Network architectures that are each good at different types of tasks. The main types with some relevance to nowcasting and Bayesian Deep Learning are feed-forward neural networks, Recurrent Neural Networks (RNN), and Convolutional Neural Networks (CNN). Feed-forward neural networks are the earliest type of NN, with linear transformations of data in layers of neurons, where neurons of successive layers are all connected to each others.

RNN are a modification of feed-forward neural networks with recurrent connections along units, forming a network along a temporal sequence. This makes them widely used for the forecasting of time-series data and natural-language processing. Important improvements over original RNN are Long-Short Term Memory (LSTM) networks, with units having internal states allowing learning longer-range temporal dependencies, and Gated Recurrent Unit (GRU) modifying LSTM units to make them more lightweight.

CNN have layers that are convolutional filters applied over the input data. CNN are great for use in computer vision and other image related tasks, as they take advantage of the fact that learning short-range dependencies is enough to perform well in many of those tasks, thus reducing the hypothesis space of learnable representations in an informed way. One CNN architecture of particular interest in the context of this work is U-Net, that has shown excellent results in e.g. semantic segmentation tasks, that are important for biomedical imaging and autonomous vehicle applications. U-Net follows a simple encoder-decoder architecture. The encoder consists of intertwined downsampling and convolution layers, while the decoder similarly consists of intertwined upsampling and convolution layers. The next section will describe current applications of artificial neural networks to precipitation

nowcasting.

2.2.1 Deep Learning Approaches to Nowcasting

There exists now many methods for producing deterministic nowcasts with machine learning. Most suffer the same problem of producing overly blurry nowcasts, i.e. losing high-resolution details, only after a few timesteps. While there has been marginal improvement with regards to this problem over the years, it remains one important challenge when it comes to producing deterministic nowcasts. The following chapters will present some of the main advancements in Deep Learning for nowcasting since its introduction.

Basic Approaches

The first dedicated Deep Learning model for precipitation nowcasting is ConvLSTM by Shi et al. [62], published in 2015 and building upon the work of Oh et al. [40]. ConvLSTM is a neural network combining the image feature encoding capacities of CNN and temporal prediction capacities of LSTM into a single network fusing the two. ConvLSTM was shown to outperform an optical flow based model at predicting precipitation exceeding a low rainrate of 0.5 mm/h. A further model inspired by ConvLSTM called TrajGRU is developed by Shi et al. [63] in 2017 by replacing the LSTM component with GRU and making recurrent connections dynamically location-variant. These connections improve the predictive skill over over invariant ones (ConvGRU), but the model was not tested against ConvLSTM.

Another class of models that have gained traction lately for nowcasting are simple U-Net based CNN, that adopts an image-to-image translation perspective on the problem. These networks are fed a sequence of radar images and output either one or several future frames. One example of this approach is that of Agrawal et al. [4]. In the case where only one frame is outputted, predictions for several leadtimes may be produced by iteratively feeding back predictions into the network as input. This is the approach taken with the RainNet model of Ayzel et al. [8]. It is more flexible but has the disadvantage of exacerbating the blurring problem. The *pure* CNN models in general also have the advantage of being somewhat computationally less expensive when compared to ConvLSTM variants.

Notable Recent Improvements

Recently, Pan et al. improved the nowcasting of convective evolution by adding polarimetric variables in addition to reflectivity scans as inputs to a

U-Net based model [42]. These polarimetric variables are derived from dual-polarization weather radars and their values are strongly associated with life-cycles of convective cells. This work showcases the importance of multi-channel data and not only relying on reflectivity if one wants to accurately model nonlinear evolution of precipitation.

The previously introduced NN-based models suffer badly the the problem of blurring stated earlier, as do all current models based on discriminative neural networks, which are networks mapping one input to one output. Prediction blurring is in part related to loss functions used, as losses like Mean Squared Error (MSE) and Mean Absolute Error (MAE) losses tend to act that way when minimized with some uncertainty present. This excessive blurring also hinders the prediction of useful patterns such as heavy localized rainfall. This phenomenon takes roots in the quick loss of predictability in smaller spatial scales, which has for effect of averaging them out over time. This is actually the same phenomenon as the blurring happening with the classical models containing multiscale autoregressive processes, such as S-PROG or STEPS. Multi-Scale Structural Similarity Index (MS-SSIM), originally an image quality assessment metric [70], has recently started to be used as a loss function in deep learning, particularly in image reconstruction tasks. One of its main assets is that its single-scale counterpart (SSIM) has been shown to reduce blurring in image reconstruction [73] making it a good candidate loss function for nowcasting with neural networks. Indeed in recent years, there has been some cases where MS-SSIM or SSIM started have started to be used in Deep learning based nowcasting models, such as the one of Yin et al. [72] in 2021. Here both SSIM and MS-SSIM produced results surpassing MSE, with MS-SSIM giving the best results of the two. Most importantly, these loss functions seemed to better preserve high rainrates, which tended to vanish with traditional losses at higher leadtimes.

Perhaps one of the biggest breakthroughs made yet, is the use of Generative Adversarial Networks (GAN) for precipitation nowcasting by Ravuri et al. [49] last year. This approach of using a generative model, i.e. one that generates samples from a probability distribution conditioned on past radar measurements, manages to solve the problem of nowcast blurring. GAN are a class of networks based on the competition between a generator network that generates the samples and discriminator(s) networks trying to discern whether these generated samples are real or fake. The generator then tries to fool the discriminators in a zero-sum game. Ravuri et al. use a generator network based on a convGRU architecture, nowcasting 90 minutes at once, two discriminators, and a regularization term penalizing deviations of generated samples at the grid level. The first discriminator ensures spatial consistency and the lack of blurring, while the second one ensures tempo-

ral consistency in generated sequences. The resulting generative model has slightly superior skill compared to existing approaches, and importantly preserves high-resolution features while producing useful probabilistic nowcasts. [49]

In addition to this, there has so far been limited work in probabilistic nowcasting with Deep Learning. One example is a probabilistic nowcasting model called MetNet that was developed by Sønderby et al. [67]. The model is based on Axial Self-Attention by Ho et al. [27] and is capable of outperforming HRRR on probabilistic verification metrics for leadtimes of up to 8 hours.

2.3 Bayesian Deep Learning

Neural networks are extremely useful model that on the other hand are very complex, in the sense that they have many optimizable parameters. The effect of this is that they are sensible to *overfitting*, meaning that if left unchecked they will learn spurious patterns in the data, over-adapting to the training dataset and worsening generalization ability. In order to counter this, different mechanisms exist that bias the neural network towards learning simpler representations that have better generalization ability when presented with out-of-training-data examples. This concept is known as *regularization*. Over the past decades, many regularization mechanisms have been successfully developed and applied to the training of deep neural networks. Notable examples include Dropout and weight decay, also known as L_2-regularization. [11, 65]

One caveat of these classical regularization methods is that they do not enable representing the uncertainty of neural networks, which is particularly interesting in unexplored regions, although they do improve model performance in them. As it has understandably many benefits to make a neural network able to say "I don't know", a way to represent different plausible scenarios arising with novel data is needed.

There are two ways of approaching uncertainty estimation in neural networks. One is to directly model the uncertainty of predictions, and the other is to model the uncertainty of model parameters. These two are often complementary, but The latter approach has the benefit of providing implicit regularization to the network and is indeed the primary focus of this work. Learning this uncertainty of model parameters is most often accomplished using Bayesian Neural Networks (BNN). Strictly speaking, BNN are a class of stochastic neural networks, that is NN with stochastic components, where the parameters are probability distributions that are estimated using Bayesian inference. Bayesian Neural Networks in their current form were introduced

by MacKay [35] in 1992, after early works starting in 1987 by Denker et al. [17], Tishby et al. [68], Denker and LeCun [16], and Buntine [14].

Bayesian inference is one of the two paradigms for statistical inference along with frequentist inference. Statistical inference is the process of using data in order to infer properties of the underlying statistical distribution. The defining feature of Bayesian inference is that it bases itself on Bayes' rule defined as

$$P(H \mid E) = \frac{P(H)P(E \mid H)}{P(E)}, \tag{2.4}$$

where H is the statistical hypothesis, E is the evidence or data, and P refers to a probability distribution. Bayes' rule serves to update the hypothesis given previous knowledge and new data. The result of the update $P(H \mid E)$ is called the *posterior*, meaning the updated probability of the hypothesis conditioned on the new evidence. $P(H)$ refers on the other hand to probability of the hypothesis a priori, i.e. before observing the new evidence, which is why it is called the *prior*. $P(E \mid H)$ is the *likelihood*, that is the probability of observing the evidence conditioned on the existing hypothesis. Finally, $P(E)$ is the probability of observing the evidence regardless of the hypothesis, i.e. integrating, or in other words marginalizing over all possible hypotheses, which is why it is called the *marginal likelihood*.

In BNN, posterior distributions of parameters are estimated conditioned on data and a prior distribution over parameters, which is chosen beforehand. The hypothesis takes as such the form of distributions over weights, and the evidence that of the training data provided to the Neural Network. Because exact Bayesian inference involves dealing with intractable integrals, specifically regarding the computation of the marginal likelihood, it is not doable to solve them for real-world neural networks having thousands to millions of parameters. As such, two broad categories of inference methods have been developed for use in BNN. The first one is using Markov-Chain Monte Carlo (MCMC) to sample from posterior distributions. MCMC works well for smaller-scale models, but it is limited to only thousands of parameters because of the computational cost of Markov-Chain simulations scaling up with the number of parameters.

The second inference method category is *Variational Inference* (VI). The main idea behind variational inference is to turn the problem of finding the intractable true Bayesian posterior into that of finding the closest distribution from a limited distribution family (the variational family) with regards to the true posterior. This turns the problem of solving an integral into that of optimization, allowing the use of classical unconstrained optimization methods.

Dropout was surprisingly shown to be an instance of variational inference with a Bernoulli distributed posterior by Gal and Ghahramani [23] in 2016. This legitimizes the use of Monte Carlo dropout, a technique used to get predictive uncertainty estimates from networks with dropout layers, by simply not switching off the layers for inference and thus drawing Monte Carlo samples of the data distribution. Although not as expressive as explicit Bayesian NN, MC dropout has no computational overhead and is simple to implement, making it a very popular alternative to BNN.

In the context of Bayesian Deep Learning, the underlying neural network architecture will be referred to as the functional architecture or the functional model. Other components such as the inference method, the prior, or the posterior modeling will be commonly referred to as the stochastic model.

2.3.1 Variational Inference

Variational inference as a means of training Bayesian Neural Networks was first introduced by Hinton and van Camp [26] in 1993. However it was not used for a long time before breakthroughs by Graves [25] in 2011 and Blundell et al. [12] in 2015 permitted its use in large-scale neural networks. The following sections will introduce the loss function and the optimization algorithm used, mostly based on the work of Blundell et al. [12].

Evidence Lower Bound Loss Function

Finding the variational posterior $q(\boldsymbol{w} \mid \theta)$ best approximating the true posterior is formalized as minimizing the Kullback-Leiber (KL)-divergence between the two distributions as

$$\theta^* = \arg\min_{\theta} \mathrm{KL}\left[q(\boldsymbol{w} \mid \theta) \mid\mid P(\boldsymbol{w} \mid \mathcal{D})\right], \tag{2.5}$$

where θ refers to variational posterior parameters,$\boldsymbol{w}$ to the weights, $\mathcal{D}$ to the data, q to the variational distribution, and P to the true distribution. The KL-divergence is defined as [11]

$$\mathrm{KL}\left[q(\boldsymbol{w} \mid \theta)||P(\boldsymbol{w} \mid \mathcal{D})\right] = \int_{\Theta} q(\boldsymbol{w} \mid \theta) \log \frac{q(\boldsymbol{w} \mid \theta)}{P(\boldsymbol{w} \mid \mathcal{D})} \mathrm{d}\theta, \tag{2.6}$$

where Θ refers to the space spanned by θ parameters. Rewriting this definition as the expected value over $q(\boldsymbol{w} \mid \theta)$ using the chain rule of probability $P(\boldsymbol{w}, \mathcal{D}) = P(\boldsymbol{w} \mid \mathcal{D})P(\mathcal{D})$ and rearranging, the minimization objective can be turned into

$$\arg\min_{\theta} \mathbb{E}_{q(\boldsymbol{w}|\theta)}[\log q(\boldsymbol{w} \mid \theta)] - \mathbb{E}_{q(\boldsymbol{w}|\theta)}[P(\boldsymbol{w}, \mathcal{D})] + \log P(\mathcal{D}), \tag{2.7}$$

where $\mathbb{E}_{q(\boldsymbol{w}|\theta)}$ denotes the expected value of the following expression over $q(\boldsymbol{w} \mid \theta)$. Here the evidence $P(\mathcal{D})$ is very difficult to estimate. Luckily though, the minimization objective does not depend on it, so a new objective called the evidence lower bound (ELBO) or variational free energy is used instead, which is defined as

$$\text{ELBO}(\mathcal{D}, \theta) = -\left[\text{KL}\left[q(\boldsymbol{w} \mid \theta) \mid\mid P(\boldsymbol{w} \mid \mathcal{D})\right] - \log P(\mathcal{D})\right]. \tag{2.8}$$

This is a maximization objective equivalent to substracting $\log P(\mathcal{D})$ from Eq. (2.5). Solving for the evidence $\log P(\mathcal{D})$, it is clear that the ELBO is a veritable lower bound for the it, as KL-divergences can not be negative.

Reopening up Eq. (2.8) in the same way as for getting Eq. (2.7), and using $P(\boldsymbol{w} \mid \mathcal{D})P(\mathcal{D}) = P(\mathcal{D} \mid \boldsymbol{w})P(\boldsymbol{w})$, The ELBO can be rewritten as

$$\mathcal{F}(\mathcal{D}, \theta) = \mathbb{E}_{q(\boldsymbol{w}|\theta)}\left[\log q(\boldsymbol{w} \mid \rho) - \log P(\boldsymbol{w}) - \log P(\mathcal{D} \mid \boldsymbol{w})\right] \tag{2.9}$$

which is a form more suitable for the development of an optimization method. Here and from now on in this section, The ELBO will be referred to with the $\mathcal{F}$ symbol. For the rest of this work, the variational posterior family will be assumed to be that of Gaussian distributions, as the *Bayes-by-Backprop* (BBB) algorithm described in the following section works on this posterior family.

The Bayes-by-Backprop Algorithm

In order to apply backpropagation to Bayesian Neural Networks with variational inference, any sampling operation must be made independent of the network in order to allow gradients to flow backwards [12]. This is accomplished by reparametrizing the positive standard deviation σ_q of posterior distributions to a parameter $\rho \in \mathbb{R}$. This is accomplished with the transformation $\sigma_q = \log(1+\exp(\rho)) \in (0, \infty)$. This is known as the reparametrization trick. Weights $\boldsymbol{w} = t(\sigma_q, \epsilon)$ are made a deterministic function t of posterior parameters and an external stochastic noise parameter ϵ. Adding this parameter ϵ is what allows gradients to flow through the network by making sampling operation external to the network.

Using the the proposition

$$\frac{\partial}{\partial\theta}\mathbb{E}_{q(\boldsymbol{w}|\theta)}[f(\boldsymbol{w}, \theta)] = \mathbb{E}_{q(\epsilon)}\left[\frac{\partial f(\boldsymbol{w}, \theta)}{\partial \boldsymbol{w}} + \frac{\partial \boldsymbol{w}}{\partial \theta}\frac{\partial f(\boldsymbol{w}, \theta)}{\partial \boldsymbol{w}}\right] \tag{2.10}$$

from Blundell et al. [12], the derivative of an expectation can be turned into the expectation of a derivative. This allows to approximate the ELBO as

$$\mathcal{F}(\mathcal{D}, \theta) \approx \sum_{i=1}^{n} \log q(\boldsymbol{w}^{(i)} \mid \theta) - \log P(\boldsymbol{w}^{(i)}) - \log P(\mathcal{D} \mid \boldsymbol{w}^{(i)}), \tag{2.11}$$

which is an unbiased Monte Carlo estimator of the ELBO. Here $\boldsymbol{w}^{(i)}$ refers to the i:th Monte Carlo sample drawn from the posterior distribution. The proposition (2.10) follows from the reparametrization trick, as demonstrated by Blundell et al. [12].

From the terms in Eq. (2.11):

1. $\log q(\boldsymbol{w}^{(i)} \mid \theta)$ is the log likelihood of weights given the current posterior distribution.
2. $-\log P(\boldsymbol{w}^{(i)})$ is the negative log likelihood of the weights given the prior, which is usually not learned and serves as a regularization mechanism.
3. $-\log P(\mathcal{D} \mid \boldsymbol{w}^{(i)})$ is the likelihood term, dependent on the data $\mathcal{D}$

The first two terms are grouped together as the complexity cost, while the last term is often called the likelihood cost. With all of the conditions in place, optimization takes place in pretty much the same way as for basic backpropagation, only updating two variables instead of one (μ_q and ρ instead of point estimates). The pseudocode for one optimization step using the Bayes-By-Backprop algorithm is shown in Algorithm 1, where $\circ$ refers to element-wise multiplication. On line 1, sampling stochastic noise ϵ is performed in each layer. On line 2, weights are set using ϵ. On line 3, the data $\mathcal{D}$ goes through the network and the ELBO is calculated, then on lines 4 and 5, posterior mean and variance gradients are calculated through backpropagation. Finally on lines 6 and 7, the posterior parameters are updated with these gradients.

Complexity Cost Weighting and Priors

Assuming minibatch optimization, the ELBO from (2.9) for the i:th minibatch can be again rewritten as

$$\mathcal{F}_i^{\pi}(\mathcal{D}_i, \theta) = \pi_i \left[\log q(\boldsymbol{w} \mid \rho) - \log P(\boldsymbol{w})\right] - \log P(\mathcal{D}_i \mid \boldsymbol{w}), \tag{2.12}$$

here π_i being the relative weighting of the complexity cost and $\mathcal{D}_i$ the data in the i:th minibatch. There are many ways to weigh the complexity cost

Algorithm 1 Bayes-By-Backprop

1: Sample $\epsilon \sim \mathcal{N}(0, 1)$ ▷ Reparametrization
2: $\boldsymbol{w} \leftarrow \mu_q + \log(1 + \exp(\rho)) \circ \epsilon$, $\theta \leftarrow (\mu_q, \rho)$ ▷ Weights
3: $\mathcal{F}(\boldsymbol{w}, \theta) \leftarrow \log(q(\boldsymbol{w} \mid \rho)) - \log(P(\boldsymbol{w}) - \log P(\mathcal{D} \mid \boldsymbol{w}))$ ▷ Loss
4: $\Delta_\mu \leftarrow \frac{\partial \mathcal{F}(\boldsymbol{w},\theta)}{\partial \boldsymbol{w}} + \frac{\partial \mathcal{F}(\boldsymbol{w},\theta)}{\partial \mu_q}$ ▷ Gradient calculations
5: $\Delta_\rho \leftarrow \frac{\partial \mathcal{F}(\boldsymbol{w},\theta)}{\partial \boldsymbol{w}} \frac{\epsilon}{1+\exp(-\rho)} + \frac{\partial \mathcal{F}(\boldsymbol{w},\theta)}{\partial \rho}$
6: $\mu_q \leftarrow \mu_q - \alpha\Delta_\mu$ ▷ Posterior parameter updates
7: $\rho \leftarrow \rho - \alpha\Delta_\rho$

against the likelihood cost, but an usual constraint is that $\pi \in [0, 1]^M$ and $\sum_{i=1}^{M} \pi_i = 1$. Graves [25] used equal weighting as $\pi_i = 1/M$, but Blundell et al. [12] found the scheme $\pi_i = \frac{2^{M-i}}{2^M-1}$ to offer better performance in their experiments.

The prior distribution $P(\boldsymbol{w})$ is often chosen to be a diagonal Gaussian distribution, because it allows calculating the KL-divergence with regards to the Gaussian posterior analytically. A Gaussian prior placed on weights would be equivalent to L_2- regularization, also known as weight decay. Recently, despite of the inability to analytically calculate KL-divergences using it, Gaussian scale mixture priors have also been used [12, 64] because of their properties facilitating optimization. These scale mixture priors, if containing two scales are defined as

$$P(\boldsymbol{w}) = \prod_j \alpha\mathcal{N}(\boldsymbol{w}_j \mid 0, \sigma_1^2) + (1 - \alpha)\mathcal{N}(\boldsymbol{w}_j \mid 0, \sigma_2^2), \tag{2.13}$$

where $\sigma_1 > \sigma_2$ and often $\sigma_2 \ll 1$, α is the relative weights of the two scales, and $\boldsymbol{w}_j$ is the j:th weight of the neural network.

Alternative Reparametrizations

One problem with conventional Bayes-By-Backprop is that the same ϵ is used for each parameter in a layer. One side effect of this is that gradients of different weights are correlated, hindering the training.

Kingma et al. [30] introduced a modification to the reparametrization trick, called the *local reparametrization trick* (LRT). This modification works similarly but rather than weights, pre-activation layer outputs are sampled using ϵ, with a different ϵ for each output. This reduces variances in a mini-batch, allowing for faster training overall. Kingma et al. [30] showed that the unbiased Monte Carlo estimator of log likelihood in Blundell et al. [12] has variance that does not decrease with batch size because of the contribu-

tion of batch member covariances due to shared ϵ, and consequently designs an estimator with zero covariance, leading to the method above. Gradient variances with LRT thus are inversely proportional to batch size, leading to easier optimization with larger batch sizes.

Flipout by Wen et al. [71] is another modification to Bayes-by-Backprop, attempting to solve the above problem. While LRT can only be used for fully-connected feed-forward neural networks with no weight sharing, Flipout can be used on other architectures too [71]. Flipout works the same as BBB, except for the fact that it multiplies the sampled ϵ by a random sign matrix of the size of the parameter space. This way the ϵ are to some degree made pseudo-random, decreasing the gradient variances for a very meager addition to the computational cost.

2.3.2 Uncertainty Decomposition in Deep Learning

There exists techniques to decompose predictive uncertainty into aleatoric and epistemic uncertainties in deep learning which can be used with Bayesian Neural Networks. To model the epistemic component, these techniques base themselves on ensembles just like those produced by BNN, or the variability between ensemble members to be more precise. Strategies for modeling the aleatoric part differs between classification and regression tasks. In classification, heteroscedastic aleatoric uncertainty can be directly inferred from class logits without any network modification [32, 64].

For regression on the other hand, there is a need to encode the aleatoric uncertainty as it is not intrinsically present in network outputs. By modeling the likelihood as homoscedastic Gaussian likelihood, one can learn a common homoscedastic uncertainty term for the dataset. By modeling the data as heteroscedastic Gaussian likelihood on the other hand, one can learn a different uncertainty term for each data point. This is done by separating the output of the network and its aleatoric uncertainty into two different channels towards the end of the network, and plugging these in the Gaussian likelihood accordingly [29].

Chapter 3

Materials and Methods

From now on are presented the data and models used for experiments, and how those nowcasting models were verified. The training, validation and verification datasets are presented in Section 3.1. The baseline CNN (RainNet) model is described in Section 3.2.1, and the BCNN Bayesian extension as well as its implemented variants are described in Section 3.2.2. After these models are trained, predictions are calculated for 3 hours with the BCNN variants and all baselines models of Sections 3.3.1 and 3.3.2. After that, deterministic verification metrics described in Section 3.4.1 are computed for the predictions of deterministic models and probabilistic model prediction ensemble averages. Then, probabilistic metrics from Section 3.4.3 are calculated for the predictions of BCNN and probabilistic models of Section 3.3.2. The metrics are then averaged over the test set for each leadtime of interest and visualized. Additional details on the common training procedure of the DL models are described in Section 3.2.3, and additional details on prediction and metric calculations in Section 3.5. Lastly, the software and various resources used are presented in Section 3.6.

3.1 Dataset and Data Selection

As input data, lowest elevation angle radar reflectivity composites with original 1km spatial resolution and 5 minute temporal resolution from the Finnish Meteorological Institute, cropped into a 512x512 km region covering southern Finland are used. The inputs are in this work *downsampled once, lowering the image size to* 256×256 *pixels and the effective spatial resolution to 2km.* The domain and bounding box are illustrated in Figure 3.1. The radar network consists of C-band dual polarization radars.

Because lowest elevation angle horizontal reflectivity is a good estimator

of precipitation at ground level, it is a natural data choice for building a nowcasting model. Radar composites archived from these radar scans are readily available at FMI which facilitated their retrieval for this work. Although the bounding box covering southern Finland did have some data quality issues, it was still deemed to be the best choice, when compared to other candidate datasets such as TAASRAD19 [22], as the problems were minor and did not disturb the training process. For example, missing pixels were correctly labeled, which allowed trivial removal of composites with some included.

Some of the problems with the chosen data are related to insufficient data quality control. Specifically, polarimetric information was not used, making the removal of non-meteorological echos less accurate. Also this absence means that attenuation correction could not be performed, having for effect that echoes originating from behind other strong echoes are attenuated and are shown as weaker than they really are. This is partly mitigated by having multiple radars in the composite, but it does not cover all needed scenarios. Another problem with data quality is that even though summer months in southern Finland were chosen making the precipitation likely rainfall, it can not be said for sure as the phase of precipitation in the dataset was not checked. Again, this could have been checked with polarimetric information.

The dataset was chosen so that at first, a selection of rainy days were chosen as to correspond to the 100 days with the most pixels exceeding a 35 dBZ reflectivity threshold during the summer period spanning from May to September during years 2019, 2020, and 2021. The present threshold was chosen as a value which convective storms usually exceed during their whole lifetime [69], as the events that are of the most interest in the context of this work are such convective storms. Because the dataset used for this work is concentrated to summer months in southern Finland, all precipitation from the dataset is assumed to be rainfall from now on (which is mostly a valid assumption, except for exceptional hail), and the terms will thus be used interchangeably.

This dataset is then cleaned and filtered, after which it is divided into training, validation, and test sets. Cleaning the data involves first going through all timestamps and removing those with either partially or completely missing data. The filtering part consists of removing timestamps with less than 1% of pixels containing reflectivity values exceeding 20 dBZ. Splitting of data into training, validation, and testing sets is performed by using a block sampling strategy [59] with 6 hour long blocks to prevent autocorrelation between consecutive radar images from invalidating independence between splits. The final split sizes were 15840 radar images for the training split, 2664 for the validation split, and 2448 for the test split. The radar images are always downsampled one time using average-pooling before being

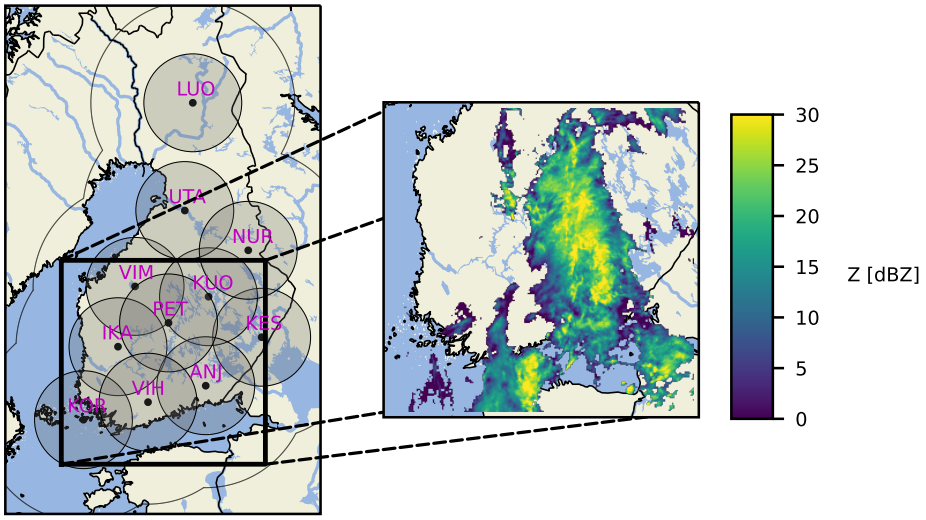


Figure 3.1: FMI radar domain and the chosen bounding box covering southern Finland. The three letter codes identify the radars, inner circles their worst case effective range (usu. during summer), and outer circles their best case effective range (usu. during winter). The lowest level reflectivity composite from the 30th of June 2020 at 00:05:00 is depicted on the right.

used in this work. This is done to alleviate the computational load of training and making predictions. As a consequence, the dimensions of the images drops to 256×256 and their effective spatial resolution then becomes two kilometers instead of one kilometer for full-scale composites.

From radar reflectivities, rainrate estimates were calculated using Eq. (2.1) solved for R. In this work, Empirically determined Z-R relationship parameters from Leinonen et al. [33] are used, where $A = 223$ and $b = 1.53$. Finally, $z = 10^{Z/10}$, giving a formula of

$$R = (10^{Z/10}/223)^{1/1.53} \tag{3.1}$$

for estimating the rainrate R (mm/h) from reflectivities Z (dBZ) with the current FMI data.

3.2 Model

3.2.1 The Baseline: A U-Net Based CNN

For the implementation of the Bayesian Convolutional Network the same architecture as for RainNet by Ayzel et al. [8] is used. Being itself heavily inspired by U-Net and SegNet model families, RainNet adopts a U-shaped encoder-decoder architecture just like them. The encoder is composed of successive intertwined pooling and convolutional layers, reducing image sizes passed to the next layer while increasing the number of filters. The decoder part adopts a mirror architecture of successive upsampling and convolutional layers, increasing the image size while reducing the number of filters. There are five levels in the encoder-decoder structure, just like in SegNet. Additionally, there are skip connections from the encoder to the decoder branch, carrying higher-level filter maps through, just like in U-Net. This is done to counteract the loss of smaller spatial scale details occurring with pooling. The network architecture is presented in Figure 3.2.

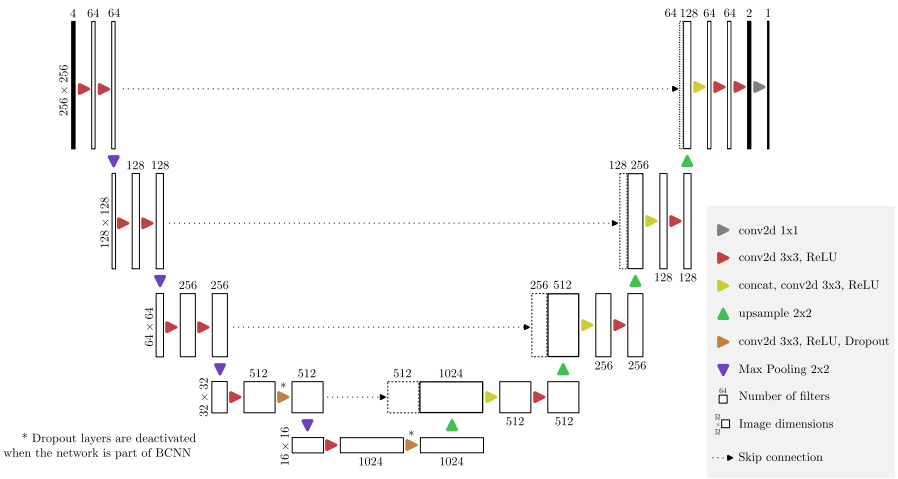


Figure 3.2: The functional CNN architecture used for this work, identical to that of RainNet from Ayzel et al. [8]. `concat` refers to the concatenation of feature maps along the channel dimension.

The convolutional filters have a filter size of three, with padding designed to preserve image shape, and a stride of one. Finally, the last convolutional layer in the network has a filter size of one with no padding, and it is followed

by a linear activation output layer, i.e. no (nonlinear) activation function. The network does not contain any fully-connected layers, and in total consists of 31.4M parameters. RainNet works by feeding in four consecutive radar images, with the physically speaking temporal dimension represented in the channel dimension of the tensor. As the output of the network, one image is obtained, representing the predicted frame of the next timestep. In order to make predictions at further leadtimes, the predictions are fed back to the network by appending them to the inputs while removing the oldest image of the sequence in an iterative process.

The network is trained by calculating a loss function between observed and predicted radar images. This loss function is originally the LogCosh loss in Ayzel et al. [8]. However in this work, the loss function is swapped for homoscedastic Gaussian Negative Log Likelihood (Gaussian NLL) loss. This is to ensure comparability of the results because it is the loss function used with BCNN. The Gaussian NLL loss [29] is defined by

$$\mathcal{L}_{\text{Gauss}}(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2\sigma_\ell^2} ||y_i - \hat{y}_i||^2 + \frac{1}{2} \log \sigma_\ell^2, \tag{3.2}$$

where $i = 1 \ldots N$ indexes the number of pixels in the prediction $\hat{y}$ and ground-truth y image or image sequences. Here σ_ℓ refers to the observational noise parameter. In homoscedastic Gaussian negative log likelihood, aleatoric uncertainty, characterized by σ_ℓ is assumed to be the same for all data. This parameter can be learned or left constant. In this work, it is tuned using RainNet, by selecting the best value based on validation predictive skill between candidates of 10^{-2}, 10^{-3}, and 10^{-4}. The choices were picked non-rigorously, trying to estimate which order of magnitude would be close to the true uncertainty at different rainrates.

The input rainrates are scaled using a logarithmic transformation $x_{\log} = \ln(x+0.01)$ as in Ayzel et al. [8], after which they are additionally in this work shifted upwards by a factor of five and then scaled down by a factor of 10 to fit into the range of zero to one. The logarithmic transformation of the data is needed because rainrates in themselves are lognormally distributed. This transformation makes them normally distributed, and thus more suitable inputs to neural networks. Additionally, shifting and scaling the inputs to the range from zero to one helps with convergence of RainNet and BCNN.

Because of the nature of predicting only the next frame, it is very challenging to get a stable nowcast at longer leadtimes. In practice, the precipitation values of nowcasts will often tend to diverge either to zero or infinity. This behavior is alleviated by calculating the loss function for nowcasts done over several leadtimes, which should make the resulting learned network more

temporally stable. Hence for training, after pretraining the network with one predicted frame, the training predictions are calculated for a 30 minutes leadtimes (six frames) and the ELBO loss is calculated such that each frame is weighted equally.

The reason for adopting RainNet as the functional architecture of the Bayesian Neural Network is that compared to for example ConvLSTM based models, it has faster inference without losing much in skill. This is important when building upon a model, especially when having limited resources, as added components make the training and inference slower.

3.2.2 BCNN: a Bayesian Extension to RainNet

BCNN is the Bayesian extension made for this work of the U-Net based architecture described in Section 3.2.1. In the network, posterior probability distributions of network parameters are modeled as diagonal Gaussian distributions and optimization is carried out using variants of the Bayes-by-Backprop algorithm described by Blundell et al. [12], minimizing the ELBO loss function as described in Section 2.3.1. Posteriors for all parameters are initialized identically to a mean μ_q of 0 and a variance σ_q of 10^{-3}. Parametrizing the posteriors as Gaussian distributions doubles the number of learnable parameters to 62.8M. Flipout reparametrization [71] is used in an attempt to reduce gradient variance and speed-up optimization.

As for the prior distribution, four variants are implemented. two Gaussian priors having $\mu_P = 0$ and alternatively $\sigma_P = 10^{-1}$ or $\sigma_P = 10^{-3}$, and two Gaussian Scale Mixture (GSM) priors having $\alpha = 0.5$ and alternatively $\sigma_1, \sigma_2 = 3 \times 10^{-1}, 3 \times 10^{-2}$ or $\sigma_1, \sigma_2 = 10^{-1}, 10^{-3}$. The scale-mixture prior might enable faster initialization and more accurate asymptotic behavior, but the Gaussian prior has more convenient mathematical properties, as it enables calculating KL-divergences in closed form when combined with a Gaussian posterior as in here, using the analytical expression

$$D_{\mathrm{KL}}(q(\mathcal{D}|\boldsymbol{w})||P(\boldsymbol{w})) = \frac{1}{2}\left[\log\frac{|\sigma_P|}{|\sigma_q|} - d + (\mu_q - \mu_P)^\top \sigma_P^{-1}(\mu_q - \mu_P) + tr\{\sigma_P^{-1}\sigma_q\}\right], \quad (3.3)$$

where q and P denote the multivariate diagonal posterior and prior, μ_q and μ_P their respective means, σ_P and σ_q their respective standard deviations and d the identity matrix of the number of dimensions equal to that of the distributions.

In BCNN, the radar images undergo the same scaling and preprocessing as described for RainNet in Section 3.2.1. The data likelihood cost is modeled as Homoscedastic Gaussian negative log likelihood (NLL) which is the same as the loss function used for RainNet. The σ_ℓ parameter for BCNN is chosen as the one with the best validation performance with RainNet, based on the ETS metric (see Section 3.4.1) at different thresholds, and is kept constant for most experiments afterwards.

Multiple procedures for weighting the complexity cost against the likelihood cost, some presented in Section 2.3.1, are implemented. In addition to the equal weighting scheme from Graves [25] abbreviated `equal` and the batch index dependent scheme from Blundell et al. [12] abbreviated `blundell`, a scheme reducing the weight of the complexity cost each epoch abbreviated `epoch`, defined as $\pi_j = 2^{-j}$, where $j = 0, 1, \ldots$ is the current epoch, is implemented to try and provide potentially better potential asymptotic nowcasting skill. This emanates from the fact that in preliminary experiments, `equal` has sometimes provided excessive regularization leading to a sub-optimal models, and `blundell` has had problems with converging when using batch sizes of one or two as is sometimes used in this work.

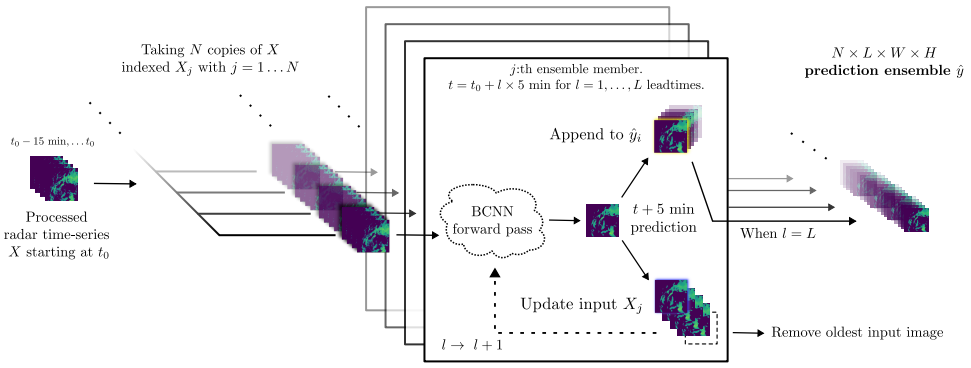


Figure 3.3: Inference procedure for generating ensemble nowcasts with the BCNN. Each training sample undergoes the same stochastic iterative procedure independently of one another. The input is in the present work always of dimension $L \times W \times H = 4 \times 256 \times 256$, and the resulting ensemble nowcast of dimension $N \times L \times W \times H = N \times L \times 256 \times 256$, where N could be e.g. one or four in training or 48 in verification, and L one or six in training, or 36 in verification.

With BCNN, predictions are generated by simply doing a forward pass through the network, which samples the parameters and produces a stochastic ensemble member. Predictions for further leadtimes are made in the same

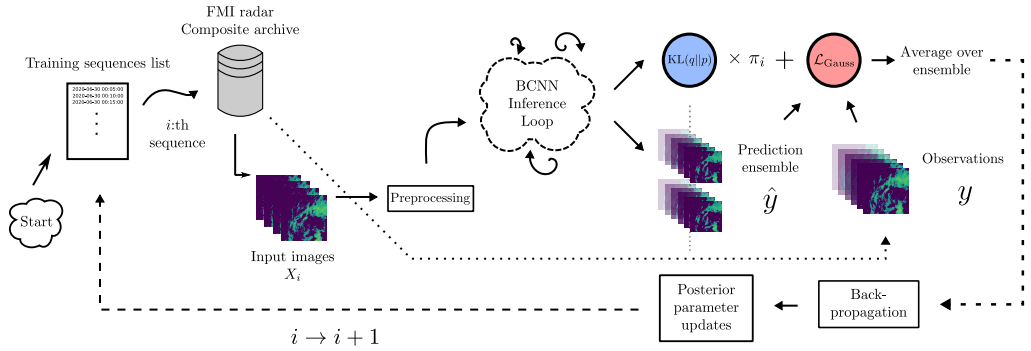


Figure 3.4: Training loop for the BCNN illustrated. One loop corresponds to one training minibatch. During the training, the KL-loss of each layer is accessed as a cached property of the network from the previous forward passes. Training sequences formable from successive timestamps are indexed with i. Additionally, although oversimplified, $\mathrm{KL}(q \mid\mid p)$ refers in this diagram to the cached complexity cost, π_i to the complexity cost weighting associated with the i:th sequence, and $\mathcal{L}_{\text{Gauss}}$ to the Gaussian NLL loss.

way as with the deterministic RainNet, that is by iteratively re-plugging the previous prediction into the network appended to the input sequence, with the oldest image removed. It is important to note that stochasticity is preserved because parameters are re-sampled at each step of the iterative prediction. Ensembles are produced by repeating this process multiple times independently, that is such that one initial ensemble member at timestep t always leads to a no more than a single new ensemble member at the next timestep $t+1$. Sampling is done this way to avoid the problem of exponential growth that would appear if prediction at timestep t would yield more than one new prediction at timestep $t+1$. The inference procedure is illustrated in Figure 3.3.

Training is performed first on a single leadtime (+ 5 minutes), meaning that during one training step, predictions are made and the loss calculated for only one leadtime. After this, the training is continued on six leadtimes, so using predictions until thirty minutes in the future. This is done in an attempt to speed up model convergence compared to training the model on many leadtimes right from the start. At each training step, input images are transformed, after which the inference procedure in Figure 3.3 is ran for the data, yielding a prediction ensemble and the complexity cost. Complexity and likelihood costs are averaged over the ensemble, and are used to calculate the ELBO loss and subsequent parameter updates after backpropagation. A

diagram illustrating the training process is shown in Figure 3.4.

The model selection process is performed hierarchically as follows: Starting with the optimal Gaussian NLL parametrized by the uncertainty parameter obtained through RainNet training experiments and `equal` complexity cost weighting, BCNN models with the four selected priors are trained. The best performing model is selected based on the validation likelihood cost, and it is retrained using the `epoch` and `blundell` complexity cost weighting. Out of them, the best performing model overall is selected and named `BCNN lt5`. This model has its training continued with six leadtimes, and the resulting model is called `BCNN lt30`. After that, an alternative training regime using four training samples (meaning using four-member prediction ensembles for the training) instead of only one is tried for `lt5`, with the resulting model called `BCNN lt5 new`. A further experiment for `lt5` is carried out again with four training samples but this time also with the Gaussian NLL data uncertainty parameter σ_ℓ set to an alternative, considerably bigger value than the optimal one from the likelihood perspective. This model is called `BCNN lt5 v2`. The training for that model is also continued for six leadtimes, resulting in `BCNN lt30 v2`. The batch size is chosen as the biggest power of two fitting into VRAM (32GB). Consequently, `BCNN lt5` is trained with a batch size of 32, `BCNN lt30`, `BCNN lt5 new` and `BCNN lt5 V2` using a batch size of 8, and finally `BCNN lt30 v2` using a batch size of two.

3.2.3 Additional Hyperparameters for the Neural Networks

Both for RainNet and BCNN, the Adam optimizer is used with an initial learning rate of 10^{-4} without any learning rate scheduler. For RainNet, other hyperparameters were left to their `torch.optim.Adam` default values, but for BCNN, the momentum parameter β_1 was increased to 0.95 to better accommodate the increased stochasticity of SVI compared to non-Bayesian NN [1]. Early stopping was set for both models with for condition that the validation loss does not decrease for 3 epochs. For BCNN, only the likelihood part of the loss was taken into account in the early stopping criterion.

3.3 Baseline Models

In this section are concisely presented all deterministic and probabilistic models used as baselines for benchmarking BCNN skill. The parameters used are stated for ensuring reproducibility, and a short reasoning for the choice of each model is given.

3.3.1 Deterministic Models

In order to verify the skill of BCNN, multiple deterministic models were used to produce nowcasts against which those produced by BCNN are compared. First and foremost, the ensemble averages of BCNN are compared against those of RainNet trained with both one and six frames for training, with the configuration outlined in Sections 3.2.1 and 3.2.3. It is interesting to compare the skill between the classical predictions and Bayesian version ensemble means, as underlying functional architectures and loss functions are the same, with the only change being the effect of the Bayesian prior on training and the uncertainty related to the Gaussian posteriors at evaluation time.

In addition, predictions were calculated with a few non-deep-learning models introduced in Section 2.1.1. These are Lagrangian Persistence (extrapolation nowcast) and LINDA-D, the deterministic version of LINDA. Lagrangian persistence represents in practice a very pragmatic baseline because of its ubiquity and simplicity, while having good skill. It is apparent that most (deterministic) models that can claim to be useful have to at least in a certain way outperform Lagrangian Persistence. LINDA-D on the other hand represents one such model, particularly outperforming Lagrangian Persistence on heavy rainfall. In the event that BCNN outperforms Lagrangian Persistence, LINDA-D serves to assess better the degree of performance achieved.

Concerning baseline models other than RainNet, before all predictions, the bounding box is applied, input data is converted to rainrate using Eq. (3.1) and thresholded at 0.1 mm/h and possible NaN values are set to zero. After predictions are made, these are thresholded at 0.1 mm/h and converted to back to reflectivities by first using Eq. (2.1) with parameters A and b stated in Section 3.1 and then converting Z to dBZ for storage. When predictions are read for metric calculation, they are once again converted to rainrate before metric calculations.

Both extrapolation and LINDA-D, as well as models described in Section 3.3.2 use Lucas-Kanade optical flow for advection field estimation with Pysteps [45] 1.6.1 default parameters and four input frames with field interpolation. These models also perform advection using the Semi-Lagrangian backward scheme of Pysteps using cubic interpolation and with other parameters left to default values. LINDA-D has stochastic perturbations set off with the `add_perturbations` flag and otherwise uses default parameters.

3.3.2 Probabilistic Models

Probabilistic skill verification was performed against STEPS and LINDA-P. The former was chosen as it is a well-established and broadly used model

offering good performance for a relatively low inference cost of 1-2 minutes to produce a 3 hours nowcast of 24-48 ensemble members on a modern CPU for a domain of 512×512 pixels [45]. It thus makes sense to want BCNN to perform at least on the same level as STEPS, considering that the time required for inference with it might not be much lower than that. LINDA-P, also known as the probabilistic variant of LINDA, is computationally more expensive but represents the state-of-the-art in terms of probabilistically predicting localized high-intensity rainfall. Consequently, LINDA-P makes for an interesting benchmark for those cases.

The same preprocessing and postprocessing pipelines as for deterministic cases (Section 3.3.1) are used with baseline probabilistic models. Also, the same optical flow and extrapolation parameters are used, and again, Pysteps 1.6.1 default values for methods are used unless stated otherwise.

Probabilistic nowcasts are set to produce 48 ensemble member nowcasts, just as BCNN. STEPS has parameters set to match the input data, that is `km_per_pixel` is 2.0 and `timestep` is 5.0, also `R_thr` is set to -10. LINDA-P has the same parameters as LINDA-D, except that naturally stochastic perturbations are set to `True` with the `add_perturbations` flag. It has to be noted that using the current configuration, STEPS is set to add advection field stochastic perturbations while LINDA-P does not. This affects the characteristics of the nowcasts generated, making STEPS nowcasts tend to have higher predictive uncertainty than those of LINDA-P.

3.4 Verification Visualizations and Metrics

Here onward are presented the verification metrics and visualization methods used to benchmark the performance of BCNN. The metrics are divided into deterministic and probabilistic ones. Deterministic metrics assess the performance of ensemble prediction means and deterministic predictions, whereas probabilistic metrics take ensemble uncertainty quality into account in their evaluation. This uncertainty estimation quality is also evaluated visually through different images of predictions.

3.4.1 Deterministic Prediction Skill Evaluation Metrics

Deterministic prediction skill scores are calculated in order to compare the raw predictive skill of ensemble nowcasts to the skill of deterministic models. Such comparison is an important facet of verification as low skill in deterministic scores would even for an otherwise competent model mean that it

would benefit from being complemented by a stronger deterministic model. In the case of ensemble nowcasting, deterministic scores were calculated for ensemble means. Implemented deterministic metrics are divided into four different categories, roughly complementing each others. These are continuous, categorical, and spatial scores, as well as radially-averaged power spectral density.

Continuous scores scores are as their name suggests distance metrics used for the evaluation of continuously valued predictions, i.e. regression tasks. The continuous scores used are Mean Error (ME) and Mean Absolute Error (MAE), and they are calculated for leadtimes up until two hours. ME is defined as

$$\mathrm{ME} = \frac{1}{N}\sum_{i=1}^{N} y_i - \hat{y}_i, \tag{3.4}$$

where summation is performed over the $i = 1, \ldots, N$ pixels in the radar image, y denotes ground truth and $\hat{y}$ the prediction made. The principal utility of Mean Error is detection whether predictions are biased towards too low or too high rainrates at a certain point in time. On the other hand,

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i|, \tag{3.5}$$

only cares about the magnitude of the errors in predictions by taking the absolute value, so it gives an idea of the prediction skill over the images *on average*. Nevertheless, this is not enough to accurately assess the skill of a nowcast method, because not all pixels are equally important, as most of them have no rain or very low rainrates that are not interesting to predict.

Additionally, A poor forecast may have good MAE and vice versa. To illustrate this, a forecast failing to predict localized intense precipitation, but otherwise accurately capturing light precipitation over large areas will usually have a small MAE but will have low operational usefulness. This limited utility of continuous

Precipitation is ...	**Observed**	**Not observed**
Predicted	TP (Hits)	FP (False Alarms)
Not predicted	FN (Misses)	TN

Table 3.1: Contingency table of categories used. Alternative nomenclature is shown in parentheses.

scores serves as a motivation
to introduce categorical scores. These are based on the principle of comparing the presence or absence of a rain event in observations and predictions as defined by having a pixel exceeding a threshold value R_{THR}. The categorical scores are defined by dividing events into four categories, that are true positives (TP), i.e rain events that were correctly predicted, true negatives (TN), i.e. lack of rain event that was correctly predicted, false negatives (FN), i.e. rain that wasn't successfully predicted, and false positives (FP), i.e rain that was erroneously predicted. These categories and their relations are illustrated in Table 3.1. Scores derived from those quantities that are used are probability of detection (POD) [56] defined as

$$\mathrm{POD} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{3.6}$$

which simply tells the probability that an event really occurring is correctly predicted. Next, false alarm rate (FAR) [56] is calculated as

$$\mathrm{FAR} = \frac{\mathrm{FP}}{\mathrm{TP} + \mathrm{FP}}, \tag{3.7}$$

which reversely indicates the percentage of positively predicted events not actually happening. Critical success index (CSI) [56], which is defined as

$$\mathrm{CSI} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP}} \tag{3.8}$$

is computed and aims to generally assess the performance of the forecast by taking the proportion of correct positive event predictions out of critically important cases, that is those excluding true negatives but including both false alarms (FP) and misses (FN). Lastly, the equitable threat score (ETS) [28] is defined as

$$\begin{aligned} \mathrm{ETS} &= \frac{\mathrm{TP} - rnd}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP} - rnd}, \\ \text{where } rnd &= \frac{(\mathrm{TP} + \mathrm{FN})(\mathrm{TP} + \mathrm{FP})}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP} + \mathrm{TN}} \end{aligned} \tag{3.9}$$

is computed. ETS aims to improve CSI assessment of forecast skill, by attempting to estimate the amount of random TP among the prediction using the term rnd, and remove that number of data points from the calculations.

The R_{thr} thresholds chosen for the performing verification are 0.5, 5.0, and 20.0 mm/h. 0.5 mm/h corresponds to very light rainfall and should be easy to predict, 5.0 mm/h to moderately heavy rainfall while 20.0 mm/h

corresponds to very heavy and rare rainfall, which is very difficult to predict even for short leadtimes. These metrics are calculated for up to two hours leadtimes.

In addition to evaluating prediction skill above rainrate thresholds, it is also important to be able to evaluate the nowcast predictive performance at multiple spatial scales. The reason for this is that larger spatial scales have more predictability, and so predicting smaller scales is more difficult while also being of high importance in the context of heavy localized rainfall.

As such, a spatial verification score, namely Fraction Skill Score (FSS) [52] is added to the panoply of metrics used. FSS is a verification metric aiming to estimate the prediction skill above a certain threshold at different spatial scales. It works by calculating a binary threshold exceedance map for forecasts and observations, averaging it over gaussian windows of different lengths representing scales, and calculating for each of those a Mean Squared Error (MSE) skill score relative to a reference low-skill forecast [52]. FSS is calculated for thresholds of 0.5, 5.0, and 20.0 mm/h and spatial scales of four and 16 kilometers, up until two hours again.

Last but not least, the ability to maintain small-scale details as the forecast leadtime increases is related to the skill of the nowcast with regards to the spatial scale. Failure in maintaining those details will results in low skill at small scales and a blurred forecast demonstrated by a dip in nowcast small-frequency power spectral density. Hence the last deterministic verification metric chosen is Radially-Averaged Power Spectral Density (RAPSD). This metric as its name suggests provides a way of calculating power spectral densities for 2D images such as nowcasts, independently of direction. RAPSD characterizes the amount and type of blurring occurring in deep-learning based models verified. It is calculated at 15, 60, and 120 minute leadtimes for all predictions and it is compared to observation RAPSD at those instants.

3.4.2 Visual Evaluation of Nowcast Predictive Uncertainty and Skill

In order to visually assess ensemble nowcasts, ensemble mean and predictive uncertainty equivalent to two standard deviations of nowcasts are visualized and compared to radar observations. These quantities are calculated for prediction ensembles after being converted to rainrate units. In addition, rainrate exceedance probability estimations for the ensemble are visualized for thresholds of 0.5, 5.0, and 20.0 mm/h. Leadtimes chosen for these visualizations include 5, 15, 30, and 60 minutes. This allows to assess visually the

evolution of the accuracy and uncertainty of the ensemble from very short to medium-long leadtimes.

These visualizations are made for two distinct cases from the verification (test) set containing a mix of stratiform and convective rainfall. The first one (Case 1) contains a fast north-moving heterogeneous front, and starts on the 25th of May 2019 at 13:00 local (Helsinki) time. The second one (Case 2), starting on the 18th of May 2021 at 21:00 local (Helsinki) time, represents again a heterogeneous front, but this time it exhibits large scale spinning motion rather than the relatively uniform motion of Case 1. The presence of both types of precipitation serves to assess differences in the developed method with regards to the rainfall type, knowing that convective events should be notoriously harder to predict. Case 1 nowcast examples are presented for `BCNN lt5`, `BCNN lt30`, `BCNN lt5 new`, and the STEPS baseline, whereas a Case 2 nowcast example is only presented for `BCNN lt5`.

3.4.3 Probabilistic Prediction Skill Evaluation Metrics

Deterministic verification scores are not enough to accurately assess ensemble forecast skill, because the advantage brought by ensembles does not lie in their mean value, but rather in the breath and quality of their distributions, allowing to weight in multiple possible scenarios. Consequently, specialized metrics designed to assess probabilistic forecasts are needed. For this work, four different probabilistic metrics are used for verification. These are the Continuous Ranked Probability Score (CRPS), rank histograms, reliability diagrams, and Receiver operating characteristic (ROC) curves.

CRPS is a metric generalizing MAE for probabilistic forecast. It is defined as

$$\mathrm{CRPS}(F, x) = \int_{-\infty}^{\infty} (F(y) - \mathbb{1}(x \geq y))^2 \mathrm{d}y, \tag{3.10}$$

where F is the forecast cumulative density function (CDF) and $\mathbb{1}(x \geq y)$ is the indicator function, representing the empirical CDF of the observation x. CRPS aims thus to represent the distance between those cumulative distributions as a proxy to estimate ensemble forecast skill.

A rank histogram is a measure that from the rank of the radar observation among ensemble members, builds a histogram. For each pixel, the rank is determined and and a bin is incremented accordingly. The shape of the histogram is indicative of whether the spread of the ensemble is representative of the true spread of observations. Variability in the ensemble equal to the uncertainty of observations would make a flat histogram. A convex histogram would mean that ensemble spread under-estimates true uncertainty, whereas

a concave histogram would mean that uncertainty is over-estimated, that is that the ensemble is more uncertain that it should be. Furthermore, skewness of the histogram gives a hint on whether there is any kind of bias in the predictions of the ensemble. This goes a step further than ME, as it does not simply tell the average error, but estimates its distribution in a sense.

ROC curves [37] are a verification method assessing the ability of a forecast to discern between positive and negative events, that is in the present case pixels with or without exceedance of a particular rainrate threshold. This is accomplished by keeping track of false alarm rates (FAR) against probability of detection (POD) of an event. Probabilities are divided into a certain number of bins over which corresponding FAR are averaged, and a curve is formed. A random forecast corresponding to no skill corresponds to a line, and an increasing area under the curve (AUC) indicates better discernment ability.

A reliability diagram presents observed relative frequencies of events (rainrates exceeding a certain threshold value) against their average forecast probabilities. When these two values are close to each others, the forecast is said to be reliable. This means that a reliable forecast is defined as having its reliability diagram close to the straight line corresponding to observed relative frequencies and forecast probabilities output by the model being equal. In practice, reliability diagrams are built by dividing forecast probabilities into bins with associated observed relative frequencies. The diagram is often accompanied by a histogram of event counts in those bins called the sharpness histogram. A model's forecasts can be said to be sharp if most predictions are close to zero or one probabilities. Sharpness can be seen as a measure of the "decisiveness" of a forecast.

ROC curves are conditioned on observation of an event, while reliability diagrams are conditioned on its forecast. Because of this they complement each others well in the evaluation of ensemble prediction skill. For ROC curves and reliability diagrams, the same rain intensity thresholds as in deterministic metrics, namely 0.5, 5.0, and 20.0 mm/h are used. CRPS is calculated for the first 12 timesteps covering leadtimes up until one hour, while other metrics are shown solely for a one hour leadtime. Additionally, the AUC of the ROC curve is shown for all models except LINDA-P for leadtimes of 0.5, 1.0, 5.0, 10.0, 20.0, and 30.0 mm/h and leadtimes of 15, 30, 60, and 120 minutes. Other metrics are calculated and shown for all models, including LINDA-P.

With all of the scores and visualizations described in Sections 3.4.1, 3.4.2, and 3.4.3 complementing each other, forecast skill is estimated as their combination, as no single one of them is capable of capturing all of the facets of a skillful nowcast by itself.

3.5 Experimental Setup Details

In order to make sure that results are valid, all timestamps having any (even a single) observation missing are discarded, and similarly timestamps having any prediction from any model missing are also removed from calculations. Additionally, only pixels where data is present in the predictions of all models are counted in metric calculations. This is accomplished by calculating a common NaN (missing value) mask using the logical OR operation over NaN values of each model, and subsequently applying that common mask to each prediction.

Regarding probabilistic nowcasts, uncertainties present in the data are coming from a diverse set of sources, including both aleatoric and epistemic uncertainties. In order to accurately represent the data distribution resulting from this compound uncertainty, a large ensemble size is needed. This is why the ensemble size was chosen to be 48 for all models. Verification might be even more reliable with bigger ensembles, but this would be at the expense of too much storage space needed for predictions, which would be very inconvenient. Chosen sizes were deemed to be a good compromise regarding this dilemma.

Intermediate results are saved such that predictions are saved in HDF5 archives, in a format where each predicted radar image is a separate dataset in deterministic models, and all ensemble members of predicted radar images for a certain leadtime is its separate dataset in ensemble nowcast models. In this storage procedure, predicted float reflectivity values are first compressed into 8-bit unsigned integers with a scale-offset scheme, then thresholded to 0.1 mm/h to facilitate further lossless compression which is finally carried out using GZIP. Trying to reduce the storage space needed is essential because predictions in this work take in total a very large ($>$ 100 GB after these procedures) amount of space on disk. This is a consequence of combining a big number of timestamps, ensemble members and leadtimes for multiple models. Raw metrics (numerical values) do not suffer from the same problem and are saved to disk in a binary Numpy format.

3.6 Software and Resources

The overall workflow of building models and making experiments, along with what software and environment was used for which task, is summarized in Figure 3.5.

The deep learning models were all implemented using the PyTorch framework and the PyTorch Lightning wrapper [20]. These libraries were chosen

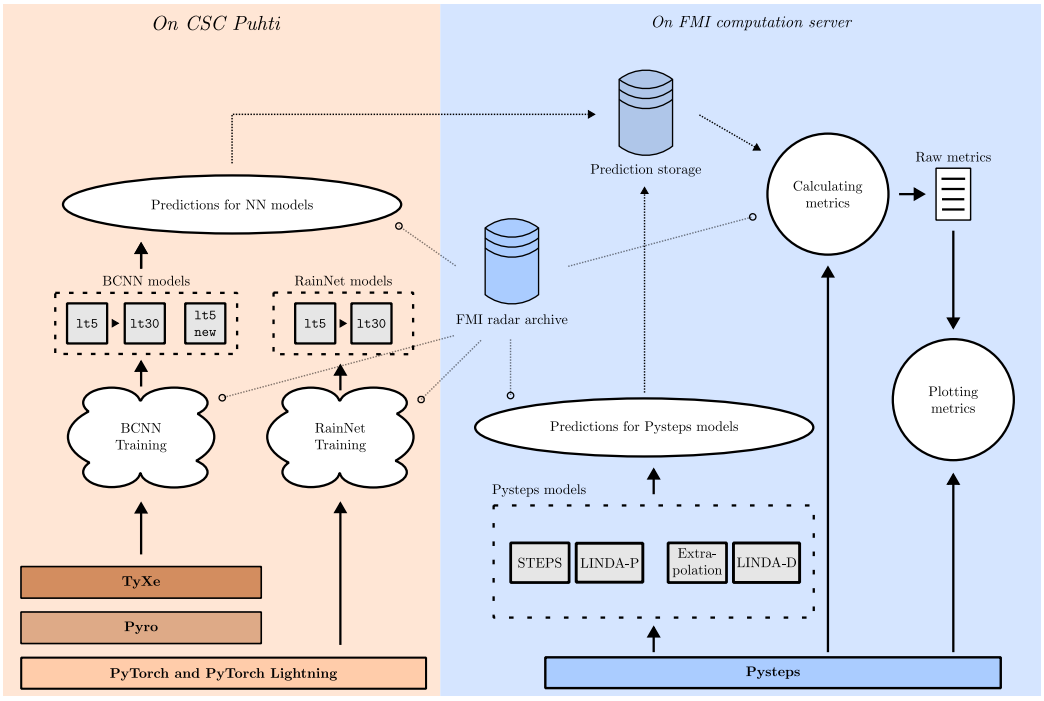


Figure 3.5: Overall workflow showing the process of this work, tied with used libraries for different components. Models and components shown are non-exhaustive.

because of the combined ease of prototyping brought by PyTorch Lightning, and the maturity and flexibility of the parent framework PyTorch. RainNet was ported to PyTorch following the original TensorFlow [3] implementation available on Github by Ayzel [7].

As for the implementation of the probabilistic inference mechanisms for Bayesian Neural Networks, choice was made not to implement them by hand, but to rely on the machinery contained in the Probabilistic Programming Language (PPL) Pyro [10], which is itself built on top of PyTorch and includes a fully-featured implementation of Stochastical Variational Inference (SVI). In order to facilitate the implementation task, the TyXe package [51] was used. TyXe is a library designed to provide an interface simplifying the implementation of Bayesian Neural Networks using PyTorch and Pyro. A few of the reasons why TyXe was chosen are that it permits easily turning existing neural networks into BNNs without having to use hard-coded bayesian layers, and dynamically switching on-and-off the local reparametrization trick and flipout in layers. Some problems encountered include components of TyXe having difficulties working together with PyTorch Lightning abstractions.

Verification experiments and non-deep-learning baseline models were ran and implemented using the open-source library Pysteps [45]. It provides implementations for all non-deep-learning models described as well as implementation of verification metric primitives used in this work, and tools for their visualization.

The computational resources from the Finnish IT Center for Science (CSC) were used for GPU intensive tasks such as training and calculating predictions with deep-learning models, and one of FMI's computational servers was used for performing other, more CPU-intensive operations such as predicting with baseline non-deep-learning models and calculating verification metrics. The models are trained on the CSC Puhti supercomputer, using one Nvidia V100 GPU with 32GB of VRAM, 64GB of RAM, and 10 cores from a 2.1 GHz Intel Xeon Gold 6230 CPU. As for the FMI server, it contains two Intel Xeon Gold 6138 2.0 GHz CPUs with each 20 cores and two threads by core, with 192GB of RAM.

Chapter 4

Results

This chapter presents the results obtained through the experiments performed. First, the results of the model hyperparameter optimization process are shown. Then, a nowcasting example is given for two meteorologically interesting cases for one selected model, with other versions and results for an existing baseline (STEPS) given in the appendix. The main version is `BCNN lt5`. The alternative versions are `BCNN lt30` based on the pretrained `BCNN lt5` and `BCNN lt5 new`. Unfortunately, `BCNN lt5 V2` and `BCNN lt30 V2` were left out of the results because an error in the training procedure and was only realized afterwards (only 20% of the training data was considered). Lastly, both deterministic and probabilistic performance of the BCNN variants and baselines models is presented.

4.1 BCNN Hyperparameter Optimization

Gaussian NLL loss data uncertainty parameter σ_ℓ was chosen to be 10^{-4} among choices of 10^{-2}, 10^{-3}, and 10^{-4} with the procedure defined in Section 3.2.1. As Gaussian NLL loss varies in magnitude when varying σ_ℓ, the choice was done by observing the ETS skill score of a deterministic RainNet at different precipitation thresholds on leadtimes from five to 30 minutes, with the scores depicted in Figure A.1.

The first hyperparameter to be optimized for the Bayesian CNN is the prior. Here, it was chosen based on validation Gaussian NLL loss at model convergence. To give an idea about the natural distribution of network parameters under no regularizing prior, the parameter distribution for the deterministic RainNet with $\sigma_\ell = 10^{-4}$ is shown in Figure 4.1. The distribution is centered at zero and has high kurtosis, mostly containing parameter values between -0.2 and 0.2.

The four priors used as candidates presented in Section 3.2.2 were chosen based on knowledge of the "natural" parameter distribution in Figure 4.1. Two are Gaussian with $\mu_P, \sigma_P = 0, 10^{-1}$: `BCNN Gauss 1e-1 EQ`(1) and $\mu_P, \sigma_P = 0, 10^{-3}$: `BCNN Gauss 1e-3 EQ` (2); and two are Gaussian Scale Mixtures with $\sigma_1, \sigma_2 = 10^{-1}, 10^{-3}$: `BCNN GSM BD EQ` (3) and $\sigma_1, \sigma_2 = 3\times10^{-1}, 3\times10^{-2}$: `BCNN GSM SD EQ` (4). Prior (1) serves as a weak prior, allowing most "natural" parameter values. Prior (2) on the other hand is much stricter, restricting the range of possible values to be much smaller than it would be without it. Prior (3) mixes (1) and (2) to try to figure out if the approach of scale mixture has benefits. Prior (4) finally is based on the observation that the empirical distribution of deterministic RainNet models is relatively close the PDF of this prior, making it interesting to look at whether using it results in performance improvements.

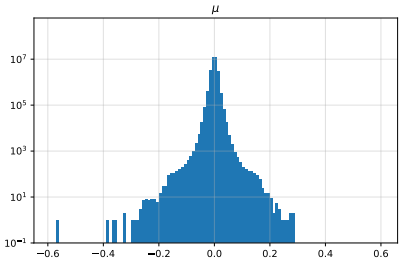
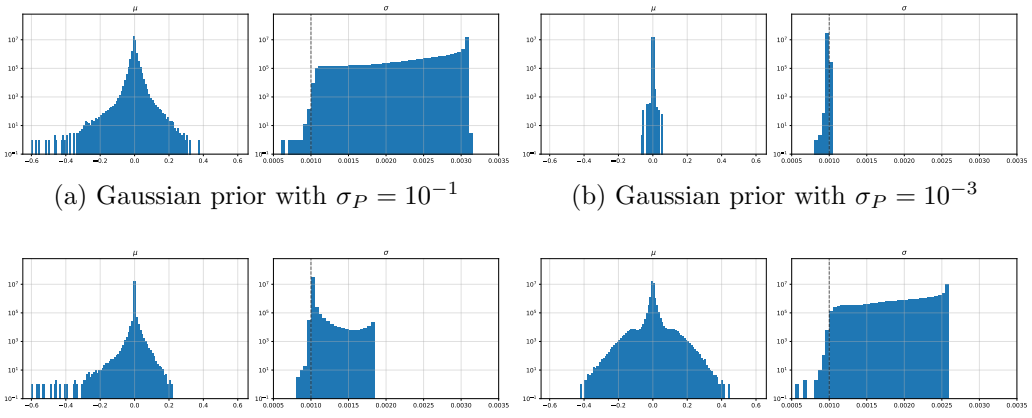


Figure 4.1: Parameter value distribution of RainNet trained with one leadtime and Gaussian NLL $\sigma_\ell = 10^{-4}$ after training.



(a) Gaussian prior with $\sigma_P = 10^{-1}$

(b) Gaussian prior with $\sigma_P = 10^{-3}$

(c) Gaussian Scale Mixture Prior: $\sigma_1, \sigma_2 = 10^{-1}, 10^{-3}$

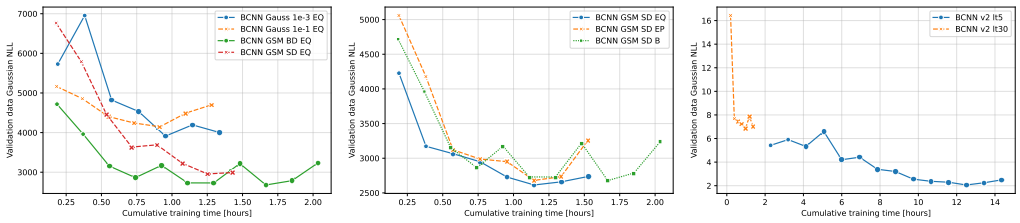
(d) Gaussian Scale Mixture Prior: $\sigma_1, \sigma_2 = 3 \times 10^{-1}, 3 \times 10^{-2}$

Figure 4.2: Parameter posterior distributions in relation to the prior chosen. μ_q histograms describe the parameter mean distributions, whereas σ_q distributions describe the distribution of their standard deviations. The complexity cost weighting scheme here is set to `equal` weighting. The black dashed vertical lines indicate the initialization value for standard deviations.

The effect of the prior on the parameter mean and standard deviation

distributions is portrayed in Figure 4.2. It is seen that for all priors except (2), standard deviation distributions end up tilted towards bigger values than the initial scale 10^{-3}, pointing to a possibility that it was way too small and that combined with a small learning rate 10^{-4}, there wasn't just enough time for the them to grow where they should have. Concerning means, it is observed that the breadth of the posterior mean distribution is closely associated to the prior distribution breadth. With prior (2), it is seen that most mean values μ_q are actually very close to 0 and while standard deviations σ_q are close to their initial values, very little of them exceed it. This can be intuitively understood because the value of the prior σ_P and initial posterior σ_q standard deviations are the same, and the prior prevents the growth of posterior distributions to breaths where prior violation could happen.

The results of the optimization are shown in Figure 4.3.a. Here the complexity cost weighting scheme is set to `equal` weighting and the likelihood $\sigma_\ell = 10^{-4}$. It is shown that that best performance is achieved by prior (4), making it the choice to be used in later experiments. The weaker prior (1) is slower to converge but achieves relatively good asymptotic performance, while the stricter prior (2) gives decent results more quickly but has trouble converging to low loss values. This was substantiated by prior (2) making blurrier forecasts. Prior (3) behaved as a middle-ground between priors (1) and (2). It benefited from the same fast initial convergence as prior (2) but achieved asymptotic performance somewhere between the two. Interestingly, the training time difference between Gaussian and GSM priors was not significant contrary to expectations.



(a) Prior choice (b) KL weighting choice (c) `lt5` vs `lt30`

Figure 4.3: The validation set Gaussian NLL loss of the different BCNN candidates against total training time, highlighting the model selection process.

With the prior chosen, the next hyperparameter to be optimized is the complexity cost weighting scheme. `BCNN GSM SD EQ` refers to the `equal` weighting scheme, `BCNN GSM SD EP` to the `epoch` weighting scheme, and `BCNN GSM SD B` to the `blundell` weighting scheme. The results are depicted in Fig-

ure 4.3.b. Best performance is observed with the `blundell` weighting scheme which was chosen for further experiments, with close performance achieved using `equal` weighting. Although relatively close in performance, the `epoch` weighting scheme did not deliver in the end, although it had shown promising results in preliminary experiments, where the `blundell` scheme did not work. The effect of the complexity cost weighting scheme on the weight distribution is shown in Figure A.2. Here, the `blundell` scheme differs considerably compared to others in terms of scale σ_q distribution, as σ_q values are more closely centered around the initial value and there is no shift to bigger values as observed with other schemes. This might indicate a poor learning of posterior scales.

Finally total cumulative training times for `lt5` and `lt30` variants are compared in Figure 4.3.c. The model used as an example is `BCNN lt5 V2` (for which no other verification results are shown). It is clearly seen that continuing the training with 30 minute leadtime, i.e. six predictions, requires much more computational resources than simply using 5 minute leadtime, i.e. one prediction, which is only worth if skill is improved as a result. In the current example, using a likelihood σ_ℓ parameter of 2×10^{-2}, likelihood loss is improved with `lt30`, but the opposite is found using a likelihood σ_ℓ of 10^{-4}. The increase in training time going from `lt5` to `lt30` was observed regardless of the model parameters.

4.2 Nowcasting Examples

Here, example nowcasts made with the developed model are shown for two distinct events. The main three models retained for verification are `BCNN lt5`, `BCNN lt30`, `BCNN lt5 new`. The main model showcased is `BCNN lt5`, with additional models showcases as well as results for STEPS in Appendix A.2. Predictive mean and uncertainty, as well as exceedance probabilities for 0.5, 5.0, and 20.0 mm/h are shown.

4.2.1 Case Study 1 : Rapidly Moving Mixed Stratiform and Convective Rain Event

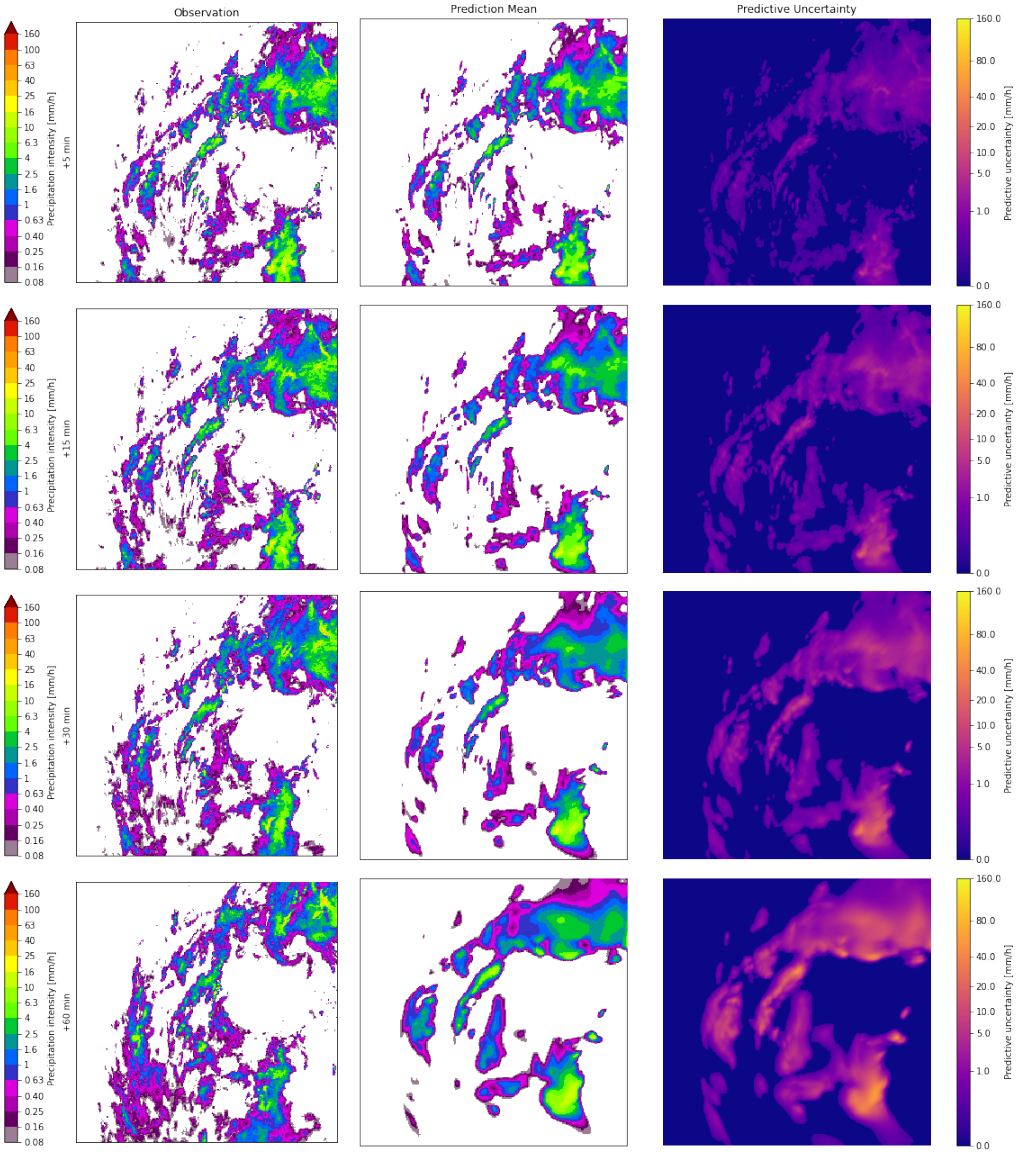


Figure 4.4: Predictive mean and uncertainty (two standard deviations) for Case 1 on the 25th of May 2019 starting at 13:00 local Helsinki time for the `BCNN lt5` model, covering the area of southern Finland.

The predictive mean and uncertainty for `BCNN lt5` on Case 1 is shown in Figure 4.4. It is observed that mean predictions get increasingly smoothed out

with time as expected, and that predictive uncertainty is highly correlated with predicted mean rainrate. Additionally, predictive uncertainty increases over time, achieving top values exceeding 50 mm/h in areas of predicted heavy rain at one-hour leadtimes. It is seen that advection and the overall movement of the front is well-modeled, while smaller-scale details even if corresponding to medium or heavy rain, are quickly lost.

Smoothing out of predictions has an important effect of reducing model capacity to predict high rainrates, which are often concentrated over small areas. This is an especially important effect because smoothing does not originate simply from averaging, but is also present in individual ensemble members, due to the combined effect of the loss function and of the iterative prediction scheme.

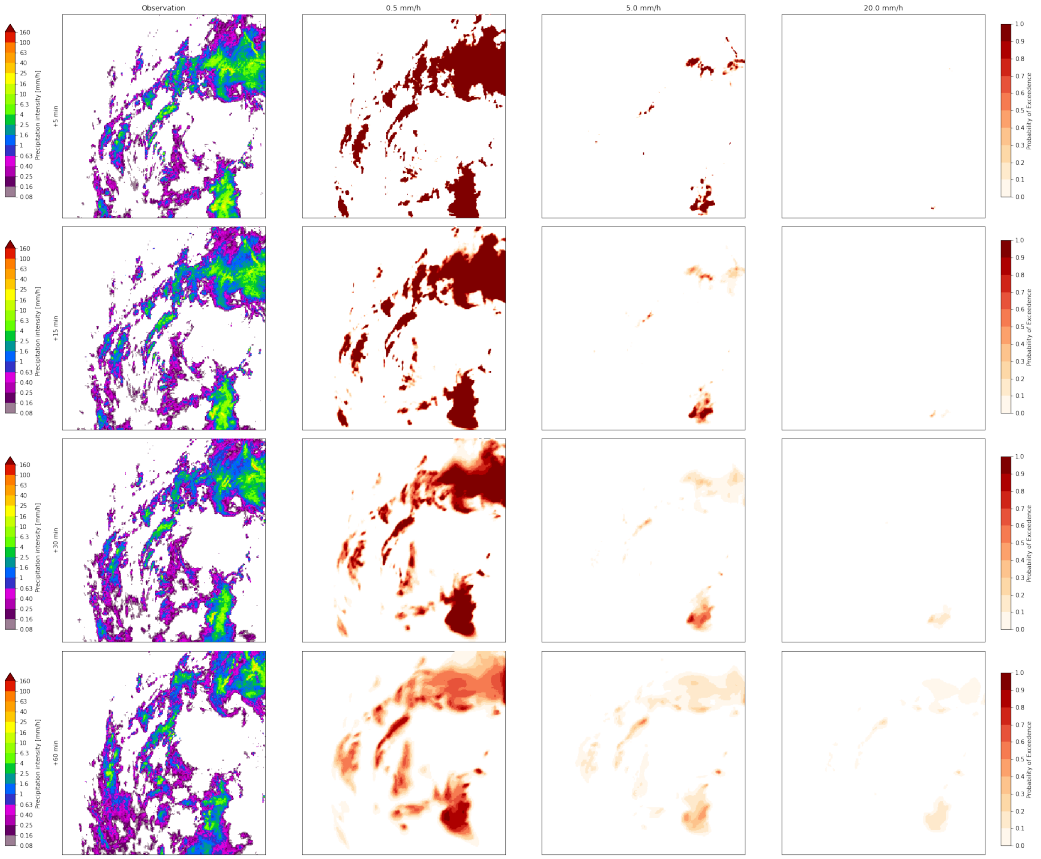


Figure 4.5: 0.5, 5.0, and 20.0 mm/h exceedance probabilities for Case 1 on the 25th of May 2019 starting at 13:00 local Helsinki time for the `BCNN lt5` model, covering the area of southern Finland.

Exceedance probabilities for `BCNN lt5` on Case 1 are shown in Figure 4.5.

Both for this model and in general, it is seen that exceedance probabilities are lower for high leadtimes and high precipitation thresholds. In the case of this model, all exceedance probabilities seem to be smoothed over time, which is indicative of the uncertainty of the extent of rain being at least to some degree taken into account. It is also seen that especially at 20.0 mm/h, the predicted exceedance probability at one-hour leadtime does not always originate from current time high rainrates. This is seen from the top of the image at +5min, where a small exceedance probability is present, but disappears at further leadtimes, before reappearing independently.

Both in the plot of prediction mean and uncertainty, as well as in that of exceedance probabilities, it is seen that because precipitation is advected towards the north (top of the image), lower parts of the image can not be predicted.

4.2.2 Case Study 2 : Mixed Stratiform and Convective Rain Event with Spinning Motion

The predictive mean and uncertainty for `BCNN lt5` on Case 2 is shown in Figure 4.6. In this case, the same overall features are observed as in Case 1. Here, the spinning motion is correctly predicted and the mean rainrates are relatively close to ground truth. As in Case 1 but especially here, many small features are lost and especially small convective cell evolution is not correctly predicted.

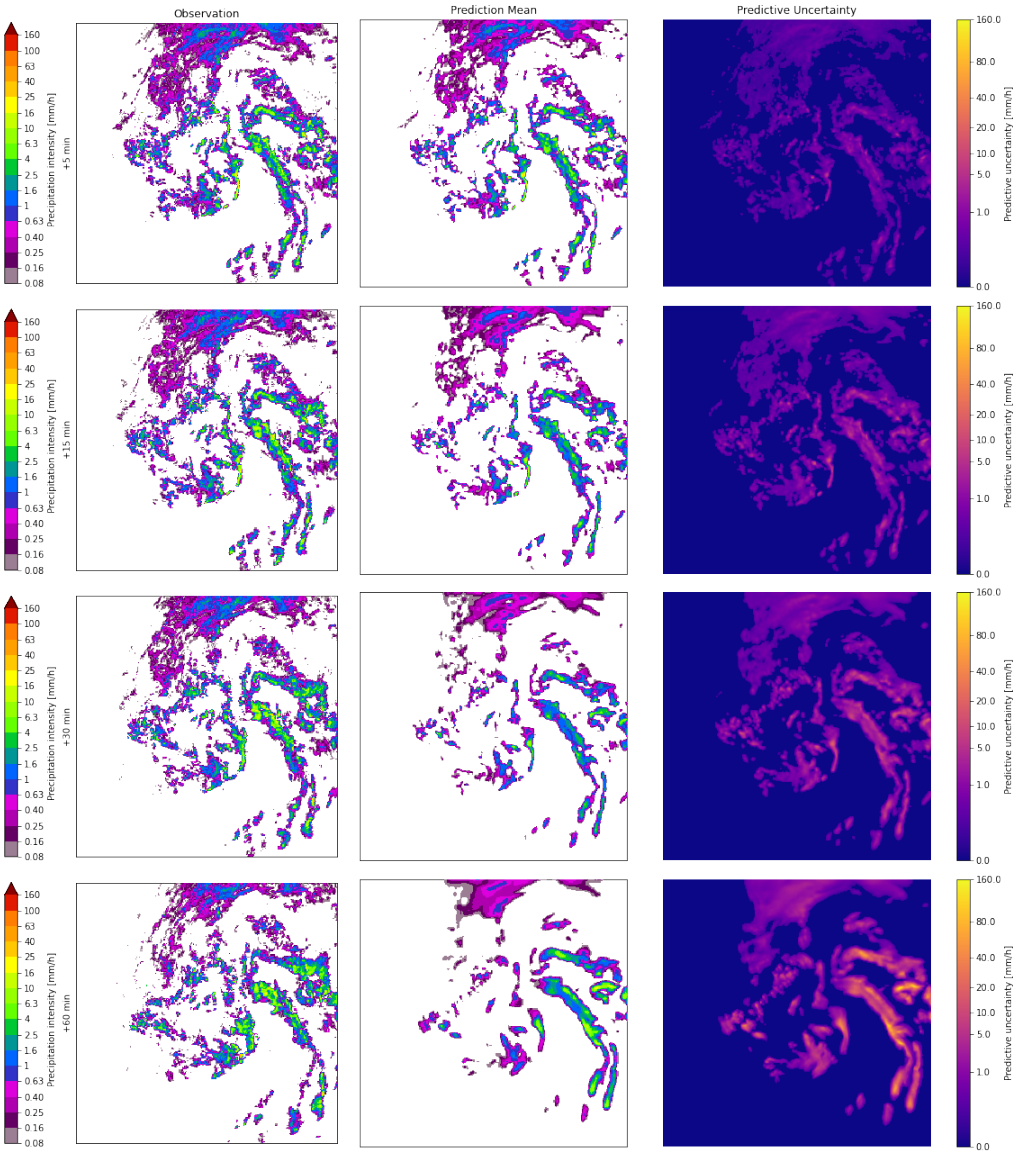


Figure 4.6: Predictive mean and uncertainty (two standard deviations) for Case 2 on the 18th of May 2021 starting at 21:00 local Helsinki time for the `BCNN 1t5` model, covering the area of southern Finland.

The exceedance probabilities are shown in Figure 4.7. Many things happening here are similar to Case 1, but especially striking is lower exceedance probabilities at one-hour leadtime for the 0.5 mm/h threshold. Here, in many samples the spinning "arms" were lost by that time because of their thinness and the tendency of thin or small features to disappear.

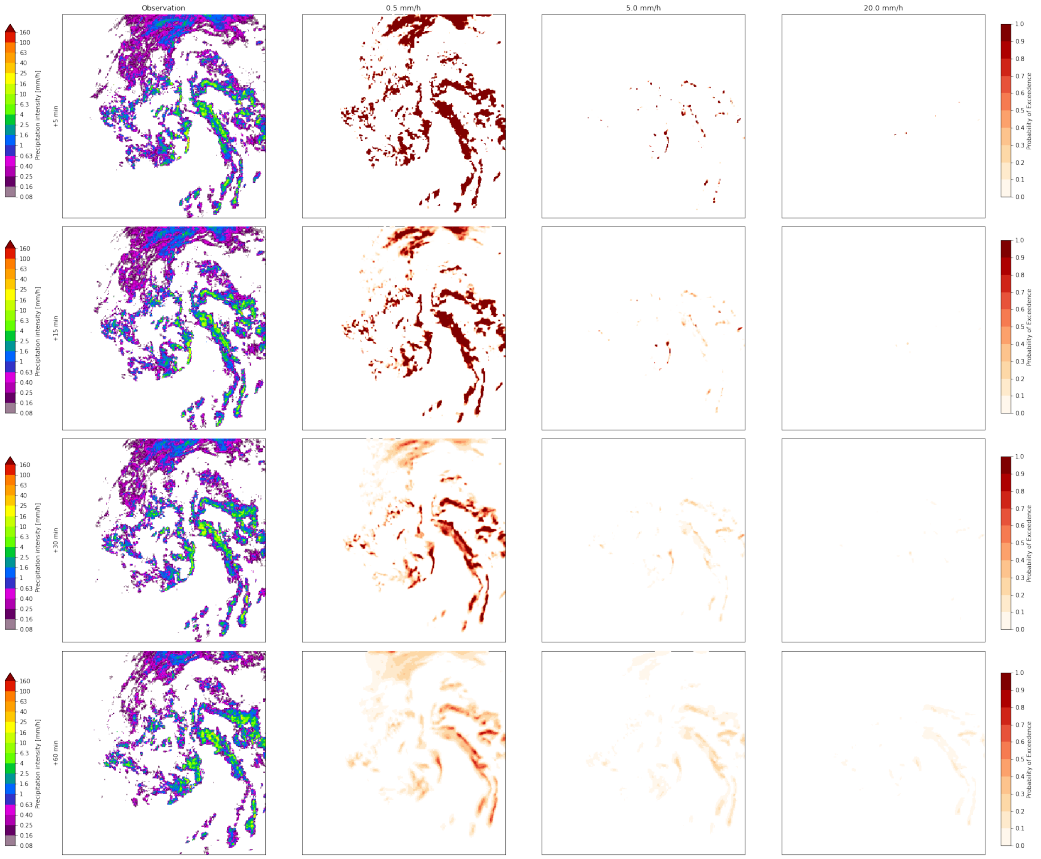


Figure 4.7: 0.5, 5.0, and 20.0 mm/h exceedance probabilities for Case 2 on the 18th of May 2021 starting at 21:00 local Helsinki time for the `BCNN lt5` model, covering the area of southern Finland.

4.2.3 Additional Case Study 1 Results

Case 1 prediction mean and uncertainty, as well as exceedance probabilities are respectively visualized for `BCNN lt5 new` in Figures A.3 and A.4, `BCNN lt30` in Figures A.5 and A.6, and finally STEPS in Figures A.7 and A.8. For `BCNN lt5 new`, results are similar as in the `BCNN lt5` case, but exhibit a much bigger bias towards predicting strong rain, with more ensemble members predicting increasingly high rainrates. From exceedance probabilities, this can be seen as overall higher values, making the model more susceptible to predicting stronger rainrates.

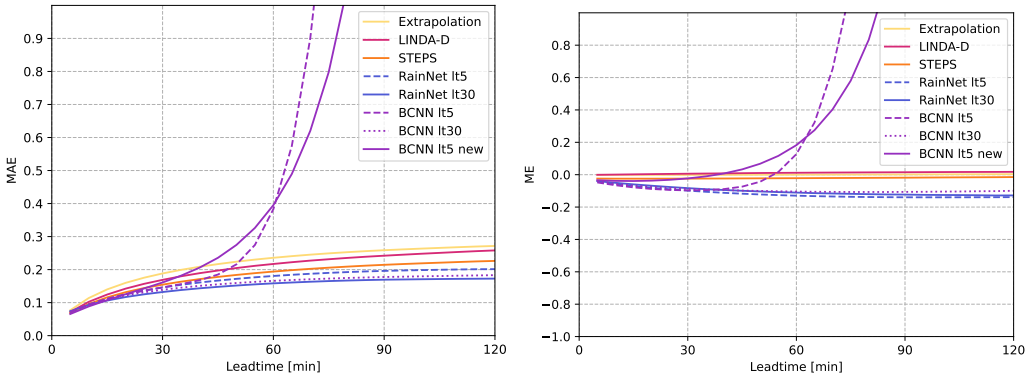
For `BCNN lt30`, it can be seen from both figures that the overall skill is lower than for `lt5` cases. In particular, predictions are more smoothed out and more details are lost, hinting to the conclusion that `lt30` training did

not work as expected. Here, predictive uncertainty does not increase with leadtime, and higher rainrates are often not predicted at all. However it can be seen that the network predicts consistently high exceedance probabilities, and that those are not (esp. 0.5 mm/h) smoothed out spatially with increased leadtime.

Finally, looking out at figures of the STEPS baseline, it is easily seen that despite being a simpler model, it predicts better spatial uncertainty in future rain. However STEPS has a hard time predicting exceedance probabilities for high (esp. 20.0 mm/h) thresholds. One feature is also that in the case of STEPS, predictive uncertainty is more spread out than predictive mean, which is a behavior not observed in any model developed here. This is likely due to the advection field perturbations applied to STEPS.

4.3 Deterministic Prediction Skill

The following results describe the skill of ensemble means of developed models against baseline models.

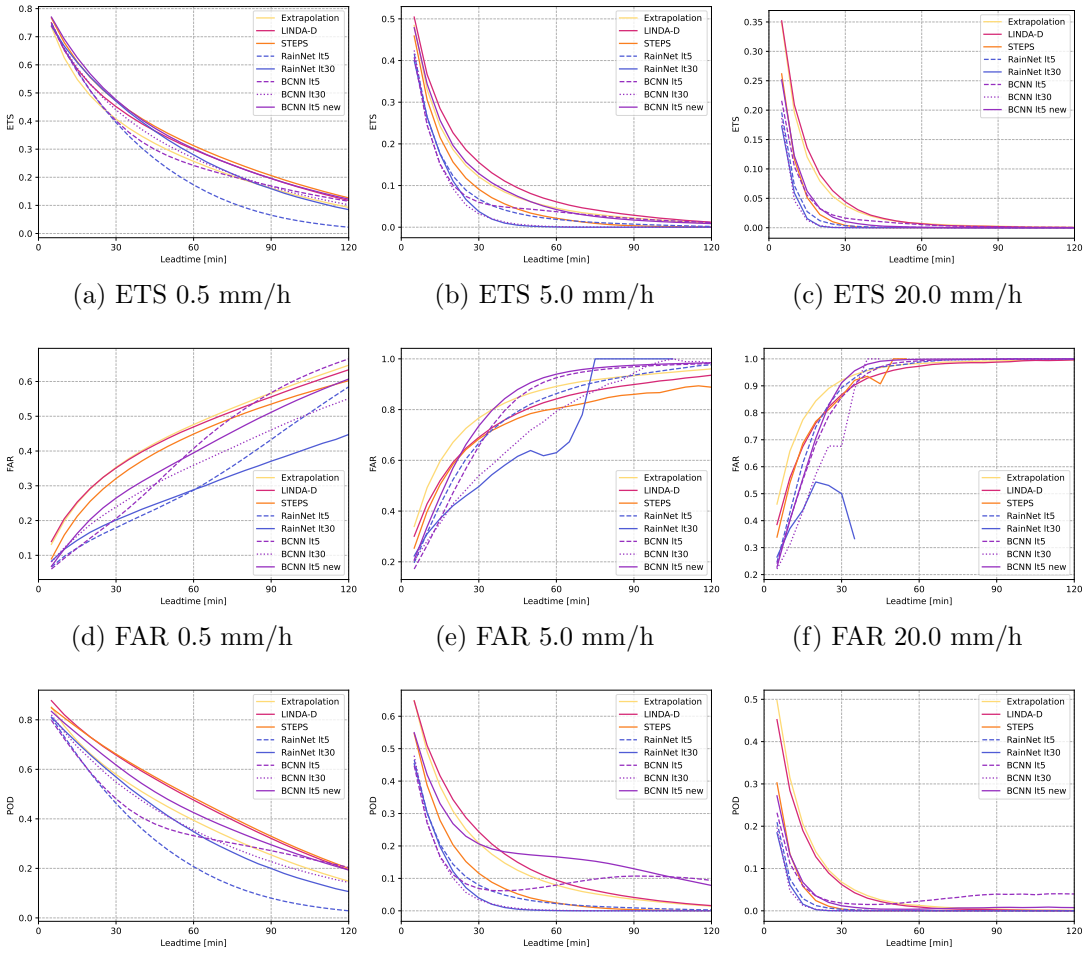


(a) MAE (mm/h) (b) ME (mm/h)

Figure 4.8: Continuous Metrics (MAE and ME) for BCNN variants and baselines, up until two-hour leadtime.

Starting off with the continuous metrics, their values are shown in Figure 4.8. BCNN and RainNet models have similar MAE and ME, except for `BCNN lt5` and `BCNN lt5 new` cases, where ensemble means exhibited clear bias towards too strong rainrate predictions after 30 minutes. These models shall be called "positively biased". ME was around zero for classical models and tended lower with other RainNet and BCNN models, showing bias towards too weak predictions ("negatively biased"), mainly due to smoothing. Still,

BCNN and RainNet gave mostly clearly superior MAE compared to classical models, which is understandable as Gaussian NLL is a loss function part of the same class as MAE, which tends to optimize for minimizing pixel-per-pixel deviations and by extension MAE. There is also reason to believe that because of the asymmetry caused by the log-transformation on NN input data, both negative biases shrink slower and positive biases grow faster in rainrate estimates than in transformed data, contributing to the observed behavior.



(a) ETS 0.5 mm/h (b) ETS 5.0 mm/h (c) ETS 20.0 mm/h

(d) FAR 0.5 mm/h (e) FAR 5.0 mm/h (f) FAR 20.0 mm/h

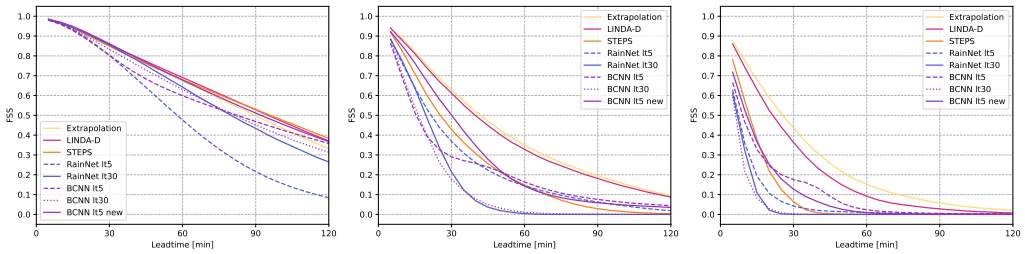
(g) POD 0.5 mm/h (h) POD 5.0 mm/h (i) POD 20.0 mm/h

Figure 4.9: Categorical metrics (ETS. FAR, and POD) for BCNN variants and baselines, up until two-hour leadtime at thresholds of 0.5, 5.0, and 20.0 mm/h.

POD, FAR, and ETS metric values are shown in Figure 4.9. For these

metrics, the strongest overall result among developed models is achieved by `BCNN lt5 new`, which is strongly positively biased. For POD, a resurgence at further leadtimes is seen in `BCNN lt5` and `BCNN lt5 new` models, because of the phenomenon of "exploding rainrates" where these attain very high ever-increasing values over large areas. In other cases, POD is better for classical baselines than CNN based models, when negatively biased. Accompanying high POD of "exploding rainrates" of positively biased models is a very high FAR, demonstrating that this high POD is not necessarily indicative of skill. FAR is interestingly worse for extrapolation based baselines than CNN based models at 0.5 mm/h threshold, a phenomenon that disappears at higher thresholds.

ETS summarizes the categorical skill of the models. Especially at 5.0 and 20.0 mm/h, LINDA-D has the best predictive skill, while CNN based models and STEPS lag far behind. Overall, no conclusion can be made about whether the Bayesian extension improved the categorical predictive skill, as the the only skillful outlier `BCNN lt5 new` didn't have a RainNet model with a corresponding training routine to compare to. CSI scores are shown in Figure A.9, but they do not differ much from ETS.



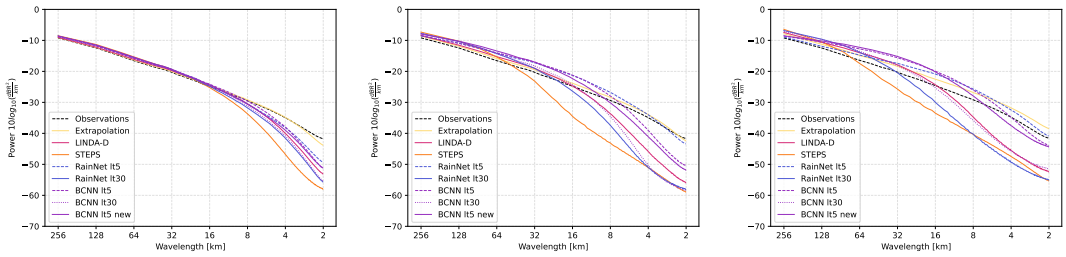
(a) 0.5 mm/h (b) 5.0 mm/h (c) 20.0 mm/h

Figure 4.10: FSS metric at a 16km scale for BCNN variants and baselines, up until two-hour leadtime at thresholds of 0.5, 5.0, and 20.0 mm/h.

The FSS metric results at a 16km spatial scale for the thresholds of interest are shown in Figure 4.10. Here, it is found that the difference between extrapolation based deterministic models and others is strikingly big. FSS at a scale of 4km is shown in Figure A.10, but the results do not differ much from ETS and CSI at the original scale of two kilometers, as the change is only two-fold.

RAPSD is shown in Figure 4.11 for leadtimes of 15, 60, and 120 minutes. As expected, Extrapolation nowcast which preserves details has a PSD very close to observations, while other models show drop in power at short wavelengths, with steeper drops at further leadtime. Interestingly, this drop

in power is bigger at 60 minutes than at 120 minutes, which might have something to do with the OR-masking of NaN values, reducing significantly the area used for calculation at two hours. The biggest drops in small-scale details are experienced by STEPS and `BCNN lt30` as well as `RainNet lt30` models. All `lt5` CNN models experienced less loss in details compared to `lt30` models. It is difficult to conclude whether taking ensemble means had any significant effect on the loss of small-scale details of CNN approaches without looking at individual ensemble members and ditching the OR-masking.



(a) 15 min (b) 60 min (c) 120 min

Figure 4.11: Radially-averaged power spectral density (RAPSD) for BCNN variants and baselines, at leadtimes of 15 minutes, one hour, and two hours.

4.4 Probabilistic Prediction Skill

The following section shows probabilistic prediction skill assessment results of BCNN models against STEPS and LINDA-P baselines.

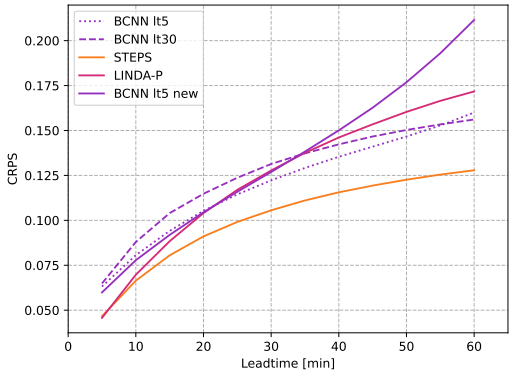
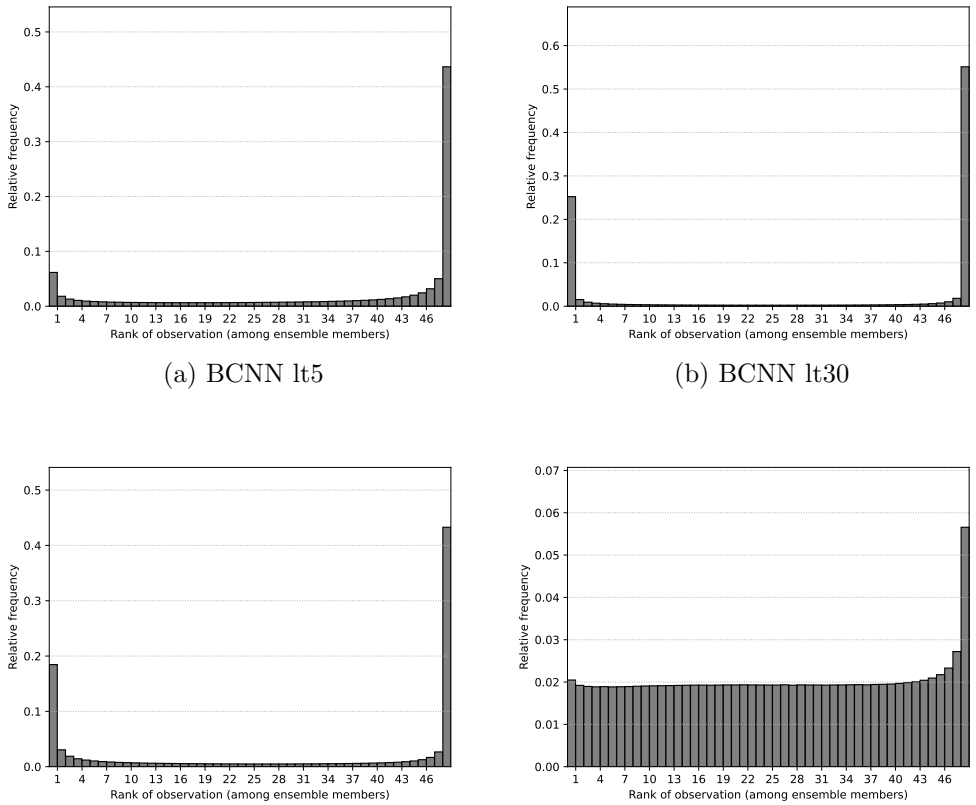


Figure 4.12: CRPS metric for all ensemble models at leadtimes up until one hour.

The continuously ranked probability score for all ensemble models is shown in Figure 4.12 for leadtimes up until one hour. Overall, STEPS achieves best performance, likely owing to advection field perturbations. BCNN models consistently perform worse than STEPS, and only LINDA-P declines to the performance range of BCNN models after 20 minutes.



(a) BCNN lt5

(b) BCNN lt30

(c) BCNN lt5 new

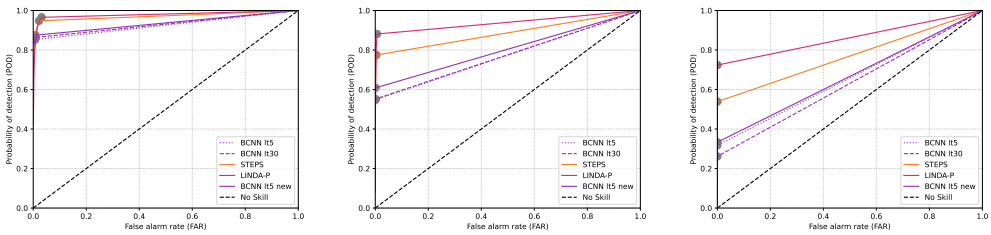
(d) STEPS

Figure 4.13: Rank histograms for BCNN and STEPS model nowcasts at a one-hour leadtime, for precipitation events exceeding 0.1 mm/h.

Rank histograms for BCNN and STEPS model nowcasts at one hour are shown in Figure 4.13, and the corresponding plot for LINDA-P is depicted in Figure A.11. It is seen that LINDA-P and especially STEPS have very flat histograms, which is indicative of well-calibrated uncertainty, i.e. no under- or over-estimation of true uncertainty is made, as observations are ranked with close to equal probability amongst ensemble members. Only a slight "negative bias" towards weak precipitation nowcasts can be detected by

observing that the rightmost bin is slightly over-represented. That is cases where the observation had the highest rainrate compared to all ensemble members.

Compared to those baselines, BCNN models had all strongly convex histograms, severely under-estimating the true uncertainty of data. Furthermore, BCNN models had the rightmost bin over-represented too, with 40 to 50% of the time, the observation having a higher rainrate than any ensemble member. Additionally, `BCNN lt5 new` and `BCNN lt30` had around 20% of observations where the observation was smaller than any ensemble member, hinting to many cases where the network is vastly over-confident about its high rainrate prediction.

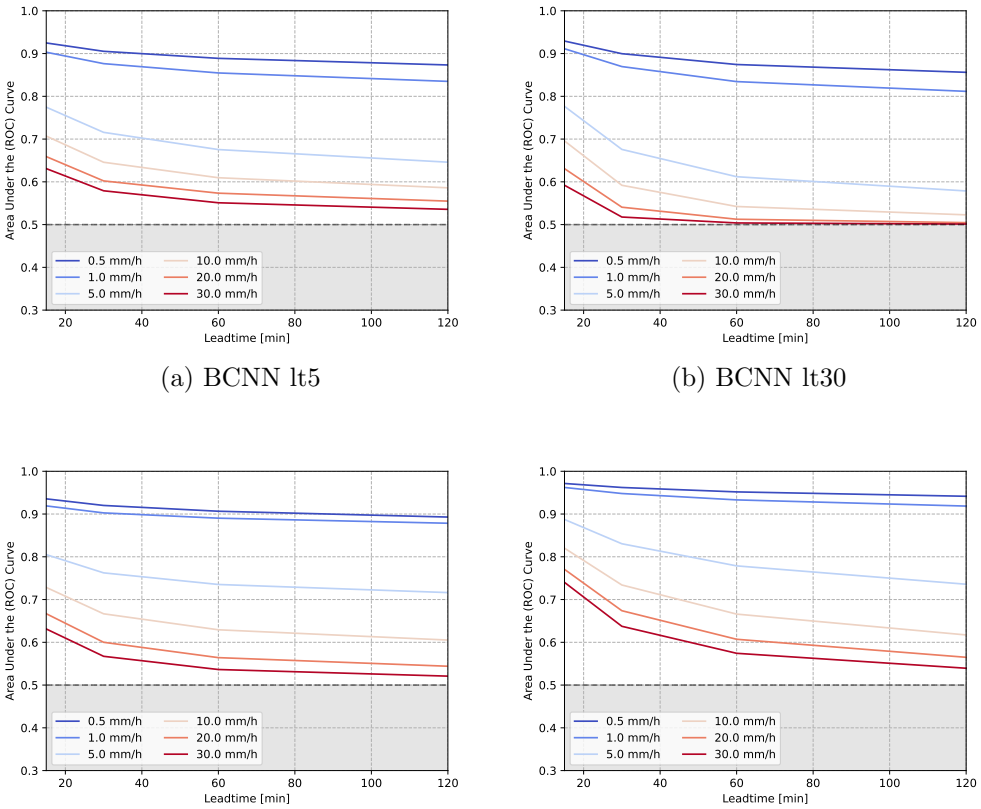


(a) ROC 0.5 mm/h (b) ROC 5.0 mm/h (c) ROC 20.0 mm/h

Figure 4.14: ROC curves for one-hour exceedance probabilities at 0.5, 5.0, and 20.0 mm/h precipitation thresholds. The probability thresholds used are ten thresholds uniformly distributed between zero and one.

The ROC curve for models at one hour for exceedance probabilities at thresholds of interest is shown in Figure 4.14, and an overview of discrimination power as a function of leadtime and threshold proxied by the ROC area under the curve (AUC) for BCNN models and STEPS is shown in Figure 4.15. From the ROC curves, it is seen that best discriminative power is achieved by LINDA-P, overwhelmingly so at high thresholds. Second was STEPS, followed by the three BCNN models lagging far behind. From the AUC plots, it is visible that for all models, leadtime exhibits a stronger effect on discrimination power for higher thresholds. The AUC plots come to confirm that the discrimination power difference between models derived from ROC curves is consistent varying leadtime and threshold, with perhaps the exception of `BCNN lt5` compared to `BCNN lt5 new` performing better on higher, but worse on lower thresholds.

Lastly, the reliability diagram accompanied by its sharpness histogram for each model, at the same precipitation thresholds at a one-hour leadtime is shown in Figure 4.16. Starting with the histogram, the sharpness of nowcasts



(a) BCNN lt5

(b) BCNN lt30

(c) BCNN lt5 new

(d) STEPS

Figure 4.15: ROC Area under the curves summarized for BCNN and STEPS models. The dashed lines correspond to a random forecasts, and grayed out areas to results worse than that of a random (no skill) forecast.

seems to generally increase with the threshold. At 0.5 mm/h, all models have a relatively similar U-shaped sharpness histogram, indicative of moderate sharpness, with BCNN models perhaps a little sharper. Higher thresholds see all models adopt distributions of exceedance probabilities skewed towards smaller values. Generally it seems to be that it is at least one order of magnitude less common to have occurrences of forecast probabilities close to 0.5 for BCNN models compared to LINDA-P and STEPS.

Now turning towards the reliability diagram, it is clear that nowcasts get less reliable for higher thresholds. One will notice that LINDA-P and STEPS both stay both much closer to the black dashed line, indicator of perfect reliability. BCNN models tend here to assign high forecast probabilities to events that will really take place less often, and conversely assign too low

forecast probabilities to events that will take place more often in practice. This expresses a lack of reliability. Furthermore, one can see that the reliability diagram of BCNN models gets non monotonous at 20.0 mm/h. This is due to too small sample sizes in the middle probabilities.

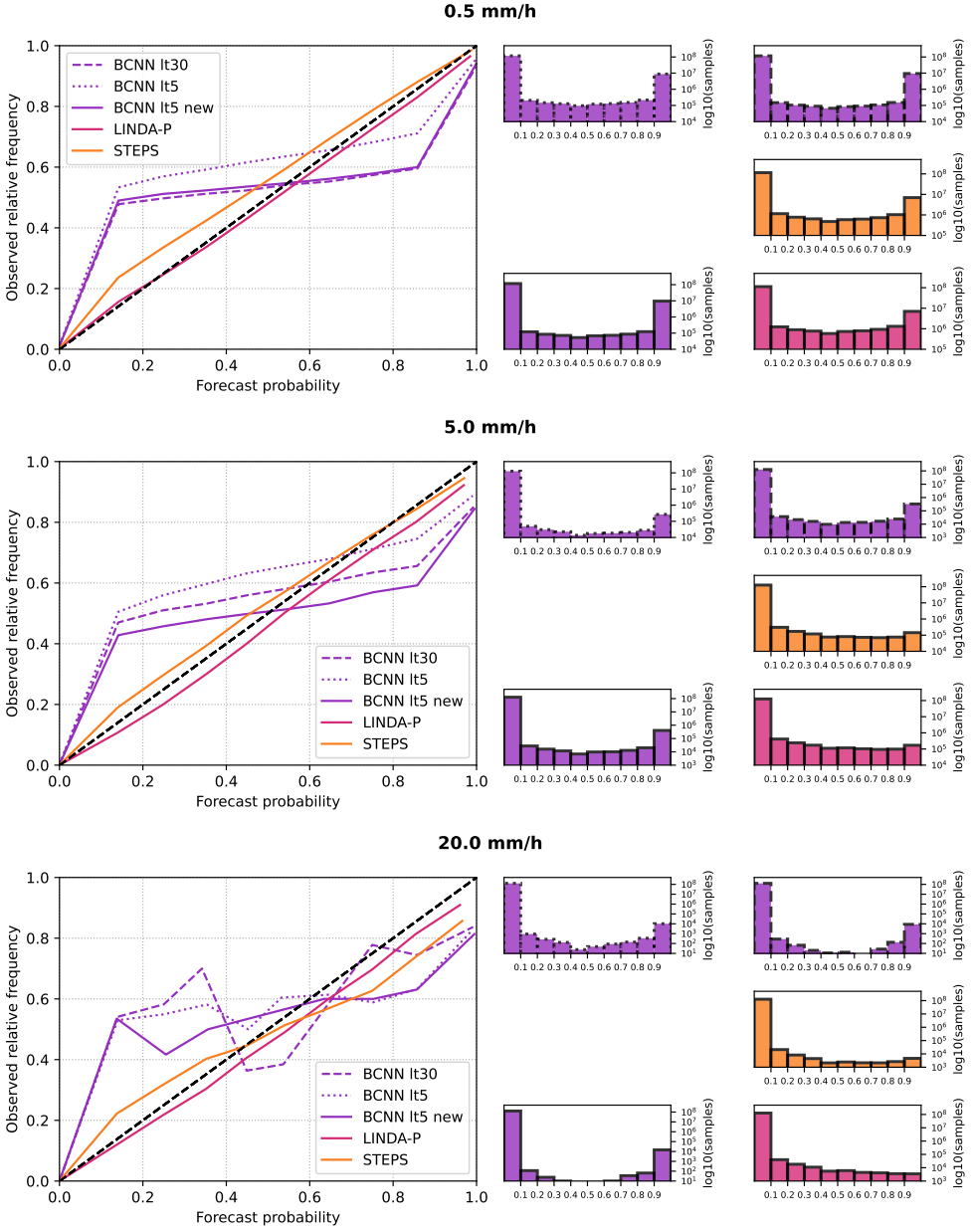


Figure 4.16: Reliability diagrams and sharpness histogram for one-hour exceedance probabilities at 0.5, 5.0, and 20.0 mm/h precipitation thresholds.

Chapter 5

Discussion

5.1 Critical Evaluation of Results

BCNN ensemble means produced acceptable deterministic skill, with properties in line with the RainNet baseline [8] and other CNN based models, with characteristic blurring, tendency to be biased or unstable at further leadtimes, difficulties in skillfully nowcasting high rainrate thresholds, but high success at lower thresholds and very good MAE. It should be kept in mind that achieving high predictive skill both with low threshold/MAE and high threshold is a difficult problem which is the focus of active research, and is related in the Deep Learning approach to the data imbalance problem of strong precipitation. This means that failure to predict strong precipitation is related to those events representing only a small part of the training data, while being of high importance. Trying to accomodate this by means of event importance weighting is a possibility, but this weighting might easily lead to the network performing worse on all non-extreme cases.

Interestingly, `lt30` training after `lt5` pre-training not only did not improve, but worsened the predictive skill with Gaussian NLL loss. In unrelated experiments to this thesis, `lt30` used as a standalone has shown much better skill than `lt5` training using the LogCosh loss function, owning to a more stable iterative process. One hypothesis why the scheme didn't work out in the present case is that the network should not have been pre-trained with `lt5` but rather directly trained with `lt30`. One related explanation could be that the non-improvement might be due to the constant Gaussian NLL σ_ℓ data uncertainty parameter. Here with `lt5` training the parameter could indeed be assumed constant for each input to some degree, but it can be easily seen why iterative re-plugging in of the outputs breaks that assumption, as more uncertainty is brought into the inputs. However, even the assumption of con-

stancy with `lt5` training is not really valid as aleatoric uncertainty strongly dependent on pixel position and rainrate values. For example places far from rain front have low uncertainty while convective cells have big uncertainty.

From a probabilistic nowcasting point of view, BCNN did not live up to expectations. Only CRPS is relatively close to baselines, but all other metrics show that BCNN makes relatively unskillful predictions compared to STEPS and LINDA-P. Rank histograms show that predictive uncertainty is most likely severely under-estimated. Reliability diagrams demonstrate that their predicted exceedance probabilities are not very reliable, and thus can not be trusted. ROC curves on the other hand show that the discriminative power of BCNN is also considerably worse than baseline, keeping in mind that the ROC curves suffer from too little probability categories, a subject which will be covered below. These results highlight the fact that in its current form, BCNN is not yet a useful probabilistic nowcasting method.

Looking at ensembles and rank histograms it is noticeable that ensemble variability seems low. This might be related to hyperparameters or more generally to the stochastic model as will be expanded on below, or it might have something to do with the small-scale detail loss occurring, as quick disappearance of them leads to only slowly evolving large-scale features remaining, which might undermine the variability at further leadtimes. If details were preserved such as in generative model based nowcasting methods, it could help assuming that is indeed part of the problem.

5.2 Validity of Assumptions and Experiments

The way that hyperparameter optimization was carried out in this work is a possible source of problems, potentially having led to sub-optimal solutions. This and other validity issues will be covered in this section.

5.2.1 SVI Issues

When it comes to the Variational Inference method employed, there might have been unguarded assumptions regarding the prior and posterior families chosen. In particular, the problem in question might have benefited from a more careful choice of those families based on physical constraints and known behavior of precipitation. Related to this matter, it is known that Variational approximations tend to underestimate uncertainty of learned distributions. [11, 39]. However, it is hard to say if this is the only factor at play here when observing the rank histograms and reliability diagrams of BCNN models. As their rank histogram shape is generally concave compared to models based

on stochastic perturbations, the variability of ensembles is indeed less than that of observations most pronouncedly in variational models, as one might expect.

Nevertheless, the magnitude of the effect leaves the possibility that some parts of the model might have been non-optimal for the task, even within the SVI framework. The reason for this is that in part of preliminary experiments performed, uncertainty estimates for BCNN precursors were way more reliable than in current experiments. After performing the experiments, it was realized that one hyperparameter that was different and left un-optimized here is the initialization value of parameter posterior scales, which was set to $1e^{-3}$ here and $1e^{-2}$ in these preliminary experiments. Setting the parameter too high led to difficulties for the model to converge and unstable predicted rainrates, but while it was also known that setting it lower leads to initially smaller uncertainty estimates, no attention was paid while performing experiments to whether this might affect the asymptotic behavior of the estimates. Looking at Figure 4.2, it is seen that scales only deviate from their initial values by a factor of two to three at maximum, hinting at the fact that asymptotic marginal uncertainty estimates may very well be connected to the initial scale of parameters.

5.2.2 Likelihood Cost and Uncertainty

It is possible that the magnitude of the Gaussian NLL loss observational noise parameter σ_ℓ influenced the predictive uncertainty estimates. With regards to this parameter, optimizing for it using conventional RainNet was questionable, because the interactions between the complexity and likelihood terms of the ELBO are not thus taken into account, and the choice is only based on deterministic skill, which is only a small part and sometimes in contradiction with other aspects of what an ensemble probabilistic model should aim for. 10^{-4} is also much smaller than the realistic average uncertainty in data in the value range of interest, i.e. heavy rainfall. This uncertainty is expressed in terms of (log-transformed) rainrate (mm/h) and the data has a resolution expressed in reflectivity of 0.5 dBZ, which has for effect that the resolution of the rainrate data is dependent on rain intensity. This calls for non-constant data dependent, e.g. heteroscedastic uncertainties. Another facet of the model making the homoscedasticity assumption questionable is the iterative prediction scheme. This is because the outputs re-fed into the network will inherently have higher aleatoric uncertainty than original inputs, as model (epistemic) uncertainty of previous steps increments data uncertainty of next steps. Also, smaller-scale features often including high-intensity precipitation, have worse predictability and will suffer from higher

uncertainty than large-scale features.

5.2.3 Input Data

Concerning the input data, the downsampling by a factor of two performed for computational performance reasons might have been detrimental for the validity of the verification of small-scale and especially high-intensity precipitation. There are two factors at play here. One is that because of the scale-dependence of precipitation lifetime, NN based nowcasting model performance might not scale the same way when changing scale as extrapolation-based classical models, which might make interpreting the performance obtained at bigger scales more difficult. The other factor is that because average pooling is used for downsampling, higher rainrates get diluted, an effect exacerbated by the on average small spatial extent of intense precipitation. Hence, downsampled performance may not necessarily extrapolate to higher spatial resolutions. This is important because in the end, developed precipitation nowcasting models are usually to be used with input data having a grid resolution of 500 meters to one kilometer.

In addition to the downsampling issue, performing the train/validation/test split using six hour blocks might not have been enough to discard all harmful temporal correlations between different time-series, as input images from the end of one block are still correlated to images at the start of the next block, which is part of another split. Thus, even though the correlations are reduced, they do not disappear and the validation and test split performance is still over-estimated for Deep Learning models. A more robust split would have been using non-successive days as units.

5.2.4 Verification Experiments

RAPSD results suffer from potentially spurious gain in small wavelength power at 120 minutes. This might have something to do with the OR-masking of precipitation reducing too much the area used, or it might be some other unknown effect or mistake in the calculations, which is why the credibility of RAPSD results here is doubtful.

ROC curves of Figure 4.14 in probabilistic verification suffer from too few probability categories, as the curves exhibit apparent discontinuity, while they should approximately follow a bi-normal distribution [37]. This undermines their accuracy for determining forecast accuracy and resolution.

5.3 Directions for Further Work

First and foremost, further work should aim to produce more reliable uncertainty estimates in the current framework. This could be approached by first examining the effect on initial posterior scales on ensemble variance. Secondly, BCNN would have to be trained and tested using non-downscaled 1km grid resolution reflectivity composites. If these do not resolve most current issues and more time and resources are available, it could be possible to implement other methods for uncertainty estimation using NN and compare them to Bayesian NN with Variational Inference. Different priors, maybe more informed ones or some with varying degrees of strictness, perhaps even non identically distributed among parameters, could be tried to see if that improves predictive skill.

Some other simple tricks that might improve performance of BCNN are pre-learning the parameter means as point estimates and using the Bayesian approach to only learn posterior scales while keeping the means fixed. Assuming a non-biased deterministic model, this could make the learning process much more efficient, raising the quality of uncertainty estimates. Another trick would be to take a pre-trained BCNN model underestimating uncertainty, and add post-processing to the predictions, aiming to calibrate the bias and add stochastic noise for increasing predicted uncertainty. Lastly, despite not improving model performance, weight pruning could be tried for reducing computational cost of making nowcasts. Blundell et al. [12] show that Bayesian Neural Networks with a high number of parameters are very sparse thus and amenable to extensive weight pruning without much loss in skill. Although that study concentrates on fully-connected Neural Networks, it is worth considering if skillful pruning could be efficiently implemented in CNN architectures too, as Shridhar et al. [64] seems to have experienced success doing that. Looking at a completely different direction, it might be a good idea to try and implement BNN versions of other established functional architectures, such as ConvLSTM or TrajGRU.

Despite being one very interesting research direction, decomposition of predictive uncertainty into aleatoric and epistemic uncertainties is not performed, as it was deemed out of scope for this work. Indeed, trying to model the uncertainty the radar image inputs is a separate problem which is not so much of interest in the domain of radar-based precipitation, as so much more can be done to improve performance by ameliorating radar scan fidelity, preprocessing, and choice. Despite of this, one could still view aleatoric uncertainty as a proxy to non-predictability of precipitation given starting conditions represented by the input frames. This could in the case

of the iterative models presented at least serve as a metric to quantify the uncertainty created by feeding back in previous model outputs and could potentially allow the compound aleatoric and epistemic ensemble spread to better match the true distribution of the data. If modeling the uncertainty as data dependent and using heteroscedastic Gaussian likelihood, it would be interesting to take the same approach as Kendall and Gal [29] and learn heteroscedastic uncertainties as a separate output channel. This uncertainty learning would eliminate the hairy problem of choosing a homoscedastic data uncertainty value a priori, and possibly make for more skillful probabilistic nowcasts. How the separate channel aleatoric uncertainty would be integrated with the ensembles for predictions is still an unsolved problem.

Chapter 6

Conclusions

In this work, Bayesian Convolutional Neural Network (BCNN) model variants were formulated and developed in an attempt to produce skillful and reliable ensemble based probabilistic precipitation nowcasts. It was shown that BCNN can be used to produce both deterministic and probabilistic precipitation nowcasts, and that hyperparameters such as the prior distribution posed on weights can have a big impact on the performance of the network and must be chosen wisely. Two facets of BCNN were evaluated. Firstly the skill of its ensemble means used as point estimates for nowcasting precipitations, and secondly the quality of its uncertainty estimates were evaluated to measure the fitness of the model for deterministic and probabilistic nowcasting. These facets were compared to existing baseline models, both extrapolation based and Neural Network based.

BCNN ensemble averages performed acceptably as deterministic precipitation nowcasts, exhibiting many of the known distinctive features of Neural Network based nowcasts, with both good and bad aspects. Judging from the results, the regularizing effect of the Bayesian approach seems to have worked out to at least some degree. From a probabilistic nowcast point of view on the other hand, the skill of BCNN was poor, characterized by a striking lack of sample diversity, unreliability and low discriminatory skill at all precipitation exceedance probability thresholds studied.

Some parts of the model selection and training processes had their problems, leading to a high margin for improvement capacity with possibly little effort, especially in the realm of making reliable uncertainty estimates. Still, these issues are diverse and have unclear roots, meaning that although they might resolve easily, this is not guaranteed.

The hopeful prospects in this work are that the developed models are indeed fast at making inference while making great use of the capacity of Neural Networks to fit nonlinear patterns, and most importantly do so while

providing an early iteration of uncertainty estimates. The current model has much potential for improvement as outlined in the Discussion Chapter 5. Although it is not yet ready for operative use, future approaches to probabilistic nowcasting using Neural Networks, regardless of whether or not they are based on this work, may very well attain sufficient proficiency for that purpose, as there is an ever-increasing demand and need for skillful probabilistic precipitation nowcasting methods.

Chapter 7

References

[1] SVI Part IV: Tips and Tricks - Pyro Tutorials 1.8.1 documentation. URL `https://pyro.ai/examples/svi_part_iv.html`. Online, accessed July 27, 2022.

[2] Next Generation Weather Radar (NEXRAD), sep 2020. URL `https://www.ncei.noaa.gov/products/radar/next-generation-weather-radar`. Online, accessed July 27, 2022.

[3] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. TensorFlow: A System for Large-Scale Machine Learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[4] Shreya Agrawal, Luke Barrington, Carla Bromberg, John Burge, Cenk Gazen, and Jason Hickey. Machine Learning for Precipitation Nowcasting from Radar Images. December 2019.

[5] C Alexander, David C Dowell, Ming Hu, Joseph Olson, Tanya Smirnova, Therese Ladwig, Steve Weygandt, Jaymes S Kenyon, Eric James, Haidao Lin, et al. Rapid refresh (RAP) and high resolution rapid refresh (HRRR) model development. In *100th American Meteorological Society Annual Meeting*. AMS, 2020.

[6] Tage Andersson and Karl-Ivar Ivarsson. A Model for Probability Nowcasts of Accumulated Precipitation Using Radar. *Journal of Applied Meteorology and Climatology*, 30(1):135–141, January 1991.

[7] Georgy Ayzel. Rainnet: a convolutional neural network for radar-based

precipitation nowcasting, 2020. URL `https://github.com/hydrogo/rainnet`. Online, accessed July 27, 2022.

[8] Georgy Ayzel, Tobias Scheffer, and Maik Heistermann. RainNet v1.0: a convolutional neural network for radar-based precipitation nowcasting. *Geoscientific Model Development*, 13(6):2631–2644, 2020.

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

[10] Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D. Goodman. Pyro: Deep Universal Probabilistic Programming. *Journal of Machine Learning Research*, 2018.

[11] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[12] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight Uncertainty in Neural Networks. In *International conference on machine learning*, pages 1613–1622. PMLR, 2015.

[13] Neill E. Bowler, Clive E. Pierce, and Alan W. Seed. STEPS: A probabilistic precipitation forecasting scheme which merges an extrapolation nowcast with downscaled NWP. *Quarterly Journal of the Royal Meteorological Society*, 132(620):2127–2155, 2006.

[14] Wray Buntine. Bayesian back-propagation. *Complex systems*, 5:603–643, 1991.

[15] Qing Cao, Michael Knight, Michael Frech, and Theodor Mammen. Measurement uncertainty and system assessment of weather radar network in Germany. In *2016 IEEE Radar Conference (RadarConf)*, pages 1–5, May 2016.

[16] John Denker and Yann LeCun. Transforming Neural-Net Output Levels to Probability Distributions. In *Advances in Neural Information Processing Systems*, volume 3. Morgan-Kaufmann, 1990.

[17] John Denker, Daniel Schwartz, Ben Wittner, Sara Solla, Richard Howard, Lawrence Jackel, and John Hopfield. Large automatic learning, rule extraction, and generalization. *Complex Systems*, 1(5):877–922, 1987.

[18] Michael Dixon and Gerry Wiener. Titan: Thunderstorm identification, tracking, analysis, and nowcasting a radar-based methodology. *Journal of atmospheric and oceanic technology*, 10(6):785–797, 1993.

[19] Frédéric Fabry. *Radar Meteorology: Principles and Practice*. Cambridge University Press, March 2018.

[20] William Falcon and The PyTorch Lightning team. PyTorch Lightning, mar 2019. URL `https://github.com/Lightning-AI/lightning`. Online, accessed July 27, 2022.

[21] Neil I Fox and Christopher K Wikle. A bayesian quantitative precipitation nowcast scheme. *Weather and forecasting*, 20(3):264–275, 2005.

[22] Gabriele Franch, Valerio Maggio, Luca Coviello, Marta Pendesini, Giuseppe Jurman, and Cesare Furlanello. TAASRAD19, a high-resolution weather radar reflectivity dataset for precipitation nowcasting. *Scientific Data*, 7(1):234, July 2020.

[23] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1050–1059. PMLR, June 2016.

[24] Urs Germann and Isztar Zawadzki. Scale-dependence of the predictability of precipitation from continental radar images. part i: Description of the methodology. *Monthly Weather Review*, 130(12):2859–2873, 2002.

[25] Alex Graves. Practical Variational Inference for Neural Networks. In *Advances in neural information processing systems*, volume 24. NIPS, 2011.

[26] GE Hinton and Drew van Camp. Keeping Neural Networks Simple by Minimizing the Description Length of the Weights. In *Proceedings of COLT-93*, pages 5–13, 1993.

[27] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial Attention in Multidimensional Transformers. *ArXiv preprint arXiv:1912.12180*, December 2019.

[28] Robin J. Hogan, Christopher A. T. Ferro, Ian T. Jolliffe, and David B. Stephenson. Equitability Revisited: Why the "Equitable Threat Score" Is Not Equitable. *Weather and Forecasting*, 25(2):710–726, April 2010.

[29] Alex Kendall and Yarin Gal. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In *Advances in neural information processing systems*, volume 30. NIPS, October 2017.

[30] Diederik P. Kingma, Tim Salimans, and Max Welling. Variational Dropout and the Local Reparameterization Trick. In *Advances in neural information processing systems*, volume 28. NIPS, December 2015.

[31] Armen Der Kiureghian and Ove Ditlevsen. Aleatory or epistemic? Does it matter? *Structural Safety*, 31(2):105–112, March 2009.

[32] Yongchan Kwon, Joong-Ho Won, Beom Joon Kim, and Myunghee Cho Paik. Uncertainty quantification using Bayesian neural networks in classification: Application to biomedical image segmentation. *Computational Statistics & Data Analysis*, 142:106816, February 2020.

[33] Jussi Leinonen, Dmitri Moisseev, Matti Leskinen, and Walter A. Petersen. A Climatology of Disdrometer Measurements of Rainfall in Finland over Five Years with Implications for Global Radar Observations. *Journal of Applied Meteorology and Climatology*, 51(2):392–404, 2012.

[34] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, volume 81, pages 674–679, August 1981. Vancouver, British Columbia.

[35] David JC MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

[36] JS Marshall and WM Palmer. The size distribution of raindrops. *Journal of Atmospheric Sciences*, 5:165–166, 1948.

[37] Ian Mason. A model for assessment of weather forecasts. *Aust. Meteor. Mag*, 30(4):291–303, 1982.

[38] Rowan McAllister, Yarin Gal, Alex Kendall, Mark van der Wilk, Amar Shah, Roberto Cipolla, and Adrian Weller. Concrete Problems for Autonomous Vehicle Safety: Advantages of Bayesian Deep Learning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 4745–4753, 2017.

[39] Thomas P Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2001.

[40] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard Lewis, and Satinder Singh. Action-Conditional Video Prediction using Deep Networks in Atari Games. *Advances in neural information processing systems*, 28, December 2015.

[41] Shigenori Otsuka, Gulanbaier Tuerhong, Ryota Kikuchi, Yoshikazu Kitano, Yusuke Taniguchi, Juan Jose Ruiz, Shinsuke Satoh, Tomoo Ushio, and Takemasa Miyoshi. Precipitation Nowcasting with Three-Dimensional Space - Time Extrapolation of Dense and Frequent Phased-Array Weather Radar Observations. *Weather and Forecasting*, 31(1): 329–340, February 2016.

[42] Xiang Pan, Yinghui Lu, Kun Zhao, Hao Huang, Mingjun Wang, and Haonan Chen. Improving Nowcasting of Convective Development by Incorporating Polarimetric Radar Variables Into a Deep-Learning Model. *Geophysical Research Letters*, 48(21):e2021GL095302, 2021.

[43] Lutz Prechelt. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pages 55–69. Springer, 1998.

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

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

[46] Seppo Pulkkinen, V. Chandrasekar, Annakaisa von Lerber, and Ari-Matti Harri. Nowcasting of Convective Rainfall Using Volumetric Radar Observations. *IEEE Transactions on Geoscience and Remote Sensing*, 58(11):7845–7859, November 2020.

[47] Seppo Pulkkinen, V. Chandrasekar, and Tero Niemi. Lagrangian Integro-Difference Equation Model for Precipitation Nowcasting. *Journal of Atmospheric and Oceanic Technology*, 38(12):2125–2145, December 2021.

[48] Chandrasekar Radhakrishnan and V Chandrasekar. Casa prediction system over dallas–fort worth urban network: Blending of nowcasting

and high-resolution numerical weather prediction model. *Journal of Atmospheric and Oceanic Technology*, 37(2):211–228, 2020.

[49] Suman Ravuri, Karel Lenc, Matthew Willson, Dmitry Kangin, Remi Lam, Piotr Mirowski, Megan Fitzsimons, Maria Athanassiadou, Sheleem Kashem, Sam Madge, Rachel Prudden, Amol Mandhane, Aidan Clark, Andrew Brock, Karen Simonyan, Raia Hadsell, Niall Robinson, Ellen Clancy, Alberto Arribas, and Shakir Mohamed. Skilful precipitation nowcasting using deep generative models of radar. *Nature*, 597(7878): 672–677, September 2021.

[50] R. E. Rinehart and E. T. Garvey. Three-dimensional storm motion detection by conventional weather radar. *Nature*, 273(5660):287–289, May 1978.

[51] Hippolyt Ritter and Theofanis Karaletsos. TyXe: Pyro-based Bayesian neural nets for Pytorch. *arXiv preprint arXiv:2110.00276*, 2021.

[52] Nigel M. Roberts and Humphrey W. Lean. Scale-Selective Verification of Rainfall Accumulations from High-Resolution Forecasts of Convective Events. *Monthly Weather Review*, 136(1):78–97, January 2008.

[53] Soorok Ryu, Geunsu Lyu, Younghae Do, and GyuWon Lee. Improved rainfall nowcasting using Burgers' equation. *Journal of Hydrology*, 581: 124140, February 2020.

[54] Hidetomo Sakaino. Spatio-Temporal Image Pattern Prediction Method Based on a Physical Model With Time-Varying Optical Flow. *IEEE Transactions on Geoscience and Remote Sensing*, 51(5):3023–3036, May 2013.

[55] Elena Saltikoff, Günther Haase, Laurent Delobbe, Nicolas Gaussiat, Maud Martet, Daniel Idziorek, Hidde Leijnse, Petr Novák, Maryna Lukach, and Klaus Stephan. OPERA the Radar Project. *Atmosphere*, 10(6):320, June 2019.

[56] Joseph T. Schaefer. The Critical Success Index as an Indicator of Warning Skill. *Weather and Forecasting*, 5(4):570–575, December 1990.

[57] Franziska Schmid, Yong Wang, and Abdoulaye Harou. Nowcasting Guidelines - A Summary. World Meteorological Institute Bulletin n$^{\underline{o}}$ 68, 2019.

[58] W Schmid, S Mecklenburg, and J Joss. Short-term risk forecasts of heavy rainfall. *Water science and technology*, 45(2):121–125, 2002.

[59] M. G. Schultz, C. Betancourt, B. Gong, F. Kleinert, M. Langguth, L. H. Leufen, A. Mozaffari, and S. Stadtler. Can deep learning beat numerical weather prediction? *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2194):20200097, April 2021.

[60] A. W. Seed. A Dynamic and Spatial Scaling Approach to Advection Forecasting. *Journal of Applied Meteorology and Climatology*, 42(3): 381–388, March 2003.

[61] Alan W. Seed, Clive E. Pierce, and Katie Norman. Formulation and evaluation of a scale decomposition-based stochastic precipitation nowcast scheme. *Water Resources Research*, 49(10):6624–6641, 2013.

[62] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In *Advances in neural information processing systems*, volume 28. NIPS, September 2015.

[63] Xingjian Shi, Zhihan Gao, Leonard Lausen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Deep Learning for Precipitation Nowcasting: A Benchmark and A New Model. In *Advances in neural information processing systems*, volume 30. NIPS, October 2017.

[64] Kumar Shridhar, Felix Laumann, and Marcus Liwicki. A Comprehensive guide to Bayesian Convolutional Neural Network with Variational Inference. *arXiv preprint arXiv:1901.02731*, January 2019.

[65] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1): 1929–1958, 2014.

[66] Andrew Staniforth and Jean Côté. Semi-Lagrangian Integration Schemes for Atmospheric Models - A Review. *Monthly Weather Review*, 119(9):2206–2223, September 1991.

[67] Casper Kaae Sønderby, Lasse Espeholt, Jonathan Heek, Mostafa Dehghani, Avital Oliver, Tim Salimans, Shreya Agrawal, Jason Hickey,

and Nal Kalchbrenner. MetNet: A Neural Weather Model for Precipitation Forecasting. *ArXiv preprint arXiv:2003.12140*, March 2020.

[68] Tishby, Levin, and Solla. Consistent inference of probabilities in layered networks: predictions and generalizations. In *International 1989 Joint Conference on Neural Networks*, pages 403–409 vol.2, 1989.

[69] Tanel Voormansik, Pekka J. Rossi, Dmitri Moisseev, Tarmo Tanilsoo, and Piia Post. Thunderstorm hail and lightning detection parameters based on dual-polarization Doppler weather radar data. *Meteorological Applications*, 24(3):521–530, 2017.

[70] Z. Wang, E.P. Simoncelli, and A.C. Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, pages 1398–1402, Pacific Grove, CA, USA, 2003. IEEE.

[71] Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse. Flipout: Efficient Pseudo-Independent Weight Perturbations on Mini-Batches. In *Sixth International Conference on Learning Representations*. ICLR, April 2018.

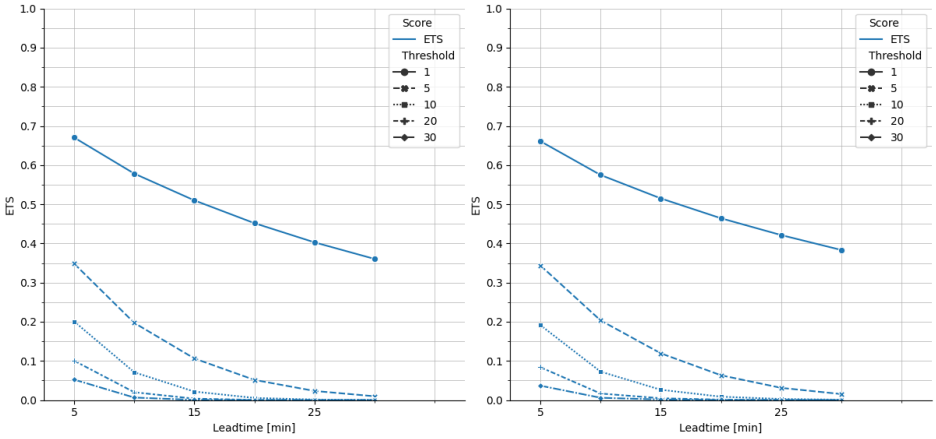
[72] Jian Yin, Zhiqiu Gao, and Wei Han. Application of a Radar Echo Extrapolation-Based Deep Learning Method in Strong Convection Nowcasting. *Earth and Space Science*, 8(8):e2020EA001621, 2021.

[73] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Loss Functions for Image Restoration With Neural Networks. *IEEE Transactions on Computational Imaging*, 3(1):47–57, March 2017.

Appendix A

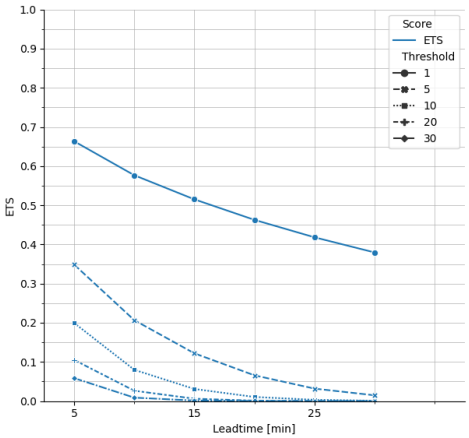
Additional Figures

A.1 Model Selection



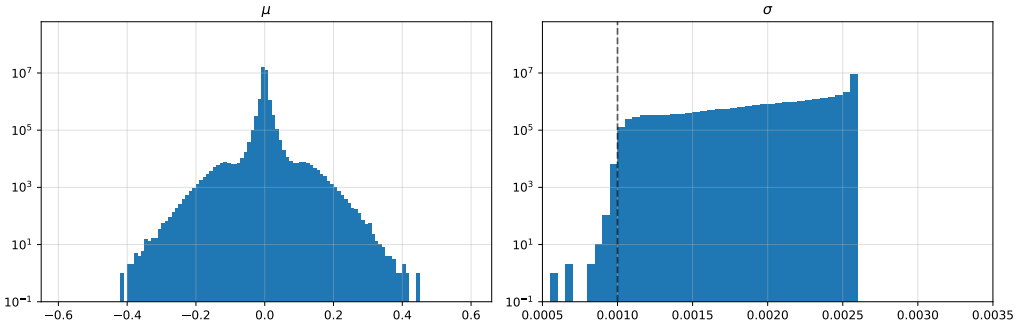
(a) $\sigma_\ell = 10^{-2}$

(b) $\sigma_\ell = 10^{-3}$

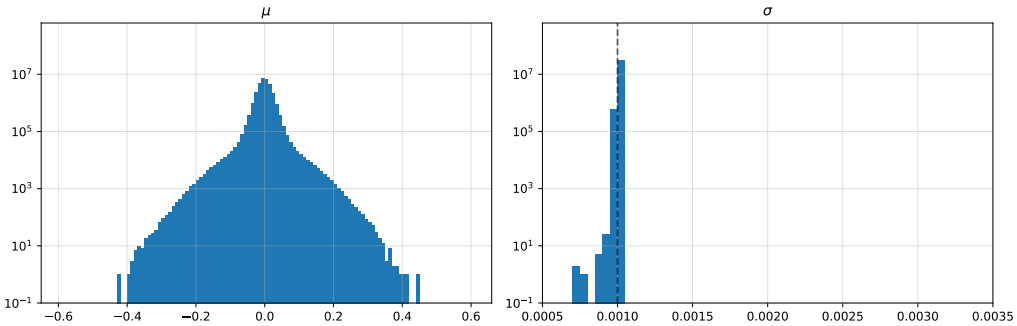


(c) $\sigma_\ell = 10^{-4}$

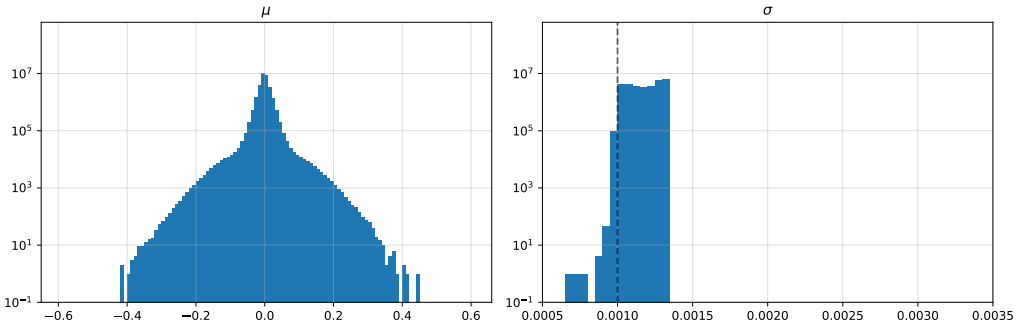
Figure A.1: ETS scores of RainNet over the validation set at model convergence used for choosing a σ_ℓ Gaussian NLL parameter. Thresholds are expressed in rainrate (mm/h.)



(a) Equal



(b) Blundell



(c) Epoch

Figure A.2: The relation of complexity cost weighting scheme to the parameter posterior distributions across the network. μ_q histograms describe the parameter mean distributions, whereas σ_q distributions describe the distribution of their standard deviations. The prior chosen is prior (4). The black dashed vertical lines indicate the initialization value for standard deviations.

A.2 Case Studies

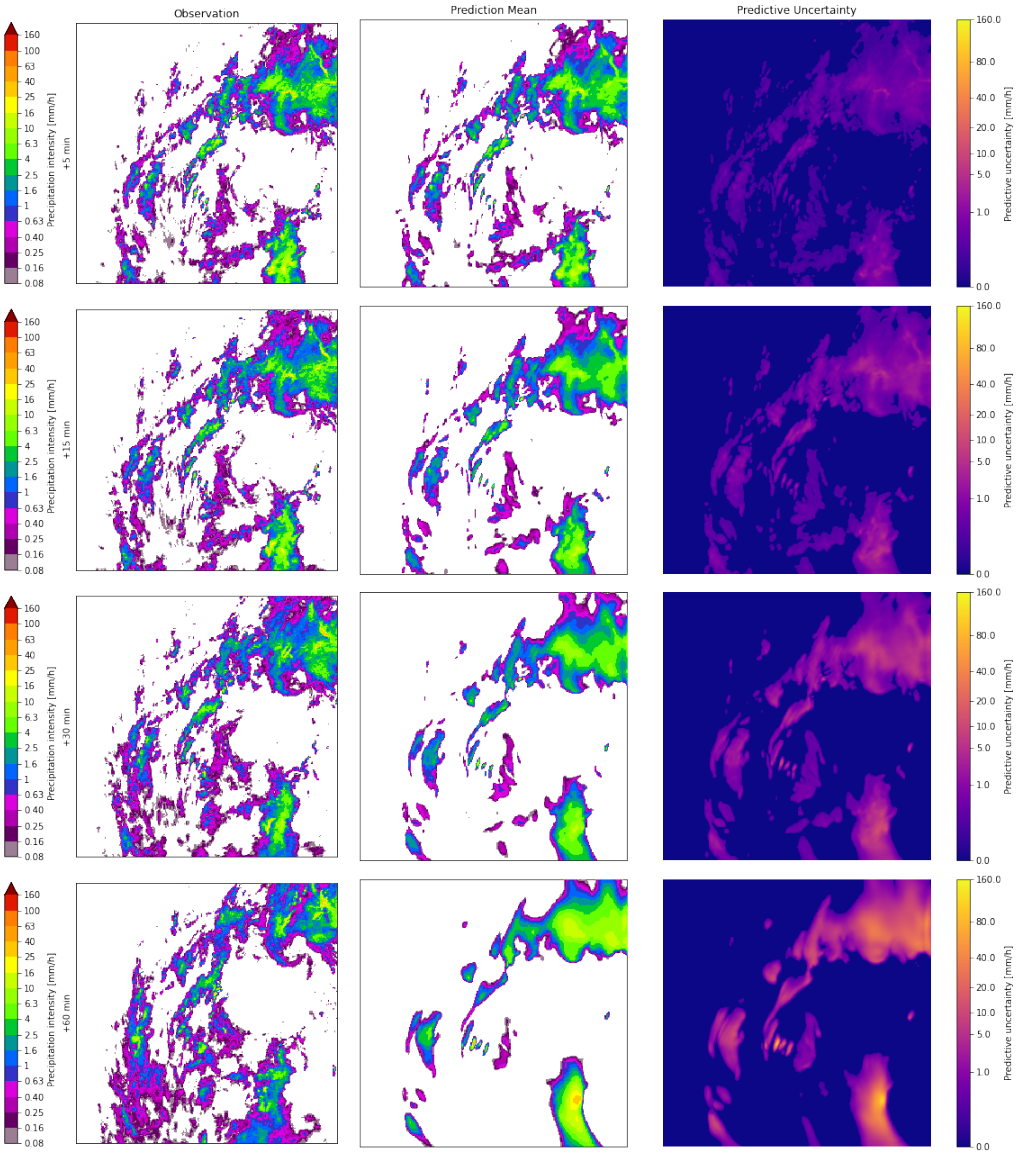


Figure A.3: Predictive mean and uncertainty (two standard deviations) for Case 1 on the 25th of May 2019 starting at 13:00 local Helsinki time for the `BCNN lt5 new` model, covering the area of southern Finland.

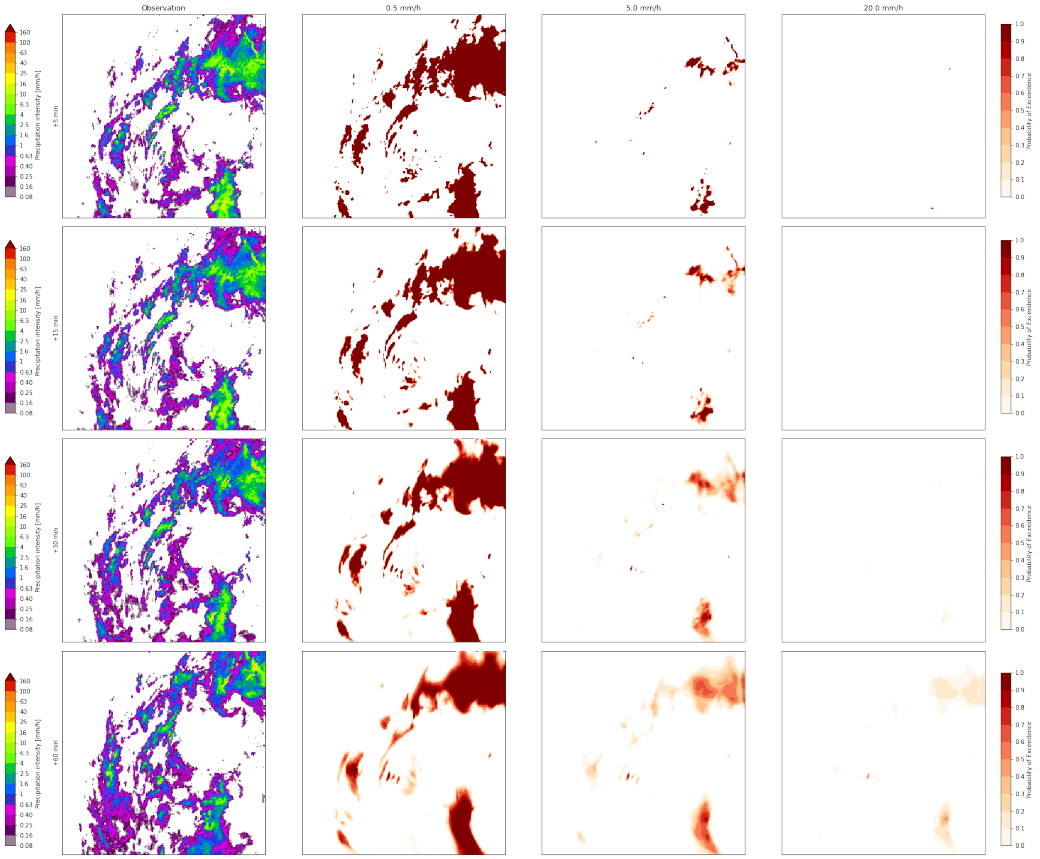


Figure A.4: 0.5, 5.0, and 20.0 mm/h exceedance probabilities for Case 1 on the 25th of May 2019 starting at 13:00 local Helsinki time for the `BCNN lt5 new` model, covering the area of southern Finland.

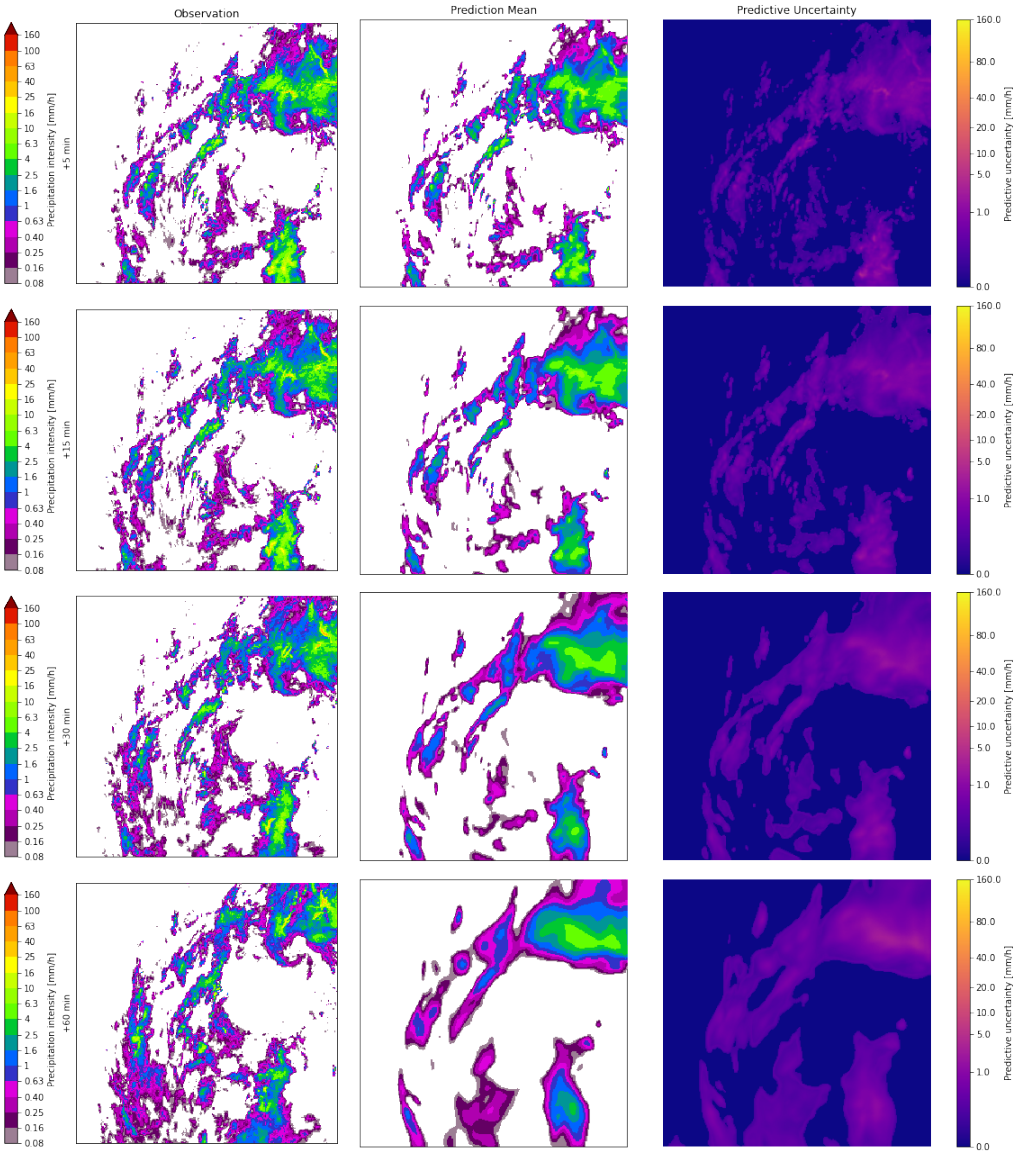


Figure A.5: Predictive mean and uncertainty (two standard deviations) for Case 1 on the 25th of May 2019 starting at 13:00 local Helsinki time for the `BCNN 1t30` model, covering the area of southern Finland.

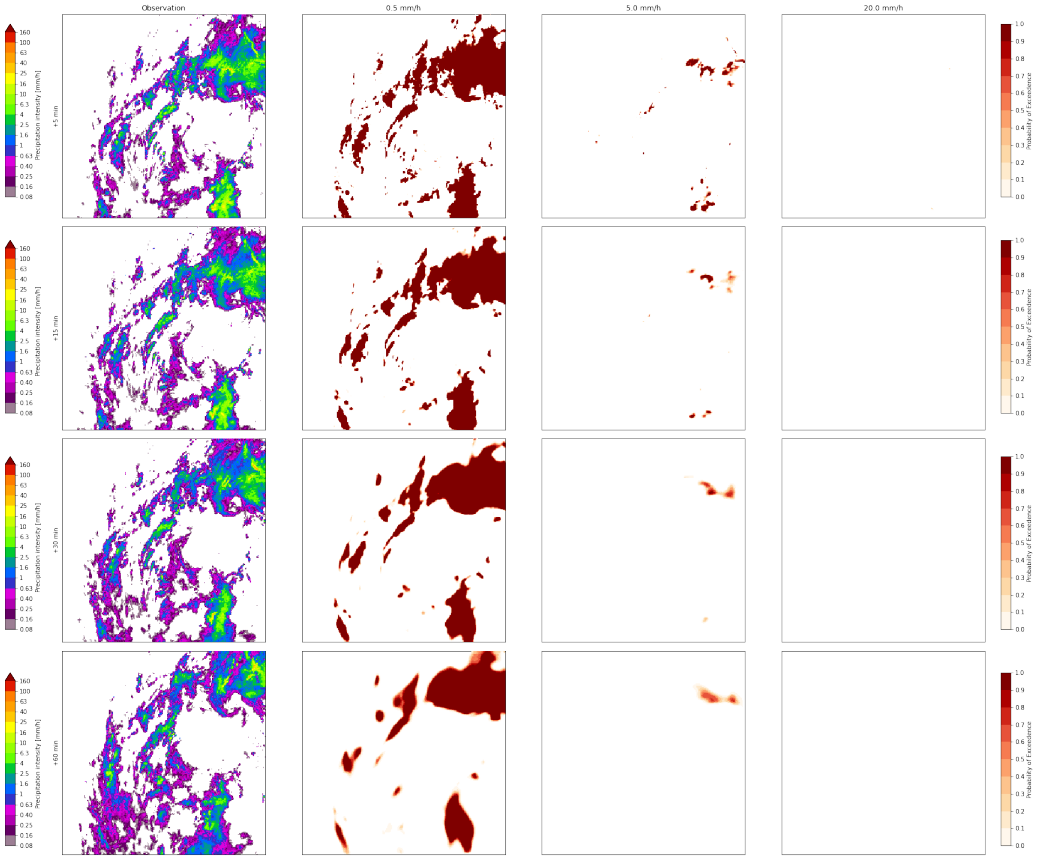


Figure A.6: 0.5, 5.0, and 20.0 mm/h exceedance probabilities for Case 1 on the 25th of May 2019 starting at 13:00 local Helsinki time for the `BCNN 1t30` model, covering the area of southern Finland.

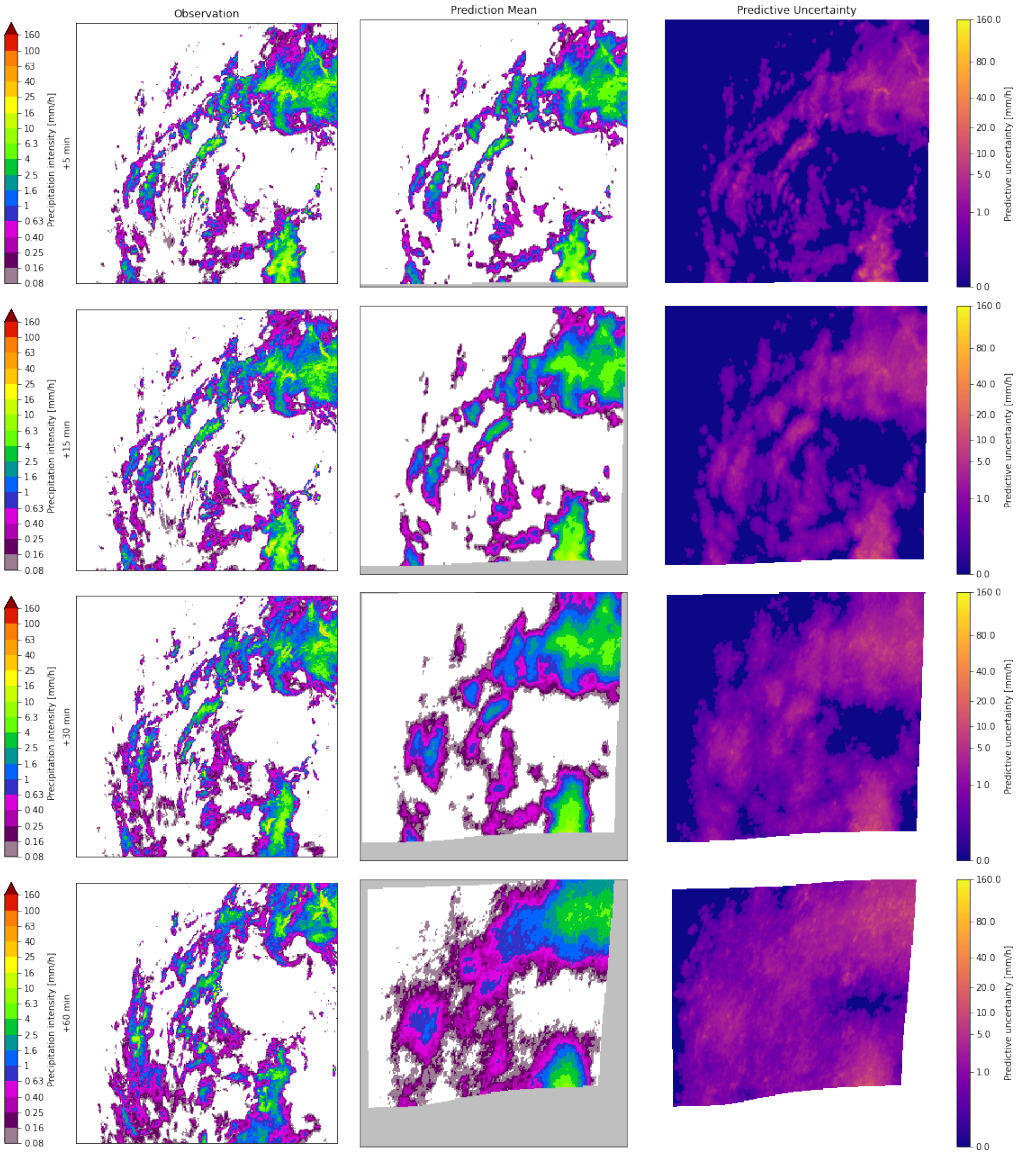


Figure A.7: Predictive mean and uncertainty (two standard deviations) for Case 1 on the 25th of May 2019 starting at 13:00 local Helsinki time for the baseline STEPS model, covering the area of southern Finland.

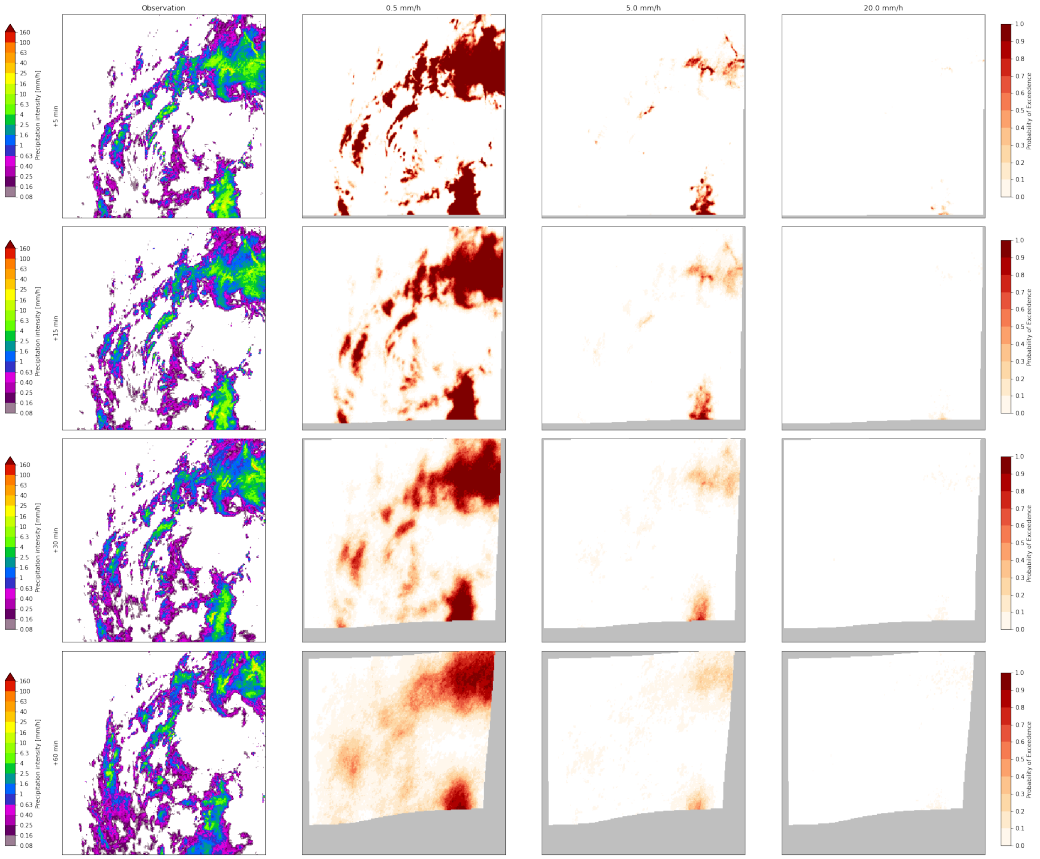
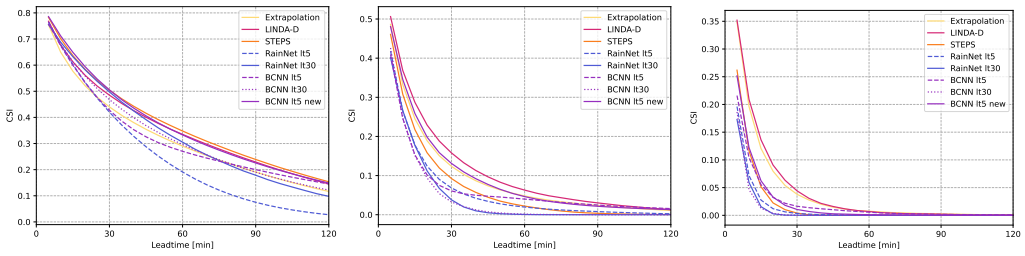


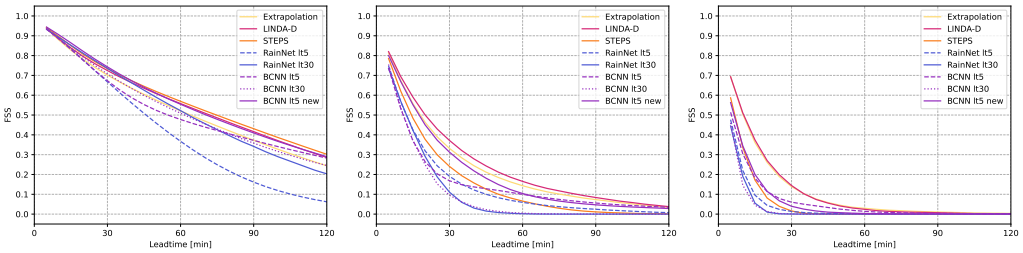
Figure A.8: 0.5, 5.0, and 20.0 mm/h exceedance probabilities for Case 1 on the 25th of May 2019 starting at 1PM for the baseline STEPS model, covering the area of southern Finland.

A.3 Deterministic Metrics



(a) 0.5 mm/h (b) 5.0 mm/h (c) 20.0 mm/h

Figure A.9: CSI scores



(a) 0.5 mm/h (b) 5.0 mm/h (c) 20.0 mm/h

Figure A.10: FSS scores (4km)

A.4 Probabilistic Metrics

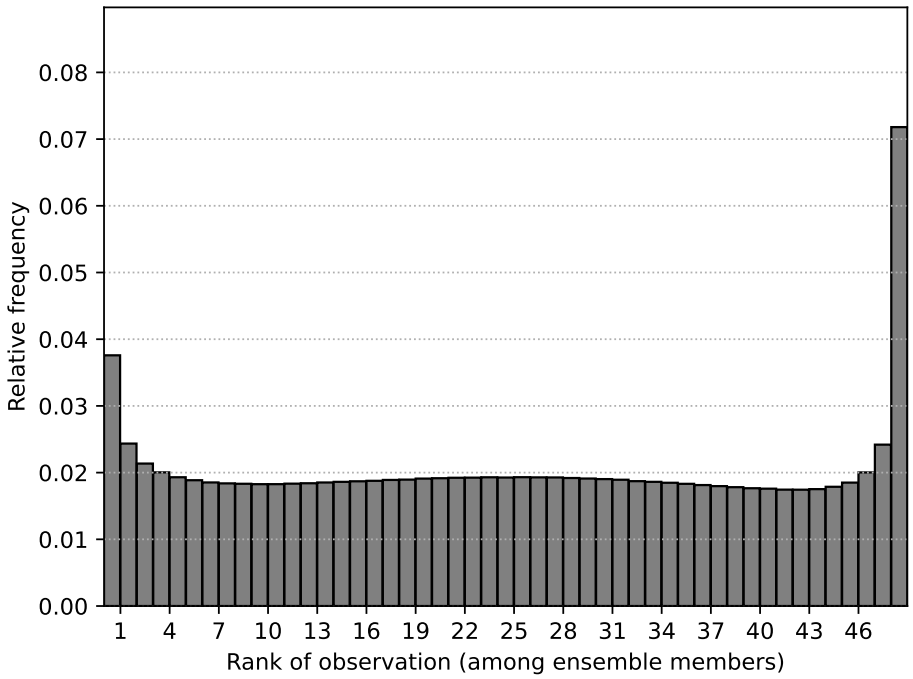


Figure A.11: Rank Histogram for LINDA-P model nowcasts at a one-hour leadtime, for precipitation events exceeding 0.1 mm/h