University of Potsdam

Cumulative Dissertation

Advancing radar-based precipitation nowcasting: an open benchmark and the potential of deep learning

by

Georgy Ayzel

Supervisors:
Maik Heistermann, Ph.D.
Prof. Dr. Tobias Scheffer

for the degree of
doctor rerum naturalium (Dr. rer. nat.)
in Geoecology
Institute of Environmental Science and Geography
Faculty of Science

submitted on November 19, 2020

defended on March 2, 2021

Advancing radar-based precipitation nowcasting: an open benchmark and the potential of deep learning

by Georgy Ayzel

Supervisor:	*Affiliation:*
Maik Heistermann, Ph.D. (*Reviewer*)	*University of Potsdam*
Co-Supervisor:	
Prof. Dr. Tobias Scheffer (*Reviewer*)	*University of Potsdam*
Mentor:	
Prof. Oliver Korup, Ph.D.	*University of Potsdam*
Assessment Committee:	
Prof. Dr. Axel Bronstert *(Chair)*	*University of Potsdam*
Prof. Dr. Bodo Bookhagen	*University of Potsdam*
Prof. Oliver Korup, Ph.D.	*University of Potsdam*
Prof. Dr.-Ing. Uwe Haberlandt (*Reviewer*)	*Leibniz University Hannover*

Publication-based dissertation submitted in fulfilment of the requirements for the degree of Doctor of Philosophy under the discipline of Geoecology in the Institute of Environmental Science and Geography Faculty of Science at the University of Potsdam.

Published online on the
Publication Server of the University of Potsdam:
https://doi.org/10.25932/publishup-50426
https://nbn-resolving.org/urn:nbn:de:kobv:517-opus4-504267

Declaration of Authorship

I, Georgy Ayzel, declare that this thesis titled, "Advancing radar-based precipitation nowcasting: an open benchmark and the potential of deep learning " and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at the University of Potsdam.
- Where any part of this dissertation has previously been submitted for a degree or any other qualification at the University of Potsdam, or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signature: ____

Date: ____

"*But, apparently, this is the way it happened in the world: when we really begin to understand something, we are too old to apply it to life, so it goes – wave after wave, generation after generation, and no one is able to learn anything from another.*"

Erich Maria Remarque. The Black Obelisk

Abstract

Precipitation forecasting has an important place in everyday life – during the day we may have tens of small talks discussing the likelihood that it will rain this evening or weekend. Should you take an umbrella for a walk? Or should you invite your friends for a barbecue? It will certainly depend on what your weather application shows.

While for years people were guided by the precipitation forecasts issued for a particular region or city several times a day, the widespread availability of weather radars allowed us to obtain forecasts at much higher spatiotemporal resolution of minutes in time and hundreds of meters in space. Hence, radar-based precipitation nowcasting, that is, very-short-range forecasting (typically up to 1 – 3 h), has become an essential technique, also in various professional application contexts, e.g., early warning, sewage control, or agriculture.

There are two major components comprising a system for precipitation nowcasting: radar-based precipitation estimates, and models to extrapolate that precipitation to the imminent future. While acknowledging the fundamental importance of radar-based precipitation retrieval for precipitation nowcasts, this thesis focuses only on the model development: the establishment of open and competitive benchmark models, the investigation of the potential of deep learning, and the development of procedures for nowcast errors diagnosis and isolation that can guide model development.

The present landscape of computational models for precipitation nowcasting still struggles with the availability of open software implementations that could serve as benchmarks for measuring progress. Focusing on this gap, we have developed and extensively benchmarked a stack of models based on different optical flow algorithms for the tracking step and a set of parsimonious extrapolation procedures based on image warping and advection. We demonstrate that these models provide skillful predictions comparable with or even superior to state-of-the-art operational software. We distribute the corresponding set of models as a software library, rainymotion, which is written in the Python programming language and openly available at GitHub (https://github.com/hydrogo/rainymotion). That way, the library acts as a tool for providing fast, open, and transparent solutions that could serve as a benchmark for further model development and hypothesis testing.

One of the promising directions for model development is to challenge the potential of deep learning – a subfield of machine learning that refers to artificial neural networks with deep architectures, which may consist of many computational layers. Deep learning showed promising results in many fields of computer science, such as image and speech recognition, or natural language processing, where it started to dramatically outperform reference methods.

The high benefit of using ”big data” for training is among the main reasons for that. Hence, the emerging interest in deep learning in atmospheric sciences is also caused and concerted with the increasing availability of data – both observational and model-based. The large archives of weather radar data provide a solid basis

for investigation of deep learning potential in precipitation nowcasting: one year of national 5-min composites for Germany comprises around 85 billion data points.

To this aim, we present RainNet, a deep convolutional neural network for radar-based precipitation nowcasting. RainNet was trained to predict continuous precipitation intensities at a lead time of 5 min, using several years of quality-controlled weather radar composites provided by the German Weather Service (DWD). That data set covers Germany with a spatial domain of 900 km $\times$ 900 km and has a resolution of 1 km in space and 5 min in time. Independent verification experiments were carried out on 11 summer precipitation events from 2016 to 2017. In these experiments, RainNet was applied recursively in order to achieve lead times of up to 1 h. In the verification experiments, trivial Eulerian persistence and a conventional model based on optical flow served as benchmarks. The latter is available in the previously developed rainymotion library.

RainNet significantly outperformed the benchmark models at all lead times up to 60 min for the routine verification metrics mean absolute error (MAE) and critical success index (CSI) at intensity thresholds of 0.125, 1, and 5 mm h^{-1}. However, rainymotion turned out to be superior in predicting the exceedance of higher intensity thresholds (here 10 and 15 mm h^{-1}). The limited ability of RainNet to predict high rainfall intensities is an undesirable property which we attribute to a high level of spatial smoothing introduced by the model. At a lead time of 5 min, an analysis of power spectral density confirmed a significant loss of spectral power at length scales of 16 km and below.

Obviously, RainNet had learned an optimal level of smoothing to produce a nowcast at 5 min lead time. In that sense, the loss of spectral power at small scales is informative, too, as it reflects the limits of predictability as a function of spatial scale. Beyond the lead time of 5 min, however, the increasing level of smoothing is a mere artifact – an analogue to numerical diffusion – that is not a property of RainNet itself but of its recursive application. In the context of early warning, the smoothing is particularly unfavorable since pronounced features of intense precipitation tend to get lost over longer lead times. Hence, we propose several options to address this issue in prospective research on model development for precipitation nowcasting, including an adjustment of the loss function for model training, model training for longer lead times, and the prediction of threshold exceedance.

The model development together with the verification experiments for both conventional and deep learning model predictions also revealed the need to better understand the source of forecast errors. Understanding the dominant sources of error in specific situations should help in guiding further model improvement. The total error of a precipitation nowcast consists of an error in the predicted location of a precipitation feature and an error in the change of precipitation intensity over lead time. So far, verification measures did not allow to isolate the location error, making it difficult to specifically improve nowcast models with regard to location prediction.

To fill this gap, we introduced a framework to directly quantify the location error. To that end, we detect and track scale-invariant precipitation features (corners) in radar images. We then consider these observed tracks as the true reference in order to evaluate the performance (or, inversely, the error) of any model that aims to predict the future location of a precipitation feature. Hence, the location error of a forecast

at any lead time ahead of the forecast time corresponds to the Euclidean distance between the observed and the predicted feature location at the corresponding lead time.

Based on this framework, we carried out a benchmarking case study using one year worth of weather radar composites of the DWD. We evaluated the performance of four extrapolation models, two of which are based on the linear extrapolation of corner motion; and the remaining two are based on the Dense Inverse Search (DIS) method: motion vectors obtained from DIS are used to predict feature locations by linear and Semi-Lagrangian extrapolation.

For all competing models, the mean location error exceeds a distance of 5 km after 60 min, and 10 km after 110 min. At least 25 % of all forecasts exceed an error of 5 km after 50 min, and of 10 km after 90 min. Even for the best models in our experiment, at least 5 percent of the forecasts will have a location error of more than 10 km after 45 min. When we relate such errors to application scenarios that are typically suggested for precipitation nowcasting, e.g., early warning, it becomes obvious that location errors matter: the order of magnitude of these errors is about the same as the typical extent of a convective cell. Hence, the uncertainty of precipitation nowcasts at such length scales – just as a result of locational errors – can be substantial already at lead times of less than 1 h. Being able to quantify the location error should hence guide any model development that is targeted towards its minimization. To that aim, we also consider the high potential of using deep learning architectures specific to the assimilation of sequential (track) data.

Last but not least, the thesis demonstrates the benefits of a general movement towards open science for model development in the field of precipitation nowcasting. All the presented models and frameworks are distributed as open repositories, thus enhancing transparency and reproducibility of the methodological approach. Furthermore, they are readily available to be used for further research studies, as well as for practical applications.

Zusammenfassung

Niederschlagsvorhersagen haben einen wichtigen Platz in unserem täglichen Leben. Und die breite Abdeckung mit Niederschlagsradaren ermöglicht es uns, den Niederschlag mit einer viel höheren räumlich-zeitlichen Auflösung vorherzusagen (Minuten in der Zeit, Hunderte von Metern im Raum). Solche radargestützten Niederschlagsvorhersagen mit sehr kurzem Vorhersagehorizont (1 – 3 Stunden) nennt man auch "Niederschlagsnowcasting." Sie sind in verschiedenen Anwendungsbereichen (z.B. in der Frühwarnung, der Stadtentwässerung sowie in der Landwirtschaft) zu einer wichtigen Technologie geworden.

Eine erhebliche Schwierigkeit in Modellentwicklung zum Niederschlagsnowcastings ist jedoch die Verfügbarkeit offener Softwarewerkzeuge und Implementierungen, die als Benchmark für den Entwicklungsfortschritt auf diesem Gebiet dienen können. Um diese Lücke zu schließen, haben wir eine Gruppe von Modellen auf der Grundlage verschiedener Tracking- und Extrapolationsverfahren entwickelt und systematisch verglichen. Es konnte gezeigt werden, dass die Vorhersagen dieser einen Skill haben, der sich mit dem Skill operationeller Vorhersagesysteme messen kann, teils sogar überlegen sind. Diese Benchmark-Modelle sind nun in Form der quelloffenen Software-Bibliothek rainymotion allgemein verfügbar (https://github.com/hydrogo/rainymotion).

Eine der vielversprechenden Perspektiven für die weitere Modellentwicklung besteht in der Untersuchung des Potenzials von "Deep Learning" – einem Teilgebiet des maschinellen Lernens, das sich auf künstliche neuronale Netze mit sog. "tiefen Architekturen" bezieht, die aus einer Vielzahl von Schichten (computational layers) bestehen können. Im Rahmen dieser Arbeit wurde daher RainNet entwickelt: ein Tiefes Neuronales Netz für radargestütztes Niederschlags-Nowcasting. RainNet wurde zunächst zur Vorhersage der Niederschlagsintensität mit einem Vorhersagehorizont von fünf Minuten trainiert. Als Datengrundlage dazu dienten mehrere Jahre qualitätskontrollierter Radarkompositprodukte des Deutschen Wetterdienstes (DWD).

RainNet übertraf die verfügbaren Benchmark-Modelle für Vorhersagezeiten bis zu 60 min in Bezug auf den Mittleren Absoluten Fehler (MAE) und den Critical Success Index (CSI) für Intensitätsschwellenwerte von 0.125, 1 und 5 mm h^{-1}. Allerdings erwies sich das das Benchmark-Modell aus dem Softwarepaket rainymotion bei der Vorhersage der Überschreitung höherer Intensitätsschwellen (10 und 15 mm h^{-1}) als überlegen. Die eingeschränkte Fähigkeit von RainNet zur Vorhersage hoher Niederschlagsintensitäten ist eine unerwünschte Eigenschaft, die wir auf ein hohes Maß an räumlicher Glättung durch das Modell zurückführen. Im Kontext der Frühwarnung ist die Glättung besonders ungünstig, da ausgeprägte Merkmale von Starkniederschlägen bei längeren Vorlaufzeiten tendenziell verloren gehen. In dieser Arbeit werden daher mehrere Optionen vorgeschlagen, um dieses Problem in der zukünftigen Forschung zur Modellentwicklung anzugehen.

Ein weiterer Beitrag dieser Arbeit liegt in der Quantifizierung einer spezifischen Fehlerquelle von Niederschlagsnowcasts. Der Gesamtfehler eines Nowcasts besteht aus einem Fehler in der vorhergesagten Lage eines Niederschlagsfeatures (Ortsfehler) sowie einem Fehler in der Änderung der Intensität eines Features über die Vorhersagezeit (Intensitätsfehler). Herkömmliche Verifikationsmaße waren bislang nicht in der Lage, das Ausmaß des Ortsfehlers zu isolieren. Um diese Lücke zu füllen, haben wir einen Ansatz zur direkten Quantifizierung des Ortsfehlers entwickelt. Mit Hilfe dieses Ansatzes wurde wir Benchmarking-Experiment auf Grundlage eines fünfminütigen DWD Radarkompositprodukts für das komplette Jahr 2016 umgesetzt. In diesem Experiment wurden vier Nowcasting-Modelle aus der rainymotion-Softwarebibliothek verwendet im Hinblick auf den Ortsfehler der Vorhersage verglichen. Die Ergebnisse zeigen, dass für alle konkurrierenden Modelle die Ortsfehler von Bedeutung sind: die Größenordnung dieser Fehler entspricht etwa der typischen Ausdehnung einer konvektiven Zelle oder einer mittelgroßen Stadt (5 – 10 km).

Insgesamt zeigt diese Arbeit die Vorteile eines ”Open Science”-Ansatzes für die Modellentwicklung im Bereich der Niederschlagsnowcastings. Alle vorgestellten Modelle und Modellsysteme stehen als offene, gut dokumentierte Repositorien zusammen mit entsprechenden offenen Datensätzen öffentlich zu Verfügung für, was die Transparenz und Reproduzierbarkeit des methodischen Ansatzes, aber auch die Anwendbarkeit in der Praxis erhöht.

Acknowledgements

I would like to express my sincere gratitude to the people who played an important role in this PhD journey.

To Maik, my supervisor, for his trust and support that helped me to keep my head above the water despite the high waves generated by life.

To Tobias, my co-supervisor, who did not let me lose my head in the clouds.

To Axel, for support in all forms and heartwarming get-togethers.

To the institutions that supported me – to Geo.X for the financial support and providing the environment for collaboration and learning; to ClimXtreme for the financial support on the last mile; to the Geoecology staff – Andi, Daniel, Silke, and Sabine – for all the help in the administrative and technical things.

To the German Weather Service and personally to Tanja Winterrath for providing radar data and support with its analysis.

To my working group – Arthur, Irene, Lisei, and Omar – for the fruitful discussions about work and everything that far beyond it.

To the Köhn-Reich family – Marcel, Lisei, and Kepa – for the support and the example of empathy I tried to learn from.

To my best friends – Sasha, Sanya, Diman, Lyokha, and their families – for the unconditional support, 24/7 availability in Telegram, and the best talks ever.

To my family, for being far away, but always right in the heart.

To Natasha, Mark, and Anna, for being always close and provided me the infinite source of strength and inspiration.

Спасибо вам от всей души!

Contents

List of Figures

Chapter 1

Introduction

How much will it rain within the next hour? The answer to this question could be given by precipitation nowcasting – a technique that provides precipitation forecasts at high spatiotemporal resolution and very short lead times. Typically, nowcasting operates at a spatial resolution from 10s to 1000s of meters and a temporal resolution from one to 15 min, while lead times do not exceed a few hours. This is achieved by utilizing heuristic models for the extrapolation of rain field motion and development, as observed by weather radar.

Nowcasts have become essential for a broad audience for planning various kinds of activities. Nowcasting applications installed in millions of smartphones worldwide help people to bring an umbrella or to organize a barbeque with friends. Furthermore, nowcasting has a strong professional relevance in many fields which require operational management, e.g., agriculture, or water management in urban environments (sewage control, underground operations). Nowcasts are particularly relevant in the context of early warning of heavy convective rainfall events and their corresponding impacts such as flash floods or landslides.

The state-of-the-art in nowcasting

Nowcasting is not the only option to predict the amount of precipitation in the next hour. Recent advances in numerical weather prediction (NWP) allow us to forecast atmospheric dynamics at a very high spatiotemporal resolution (Bauer et al., 2015; Fuhrer et al., 2018). Yet computational costs are typically prohibitive for the requirements of operational nowcasting applications with frequent update cycles (5–15 min). Furthermore, the parsimonious models of the heuristic extrapolation of rain field motion and development still appear to outperform NWP forecasts at very short lead times (Fig. 1.1; Berenguer et al., 2012; Germann et al., 2006; Jensen et al., 2015; Lin et al., 2005; Sun et al., 2014).

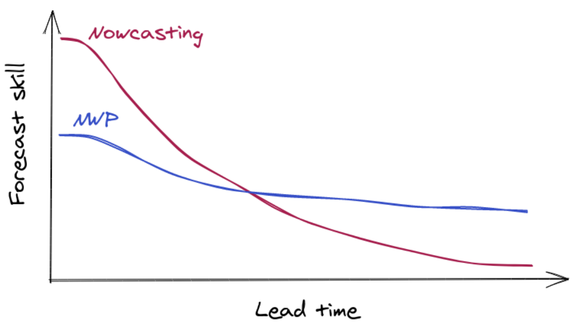

Figure 1.1: The schematic progression of the forecast skill in lead time for nowcasting and NWP models.

Today, many precipitation nowcasting systems are operational at regional or national scales, utilizing various radar products, algorithms, and blending techniques to provide forecasts up to 1–3 h, for example, ANC (Mueller et al., 2003), MAPLE (Germann and Zawadzki, 2002b), RADVOR (Winterrath et al., 2012), STEPS (Bowler et al., 2006), STEPS-BE (Foresti et al., 2016), and SWIRLS (Cheung and Yeung, 2012; Woo and Wong, 2017). Recently, nowcasting systems approached mobile and web applications making nowcasts available for billions of people whose devices have an internet connection.

The majority of these systems use models which are based on the assumption of Lagrangian persistence (Reyniers, 2008). They consist of two main computational steps: tracking and forecasting (extrapolation) (Fig. 1.2; Austin and Bellon, 1974). In the tracking step, we compute either a velocity field or the displacement of distinct features (e.g., storm cells) from a series of consecutive radar images. In the second step, we use that information to advect the most recent rain field, i.e., to displace it to the imminent future. For these two computational steps – tracking and extrapolation – the utilization of optical flow-based and semi-Lagrangian advection algorithms, respectively, is of the most common use (Bowler et al., 2004; Foresti et al., 2016; Germann and Zawadzki, 2002b; Reyniers, 2008).

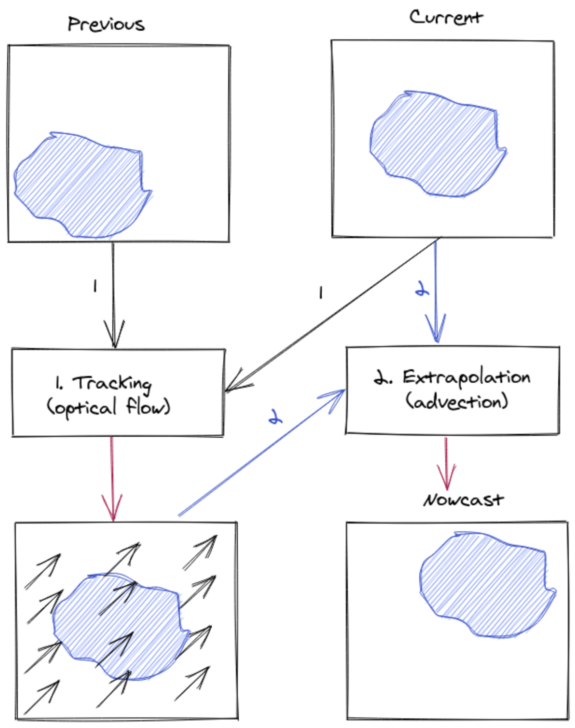

Figure 1.2: Conventional workflow for precipitation nowcasting.

The need for an open benchmark

For around two decades, techniques such as optical flow and semi-Lagrangian advection have been doing their best for state-of-the-art operational nowcasting systems worldwide. However, despite the abundance of applications and publications about different computational techniques for nowcasting applications, one may realize that a readily available benchmark model is still missing – a benchmark that is not only open-source, but also well-documented and reproducible with runtime environments and data.

Based on the (comparatively) long history of precipitation nowcasting, it is surprising why there is no available benchmark model except the most trivial one: Eulerian persistence. That is even more surprising since open-source implementations of fundamental optical flow algorithms (Brox et al., 2004; Bruhn et al., 2005b) have been around for up to 20 years – with the OpenCV library (https://opencv.org) just being the most widely known. Such libraries provide efficient implementations of various optical flow algorithms for a vast number of research and application contexts. To a lesser extent, it is relevant for open-source implementations of advection algorithms, which are not that well-developed in comparison to optical flow techniques due to the limited interest of the research community. OpenFOAM (https://openfoam.org/), pyro (https://pyro2.readthedocs.io), and fipy (https://www.ctcms.nist.gov/fipy) libraries are among the most known with a development history of over a decade. Yet none of these methods can be applied in the precipitation nowcasting context out of the box – without the need to address additional and specific challenges such as underlying assumptions and constraints of velocity fields, pre- and post-processing steps, or model parameterization.

In addition to benchmark models, we also need a set of benchmark (verification) metrics against which the value of the developed models can be measured. A large variety of verification metrics have been suggested in the literature (see, e.g., Baldwin and Kain, 2006; Ebert, 2008; Gilleland et al., 2010). Most of them, however, struggle with disentangling different sources of error: when we compare nowcast to observed precipitation fields, how can we know the cause of the disagreement? Was it our prediction of the future location of a precipitation feature, or was it how precipitation intensity changed over time? Some verification scores, such as the Fractions Skill Score (Mittermaier and Roberts, 2010), apply a metric over spatial windows of increasing size to examine how the forecast performance depends on the spatial scale. Yet we still cannot isolate and quantify the location error explicitly. This makes it difficult to benchmark and optimize the corresponding components of nowcasting models.

Thus, there is a need for full-fledged benchmark procedures that are open, transparent, reproducible, and easy-to-use, and that not only can compete with state-of-the-art but against which future advances can also be measured.

Going beyond the state-of-the-art

Considering the importance of reliable precipitation nowcasts for many stakeholders, the continuous development of models that may provide more skillful predictions for longer lead times will remain a strong focus of the research community. There are two main directions for advancing model development beyond the state-of-the-art:

1. Stay within the fundamental approach to precipitation nowcasting and try to improve either tracking or extrapolation techniques.

2. Or take a step towards exploring the potential of new methods.

Within the first approach, many techniques have been successfully used for advancing the skill of precipitation nowcasting during the last decades: blending with NWP (Sun et al., 2014), auxiliary data assimilation (Winterrath and Rosenow, 2007), probabilistic techniques (Pulkkinen et al., 2019), to name only a few. However, since the fundamental approach to nowcasting has not changed much over recent decades, the progress made in this area can be described as gradual. Yet the situation might change with the increasing popularity of a new modeling approach – deep learning.

The rise of deep learning

”Deep learning” refers to machine-learning methods for artificial neural networks with ”deep” architectures, which may consist of tens to hundreds of computational layers. Rather than relying on engineered features, deep learning derives low-level image features on the lowest layers of a hierarchical network and increasingly abstracts features on the high-level network layers as part of the solution of an optimization problem based on training data (LeCun et al., 2015). Deep learning began its rise from the field of computer science when it started to dramatically outperform reference methods in image classification (Krizhevsky et al., 2012) and machine translation (Sutskever et al., 2014), which was followed by speech recognition (LeCun et al., 2015). Three main reasons caused this substantial breakthrough in predictive efficacy: the availability of ”big data” for model training, the development of activation functions and network architectures that result in numerically stable gradients across many network layers (Dahl et al., 2013), and the ability to scale the learning process massively through parallelization on graphics processing units (GPUs). Today, deep learning is rapidly spreading into many data-rich scientific disciplines, and it complements researchers’ toolboxes with efficient predictive models, including in the field of geosciences (Reichstein et al., 2019).

The expectations of deep learning are particularly high in the atmospheric sciences (Dueben and Bauer, 2018; Gentine et al., 2018). While two of three components that determine the success of deep learning – a (deep) network architecture and the exploitation of GPUs for parallel processing – are field-agnostic, the actual properties of the big data component are certainly specific to a field. These properties mainly refer to the structure of the data, and of course the assumptions on its informative value. In particular, weather radar archives provide massive long-term and well-structured data, which is consistent in time, and resolves precipitation events at a detailed level in space and time. Thus, due to the high capacity of deep learning to discover intricate structure in large data sets (LeCun et al., 2015), it is expected to have ample potential in radar-based precipitation nowcasting.

While expectations are high, the investigation of deep learning in radar-based precipitation nowcasting is still in its infancy, and universal solutions are not yet available. Shi et al. (2015) were the first to introduce deep learning models in the field of radar-based precipitation nowcasting: they presented a convolutional long short-term memory (ConvLSTM) architecture, which outperformed the optical-flow-based ROVER (Real-time Optical flow by Variational methods for Echoes of Radar) nowcasting system in the Hong Kong area. Since then, different research groups in academia and from private companies (e.g., Google, Yandex) introduced new or modified deep learning architectures for quantitative, as well as qualitative (e.g., prediction of the exceedance of specific rainfall intensity thresholds) precipitation nowcasting (Agrawal et al., 2019; Lebedev et al., 2019; Singh et al., 2017; Shi et al., 2017, 2018). Hence, the exploration of deep learning techniques in radar-based nowcasting has begun, and the potential to overcome the limitations of standard tracking and extrapolation techniques has become apparent. There is a strong need, though, to further investigate different architectures, to set up new benchmark experiments, and to understand under which conditions deep learning models can be a viable option for operational services.

The weather radar data

The present thesis describes the model development in the field of precipitation nowcasting. Theoretically and technically, all the models presented here are data-agnostic, i.e., they may make use of several sources of spatial precipitation data products based on different remote sensing technologies, such as weather radar (Ayzel et al., 2019a, 2020), meteorological satellite (Liu et al., 2015), and commercial microwave links (Imhoff et al., 2020a). However, among the competitors, weather radar data provides higher spatial and temporal resolution, and also a higher reliability of quantitative precipitation retrieval from observed radar moments.

Weather radar does not measure precipitation directly. When the transmitted microwave pulse encounters a backscattering target (e.g., raindrops, hail, snow, but also birds, mountain tops or high buildings), some of the energy is scattered back to the radar receiver. It is then interpreted as the reflectivity factor. In turn, the reflectivity factor is a function of the distribution of the rainfall drop sizes within a unit volume of air. The rain rate (R) can be derived from the reflectivity factor (Z) by using empirically obtained $Z - R$ relationship. However, there are many potential estimation errors. They can be classified into two groups: (1) errors that are caused by artefacts in the observation of Z (e.g., attenuation, clutter, beam blockage, as well as miscalibration of the radar measurement instrument itself), and (2) errors that arise from the empirical transformation of Z to R (Crisologo and Heistermann, 2020). In the presented thesis, however, the issues of quality-control, error correction, radar calibration, and quantitative precipitation estimation are deliberately excluded. Instead, we use a quality-controlled product provided by the DWD, the so-called RY product, which is generated as part of the RADKLIM radar reanalysis of the DWD (Winterrath et al., 2017). The RY product represents a quality-controlled rainfall-depth composite of 17 operational DWD C-Band Doppler radars (Fig. 1.3). Quality control of the RY product includes a wide range of correction methods, e.g., clutter removal or accounting for a partial beam blockage (Winterrath et al., 2017). The RY product has a spatial extent of 900 km × 900 km, covers the whole area of Germany, and has been available for calendar years from 2006 to 2017. The spatial and temporal resolution of the RY product is 1 km × 1 km and 5 min, respectively.

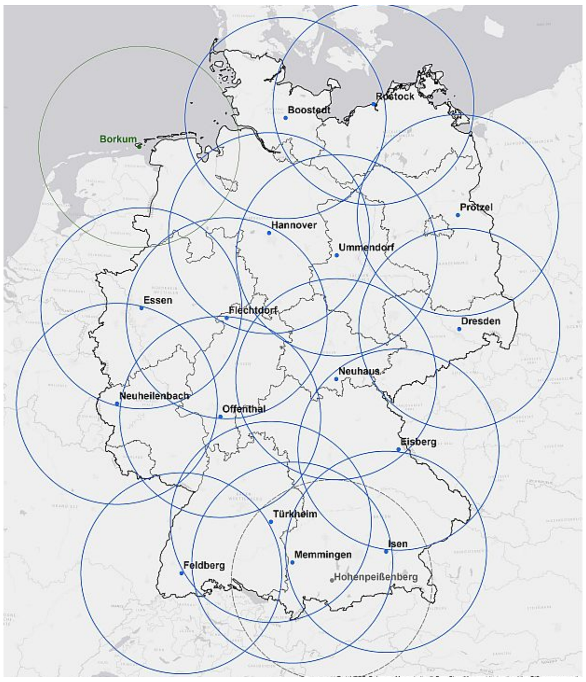

Figure 1.3: Weather radar network (Radarverbund) of the German weather service, showing the locations of the 17 C-band radars and the 150 km radii (source: DWD).

Research questions and structure

The thesis starts out from a meta-question we ask ourselves in the very beginning:

meta-RQ: How can we advance the model development for quantitative precipitation nowcasting?

Still, the present landscape of computational models for precipitation nowcasting struggles with producing open software implementations. To provide open and transparent benchmark procedures for precipitation nowcasting, the first research question asks:

RQ1: What is a suitable open benchmark for further model development – a benchmark that can compete with the state-of-the-art modeling systems?

The question is addressed in the second chapter, where we highlight the development and verification of rainymotion (Ayzel et al., 2019a) – an open-source Python library for precipitation nowcasting. The developed rainymotion models were verified against Eulerian persistence, as a trivial benchmark, and against the operational nowcasting system of the DWD, RADVOR, as a representative of state-of-the-art models.

Having a reliable benchmark at hand should pave the way for further model development, and the measurement of progress. We continued by following the most promising direction – challenging the potential of deep learning. This led us to the second group of research questions:

RQ2.1: What is a suitable deep learning architecture for radar-based precipitation nowcasting?

RQ2.2: What is the potential of a well-trained deep learning model in advancing the skill of nowcasts?

RQ2.3: What are the corresponding challenges and limitations?

These questions are answered in the third chapter, where we introduce RainNet. This deep neural network aims at learning representations of spatiotemporal precipitation field movement and evolution from a massive, open radar data archive. Chapter 3 outlines RainNet's architecture and its training and reports on a set of benchmark experiments where RainNet competes against a conventional nowcasting model from the rainymotion library – the previously introduced benchmark solution (Chapter 2). Based on these experiments, we evaluate the potential of RainNet for nowcasting, but also its limitations in comparison to conventional radar-based nowcasting techniques.

Based on the verification experiments for both rainymotion and RainNet, it turned out to be important to isolate specific error components in order to better understand the overall forecast uncertainty. The total error of a precipitation nowcast consists of an error in the predicted location of a precipitation feature and an error in the change of precipitation intensity over lead time. The isolation and quantification of a particular source of error – in this context the location error – is expected to be helpful in improving specific model components that affect the prominence of a specific error. With this, we ask:

RQ3.1: How can we isolate the forecast location error?

RQ3.2: How do standard benchmark models differ with regard to the forecast location error?

Chapter 4 provides a framework for the direct isolation and quantification of the location error of precipitation nowcasts. Based on this framework, we carried out a benchmarking case study using one year worth of weather radar composites of DWD. We evaluated the performance of four extrapolation models, two of which are based on the linear extrapolation of corner motion; another two are based on the Dense Inverse Search (DIS) method: motion vectors obtained from DIS are used to predict feature locations by linear and Semi-Lagrangian extrapolation. All these provide us with the possibility to better understand the factors that govern these errors, and hence to use that knowledge to specifically improve the extrapolation of motion patterns in existing nowcasting models.

Contribution to Publications

The following three chapters are peer-reviewed research papers which have already been accepted or published in ISI-listed journals, and represent the core of this thesis. I wish to express my gratitude to co-authors, namely Maik Heistermann, Tanja Winterrath, Tobias Scheffer, and Arthur Costa Tomaz de Souza, who substantially contributed to the production of these manuscripts.

Paper I / Chapter 2

Ayzel, G., Heistermann, M., and Winterrath, T.: Optical flow models as an open benchmark for radar-based precipitation nowcasting (rainymotion v0.1), Geosci. Model Dev., 12, 1387–1402, https://doi.org/10.5194/gmd-12-1387-2019, 2019.

GA performed the benchmark experiments, analyzed the data, and wrote the paper. MH coordinated and supervised the work, analyzed the data, and co-wrote the paper. TW assisted in the data retrieval and analysis and shared her expertise in DWD radar products.

Paper II / Chapter 3

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

GA developed the RainNet model, carried out the benchmark experiments, and wrote the paper. TS and MH supervised the study and co-wrote the paper.

Paper III / Chapter 4

de Souza, A. C. T., Ayzel, G., and Heistermann, M.: Quantifying the location error of precipitation nowcasts, Adv. in Meteorology, 2020. (accepted, in print)

A. C. T. de Souza developed the framework, carried out the benchmark experiments, and wrote the paper. GA participated in model development and data analysis and co-wrote the paper. MH supervised the study and co-wrote the paper.

In addition to these manuscripts, I contributed to the following publications in ISI-listed journals, which are not part of this thesis. I list these articles as they illustrate the vast potential of machine learning in Geosciences, specifically in Hydrology and, in the latter case, Paleoecology. Still, I decided not to make these studies a part of the thesis in order to keep a clear focus on precipitation nowcasting.

- Ayzel, G., Varentsova, N., Erina, O., Sokolov, D., Kurochkina, L., and Moreydo, V.: OpenForecast: The First Open-Source Operational Runoff Forecasting System in Russia, Water, 11(8), 1546, https://doi.org/10.3390/w11081546, 2019.

- Ayzel, G., and Izhitskiy, A.: Climate change impact assessment on freshwater inflow into the Small Aral Sea, Water, 11(11), 2377, https://doi.org/10.3390/w11112377, 2019.

- Ayzel, G., Kurochkina, L., and Zhuravlev, S.: The influence of regional hydrometric data incorporation on the accuracy of gridded reconstruction of monthly runoff, Hydrological Sciences Journal, 1–12, https://doi.org/10.1080/02626667.2020.1762886, 2020.

- Ayzel, G., and Heistermann, M.: The effect of calibration data length on the performance of conceptual versus data-driven hydrological models, Computers and Geosciences, in review.

- Foster, W. J., Ayzel, G., Isson, T. T., Mutti, M., and Aberhan, M.: Machine learning (decision tree analysis) identifies ecological selectivity patterns across the end-Permian mass extinction, Earth and Planetary Science Letters, in review. Available at: https://doi.org/10.1101/2020.10.09.332999.

Chapter 2

Optical flow models as an open benchmark for radar-based precipitation nowcasting (rainymotion v0.1)

This chapter is published as:

Ayzel, G., Heistermann, M., and Winterrath, T.: Optical flow models as an open benchmark for radar-based precipitation nowcasting (rainymotion v0.1), Geosci. Model Dev., 12, 1387–1402, https://doi.org/10.5194/gmd-12-1387-2019, 2019.

Abstract

Quantitative precipitation nowcasting (QPN) has become an essential technique in various application contexts, such as early warning or urban sewage control. A common heuristic prediction approach is to track the motion of precipitation features from a sequence of weather radar images, and then to displace the precipitation field to the imminent future (minutes to hours) based on that motion, assuming that the intensity of the features remains constant (”Lagrangian persistence”). In that context, ”optical flow” has become one of the most popular tracking techniques. Yet the present landscape of computational QPN models still struggles with producing open software implementations. Focusing on this gap, we have developed and extensively benchmarked a stack of models based on different optical flow algorithms for the tracking step, and a set of parsimonious extrapolation procedures based on image warping and advection. We demonstrate that these models provide skillful predictions comparable with or even superior to state-of-the-art operational software. Our software library (”rainymotion”) for precipitation nowcasting is written in the Python programming language, and openly available at GitHub (https://github.com/hydrogo/rainymotion, Ayzel et al., 2019b). That way, the library may serve as a tool for providing fast, free and transparent solutions that could serve as a benchmark for further model development and hypothesis testing – a benchmark that is far more advanced than the conventional benchmark of Eulerian persistence commonly used in QPN verification experiments.

2.1 Introduction

How much will it rain within the next hour? The term ”quantitative precipitation nowcasting” refers to forecasts at high spatiotemporal resolution (60-600 s, 100-1000 m) and short lead times of only a few hours. Nowcasts have become important for broad levels of the population for planning various kinds of activities. Yet, they are particularly relevant in the context of early warning of heavy convective rainfall events, and their corresponding impacts such as flash floods, landslides, or sewage overflow in urban areas.

While recent advances in numerical weather prediction (NWP) allow us to forecast atmospheric dynamics at very high resolution (Bauer et al., 2015), computational costs are typically prohibitive for the requirements of operational nowcasting applications with frequent update cycles. Furthermore, the heuristic extrapolation of rain field motion and development, as observed by weather radar, still appears to outperform NWP forecasts at very short lead times (Berenguer et al., 2012; Jensen et al., 2015; Lin et al., 2005). Today, many precipitation nowcasting systems are operational at regional or national scales, utilizing various radar products, algorithms, and blending techniques in order to provide forecasts up to 1 – 3 h, for example: ANC (Mueller et al., 2003), MAPLE (Germann and Zawadzki, 2002b), RADVOR (Winterrath et al., 2012), STEPS (Bowler et al., 2006), STEPS-BE (Foresti et al., 2016), and SWIRLS (Cheung and Yeung, 2012; Woo and Wong, 2017). For an extensive review of existing operational systems, please refer to Reyniers (2008).

A variety of radar-based precipitation nowcasting techniques can be classified into three major groups based on assumptions we make regarding precipitation field characteristics (Germann and Zawadzki, 2002b). The first group – climatological persistence – provides nowcasts by using climatological values (mean or median). The second group – Eulerian persistence – is based on using the latest available observation as a prediction, and is thus independent of the forecast lead time. The third group – Lagrangian persistence – allows the extrapolation of the most recent observed precipitation field under the assumption that intensity of precipitation features and the motion field are persistent (Germann and Zawadzki, 2002b; Woo and Wong, 2017). In addition, we can classify nowcasting methods based on how predictive uncertainty is accounted for: in contrast to deterministic approaches, ensemble nowcasts attempt to account for predictive uncertainty by including different realizations of the motion field and the evolution of rainfall intensity itself (Berenguer et al., 2011). In this study, we focus our model development around the group of Lagrangian persistence models which provide deterministic precipitation nowcasts.

Lagrangian methods consist of two computational steps: tracking and forecasting (extrapolation) (Austin and Bellon, 1974). In the tracking step, we compute a velocity field from a series of consecutive radar images, either on a per pixel basis (Germann and Zawadzki, 2002b; Grecu and Krajewski, 2000; Liu et al., 2015; Zahraei et al., 2012), or for contiguous objects (Zahraei et al., 2013). In the second step, we use that velocity field to advect the most recent rain field, i.e., to displace it to the imminent future based on its observed motion. That step has been implemented based on semi-Lagrangian schemes (Germann and Zawadzki, 2002b), interpolation procedures (Liu et al., 2015), or mesh-based models (Bellerby, 2006; Zahraei et al., 2012). Different algorithms can be used for each step – tracking and forecasting – in order to compute an ensemble forecast (Berenguer et al., 2011; Foresti et al., 2016; Grecu and Krajewski, 2000).

One of the most prominent techniques for the tracking step is referred to as ”optical flow”. The original term was inspired by the idea of an *apparent* motion of brightness patterns observed when a camera or the eyeball is moving relative to the objects (Horn and Schunck, 1981). Today, optical flow is often understood as a group of techniques to infer motion patterns or velocity fields from consecutive image frames, e.g. in the field of precipitation nowcasting (Bowler et al., 2004; Liu et al., 2015; Woo and Wong, 2017). For the velocity field estimation, we need to accept both the brightness constancy assumption and one of a set of additional optical flow constraints (OFCs). The spatial attribution of OFC marks the two main categories of optical flow models: local (differential) and global (variational) (Cheung and Yeung, 2012; Liu et al., 2015). Local models try to set an OFC only in some neighborhood, while global models apply an OFC for a whole image. There is also a distinct group of spectral methods where the Fourier transform is applied to the inputs, and an OFC is resolved in the spectral (Fourier) domain (Ruzanski et al., 2011). Bowler et al. (2004) introduced the first local optical flow algorithm for precipitation nowcasting, and gave rise to a new direction of models. Bowler's algorithm is the basis of the STEPS (Bowler et al., 2006) and STEPS-BE (Foresti et al., 2016) operational nowcasting systems. Liu et al. (2015) proposed using a local Lucas–Kanade optical flow method (Lucas and Kanade, 1981) independently for each pixel of satellite imagery because they found it outperformed a global Horn–Schunck (Horn and Schunck, 1981) optical flow algorithm in the context of precipitation nowcasting from infrared satellite images. Yeung et al. (2009), Cheung and Yeung (2012), and Woo and Wong (2017) used different global optical flow algorithms (Bruhn et al., 2005a; Wong et al., 2009) for establishing the SWIRLS product for operational nowcasting in Hong-Kong.

Hence, for around two decades, optical flow algorithms have been doing their best for state-of-the-art operational nowcasting systems around the globe. Should research still care about them? It should, and the reason is that – despite the abundance of publications about different types of optical flow techniques for nowcasting applications – an open and transparent benchmark model is yet not available, except for the most trivial one: Eulerian persistence.

That is all the more surprising since open-source implementations of fundamental optical flow algorithms (Brox et al., 2004; Bruhn et al., 2005b) have been around for up to 20 years – with the OpenCV library (https://opencv.org, last access: 28 March 2019) just being the most widely known. Such libraries provide efficient implementations of various optical flow algorithms for a vast number of research and application contexts. Yet none can be applied in the QPN context out of the box – without the need to address additional and specific challenges such as underlying assumptions and constraints of velocity fields, pre- and post-processing steps, or model parameterization and verification.

The aim of this paper is thus to establish a set of benchmark procedures for quantitative precipitation nowcasting as an alternative to the trivial case of Eulerian persistence. This study does not aim to improve the standard of precipitation nowcasting beyond the state of the art, but to provide an open, transparent, reproducible, and easy-to-use approach that can compete with the state of the art, and against which future advances can be measured. To that end, we developed a group of models that are based on two optical flow formulations of algorithms for the tracking step – Sparse (Lucas and Kanade, 1981) and Dense (Kroeger et al., 2016) – together with two parsimonious extrapolation techniques based on image warping and spatial interpolation. These models are verified against Eulerian persistence, as a trivial benchmark, and against the operational nowcasting system of the Deutscher Wetterdienst (the German Weather Service, DWD), as a representative of state-of-the-art models. The different optical flow implementations are published as an open-source Python library (rainymotion, Ayzel et al., 2019b) that entirely relies on free and open-source dependencies, including detailed documentation and example workflows (https://rainymotion.readthedocs.io, last access: 28 March 2019).

The paper is organized as follows. In Sect. 2.2, we describe the algorithmic and technical aspects of the suggested optical flow models. Section 2.3 describes the data we used and provides a short synopsis of events we used for the benchmark experiment. We report the results in Sect. 2.4 and discuss them in various contexts in Sect. 2.5. Section 2.6 provides a summary and conclusions.

2.2 Description of the models and the library

The benchmark models developed in this study consist of different combinations of algorithms for the two major steps of Lagrangian nowcasting frameworks, namely tracking and extrapolation (Austin and Bellon, 1974). Table 2.1 provides an overview of the models. The values of model parameters adopted in the benchmark experiment have been heuristically determined and not yet been subject to systematic optimization. However, the rainymotion library provides an opportunity to investigate how different optical flow model parameters can affect nowcasting results or how they can be tuned to represent, e.g., the typical range of advection speeds of real precipitation fields. For a description of parameters, please refer to Sect. S1 in the Supplement or the rainymotion library documentation (https://rainymotion.readthedocs.io/, last access: 28 March 2019).

2.2.1 The Sparse group

The central idea around this group of methods is to identify distinct features in a radar image that are suitable for tracking. In this context, a ”feature” is defined as a distinct point (”corner”) with a sharp gradient of rainfall intensity. That approach is less arbitrary and scale dependent and thus more universal than classical approaches that track storm cells as contiguous objects (e.g., Wilson et al., 1998) because it eliminates the need to specify arbitrary and scale-dependent characteristics of ”precipitation features” while the identification of corners depends only on the gradient sharpness in a cell’s neighborhood. Inside this group, we developed two models that slightly differ with regard to both tracking and extrapolation.

The first model (SparseSD, for Sparse Single Delta) uses only the two most recent radar images for identifying, tracking, and extrapolating

Table 2.1: Overview of the developed nowcasting models and their computational performance. Nowcasting experiments were carried out for 1 h lead time in 5 min temporal resolution (12 resulting nowcast frames in total) using the RY radar data (spatial resolution of 1 km, grid size 900×900) and a standard office PC with an Intel® Core^{TM} i7-2600 CPU (eight cores, 3.4 GHz).

Model name	Input radar images	Default tracking algorithm	Extrapolation	Computational time (tracking/ extrapolation/ total), s
SparseSD	2	Shi–Tomasi corner detector, Lucas–Kanade optical flow	Constant delta-change, affine warping	0.2 / 1.1 / 1.3
Sparse	3-24	Shi–Tomasi corner detector, Lucas–Kanade optical flow	Linear regression, affine warping	0.1 / 1.1 / 1.2
Dense	2	DIS optical flow	Backward constant-vector advection scheme	0.2 / 5.5 / 5.7
DenseRotation	2	DIS optical flow	Backward semi-Lagrangian advection scheme	3.2 / 5.7 / 8.9

features. Assuming that t denotes both the nowcast issue time and the time of the most recent radar image, the implementation can be summarized as follows:

1. Identify features in a radar image at time $t-1$ using the Shi–Tomasi corner detector (Shi and Tomasi, 1994). This detector determines the most prominent corners in the image based on the calculation of the corner quality measure ($\min(\lambda_1, \lambda_2)$, where λ_1 and λ_2 are corresponding eigenvalues of the covariance matrix of derivatives over the neighborhood of 3×3 pixels) at each image pixel (see Sect. S1 in the Supplement for a detailed description of algorithm parameters).

2. Track these features at time t using the local Lucas–Kanade optical flow algorithm (Lucas and Kanade, 1981). This algorithm tries to identify the location of a feature we previously identified at time $t-1$ in the radar image at time t, based on solving a set of optical flow equations in the local feature neighborhood using the least-squares approach (see Sect. S1 in the Supplement for a detailed description of algorithm parameters).

3. Linearly extrapolate the features' motion in order to predict the features' locations at each lead time n.

4. Calculate the affine transformation matrix for each lead time n based on the locations of all identified features at time t and $t+n$ using the least-squares approach (Schneider and Eberly, 2003). This matrix uniquely identifies the required transformation of the last observed radar image at time t so that the nowcast images at times $t+1 \ldots t+n$ provide the smallest possible difference between the locations of detected features at time t and the extrapolated features at times $t+1 \ldots t+n$.

5. Extrapolate the radar image at time t by warping: for each lead time, the warping procedure uniquely transforms each pixel location of the radar image at time t to its future location in the nowcast radar images at times $t+1 \ldots t+n$, using the affine transformation matrix. Remaining discontinuities in the predicted image are linearly interpolated in order to obtain nowcast intensities on a grid that corresponds to the radar image at time t (Wolberg, 1990).

To our knowledge, this study is the first to apply image warping directly as a simple and fast algorithm to represent advective motion of a precipitation field. In Sect. S2 in the Supplement, you can find a simple synthetic example which shows the potential of the warping technique to replace an explicit advection formulation for temporal extrapolation.

For a visual representation of the SparseSD model, please refer to Fig. 2.1.

The second model (Sparse) uses the 24 most recent radar images, and we consider only features that are persistent over the whole period (of 24 time steps). The implementation can be summarized as follows.

1. Identify features on a radar image at time $t-23$ using the Shi–Tomasi corner detector (Shi and Tomasi, 1994).
2. Track these features in the radar images from $t-22$ to t using the local Lucas–Kanade optical flow algorithm (Lucas and Kanade, 1981).
3. Build linear regression models which independently parameterize changes in coordinates through time (from $t-23$ to t) for every successfully tracked feature;
4. Continue with steps 3–5 of SparseSD.

For a visual representation of the Sparse model, please refer to Fig. 2.2.

2.2.2 The Dense group

The Dense group of models uses, by default, the Dense Inverse Search algorithm (DIS) – a global optical flow algorithm proposed by Kroeger et al. (2016) – which allows us to explicitly estimate the velocity of each image pixel based on an analysis of two consecutive radar images. The DIS algorithm was selected as the default optical flow method for motion field retrieval because it showed, in our benchmark experiments, a higher accuracy and also a higher computational efficiency in comparison with other global optical flow algorithms such as DeepFlow (Weinzaepfel et al., 2013), and PCAFlow (Wulff and Black, 2015). We also tested the local Farnebäck algorithm (Farnebäck, 2003), which we modified by replacing zero velocities by interpolation, and by smoothing the obtained velocity field based on a variational refinement procedure (Brox et al., 2004) (please refer to Sect. S5 in the Supplement for verification results of the corresponding benchmark experiment with various dense optical flow models). However, the rainymotion library provides the option to choose any of the optical flow methods specified above for precipitation nowcasting.

The two models in this group differ only with regard to the extrapolation (or advection) step. The first model (Dense) uses a constant-vector advection scheme (Bowler et al., 2004), while the second model (DenseRotation) uses a semi-Lagrangian advection scheme (Germann and Zawadzki, 2002b). The main difference between both approaches is that a constant-vector scheme does not allow for the representation of rotational motion (Bowler et al., 2004); a semi-Lagrangian scheme allows for the representation of large-scale rotational movement while assuming the motion field itself to be persistent (Fig. 2.3).

There are two possible options of how both advection schemes may be implemented: forward in time (and downstream in space) or backward in time (and upstream in space) (Fig. 2.3). It is yet unclear which scheme can be considered as the most appropriate and universal solution for radar-based precipitation nowcasting, regarding the conservation of mass on the one hand and the attributed loss of power at small scales on the other hand (e.g., see discussion in Bowler et al., 2004; Germann and Zawadzki, 2002b). Thus, we conducted a benchmark experiment with any possible combination of forward vs. backward and constant-vector vs. semi-Lagrangian advection. Based on the results (see Sect. S6 in the Supplement), we use the backward scheme as the default option for both the Dense and DenseRotation models. However, the rainymotion library still provides the option to use the forward scheme, too.

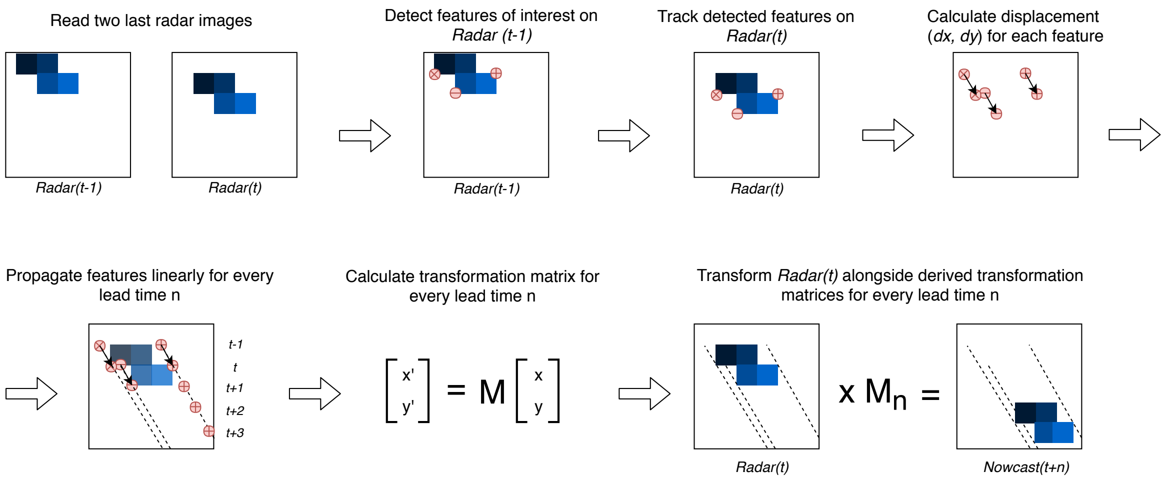
Read two last radar images
Radar(t-1)
Radar(t)
Detect features of interest on Radar (t-1)
Radar(t-1)
Track detected features on Radar(t)
Radar(t)
Calculate displacement (dx, dy) for each feature
Propagate features linearly for every lead time n
t-1
t
t+1
t+2
t+3
Calculate transformation matrix for every lead time n
[x' y'] = M [x y]
Transform Radar(t) alongside derived transformation matrices for every lead time n
Radar(t)
x Mn =
Nowcast(t+n)

Figure 2.1: Scheme of the SparseSD model.

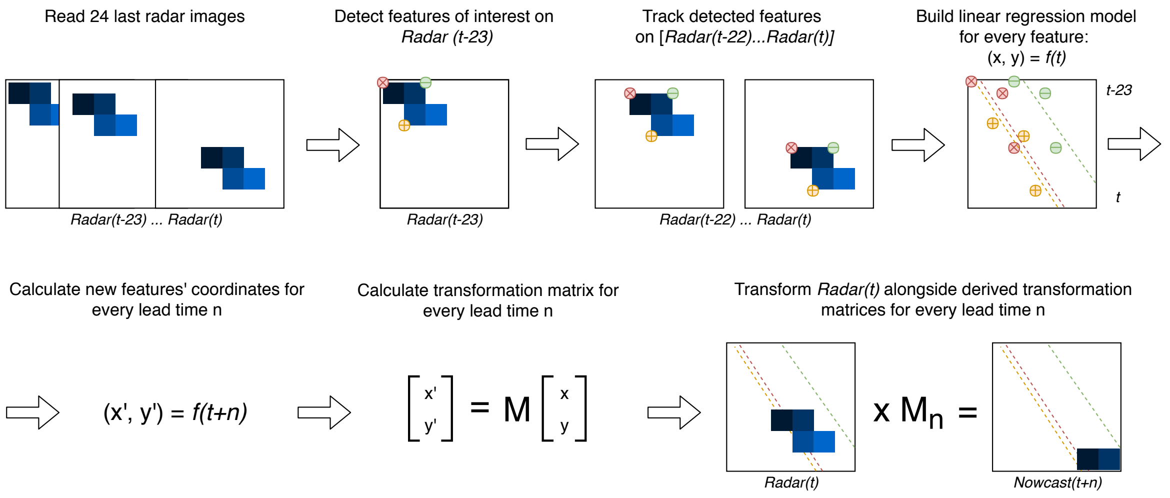
Read 24 last radar images
Radar(t-23) ... Radar(t)
Detect features of interest on Radar (t-23)
Radar(t-23)
Track detected features on [Radar(t-22)...Radar(t)]
Radar(t-22) ... Radar(t)
Build linear regression model for every feature: (x, y) = f(t)
t-23
t
Calculate new features' coordinates for every lead time n
(x', y') = f(t+n)
Calculate transformation matrix for every lead time n
[x' y'] = M [x y]
Transform Radar(t) alongside derived transformation matrices for every lead time n
Radar(t)
x Mn =
Nowcast(t+n)

Figure 2.2: Scheme of the Sparse model.

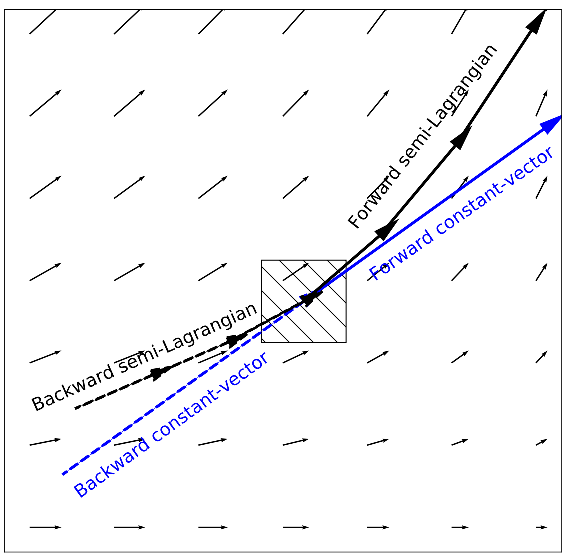

Figure 2.3: Displacement vectors of four proposed advection schemes: forward or backward constant vector and forward or backward semi-Lagrangian.

Both the Dense and DenseRotation models utilize a linear interpolation procedure in order to interpolate advected rainfall intensities at their predicted locations to the native radar grid. The interpolation procedure "distributes" the value of a rain pixel to its neighborhood, as proposed in different modifications by Bowler et al. (2004), Liu et al. (2015), and Zahraei et al. (2012). The Dense group models' implementation can be summarized as follows.

1. Calculate a velocity field using the global DIS optical flow algorithm (Kroeger et al., 2016), based on the radar images at time $t-1$ and t.

2. Use a backward constant-vector (Bowler et al., 2004) or a backward semi-Lagrangian scheme (Germann and Zawadzki, 2002b) to extrapolate (advect) each pixel according to the displacement (velocity) field, in one single step for each lead time $t + n$. For the semi-Lagrangian scheme, we update the velocity of the displaced pixels at each prediction time step n by linear interpolation of the velocity field to a pixel's location at that time step.

3. As a result of the advection step, we basically obtain an irregular point cloud that consists of the original radar pixels displaced from their original location. We use the intensity of each displaced pixel at its predicted location at time $t + n$ in order to interpolate the intensity at each grid point of the original (native) radar grid (Liu et al., 2015; Zahraei et al., 2012), using the inverse distance weighting interpolation technique. It is important to note that we minimize numerical diffusion by first advecting each pixel over the target lead time before applying the interpolation procedure (as in the "interpolate once" approach proposed by Germann and Zawadzki, 2002b). That way, we avoid rainfall features being smoothed in space by the effects of interpolation.

2.2.3 Persistence

The (trivial) benchmark model of Eulerian persistence assumes that for any lead time n, the precipitation field is the same as for time t. Despite its simplicity, it is quite a powerful model for very short lead times, and, at the same time, its verification performance is a good measure of temporal decorrelation for different events.

2.2.4 The rainymotion Python library

We have developed the rainymotion Python library to implement the above models. Since the rainymotion uses the standard format of numpy arrays for data manipulation, there is no restriction in using different data formats which can be read, transformed, and converted to numpy arrays using any tool from the set of available open software libraries for radar data manipulation (the list is available on https://openradarscience.org, last access: 28 March 2019). The source code is available in a Github repository (Ayzel et al., 2019b) and has a documentation page (https://rainymotion.readthedocs.io, last access: 28 March 2019) which includes installation instructions, model description, and usage examples. The library code and accompanying documentation are freely distributed under the MIT software license which allows unrestricted use. The library is written in the Python 3 programming language (https://python.org, last access: 28 March 2019), and its core is entirely based on open-source software libraries (Fig. 2.4): ωradlib (Heistermann et al., 2013), OpenCV (Bradski and Kaehler, 2008), SciPy (Jones et al., 2018), NumPy (Oliphant, 2006), Scikit-learn (Pedregosa et al., 2011), and Scikit-image (Van der Walt et al., 2014). For generating figures we use the Matplotlib library (Hunter, 2007), and we use the Jupyter notebook (https://jupyter.org, last access: 28 March 2019) interactive development environment for code and documentation development and distribution. For managing the dependencies without any conflicts, we recommend to use the Anaconda Python distribution (https://anaconda.com, last access: 28 March 2019) and follow rainymotion installation instructions (https://rainymotion.readthedocs.io, last access: 28 March 2019).

2.2.5 Operational baseline (RADVOR)

The DWD operationally runs a stack of models for radar-based nowcasting and provides precipitation forecasts for a lead time up to 2 h. The operational QPN is based on the RADVOR module (Bartels et al., 2005; Rudolf et al., 2012). The tracking algorithm estimates the motion field from the latest sequential clutter-filtered radar images using a pattern recognition technique at different spatial resolutions (Winterrath and Rosenow, 2007; Winterrath et al., 2012). The focus of the tracking algorithm is on the meso-β scale (spatial extent: 25–250 km) to cover mainly large-scale precipitation patterns, but the meso-γ scale (spatial extension: 2.5–25 km) is also incorporated to allow the detection of smaller-scale convective structures. The resulting displacement field is interpolated to a regular grid, and a weighted averaging with previously derived displacement fields

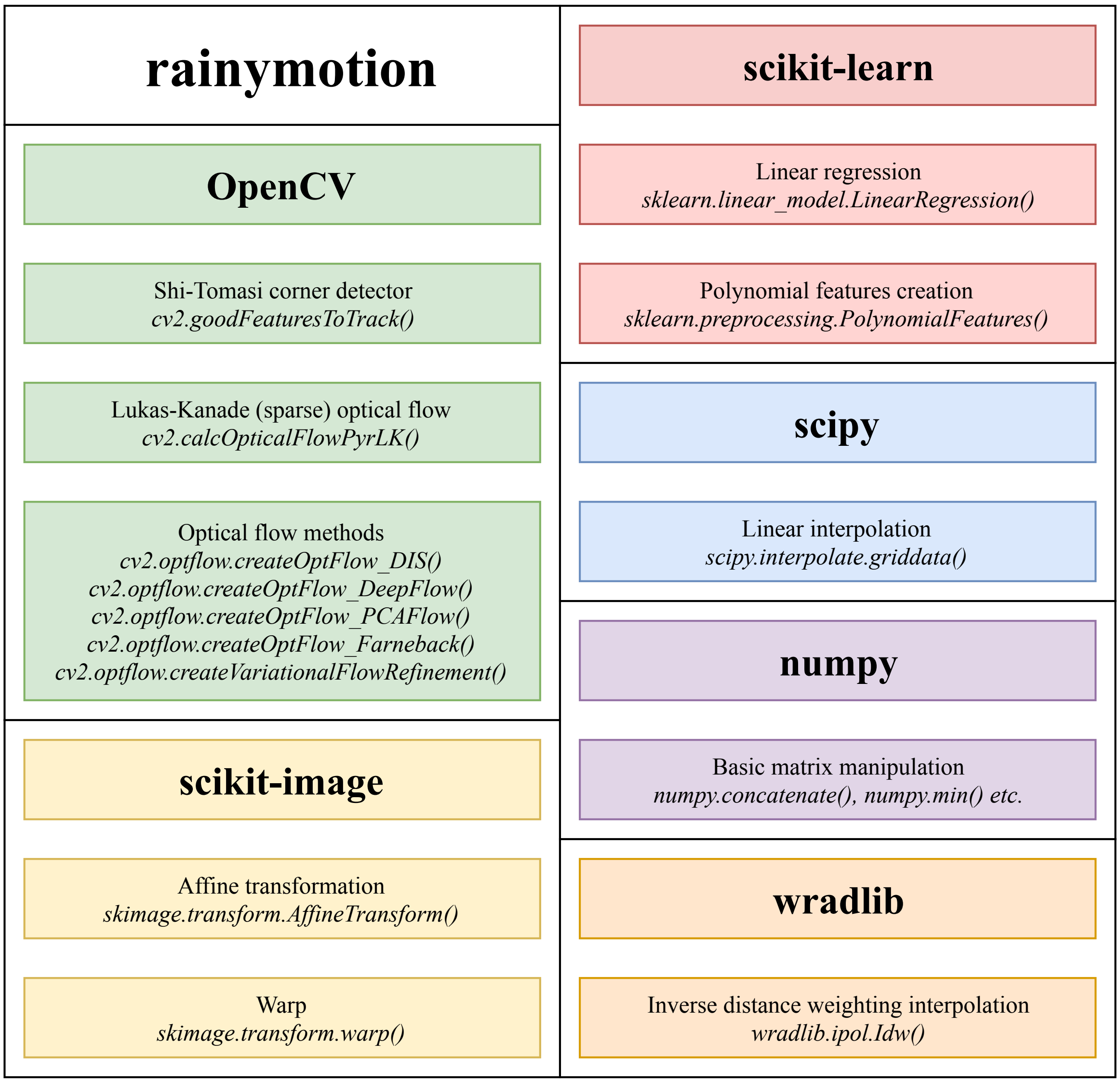
rainymotion
OpenCV
Shi-Tomasi corner detector
cv2.goodFeaturesToTrack()
Lukas-Kanade (sparse) optical flow
cv2.calcOpticalFlowPyrLK()
Optical flow methods
cv2.optflow.createOptFlow_DIS()
cv2.optflow.createOptFlow_DeepFlow()
cv2.optflow.createOptFlow_PCAFlow()
cv2.optflow.createOptFlow_Farneback()
cv2.optflow.createVariationalFlowRefinement()
scikit-image
Affine transformation
skimage.transform.AffineTransform()
Warp
skimage.transform.warp()
scikit-learn
Linear regression
sklearn.linear_model.LinearRegression()
Polynomial features creation
sklearn.preprocessing.PolynomialFeatures()
scipy
Linear interpolation
scipy.interpolate.griddata()
numpy
Basic matrix manipulation
numpy.concatenate(), numpy.min() etc.
wradlib
Inverse distance weighting interpolation
wradlib.ipol.Idw()

Figure 2.4: Key Python libraries for rainymotion library development.

is implemented to guarantee a smooth displacement over time. The extrapolation of the most recent radar image according to the obtained velocity field is performed using a semi-Lagrangian approach. The described operational model is updated every 5 min and produces precipitation nowcasts at a temporal resolution of 5 min and a lead time of 2 h (RV product). In this study we used the RV product data as an operational baseline and did not re-implement the underlying algorithm itself.

2.3 Verification experiments

2.3.1 Radar data and verification events

We use the so-called RY product of the DWD as input to our nowcasting models. The RY product represents a quality-controlled rainfall depth product that is a composite of the 17 operational Doppler radars maintained by the DWD. It has a spatial extent of 900 km $\times$ 900 km and covers the whole area of Germany. Spatial and temporal resolution of the RY product is 1 km $\times$ 1 km and 5 min, respectively. This composite product includes various procedures for correction and quality control (e.g. clutter removal). We used the ωradlib (Heistermann et al., 2013) software library for reading the DWD radar data.

For the analysis, we selected 11 events during the summer periods of 2016 and 2017. These events are selected for covering a range of event characteristics with different rainfall intensity, spatial coverage, and duration. Table 2.2 shows the studied events. You can also find links to animations of event intensity dynamics in Sect. S3 in the Supplement.

2.3.2 Verification metrics

For the verification we use two general categories of scores: continuous (based on the differences between nowcast and observed rainfall intensities) and categorical (based on standard contingency tables for calculating matches between Boolean values which reflect the exceedance of specific rainfall intensity thresholds). We use the mean absolute error (MAE) as a continuous score:

$$MAE = \frac{\sum_{i=1}^{n} |now_i - obs_i|}{n} \tag{2.1}$$

where now$_i$ and obs$_i$ are nowcast and observed rainfall rate in the i-th pixel of the corresponding radar image, and n the number of pixels. To compute the MAE, no pixels were excluded based on thresholds of nowcast or observed rainfall rate.

And we use the critical success index (CSI) as a categorical score:

$$CSI = \frac{hits}{hits + false\ alarms + misses} \tag{2.2}$$

where hits, false alarms, and misses are defined by the contingency table and the corresponding threshold value (for details see Sect. S4 in the Supplement).

Following studies of Bowler et al. (2006) and Foresti et al. (2016), we have applied threshold rain rates of 0.125, 0.25, 0.5, 1 and 5 mm h^{-1} for calculating the CSI.

These two metrics inform us about the models' performance from the two perspectives: MAE captures errors in rainfall rate prediction (the less the better), and CSI captures model accuracy (the fraction of the forecast event that was correctly predicted; it does not distinguish between the sources of errors; the higher the better). You can find results represented in terms of additional categorical scores (false alarm rate, probability of detection, equitable threat score) in Sect. S4 in the Supplement.

2.4 Results

For each event, all models (Sparse, SparseSD, Dense, DenseRotation, Persistence) were used to compute nowcasts with lead times from 5 to 60 min (in 5 min steps). Operational nowcasts generated by the RADVOR system were provided by the DWD with the same temporal settings. An example of nowcasts for lead times 0, 5, 30, and 60 min is shown in Fig. 2.5.

To investigate the effects of numerical diffusion, we calculated, for the same example, the power spectral density (PSD) of the nowcasts and the corresponding observations (bottom panels in Fig. 2.5) using Welch's method (Welch, 1967). Germann and Zawadzki (2002b) showed that the most significant loss of power (lower PSD values) occurs at scales between 8 and 64 km. They did not analyze scales below 8 (2^3) km because their original grid resolution was 4 km. We extended the spectral analysis to consider scales as small as 2^1 km. Other than Germann and Zawadzki (2002b), we could not observe any substantial loss of power between 8 and 64 km, yet Fig. 2.5 shows that both Dense and Sparse models consistently start to lose power at scales below 4 km. That loss does not depend much on the nowcast lead time, yet, the Sparse group of models loses more power at a lead time of 5 min as compared to the Dense group. Still, these results rather confirm Germann and Zawadzki (2002b): they show, as would be expected, that any loss of spectral power is most pronounced at the smallest scales and disappears at scales about 2–3 orders above the native grid resolution. For the investigated combination of data and models, this implies that our nowcasts will not be able to adequately represent rainfall features smaller than 4 km at lead times of up to 1 h.

Figure 2.6 shows the model performance (in terms of MAE) as a function of lead time. For each event, the Dense group of models is superior to the other ones. The RV product achieves an efficiency that is comparable to the Dense group. The SparseSD model outperforms the Sparse model for short lead times (up to 10–15 min), and vice versa

Table 2.2: Characteristics of the selected events.

Event no.	Start	End	Duration (h)	Maximum extent (km^2)	Extent >1 mm h^{-1} (%)
Event 1	2016-05-23 2:00	2016-05-23 8:00	6	159318	42
Event 2	2016-05-23 13:00	2016-05-24 2:30	13.5	135272	56
Event 3	2016-05-29 12:05	2016-05-29 23:55	12	160095	72
Event 4	2016-06-12 7:00	2016-06-12 19:00	12	150416	53
Event 5	2016-07-13 17:30	2016-07-14 1:00	7.5	145501	62
Event 6	2016-08-04 18:00	2016-08-05 7:00	13	168407	74
Event 7	2017-06-29 3:00	2017-06-29 5:05	2	140021	70
Event 8	2017-06-29 17:00	2017-06-29 21:00	4	182561	60
Event 9	2017-06-29 22:00	2017-06-30 21:00	23	160822	75
Event 10	2017-07-21 19:00	2017-07-21 23:00	4	63698	77
Event 11	2017-07-24 8:00	2017-07-25 23:55	16	253666	63

for longer lead times. For some events (1–4, 6, 10, 11), the performance of the RV product appears to be particularly low in the first 10 min, compared to the other models. These events are characterized by particularly fast rainfall field movement.

Figure 2.7 has the same structure as Fig. 2.6 but shows the CSI with a threshold value of 1 mm h^{-1}. For two events (7 and 10) the RV product achieves a comparable efficiency with the Dense group for lead times beyond 30 min. For the remaining events, the Dense group tends to outperform all other methods and the RV product achieves an average rank between models of the Sparse and Dense groups. For the Dense group of models, it appears that accounting for field rotation does not affect the results of the benchmark experiment much – the Dense and DenseRotation models perform very similarly, at least for the selected events and the analyzed lead times. The behavior of the Sparse group models is mostly consistent with the MAE.

Figure 2.8 shows the model performance using the CSI with a threshold value of 5 mm h^{-1}. For the majority of events, the resulting ranking of models is the same as for the CSI with a threshold of 1 mm h^{-1}. For events no. 2 and no. 3, the performance of the RV product relative to the Dense models is a little bit better, while for other events (e.g., no. 7), the Dense models outperform the RV product more clearly than for the CSI of 1 mm h^{-1}.

Table 2.3 summarizes the results of the Dense group models in comparison to the RV product for different verification metrics averaged over all the selected events and two lead time periods: 5–30, and 35–60 min. Results show that the Dense group always slightly outperforms the RV model in terms of CSI metric for both lead time periods and all analyzed rainfall intensity threshold used for CSI calculation. In terms of MAE, differences between model performances are less pronounced. For the CSI metric, the absolute differences between all models tend to be consistent with increasing rainfall thresholds.

You can find more figures illustrating the models' efficiency for different thresholds and lead times in Sect. S4 in the Supplement.

2.5 Discussion

2.5.1 Model comparison

All tested models show significant skill over the trivial Eulerian persistence over a lead time of at least 1 h. Yet a substantial loss of skill over lead time is present for all analyzed events, as expected. We have not disentangled the causes of that loss, but predictive uncertainty will always result from errors in both the representation of field motion and the total lack of representing precipitation formation, dynamics, and dissipation in a

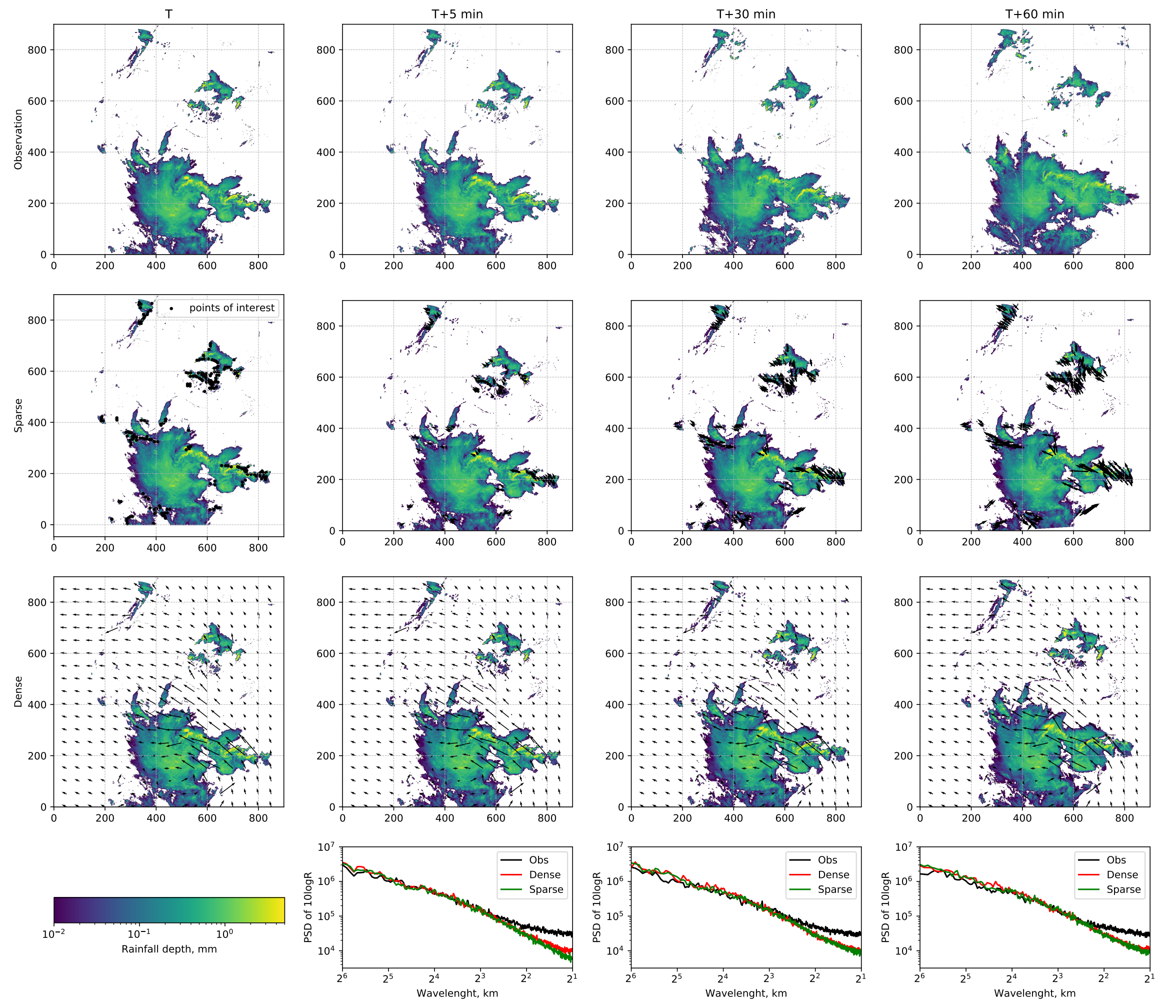

Figure 2.5: Example of the nowcasting models output (SparseSD and Dense models) for the timestep, 29 May 2016, 19:15, and corresponding level of numerical diffusion (the last row).

framework of Lagrangian persistence. Many studies specify a lead time of 30 min as a predictability limit for convective structures with fast dynamics of rainfall evolution (Foresti et al., 2016; Grecu and Krajewski, 2000; Thorndahl et al., 2017; Wilson et al., 1998; Zahraei et al., 2012). Our study confirms these findings.

For the majority of analyzed events, there is a clear pattern that the Dense group of optical flow models outperforms the operational RV nowcast product. For the analyzed events and lead times, the differences between the Dense and the DenseRotation models (or, in other words, between constant-vector and semi-Lagrangian schemes) are negligible. The absolute difference in performance between the Dense group models and the RV product appears to be independent of rainfall intensity threshold and lead time (Table 2.3), which implies that the relative advance of the Dense group models over the RV product increases both with lead time and rainfall intensity threshold. A gain in performance for longer lead times by taking into account more time steps from the past can be observed when comparing the SparseSD model (looks back 5 min in time) against the Sparse model (looks back 2 h in time).

Despite their skill over Eulerian persistence, the Sparse group models are significantly outperformed by the Dense group models for all the analyzed events and lead times. The reason for this behaviour still remains unclear. It could, in general, be a combination of errors introduced in corner tracking and extrapolation as well as image warping as a surrogate for formal advection. While the systematic identification of error sources will

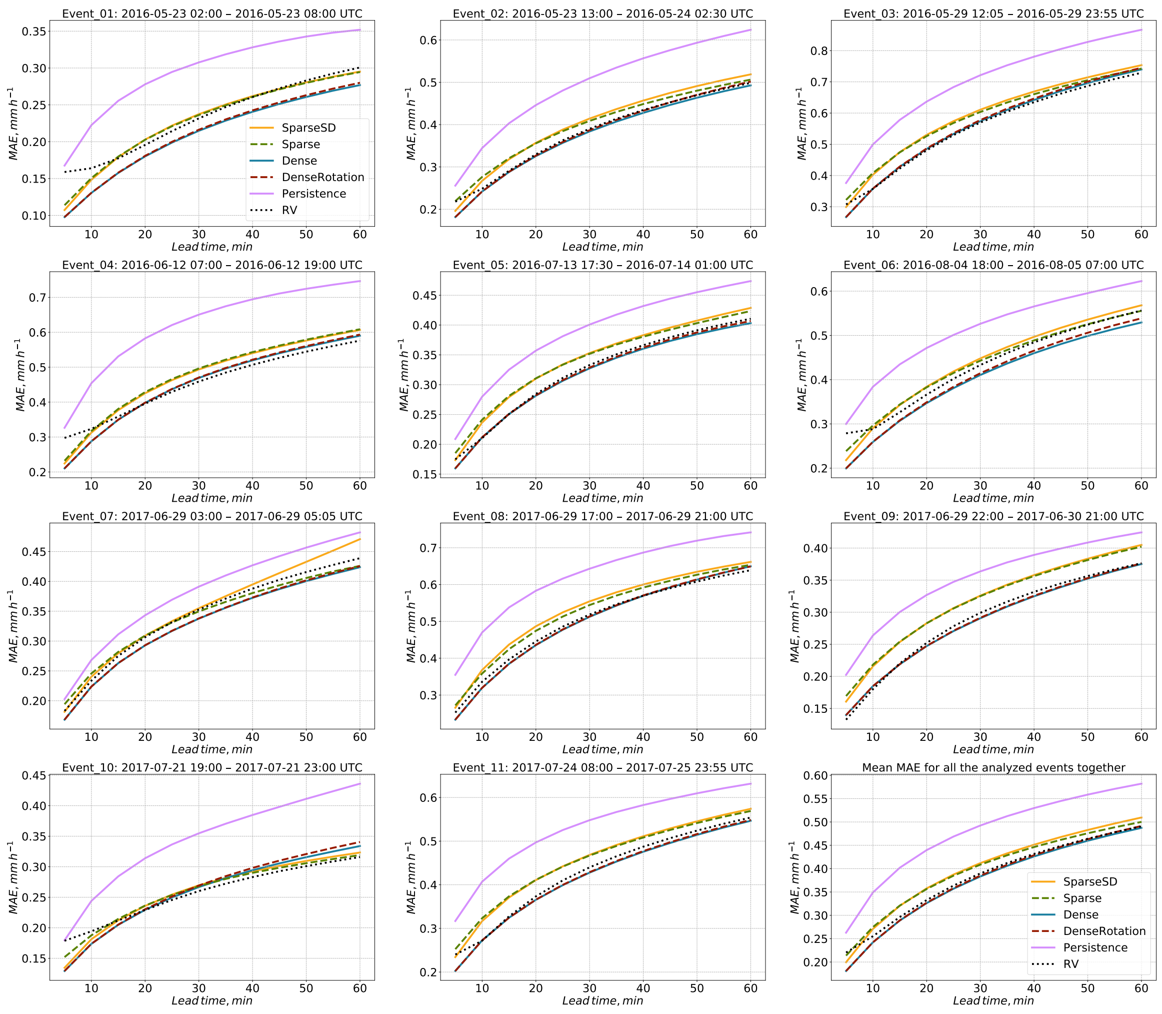

Figure 2.6: Verification of the different optical flow based nowcasts in terms of MAE for 11 precipitation events over Germany.

be subject to future studies, we suspect that the the local features (corners) identified by the Shi–Tomasi corner detector might not be representative of the overall motion of the precipitation field: the detection focuses on features with high intensities and gradients, the motion of which might not represent the dominant meso-γ-scale motion patterns.

There are a couple of possible directions for enhancing the performance for longer lead times using the Dense group of models. A first is to use a weighted average of velocity fields derived from radar images three (or more) steps back in time (as done in RADVOR to compute the RV product). A second option is to calculate separate velocity fields for low-and high-intensity subregions of the rain field and advect these subregions separately (as proposed in Golding, 1998) or find an optimal weighting procedure. A third approach could be to optimize the use of various optical flow constraints in order to improve the performance for longer lead times, as proposed in Germann and Zawadzki (2002b), Bowler et al. (2004), or Mecklenburg et al. (2000). The flexibility of the rainymotion software library allows users to incorporate such algorithms for benchmarking any hypothesis, and, for example, implement different models or parameterizations for different lead times. Bowler et al. (2004) also showed a significant performance increase for longer lead times by using NWP model winds for the advection step. However, Winterrath and Rosenow (2007) did not obtain any improvement compared to RADVOR for longer lead times by incorporating NWP model winds into the nowcasting procedure.

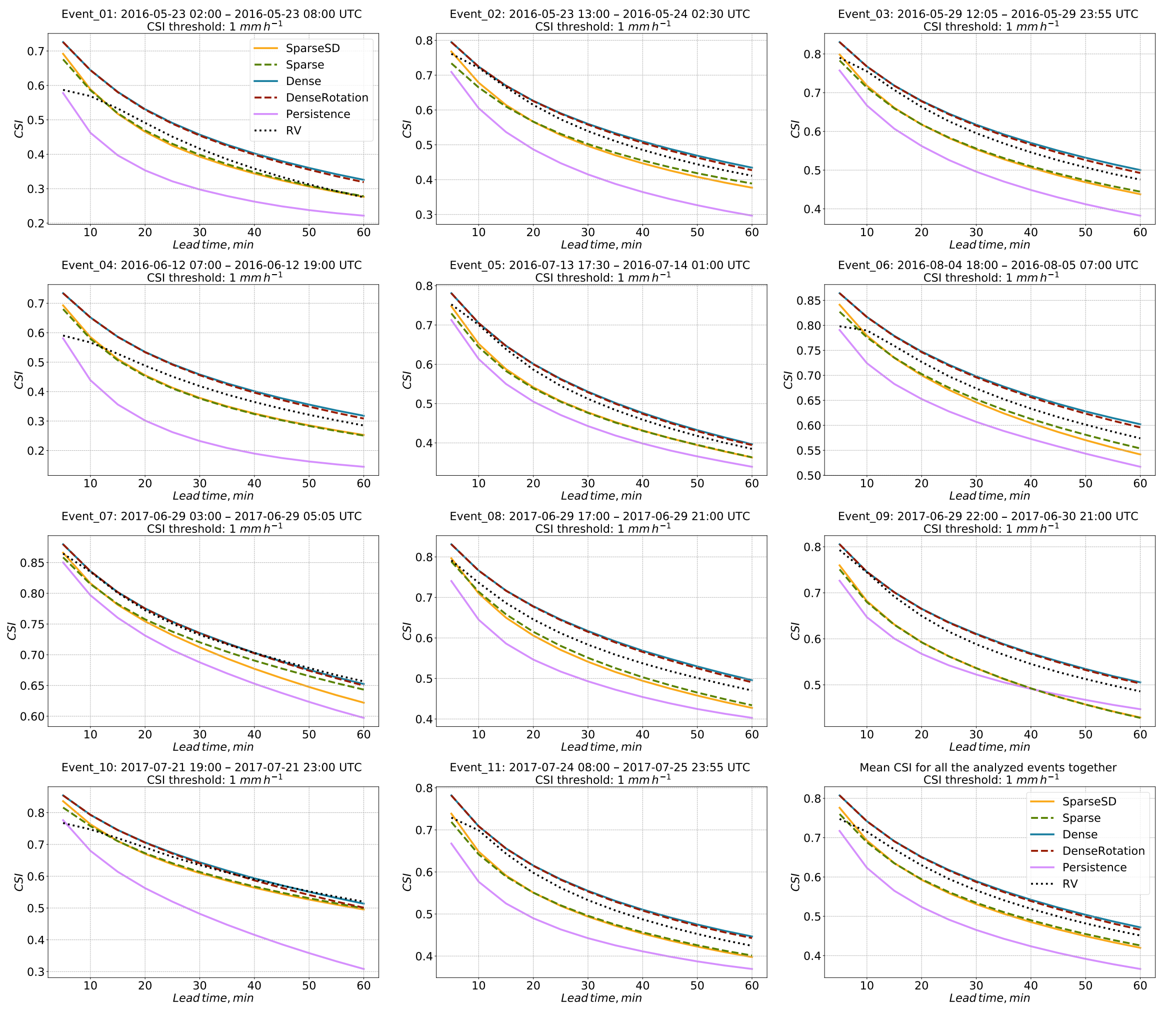

Figure 2.7: Verification of the different optical flow based nowcasts in terms of CSI for the threshold of 1 mm h^{-1} for 11 precipitation events over Germany.

2.5.2 Advection schemes properties and effectiveness

Within the Dense group of models, we could not find any significant difference between the performance and PSD of the constant-vector (Dense model) and the semi-Lagrangian scheme (DenseRotation). That confirms findings presented by Germann and Zawadzki (2002b) who found that the constant vector and the modified semi-Lagrangian schemes have very similar power spectra, presumably since they share the same interpolation procedure. The theoretical superiority of the semi-Lagrangian scheme might, however, materialize for other events with substantial, though persistent rotational motion. A more comprehensive analysis should thus be undertaken in future studies.

Interpolation is included in both the post-processing of image warping (Sparse models) and in the computation of gridded nowcasts as part of the Dense models. In general, such interpolation steps can lead to numerical diffusion and thus to the degradation or loss of small-scale features (Germann and Zawadzki, 2002b). Yet we were mostly able to contain such adverse effects for both the Sparse and the Dense group of models by carrying out only one interpolation step for any forecast at a specific lead time. We showed that numerical diffusion was negligible for lead times of up to 1 h for any model: however, as has been shown in Germann and Zawadzki (2002b), for longer lead times these effects can be significant, depending on the implemented extrapolation technique.

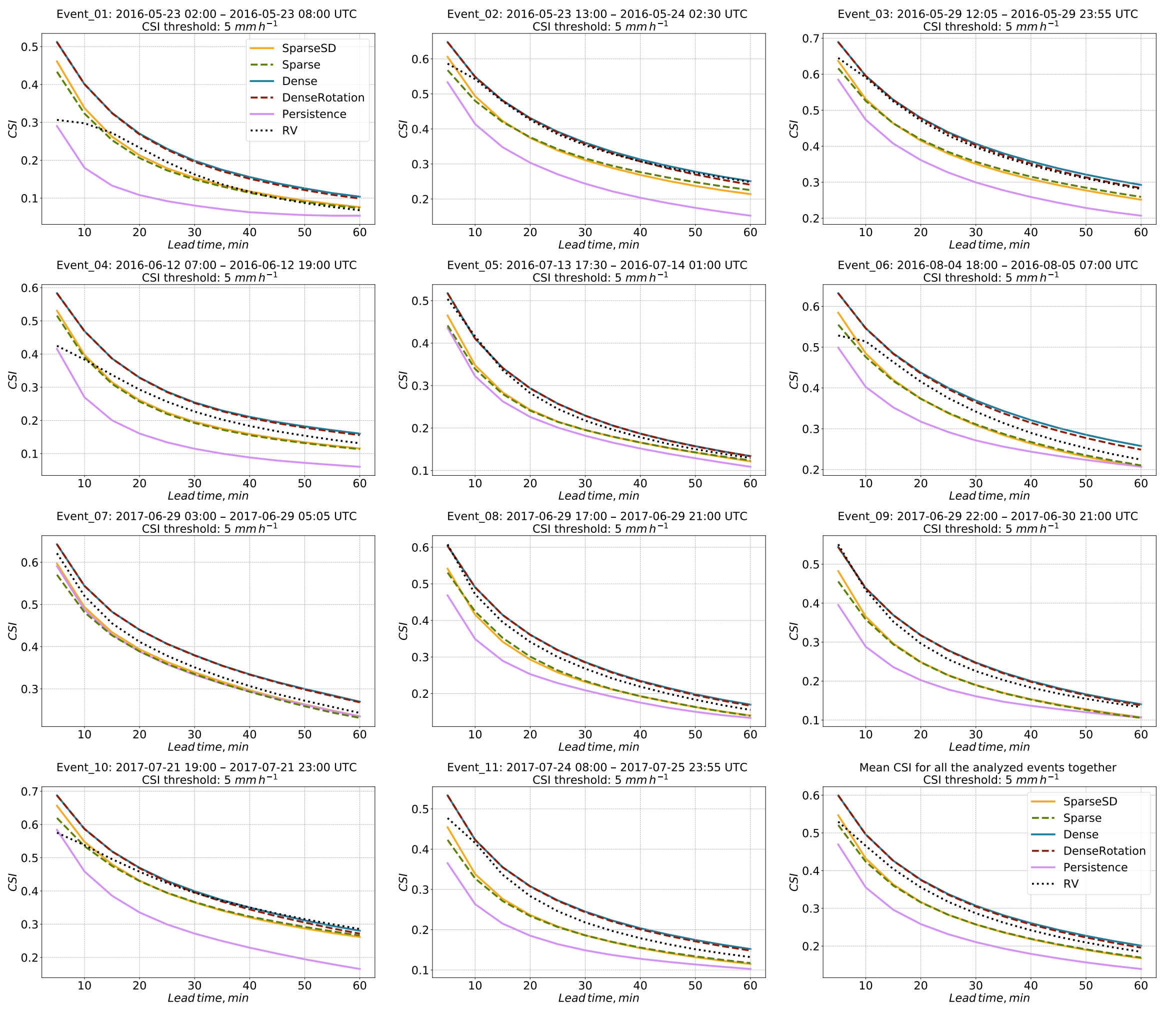

Figure 2.8: Verification of the different optical flow based nowcasts in terms of CSI for the threshold of 5 mm h^{-1} for 11 precipitation events over Germany.

2.5.3 Computational performance

Computational performance might be an important criterion for end users aiming at frequent update cycles. We ran our nowcasting models on a standard office PC with an Intel® Core™ i7-2600 CPU (eight cores, 3.4 GHz), and on a standard laptop with an Intel® Core™ i5-7300HQ CPU (four cores, 2.5 GHz). The average time for generating one nowcast for 1 h lead time (at 5 min resolution) is 1.5 – 3 s for the Sparse group and 6 – 12 s for the Dense group. The Dense group is computationally more expensive due to interpolation operations implemented for large grids (900×900 pixels). There is also potential for increasing the computational performance of the interpolation.

2.6 Summary and conclusions

Optical flow is a technique for deriving a velocity field from consecutive images. It is widely used in image analysis, and has become increasingly popular in meteorological applications over the past 20 years. In our study, we examined the performance of optical-flow-based models for radar-based precipitation nowcasting, as implemented in the open-source rainymotion library, for a wide range of rainfall events using radar data provided by the DWD.

Our benchmark experiments, including an operational baseline model (the RV product provided by the DWD), show a firm basis for using optical flow in radar-based precipitation nowcasting studies. For the majority of the analyzed events, models from the Dense group outperform the operational baseline. The Sparse group of models showed significant skill, yet they generally performed more poorly than both the Dense group and the RV product. We should, however, not prematurely discard the group of Sparse models before we have not gained a better understanding of error sources with regard to the tracking, extrapolation and warping steps. Combining the warping procedure for the extrapolation step with the Dense optical flow procedure for the tracking step (i.e., to advect corners based on a "Dense" velocity field obtained by implementing one of the dense optical flow techniques) might also be considered. This opens the way for merging two different model development branches in the future releases of the rainymotion library.

There is a clear and rapid model performance loss over lead time for events with high rainfall intensities. This issue continues to be unresolved by standard nowcasting approaches, but some improvement in this field may be achieved by using strategies such as merging with NWP results and stochastic modeling of rainfall field evolution. Admittedly, deterministic nowcasts in a Lagrangian framework account neither for precipitation intensity dynamics nor for the uncertainties in representing precipitation field motion. At least for the latter, the rainymotion library provides ample opportunities to experiment with forecast ensembles, based on various tracking and extrapolation techniques. Furthermore, we suppose that using new data-driven models based on machine and deep learning may increase the performance by utilizing and structuring common patterns in the massive archives of radar data.

We do not claim that the developed models will compete with well-established and excessively tuned operational models for radar-based precipitation nowcasting. Yet, we hope our models may serve as an essential tool for providing a fast, free, and open-source solution that can serve as a benchmark for further model development and hypothesis testing – a benchmark that is far more advanced than the conventional benchmark of Eulerian persistence.

Recent studies show that open-source community-driven software advances the field of weather radar science (Heistermann et al., 2015a,b). Just a few months ago, the pySTEPS (https://pysteps.github.io, last access: 28 March 2019) initiative was introduced "to develop and maintain an easy to use, modular, free and open source python framework for short-term ensemble prediction systems." As another piece of evidence of the dynamic evolution of QPN research over the recent years, these developments could pave the way for future synergies between the pySTEPS and rainymotion projects – towards the availability of open, reproducible, and skillful methods in quantitative precipitation nowcasting.

Table 2.3: Mean model metrics for different lead time periods.

Model	Lead time (from–to), min 5–30	35–60
	MAE, mm h^{-1}	
Dense	0.30	0.45
DenseRotation	0.30	0.45
RV	0.31	0.45
	CSI, threshold=0.125 mm h^{-1}	
Dense	0.78	0.64
DenseRotation	0.78	0.64
RV	0.76	0.61
	CSI, threshold=0.25 mm h^{-1}	
Dense	0.76	0.61
DenseRotation	0.76	0.61
RV	0.74	0.59
	CSI, threshold=0.5 mm h^{-1}	
Dense	0.73	0.57
DenseRotation	0.73	0.57
RV	0.70	0.55
	CSI, threshold=1 mm h^{-1}	
Dense	0.68	0.52
DenseRotation	0.68	0.51
RV	0.65	0.49
	CSI, threshold=5 mm h^{-1}	
Dense	0.42	0.24
DenseRotation	0.42	0.23
RV	0.39	0.22

Code and data availability

The rainymotion library is free and open-source. It is distributed under the MIT software license, which allows unrestricted use. The source code is provided through a GitHub repository (Ayzel et al., 2019b), the snapshot of the rainymotion v0.1 is also available on Zenodo: https://doi.org/10.5281/zenodo.2561583 (Ayzel, 2019), and the documentation is available on a website (https://rainymotion.readthedocs.io, last access: 28 March 2019). The DWD provided the sample data of the RY product, and it is distributed with the rainymotion repository to provide a real case and reproducible example of precipitation nowcasting.

Supplement

The supplement related to this article is available online at: https://doi.org/10.5194/gmd-12-1387-2019-supplement.

Acknowledgements

Georgy Ayzel was financially supported by Geo.X, the Research Network for Geosciences in Berlin and Potsdam. The authors thank Loris Foresti, Seppo Pulkkinen, and Remko Uijlenhoet for their constructive comments and suggestions that helped to improve the paper. We acknowledge the support of Deutsche Forschungsgemeinschaft (German Research Foundation) and the Open Access Publication Fund of Potsdam University.

Chapter 3

RainNet v1.0: a convolutional neural network for radar-based precipitation nowcasting

This chapter is published as:

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

Abstract

In this study, we present RainNet, a deep convolutional neural network for radar-based precipitation nowcasting. Its design was inspired by the U-Net and SegNet families of deep learning models which were originally designed for binary segmentation tasks. RainNet was trained to predict continuous precipitation intensities at a lead time of 5 min, using several years of quality-controlled weather radar composites provided by the German Weather Service (DWD). That data set covers Germany with a spatial domain of 900 km $\times$ 900 km, and has a resolution of 1 km in space and 5 min in time. Independent verification experiments were carried out on 11 summer precipitation events from 2016 to 2017. In order to achieve a lead time of 1 h, a recursive approach was implemented by using RainNet predictions at 5 min lead times as model inputs for longer lead times. In the verification experiments, trivial Eulerian persistence and a conventional model based on optical flow served as benchmarks. The latter is available in the rainymotion library and had previously been shown to outperform DWD's operational nowcasting model for the same set of verification events.

RainNet significantly outperforms the benchmark models at all lead times up to 60 min for the routine verification metrics mean absolute error (MAE) and the critical success index (CSI) at intensity thresholds of 0.125, 1, and 5 mm h^{-1}. However, rainymotion turned out to be superior in predicting the exceedance of higher intensity thresholds (here 10 and 15 mm h^{-1}). The limited ability of RainNet to predict heavy rainfall intensities is an undesirable property which we attribute to a high level of spatial smoothing introduced by the model. At a lead time of 5 min, an analysis of power spectral density confirmed a significant loss of spectral power at length scales of 16 km and below. Obviously, RainNet had learned an optimal level of smoothing to produce a nowcast at 5 min lead time. In that sense, the loss of spectral power at small scales is informative, too, as it reflects the limits of predictability as a function of spatial scale. Beyond the lead time of 5 min, however, the increasing level of smoothing is a mere artifact – an analogue to numerical diffusion – that is not a property of RainNet itself, but of its recursive application. In the context of early warning, the smoothing is particularly unfavorable since pronounced features of intense precipitation tend to get lost over longer lead times. Hence, we propose several options to address this issue in prospective research, including an adjustment of the loss function for model training, model training for longer lead times, and the prediction of threshold exceedance in terms of a binary segmentation task. Furthermore, we suggest additional input data that could help to better identify situations with imminent precipitation dynamics. The model code, pretrained weights, and training data are provided in open repositories as an input for such future studies.

3.1 Introduction

The term ”nowcasting” refers to forecasts of precipitation field movement and evolution at high spatiotemporal resolutions (1–10 min, 100–1000 m) and short lead times (minutes to a few hours). Nowcasts have become popular not only with a broad civil community for planning everyday activities; they are particularly relevant as part of early warning systems for heavy rainfall and related impacts such as flash floods or landslides. While the recent advances in high-performance computing and data assimilation significantly improved numerical weather prediction (NWP) (Bauer et al., 2015), the computational resources required to forecast precipitation field dynamics at very high spatial and temporal resolutions are typically prohibitive for the frequent update cycles (5–10 min) that are required for operational nowcasting systems. Furthermore, the heuristic extrapolation of precipitation dynamics that are observed by weather radars still outperforms NWP forecasts at short lead times (Lin et al., 2005; Sun et al., 2014). Thus, the development of new nowcasting systems based on parsimonious but reliable and fast techniques remains an essential trait in both atmospheric and natural hazard research.

There are many nowcasting systems which work operationally all around the world to provide precipitation nowcasts (Reyniers, 2008; Wilson et al., 1998). These systems, at their core, utilize a two-step procedure that was originally suggested by Austin and Bellon (1974), consisting of tracking and extrapolation. In the tracking step, a velocity is obtained from a series of consecutive radar images. In the extrapolation step, that velocity is used to propagate the most recent precipitation observation into the future. Various flavors and variations of this fundamental idea have been developed and operationalized over the past decades, which provide value to users of corresponding products. Still, the fundamental approach to nowcasting has not changed much over recent years – a situation that might change with the increasing popularity of deep learning in various scientific disciplines.

”Deep learning” refers to machine-learning methods for artificial neural networks with ”deep” architectures. Rather than relying on engineered features, deep learning derives low-level image features on the lowest layers of a hierarchical network, and increasingly abstract features on the high-level network layers as part of the solution of an optimization problem based on training data (LeCun et al., 2015). Deep learning began its rise from the field of computer science when it started to dramatically outperform reference methods in image classification (Krizhevsky et al., 2012) and machine translation (Sutskever et al., 2014), which was followed by speech recognition (LeCun et al., 2015). Three main reasons caused this substantial breakthrough in predictive efficacy: the availability of ”big data” for model training, the development of activation functions and network architectures that result in numerically stable gradients across many network layers (Dahl et al., 2013), and the ability to scale the learning process massively through parallelization on graphics processing units (GPUs). Today, deep learning is rapidly spreading into many data-rich scientific disciplines, and complements researchers’ toolboxes with efficient predictive models, including in the field of geosciences (Reichstein et al., 2019).

While expectations in the atmospheric sciences are high (see e.g., Dueben and Bauer, 2018; Gentine et al., 2018), the investigation of deep learning in radar-based precipitation nowcasting is still in its infancy, and universal solutions are not yet available. Shi et al. (2015) were the first to introduce deep learning models in the field of radar-based precipitation nowcasting: they presented a convolutional long short-term memory (ConvLSTM) architecture, which outperformed the optical-flow-based ROVER (Real-time Optical flow by Variational methods for Echoes of Radar) nowcasting system in the Hong Kong area. A follow-up study (Shi et al., 2017) introduced new deep learning architectures, namely the trajectory gated recurrent unit (TrajGRU) and the convolutional gated recurrent unit (ConvGRU), and demonstrated that these models outperform the ROVER nowcasting system, too. Further studies by Singh et al. (2017) and Shi et al. (2018) confirmed the potential of deep-learning models for radar-based precipitation nowcasting for different sites in the US and China. Most recently, Agrawal et al. (2019) introduced a U-Net-based deep learning model for the prediction of the exceedance of specific rainfall intensity thresholds compared to optical flow and numerical weather prediction models. Hence, the exploration of deep learning techniques in radar-based nowcasting has begun, and the potential to overcome the limitations of standard tracking and extrapolation techniques has become apparent. There is a strong need, though, to further investigate different architectures, to set up new benchmark experiments, and to understand under which conditions deep learning models can be a viable option for operational services.

In this paper, we introduce RainNet – a deep neural network which aims at learning representations of spatiotemporal precipitation field movement and evolution from a massive, open radar data archive to provide skillful precipitation nowcasts. The present study outlines RainNet's architecture and its training and reports on a set of benchmark experiments in which RainNet competes against a conventional nowcasting model based on optical flow. Based on these experiments, we evaluate the potential of RainNet for nowcasting, but also its limitations in comparison to conventional radar-based nowcasting techniques. Based on this evaluation, we attempt to highlight options for future research towards the application of deep learning in the field of precipitation nowcasting.

3.2 Model description

3.2.1 Network architecture

To investigate the potential of deep neural networks for radar-based precipitation nowcasting, we developed RainNet – a convolutional deep neural network (Fig. 3.1). Its architecture was inspired by the U-Net and SegNet families of deep learning models for binary segmentation (Badrinarayanan et al., 2017; Ronneberger et al., 2015; Iglovikov and Shvets, 2018). These models follow an encoder–decoder architecture in which the encoder progressively downscales the spatial resolution using pooling, followed by convolutional layers; and the decoder progressively upscales the learned patterns to a higher spatial resolution using upsampling, followed by convolutional layers. There are skip connections (Srivastava et al., 2015) from the encoder to the decoder in order to ensure semantic connectivity between features on different layers.

As elementary building blocks, RainNet has 20 convolutional, 4 max pooling, 4 upsampling, and 2 dropout layers and 4 skip connections. Convolutional layers aim to generate data-driven spatial features from the corresponding input volume using several convolutional filters. Each filter is a three-dimensional tensor of learnable weights with a small spatial kernel size (e.g., 3×3, and the third dimension equal to that of the input volume). A filter convolves through the input volume with a step-size parameter (or stride; stride $= 1$ in this study) and produces a dot product between filter weights and corresponding input volume values. A bias parameter is added to this dot product, and the results are transformed using an adequate activation function. The purpose of the activation function is to add nonlinearities to the convolutional layer output – to enrich it to learn nonlinear features. To increase the efficiency of convolutional layers, it is necessary to optimize their hyperparameters (such as number of filters, kernel size, and type of activation function). This has been done in a heuristic tuning procedure (not shown). As a result, we use convolutional layers with up to 1024 filters, kernel sizes of 1×1 and 3×3, and linear or rectified linear unit (ReLU; Nair and Hinton, 2010) activation functions.

Using a max pooling layer has two primary reasons: it achieves an invariance to scale transformations of detected features and increases the network's robustness to noise and clutter (Boureau et al., 2010). The filter of a max pooling layer slides over the input volume independently for every feature map with some step parameter (or stride) and resizes it spatially using the *maximum* (max) operator. In our study, each max pooling layer filter is 2×2 in size, applied with a stride of 2. Thus, we take the maximum of four numbers in the filter region (2×2) which downsamples our input volume by a factor of 2. In contrast to a max pooling layer, an upsampling layer is designed for the spatial upsampling of the input volume (Long et al., 2015). An upsampling layer operator slides over the input volume and fills (copies) each input value to a region that is defined by the upsampling kernel size (2×2 in this study).

Skip connections were proposed by Srivastava

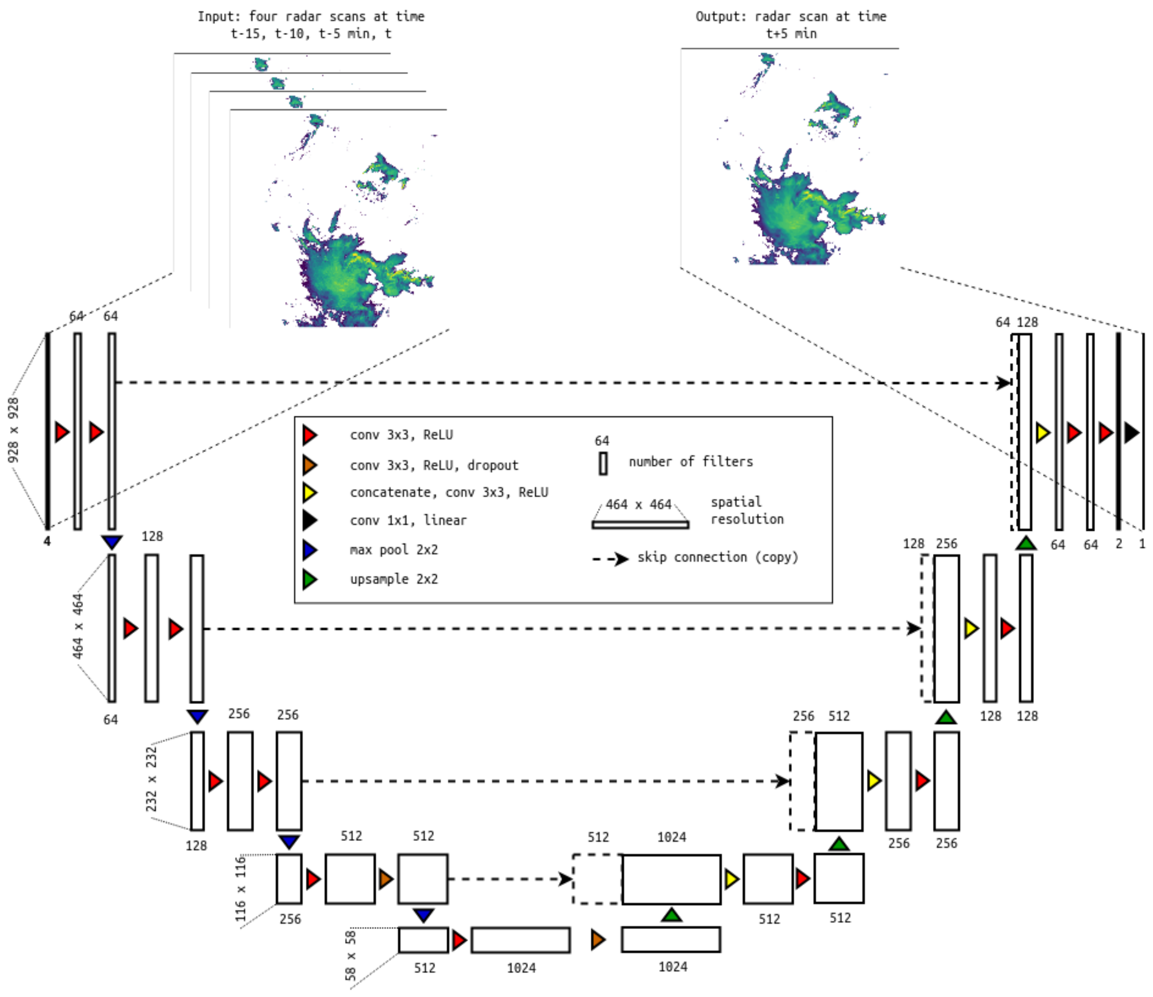

Figure 3.1: Illustration of the RainNet architecture. RainNet is a convolutional deep neural network which follows a standard encoder–decoder structure with skip connections between its branches. See main text for further explanation.

et al. (2015) in order to avoid the problem of vanishing gradients for the training of very deep neural networks. Today, skip connections are a standard group of methods for any form of information transfer between different layers in a neural network (Gu et al., 2018). They allow for the most common patterns learned on the bottom layers to be reused by the top layers in order to maintain a connection between different data representations along the whole network. Skip connections turned out to be crucial for deep neural network efficiency in recent studies (Iglovikov and Shvets, 2018). For RainNet, we use skip connections for the transition of learned patterns from the encoder to the decoder branch at the different resolution levels.

One of the prerequisites for U-Net-based architectures is that the spatial extent of input data has to be a multiple of 2^{n+1}, where n is the number of max pooling layers. As a consequence, the spatial extent on different resolution levels becomes identical for the decoder and encoder branches. Correspondingly, the radar composite grids were transformed from the native spatial extent of 900 cells $\times$ 900 cells to the extent of 928 cells $\times$ 928 cells using mirror padding.

RainNet takes four consecutive radar composite grids as separate input channels ($t-15$, $t-10$, $t-5$ min, and t, where t is the time of the nowcast) to produce a nowcast at time $t+5$ min. Each grid contains 928 cells $\times$ 928 cells with an edge length of 1 km; for each cell, the input value is the logarithmic precipitation depth as retrieved from the radar-based precipitation product. There are five almost symmetrical resolution levels for both decoder and encoder which utilize precipitation patterns at the full spatial input resolution of (x, y), at a quarter resolution $(x/2, y/2)$, at $(x/4, y/4)$, $(x/8, y/8)$, and $(x/16, y/16)$ respectively. To increase the robustness and to prevent overfitting of pattern representations at coarse resolutions, we implemented a dropout regularization technique (Srivastava et al., 2014). Finally, the output layer of resolution (x, y) with a linear activation function provides the predicted logarithmic precipitation (in mm) in each grid cell for $t + 5$ min.

RainNet differs fundamentally from ConvLSTM (Shi et al., 2015), a prior neural-network approach, which accounts for both spatial and temporal structures in radar data by using stacked convolutional and long short-term memory (LSTM) layers that preserve the spatial resolution of the input data alongside all the computational layers. LSTM networks have been observed to be brittle; in several application domains, convolutional neural networks have turned out to be numerically more stable during training and make more accurate predictions than these recurrent neural networks (e.g., Bai et al., 2018; Gehring et al., 2017).

Therefore, RainNet uses a fully convolutional architecture and does not use LSTM layers to propagate information through time. In order to make predictions with a larger lead time, we apply RainNet recursively. After predicting the estimated log-precipitation for $t + 5$ min, the measured values for $t - 10$, $t - 5$, and t as well as the estimated value for $t + 5$ serve as the next input volume which yields the estimated log-precipitation for $t + 10$ min. The input window is then moved on incrementally.

3.2.2 Optimization procedure

In total, RainNet has almost 31.4 million parameters. We optimized these parameters using a procedure of which we show one iteration in Fig. 3.2: first, we read a sample of input data that consists of radar composite grids at time $t - 15$, $t - 10$, $t - 5$ min, and t, and a sample of the observed precipitation at time $t + 5$. For both, input and observation, we increase the spatial extent to 928×928 using mirror padding and transform precipitation depth x (mm 5 min^{-1}) as follows (Eq. 3.1):

$$x_{transformed} = \ln(x_{raw} + 0.01) \tag{3.1}$$

Second, RainNet carries out a prediction based on the input data. Third, we calculate a loss function that represents the deviation between prediction and observation. Previously, Chen et al. (2018) showed that using the *logcosh* loss function is beneficial for the optimization of variational autoencoders (VAEs) in comparison to mean squared error. Accordingly, we employed the *logcosh* loss function as follows (Eq. 3.2):

$$Loss = \frac{\sum_{i=1}^{n} \ln(\cosh(now_i - obs_i))}{n} \tag{3.2}$$

$$\cosh(x) = \frac{1}{2}(e^x + e^{-x}) \tag{3.3}$$

where now$_i$ and obs$_i$ are nowcast and observation at the i-th location, respectively; $\cosh$ is the hyperbolic cosine function (Eq. 3.3), and n is the number of cells in the radar composite grid.

Fourth, we update RainNet's model parameters to minimize the loss function using a backpropagation algorithm where the Adam optimizer is utilized to compute the gradients (Kingma and Ba, 2015).

We optimized RainNet's parameters using 10 epochs (one epoch ends when the neural network has seen every input data sample once; then the next epoch begins) with a mini batch of size 2 (one mini batch holds a few input data samples). The optimization procedure converged on the eighth epoch, showing the saturation of RainNet's performance on the validation data. The learning rate of the Adam optimizer had a value of 1×10^{-1}, while other parameters had default values from the original paper of Kingma and Ba (2015).

The entire setup was empirically identified as the most successful in terms of RainNet's performance on validation data, while other configurations with different loss functions (e.g., mean absolute error, mean squared error) and optimization algorithms (e.g., stochastic gradient descent) also converged. The average training time on a single GPU (NVIDIA GeForce GTX 1080Ti, NVIDIA GTX TITAN X, or NVIDIA Tesla P100) varies from 72 to 76 h.

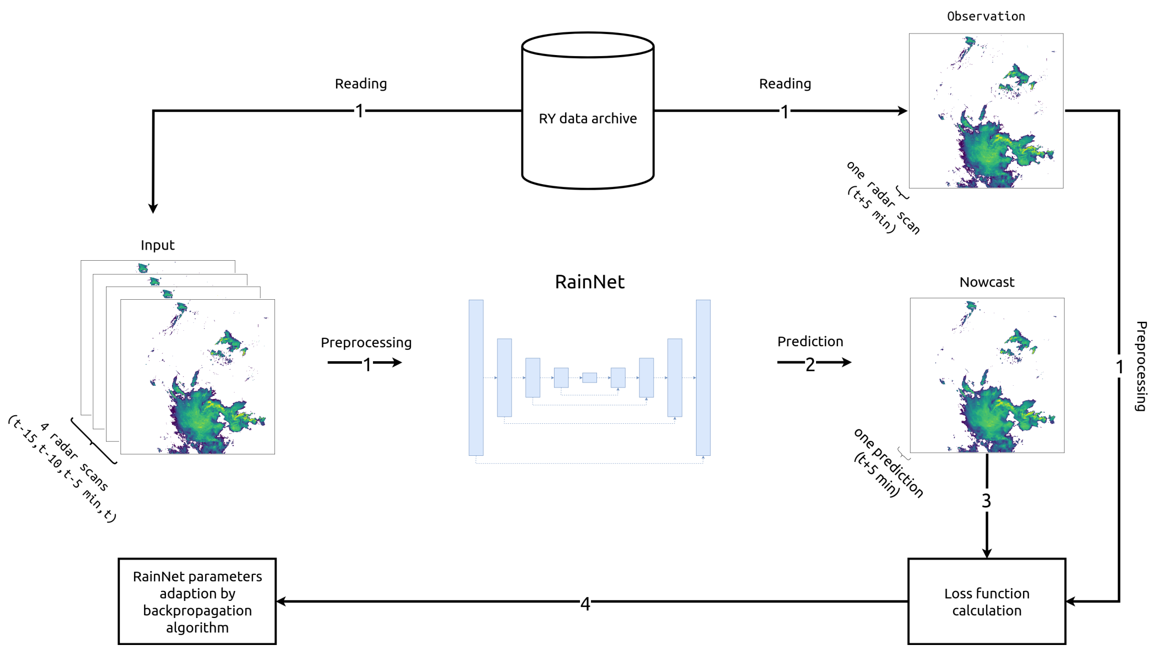

Figure 3.2: Illustration of one iteration step of the RainNet parameters optimization procedure.

We support this paper by a corresponding repository on GitHub (https://github.com/hydrogo/rainnet; last access: 10 June 2020; Ayzel, 2020a), which holds the RainNet model architecture written in the Python 3 programming language (https://python.org, last access: 28 January 2020) using the *Keras* deep learning library (Chollet et al., 2015) alongside its parameters (Ayzel, 2020b), which had been optimized on the radar data set described in the following section.

3.3 Data and experimental setup

3.3.1 Radar data

We use the RY product of the German Weather Service (DWD) as input data for training and validating the RainNet model. The RY product represents a quality-controlled rainfall-depth composite of 17 operational DWD Doppler radars. It has a spatial extent of 900 km $\times$ 900 km, covers the whole area of Germany, and is available since 2006. The spatial and temporal resolution of the RY product is 1 km $\times$ 1 km and 5 min, respectively.

In this study, we use RY data that cover the period from 2006 to 2017. We split the available RY data as follows: while we use data from 2006 to 2013 to optimize RainNet's model parameters and data from 2014 to 2015 to validate RainNet's performance, data from 2016 to 2017 are used for model verification (Sect. 3.3.3). For both optimization and validation periods, we keep only data from May to September and ignore time steps for which the precipitation field (with rainfall intensity more than 0.125 mm h^{-1}) covers less than 10 % of the RY domain. For each subset of the data – for optimization, validation, and verification –, every time step (or frame) is used once as t_0 (forecast time) so that the resulting sequences that are used as input to a single forecast (t_0 - 15 min, ..., t_0) overlap in time. The number of resulting sequences amounts to 41 988 for the optimization, 5722 for the validation, and 9626 for the verification (see also Sect. 3.3.3).

3.3.2 Reference models

We use nowcasting models from the rainymotion Python library (Ayzel et al., 2019a) as benchmarks with which we evaluate RainNet. As the first baseline model, we use Eulerian persistence (hereafter referred to as Persistence), which assumes that for any lead time n (min), precipitation at $t + n$ is the same as at forecast time t. Despite its simplicity, it is quite a powerful model for very short lead times, which also establishes a solid verification efficiency baseline which can be achieved with a trivial model without any explicit assumptions. As the second baseline model, we use the Dense model from the rainymotion library (hereafter referred to as Rainymotion), which is based on optical flow techniques for precipitation field tracking and the constant-vector advection scheme for precipitation field extrapolation. Ayzel et al. (2019a) showed that this model has an equivalent or even superior performance in comparison to the operational RADVOR (radar real-time forecasting) model from DWD for a wide range of rainfall events.

3.3.3 Verification experiments and performance evaluation

For benchmarking RainNet's predictive skill in comparison to the baseline models, Rainymotion and Persistence, we selected 11 events during the summer months of the verification period (2016–2017). These events were selected for covering a range of event characteristics with different rainfall intensity, spatial coverage, and duration. A detailed account of the events' properties is given by Ayzel et al. (2019a).

We use three metrics for model verification: mean absolute error (MAE), critical success index (CSI), and fractions skill score (FSS). Each metric represents a different category of scores. MAE (Eq. 3.4) corresponds to the continuous category and maps the differences between nowcast and observed rainfall intensities. CSI (Eq. 3.5) is a categorical score which is based on a standard contingency table for calculating matches between Boolean variables which indicate the exceedance of specific rainfall intensity thresholds. FSS (Eq. 3.6) represents neighborhood verification scores and is based on comparing nowcast and observed fractional coverages of rainfall intensities exceeding specific thresholds in spatial neighborhoods (windows) of certain sizes.

$$MAE = \frac{\sum_{i=1}^{n} |now_i - obs_i|}{n} \tag{3.4}$$

$$CSI = \frac{hits}{hits + false\ alarms + misses} \tag{3.5}$$

$$FSS = 1 - \frac{\sum_{i=1}^{n} (P_n - P_o)^2}{\sum_{i=1}^{n} P_n^2 + \sum_{i=1}^{n} P_o^2} \tag{3.6}$$

where quantities now$_i$ and obs$_i$ are nowcast and observed rainfall rate in the i-th pixel of the corresponding radar image and n is the number of pixels. Hits, false alarms, and misses are defined by the contingency table and the corresponding threshold value. Quantities P$_n$ and P$_o$ represent the nowcast and observed fractions, respectively, of rainfall intensities exceeding a specific threshold for a defined neighborhood size. MAE is positive and unbounded with a perfect score of 0; both CSI and FSS can vary from 0 to 1 with a perfect score of 1. We have applied threshold rain rates of 0.125, 1, 5, 10, and 15 mm h^{-1} for calculating the CSI and the FSS. For calculating the FSS, we use neighborhood (window) sizes of 1, 5, 10, and 20 km.

The verification metrics we use in this study quantify the models' performance from different perspectives. The MAE captures errors in rainfall rate prediction (the fewer the better), and CSI (the higher the better) captures model accuracy – the fraction of the forecast event that was correctly predicted – but does not distinguish between the sources of errors. The FSS determines how the nowcast skill depends on both the threshold of rainfall exceedance and the spatial scale (Mittermaier and Roberts, 2010).

In addition to standard verification metrics described above, we calculate the power spectral density (PSD) of nowcasts and corresponding observations using Welch's method (Welch, 1967) to investigate the effects of smoothing demonstrated by different models.

3.4 Results and discussion

For each event, RainNet was used to compute nowcasts at lead times from 5 to 60 min (in 5 min steps). To predict the precipitation at time $t+5$ min (t being forecast time), we used the four latest radar images (at time $t-15$, $t-10$, $t-5$ min, and t) as input. And since RainNet was only trained to predict precipitation at 5 min lead times, predictions beyond $t+5$ were made recursively: in order to predict precipitation at $t+10$, we considered the prediction at $t+5$ as the latest observation. That recursive procedure was repeated up to a maximum lead time of 60 min. Rainymotion uses the two latest radar composite grids ($t-5$, t) in order to retrieve a velocity field and then to advect the latest radar-based precipitation observation at forecast time t to $t+5$, $t+10$, ..., and $t+60$.

Figure 3.3 shows the routine verification metrics MAE and CSI for RainNet, Rainymotion, and Persistence as a function of lead time. The preliminary analysis had shown the same general pattern of model efficiency for each of the 11 events (Sect. S1 in the Supplement), which is why we only show the average metrics over all events. The results basically fall into two groups.

The first group includes the MAE and the CSI metrics up to a threshold of 5 mm h^{-1}. For these, RainNet clearly outperforms the benchmarks at any lead time (differences between models were tested to be significant with the two-tailed t test at a significance level of 5 %; results not shown). Persistence is the least skillful, as could be expected for a trivial baseline. The relative differences between RainNet and Rainymotion are more pronounced for the MAE than for the CSI. For the MAE, the advance of RainNet over Rainymotion increases with lead time. For the CSI, the superiority of RainNet over Rainymotion appears to be highest for intermediate lead times between 20 and 40 min. The performance of all models, in terms of CSI, decreases with increasing intensity thresholds.

That trend – a decreasing CSI with increasing intensity – continues with the second group of metrics: the CSI for thresholds of 10 and 15 mm h^{-1}. For both metrics and any of the competing methods at any lead time, the CSI does not exceed a value of 0.31 (obtained by RainNet at 5 min lead time and a threshold of 10 mm h^{-1}). That is below a value of $1/e \approx 0.37$ which had been suggested by Germann and Zawadzki (2002a) as a "limit of predictability" (under the assumption that the optimal value of the metric is 1 and that it follows an exponential-like decay over lead time). Irrespective of such an – admittedly arbitrary – predictability threshold, the loss of skill from an intensity threshold of 5 to 10 mm h^{-1} is remarkable for all competing models. Visually more apparent, however, is another property of the second group of metrics, which is that Rainymotion outperforms RainNet (except for a threshold of 10 mm h^{-1} at a lead times of 5 and 60 min). That becomes most pronounced for the CSI at 15 mm h^{-1}, while RainNet has a similar CSI value as Rainymotion at a lead time of 5 min, it entirely fails at predicting the exceedance of 15 mm h^{-1}) for longer lead times.

In summary, Fig. 3.3 suggests that RainNet outperforms Rainymotion (as a representative of standard tracking and extrapolation techniques based on optical flow) for low and intermediate rain rates (up to 5 mm h^{-1}). Neither RainNet nor Rainymotion appears to have much skill at predicting the exceedance of 10 mm h^{-1}, but the loss of skill for high intensities is particularly remarkable for RainNet which obviously has difficulties in predicting pronounced precipitation features with high intensities.

In order to better understand the fundamental properties of RainNet predictions in contrast to Rainymotion, we continue by inspecting a nowcast at three different lead times (5, 30, and 60 min), for a verification event at an arbitrarily selected forecast time (29 May 2016, 19:15:00 UTC). The top row of Fig. 3.4 shows the observed precipitation, and the second and third rows show Rainymotion and RainNet predictions. Since it is visually challenging to track the motion pattern at the scale of 900 km $\times$ 900 km by eye, we illustrate the velocity field as obtained from optical flow, which forms the basis for Rainymotion's prediction. While it is certainly difficult to infer the predictive performance of the two models from this figure, another feature becomes immediately striking: RainNet introduces a spatial smoothing which appears to substantially increase with lead time. In order to quantify that visual impression, we calculated, for the same example, the power spectral density (PSD) of the nowcasts and the corresponding observations (bottom row in Fig. 3.4), using Welch's method (Welch, 1967). In simple terms, the PSD represents the prominence of precipitation features at different spatial scales, expressed as the spectral power at different wavelengths after a two-dimensional fast Fourier transform. The power spectrum itself is not of specific interest here; it is the loss of power at different length scales, relative to the observation, that is relevant in this context. The loss of power of Rainymotion nowcasts appears to be constrained to spatial scales below 4 km, and does not seem to depend on lead time (see also Ayzel et al., 2019a). For RainNet, however, a substantial loss of power

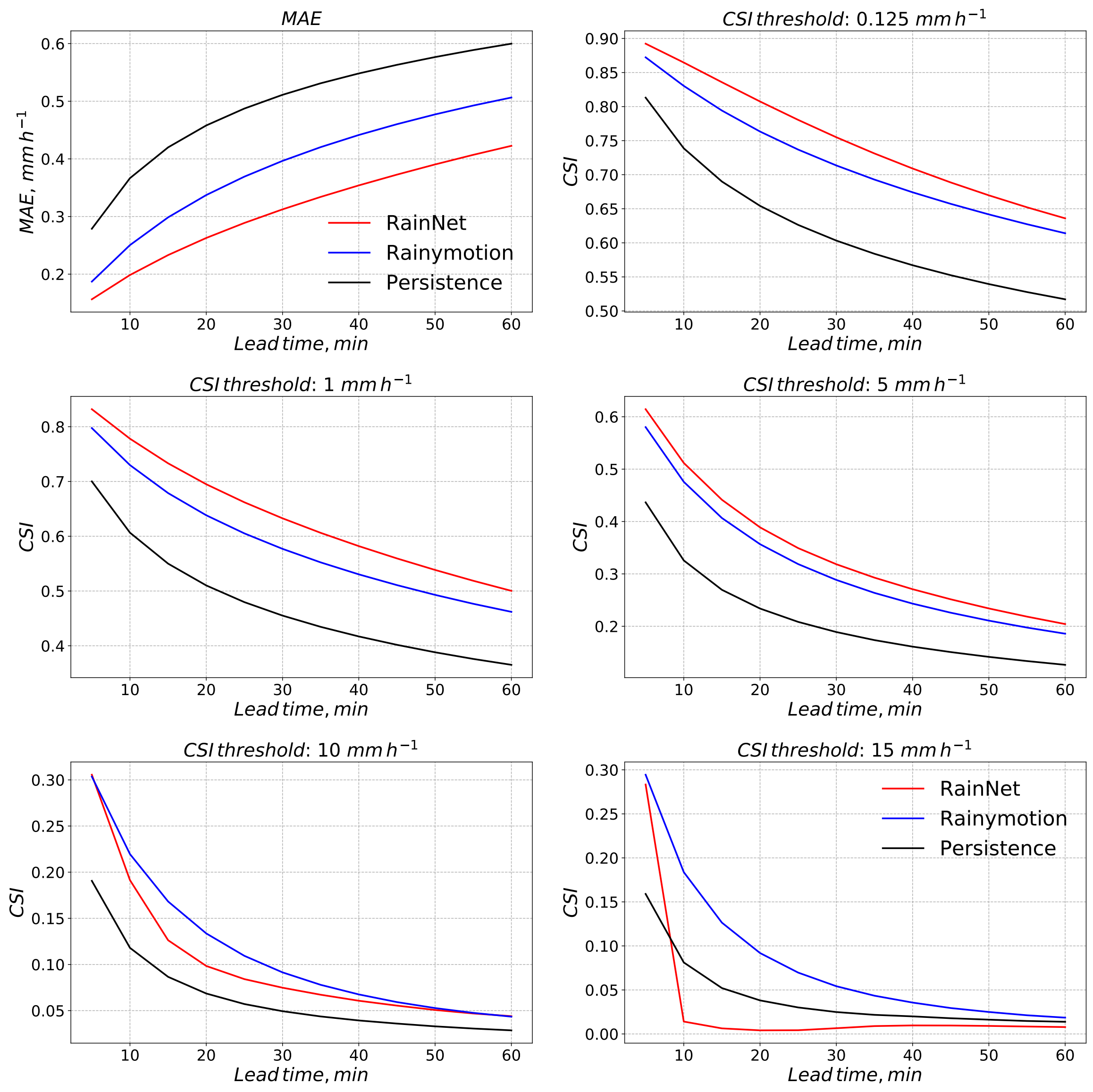

Figure 3.3: Mean absolute error (MAE) and critical success index (CSI) for five different intensity thresholds (0.125, 1, 5, 10, and 15 mm h^{-1}). The metrics are shown as a function of lead time. All values represent the average of the corresponding metric over all 11 verification events.

at length scales below 16 km becomes apparent at a lead time of 5 min. For longer lead times of 30 and 60 min, that loss of power grows and propagates to scales of up to 32 km. That loss of power over a range of scales corresponds to our visual impression of spatial smoothing.

In order to investigate whether that loss of spectral power at smaller scales is a general property of RainNet predictions, we computed the PSD for each forecast time in each verification event in order to obtain an average PSD for observations and nowcasts at lead times of 5, 30, and 60 min. The corresponding results are shown in Fig. 3.5. They confirm that the behavior observed in the bottom row of Fig. 3.4 is, in fact, representative of the entirety of verification events. Precipitation fields predicted by RainNet are much smoother than both the observed fields and the Rainymotion nowcasts. At a lead time of 5 min, RainNet starts to lose power at a scale of 16 km. That loss accumulates over lead time and becomes effective up to a scale of 32 km at a lead time of 60 min. These results confirm qualitative findings of Shi et al. (2015, 2018), who described their nowcasts as ”smooth” or ”fuzzy”.

RainNet obviously learned, as the optimal way

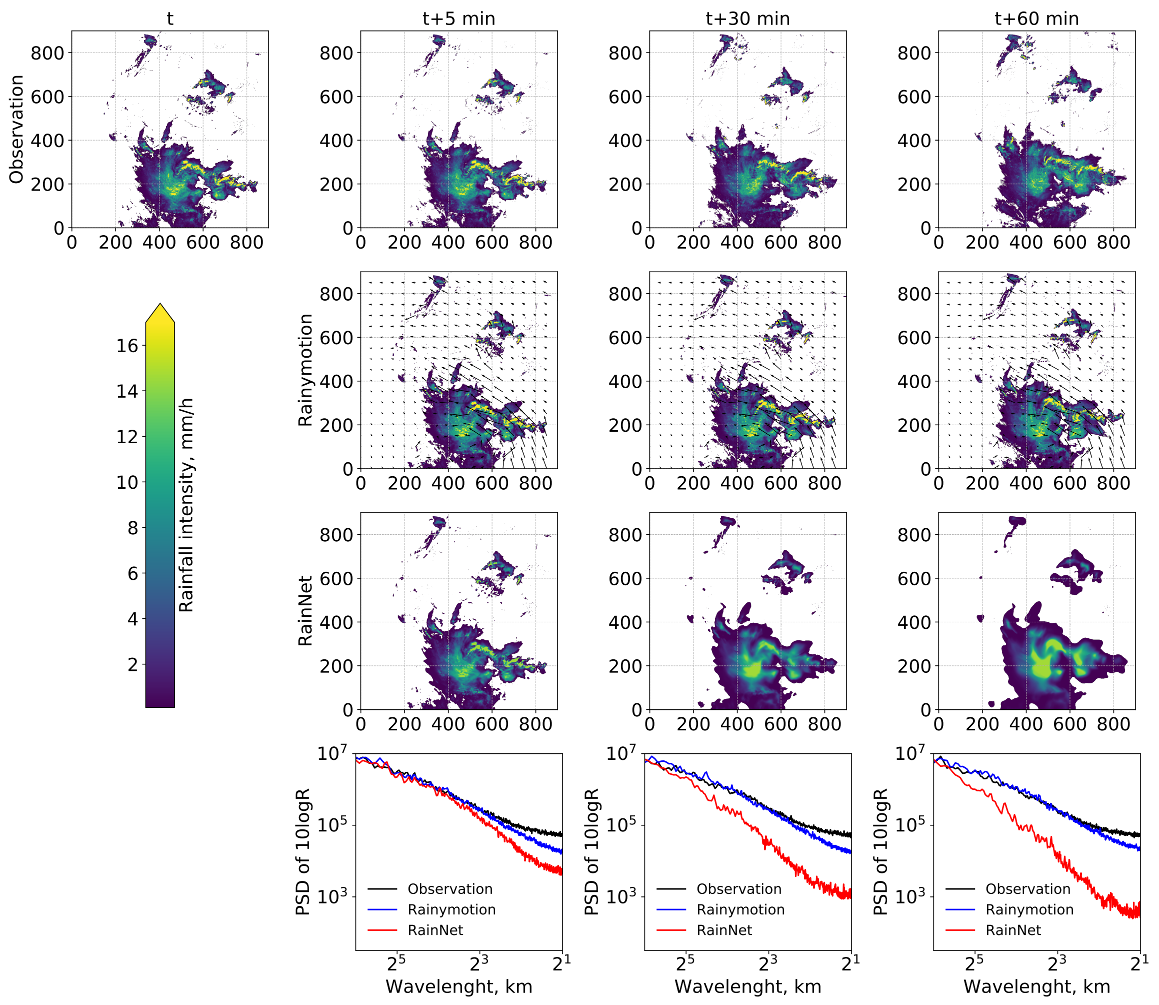

Figure 3.4: Precipitation observations as well as Rainymotion and RainNet nowcasts at $t =$ 29 May 2016, 19:15 UTC. Top row: observed precipitation intensity at time t, $t + 5$, $t + 30$, and $t + 60$ min. Ssecond row: corresponding Rainymotion predictions, together with the underlying velocity field obtained from optical flow. Bottom row: power spectral density plots for observations and nowcasts at lead times 5, 30 and 60 min.

to minimize the loss function, to introduce a certain level of smoothing for the prediction at time $t + 5$ min. It might even have learned to systematically ”attenuate” high intensity features as a strategy to minimize the loss function, which would be consistent with the results of the CSI at a threshold of 15 mm h^{-1}, as shown in Fig. 3.3. For the sake of simplicity, though, we will refer to the overall effect as ”smoothing” in the rest of the paper. According to the loss of spectral power, the smoothing is still small at a length scale of 16 km, but becomes increasingly effective at smaller scales from 2 to 8 km. It is important to note that the loss of power below length scales of 16 km at a lead time of 5 min is an essential property of RainNet. It reflects the learning outcome, and illustrates how RainNet factors in predictive uncertainty at 5 min lead times by smoothing over small spatial scales. Conversely, the increasing loss of power and its propagation to larger scales up to 32 km are *not* an inherent property of RainNet but a consequence of its recursive application in our study context: as the predictions at short lead times serve as model inputs for predictions at longer lead times, the results become increasingly smooth. So while the smoothing introduced at 5 min lead times can be interpreted as a direct result of the learning procedure, the cumulative smoothing at longer lead times has to rather be

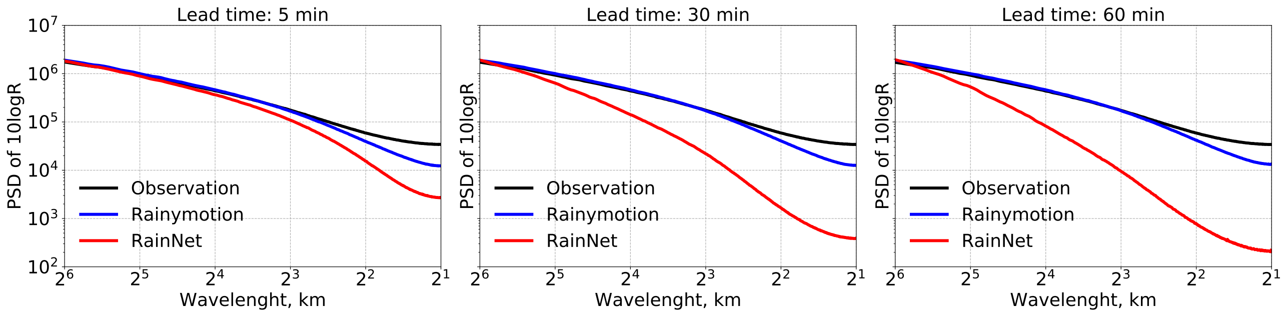

Figure 3.5: PSD averaged over all verification events and nowcasts, for lead times of 5, 30, and 60 min.

considered an artifact similar to the effect of ”numerical diffusion” in numerically solving the advection equation.

Given this understanding of RainNet's properties, we used the fractions skill score (FSS) to provide further insight into the dependency of predictive skill on the spatial scale. To that end, the FSS was obtained by comparing the predicted and observed fractional coverage of pixels (inside a spatial window / neighborhood) that exceed a certain intensity threshold (see Eq. 3.6 in Sect. 3.3.3). Figure 3.6 shows the FSS for Rainymotion and RainNet as an average over all verification events, for spatial window sizes of 1, 5, 10, and 20 km, and for intensity thresholds of 0.125, 1, 5, 10 and 15 mm h^{-1}. In addition to the color code, the value of the FSS is given for each combination of window size (scale) and intensity. In the case that one model is superior to the other, the correspondingly higher FSS value is highlighted in bold black digits.

Based on the above results and discussion of RainNet's versus Rainymotion's predictive properties, the FSS figures are plausible and provide a more formalized approach to express different behaviors of RainNet and Rainymotion in terms of predictive skill. In general, the skill of both models decreases with decreasing window sizes, increasing lead times, and increasing intensity thresholds. RainNet tends to outperform Rainymotion at lower rainfall intensities (up to 5 mm h^{-1}) at the native grid resolution (i.e., a window size of 1 km). With increasing window sizes and intensity thresholds, Rainymotion becomes the superior model. At an intensity threshold of 5 mm h^{-1}, Rainymotion outperforms RainNet at window sizes equal to or greater than 5 km. At intensity thresholds of 10 and 15 mm h^{-1}, Rainymotion is superior at any lead time and window size (except a window size of 1 km for a threshold of 10 mm h^{-1}).

The dependency of the FSS (or, rather, the difference of FSS values between Rainymotion and RainNet) on spatial scale, intensity threshold, and lead time is a direct result of inherent model properties. Rainymotion advects precipitation features but preserves their intensity. When we increase the size of the spatial neighborhood around a pixel, this neighborhood could, at some size, include high-intensity precipitation features that Rainymotion has preserved but slightly misplaced. RainNet's loss function, however, only accounts for the native grid at 1 km resolution, so it has no notion of what could be a slight or ”acceptable” displacement error. Instead, RainNet has learned spatial smoothing as an efficient way to factor in spatial uncertainty and minimize the loss function, resulting in a loss of high-intensity features. As discussed above, that effect becomes increasingly prominent for longer lead times because the effect of smoothing propagates.

3.5 Summary and conclusions

In this study, we have presented RainNet, a deep convolutional neural network architecture for radar-based precipitation nowcasting. Its design was inspired by the U-Net and SegNet families of deep learning models for binary segmentation, and it follows an encoder–decoder architecture in which the encoder progressively downscales the spatial resolution using pooling, followed by convolutional layers, and the decoder progressively upscales the learned patterns to a

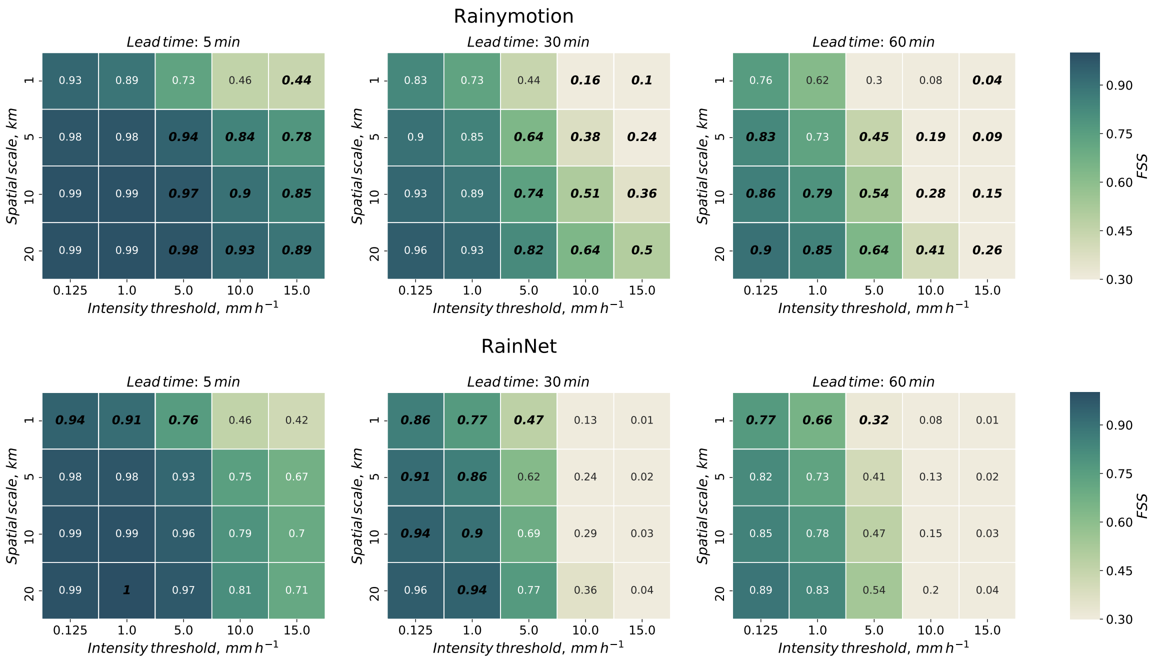

Figure 3.6: Fractions skill score (FSS) for Rainymotion (*top panel*) and RainNet (*bottom panel*), for 5, 30, and 60 min lead times, for spatial window sizes of 1, 5, 10 and 20 km, and for intensity thresholds of 0.125, 1, 5, 10 and 15 mm h^{-1}. In addition to the color code of the FSS, we added the numerical FSS values. The FSS values of the models which are significantly superior for a specific combination of window size, intensity threshold, and lead time are typed in bold black digits, and the inferior models are in regular digits.

higher spatial resolution using upsampling, followed by convolutional layers.

RainNet was trained to predict precipitation at a lead time of 5 min, using several years of quality-controlled weather radar composites based on the DWD weather radar network. Those data cover Germany with a spatial domain of 900 km $\times$ 900 km and have a resolution of 1 km in space and 5 min in time. Independent verification experiments were carried out on 11 summer precipitation events from 2016 to 2017. In order to achieve a lead time of 60 min, a recursive approach was implemented by using RainNet predictions at 5 min lead times as model inputs for longer lead times. In the verification experiments, Eulerian persistence served as a trivial benchmark. As an additional benchmark, we used a model from the rainymotion library which had previously been shown to outperform the operational nowcasting model of the German Weather Service for the same set of verification events.

RainNet significantly outperformed both benchmark models at all lead times up to 60 min for the routine verification metrics mean absolute error (MAE) and the critical success index (CSI) at intensity thresholds of 0.125, 1, and 5 mm h^{-1}. Depending on the verification metric, these results would correspond to an extension of the effective lead time in the order of 10–20 min by RainNet as compared to Rainymotion. However, Rainymotion turned out to be clearly superior in predicting the exceedance of higher-intensity thresholds (here 10 and 15 mm h^{-1}), as shown by the corresponding CSI analysis.

RainNet's limited ability to predict high rainfall intensities could be attributed to a remarkable level of spatial smoothing in its predictions. That smoothing becomes increasingly apparent at longer lead times. Yet, it is already prominent at a lead time of 5 min. That was confirmed by an analysis of power spectral density which showed, at time $t+5$ min, a loss of spectral power at length scales of 16 km and below. Obviously, RainNet has learned an optimal level of smoothing to produce a nowcast at 5 min lead times. In that sense, the loss of spectral power at small scales is informative as it reflects the limits of predictability as a function of spatial scale. Beyond the lead time of 5 min, however, the increasing level of smoothing is a mere artifact – an analogue to numerical diffusion – that is not a property of RainNet itself but of its recursive application: as we repeatedly use smoothed nowcasts as model inputs, we cumulate the effect of smoothing over time. That certainly is an undesirable property, and it becomes particularly unfavorable for the prediction of high-intensity precipitation features. As was shown on the basis of the fractions skill score (FSS), Rainymotion outperforms RainNet already at an intensity of 5 mm h^{-1}) once we start to evaluate the performance in a spatial neighborhood around the native grid pixel of 1 km $\times$ 1 km size. This is because Rainymotion preserves distinct precipitation features but tends to misplace them. RainNet, however, tends to lose such features over longer lead times due to cumulative smoothing effects – more so if it is applied recursively.

From an early warning perspective, that property of RainNet clearly limits its usefulness. There are, however, options to address that issue in future research.

- The loss function used in the training could be adjusted in order to penalize the loss of power at small spatial scales. The loss function explicitly represents our requirements to the model. Verifying the model by other performance metrics will typically reveal whether these metrics are rather in agreement or in conflict with these requirements. In our case, the *logcosh* loss function appears to favor a low MAE, but at the cost of losing distinct precipitation features. In general, future users need to be aware that, apart from the network design, the optimization itself constitutes the main difference to "heuristic" tracking-and-extrapolation techniques (such as Rainymotion) which do not use any systematic parameter optimization. The training procedure will stubbornly attempt to minimize the loss function, irrespective of what researchers consider to be "physically plausible". For many researchers in the field of nowcasting, that notion might be in stark contrast to experiences with "conventional" nowcasting techniques which tend to effortlessly produce at least plausible patterns.
- RainNet should be directly trained to predict precipitation at lead times beyond 5 min. However, preliminary training experiments with that learning task had difficulties to converge. We thus recommend to still use recursive predictions as model inputs for longer lead times during training in order to improve convergence. For example, to predict precipitation at time $t + 10$ min, RainNet could be trained using precipitation at time $t - 15$, $t - 10$, ..., t min as input, but using the recursive prediction at time $t + 5$ as an *additional* input layer. While the direct prediction of precipitation at longer lead times should reduce excessive smoothing as a result of numerical diffusion, we would still expect the level of smoothing to increase with lead time as a result of the predictive uncertainty at small scales.
- As an alternative to predicting continuous values of precipitation intensity, RainNet could be trained to predict the exceedance of specific intensity thresholds instead. That would correspond to a binary segmentation task. It is possible that the objective of learning the segmentation for *low* intensities might be in conflict with learning it for *high* intensities. That is why the training could be carried out both separately and jointly for disparate thresholds in order to investigate whether there are inherent trade-offs. From an early warning perspective, it makes sense to train RainNet for binary segmentation based on user-defined thresholds that are governed by the context of risk management. The additional advantage of training RainNet to predict threshold exceedance is that we could use its output directly as a measure of uncertainty (of that exceedance).

We consider any of those options worth pursuing in order to increase the usefulness of RainNet in an early warning context – i.e., to better represent precipitation intensities that exceed hazardous thresholds. We would expect the overall architecture of RainNet to be a helpful starting point.

Yet the key issue of precipitation prediction – the anticipation of convective initialization as well as the growth and dissipation of precipitation in the imminent future – still appears to be unresolved. It is an inherent limitation of nowcasting models purely based on optical flow: they can extrapolate motion fairly well, but they cannot predict intensity dynamics. Deep learning architectures, however, *might* be able to learn recurrent patterns of growth and dissipation, although it will be challenging to verify if they actually *did*. In the context of this study, though, we have to assume that RainNet has rather learned the representation of motion patterns instead of rainfall intensity dynamics: for a lead time of 5 min, the effects of motion can generally be expected to dominate over the effects of intensity dynamics, which will propagate to the learning results. The fact that we actually could recursively use RainNet's predictions at 5 min lead times in order to predict precipitation at 1 h lead times also implies that RainNet, in essence, learned to represent motion patterns and optimal smoothing. In that case, the trained model might even be applicable on data in another region which could be tested in future verification experiments.

Another limitation in successfully learning patterns of intensity growth and dissipation might be the input data itself. While we do not exclude the possibility that such patterns could be learned from just two-dimensional radar composites, other input variables might add essential information on imminent atmospheric dynamics – the predisposition of the atmosphere to produce or to dissolve precipitation. Such additional data might include three-dimensional radar volume data, dual-polarization radar moments, or the output fields of numerical weather prediction (NWP) models. Formally, the inclusion of NWP fields in a learning framework could be considered as a different way of assimilation, combining – in a data-driven way – the information content of physical models and observations.

Our study provides, after Shi et al. (2015, 2017, 2018), another proof of concept that convolutional neural networks provide a firm basis to compete with conventional nowcasting models based on optical flow (most recently, Google Research has also reported similar attempts based on a U-Net architecture; see Agrawal et al. (2019)). Yet this study should rather be considered as a starting point to further improve the predictive skill of convolutional neural networks and to better understand the properties of their predictions – in a statistical sense but also in how processes of motion and intensity dynamics are reflected. To that end, computational complexity and the cost of the training process still have to be considered as inhibitive, despite the tremendous progress achieved in the past years. RainNet's training would require almost a year on a standard desktop CPU in contrast to 3 d on a modern desktop GPU (although the latter is a challenge to implement for non-experts). Yet it is possible to run deep learning models with already optimized (pretrained) weights on a desktop computer. Thus, it is important to make available not only the code of the network architecture but also the corresponding weights, applicable using open-source software tools and libraries. We provide all this – code, pretrained weights, as well as training and verification data – as an input for future studies on open repositories (Ayzel, 2020a,b,c).

Code and data availability

The RainNet model is free and open source. It is distributed under the MIT software license which allows unrestricted use. The source code is provided through a GitHub repository https://github.com/hydrogo/rainnet (last access: 30 January 2020; Ayzel, 2020d); a snapshot of Rainnet v1.0 is also available at http://doi.org/10.5281/zenodo.3631038 (Ayzel, 2020a); the pretrained RainNet model and its weights are available at http://doi.org/10.5281/zenodo.3630429 (Ayzel, 2020b). DWD provided the sample data of the RY product; it is available at http://doi.org/10.5281/zenodo.3629951 (Ayzel, 2020c).

Supplement

The supplement related to this article is available online at: https://doi.org/10.5194/gmd-13-2631-2020-supplement.

Acknowledgements

Georgy Ayzel would like to thank the Open Data Science community (https://ods.ai, last access: 10 June 2020) for many valuable discussions and educational help in the growing field of deep learning. We ran our experiments using the GPU computation resources of the Machine Learning

Group of the University of Potsdam (Potsdam, Germany) and the Shared Facility Center ”Data Center of FEB RAS” (Khabarovsk, Russia). We acknowledge the support of Deutsche Forschungsgemeinschaft (German Research Foundation) and the open-access publication fund of the University of Potsdam.

Financial support

This research has been supported by Geo.X, the Research Network for Geosciences in Berlin and Potsdam (grant no. SO_087_GeoX).

Chapter 4

Quantifying the location error of precipitation nowcasts

This chapter is published as:

de Souza, A. C. T., Ayzel, G., and Heistermann, M.: Quantifying the location error of precipitation nowcasts, Adv. in Meteorology, 2020. (accepted, in print)

Abstract

In precipitation nowcasting, it is common to track the motion of precipitation in a sequence of weather radar images, and to extrapolate this motion into the future. The total error of such a prediction consists of an error in the predicted location of a precipitation feature and an error in the change of precipitation intensity over lead time. So far, verification measures did not allow to isolate the extent of location errors, making it difficult to specifically improve nowcast models with regard to location prediction. In this paper, we introduce a framework to directly quantify the location error. To that end, we detect and track scale-invariant precipitation features (corners) in radar images. We then consider these observed tracks as the true reference in order to evaluate the performance (or, inversely, the error) of any model that aims to predict the future location of a precipitation feature. Hence, the location error of a forecast at any lead time Δt ahead of the forecast time t corresponds to the Euclidean distance between the observed and the predicted feature location at $t + \Delta t$. Based on this framework, we carried out a benchmarking case study using one year worth of weather radar composites of the German Weather Service. We evaluated the performance of four extrapolation models, two of which are based on the linear extrapolation of corner motion from $t - 1$ to t (LK-Lin1) and $t - 1$ to t (LK-Lin4); another two are based on the Dense Inverse Search (DIS) method: motion vectors obtained from DIS are used to predict feature locations by linear (DIS-Lin1) and Semi-Lagrangian extrapolation (DIS-Rot1). Of those four models, DIS-Lin1 and LK-Lin4 turned out to be the most skillful with regard to the prediction of feature location while we also found that the model skill dramatically depends on the sinuosity of the observed tracks. The dataset of 376,125 detected feature tracks in 2016 is openly available to foster the improvement of location prediction in extrapolation-based nowcasting models.

4.1 Introduction

Forecasting precipitation for the imminent future (i.e., minutes to hours) is typically referred to as *precipitation nowcasting*. A common nowcasting technique is to track the motion of precipitation from a sequence of weather radar images and to extrapolate that motion into the future (Reyniers, 2008). For that purpose, we often assume that the intensity of precipitation features in the most recent image remains constant over the lead time period – an assumption commonly referred to as ”Lagrangian persistence” (Ayzel et al., 2019a). In Lagrangian *field tracking*, a velocity vector is obtained for each pixel of a precipitation field, and that vector field is used to extrapolate the motion of the entire precipitation field – as opposed to *cell tracking* in which contiguous high-intensity objects are tracked (see Pierce et al. 2012 for a discussion of both methods).

The present study focuses on nowcasts that are based on *field tracking*. The performance (or skill) of field tracking techniques is mostly verified by comparing the forecast precipitation field $F_{t+\Delta t}$

for time $t + \Delta t$ against the observed precipitation field $O_{t+\Delta t}$ at time $t + \Delta t$, where t is the forecast time and Δt is the lead time. A large variety of verification measures have been suggested in the literature (see, e.g., Baldwin and Kain, 2006; Ebert, 2008). Most of them, however, struggle with disentangling different sources of error: when we compare $F_{t+\Delta t}$ to $O_{t+\Delta t}$, how can we know the cause of the disagreement? Was it our prediction of the future location of a precipitation feature, or was it how precipitation intensity changed over time? Some verification scores, such as the Fractions Skill Score (Ayzel et al., 2020), apply a metric over spatial windows of increasing size in order to examine how the forecast performance depends on the spatial scale. Yet we still lack the ability to explicitly isolate and quantify the location error. This makes it difficult to benchmark and optimize the corresponding components of nowcast models.

In this study, we introduce an approach to directly quantify the location error of precipitation nowcasts that is based on the extrapolation of field motion. With *location error*, we refer to the spatial offset (or Euclidean distance) between the true and the forecast location of a precipitation feature (Fig. 4.1). In this context, the term "feature" does not refer to a contiguous object, but to a distinct *point* in the precipitation field, and we make use of the ability of the OpenCV library to detect and track the true motion of such distinct points. In a verification case study, we will demonstrate the ability to quantify the location error by benchmarking a set of routine extrapolation techniques for one year of quality-checked radar data in Germany.

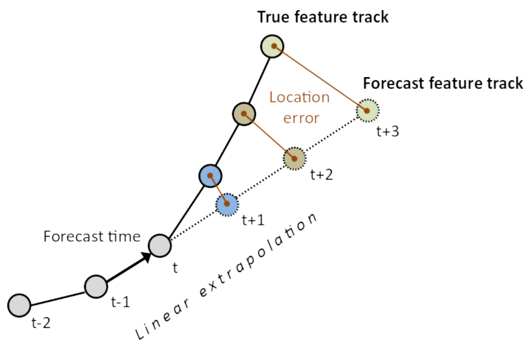

Figure 4.1: Illustration of the location error for a prediction at forecast time t that is based on the linear extrapolation of feature motion from $t-1$ to t.

Section 4.2 highlights the approach to quantify the location error, describes a set of tracking and extrapolation techniques based on optical flow, as well as the radar data for our case study. Section 4.3 presents the results of our case study, and Sect. 4.4 concludes.

4.2 Methods and data

4.2.1 Feature detection and tracking

We suggest quantifying the location error of a forecast by comparing the observed location (or displacement) of a precipitation feature against its predicted location. In visual computing, a feature is defined as a point that stands out in a local neighborhood and which is invariant in terms of scale, rotation, and brightness (Schmid et al., 2000). For a radar image, a feature (or corner) represents a point with a sharp gradient of rainfall intensity (Ayzel et al., 2019a).

In this study, features are detected using the approach of Shi and Tomasi (1994). If a feature is detected at one time step, we attempt to track that feature in any subsequent time step until it is no longer trackable. The feature tracking follows the approach of Lucas and Kanade (1981), as implemented by Bouguet (2000). The tracking *error* (or, inversely put, the robustness of tracking a feature from one radar image to the next) is quantified in terms of the minimum eigenvalue of a 2×2 normal matrix of optical flow equations (this matrix is called a spatial gradient matrix in Bouguet 2000), divided by the number of pixels in a neighborhood window. In the tracking step, that minimum eigenvalue has to exceed a threshold in order for a feature to be considered as successfully tracked. Table 4.1 provides an overview of parameters used for both feature detection and tracking. These values are based on the ones presented by Ayzel et al. (2019a). The underlying equations are well documented in OpenCV library (2020a).

In order to increase the robustness of track detection, the tracking was also performed backwards at each time step (Fig. 4.2): let p_t signify a feature which was identified at frame t and tracked to the next frame at time $t+1$ at the position p_{t+1}. The same tracking process was then applied backwards from the point p_{t+1} to time t, yielding the

Table 4.1: OpenCV function parameters used for feature detection and tracking

Parameter name	Value	Meaning
maxCorners	200	Maximum number of features
qualityLevel	0.2	Minimum accepted quality of features
minDistance	7	Minimal Euclidean distance between features
blockSize	21	Size of pixel neighborhood for covariance calculation
winSize	(20,20)	Size of the search window
maxLevel	2	Maximal number of pyramid levels

point p_t^{back}. Only the trajectories where the distance d_{back} between the source point p_t and the backwards tracked point p_t^{back} was less than 1 km (the grid resolution) were considered in our analysis.

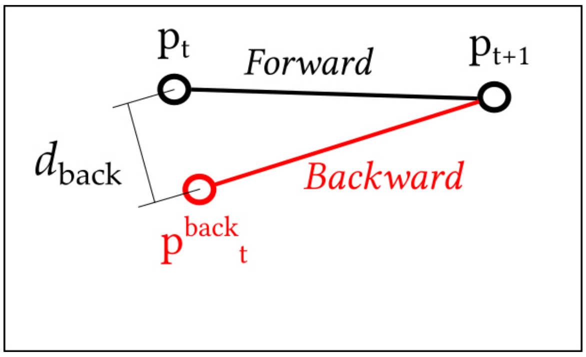

Figure 4.2: Illustration of the backward tracking test performed at each time step for all features.

To collect all feature tracks T in any given time period with a length of n time steps, we detect ”good-features-to-track” (Shi and Tomasi, 1994) at each time step $k \in [1, ..., n]$, and track these features over as many subsequent time steps as possible. Accordingly, each track $T_{i,j,k}$ could be identified by a unique tuple (i, j, k) that carries its starting point (by the grid's row and column indices, i and j) and its starting time index k. In this study, we use an analysis period of one year (2016, a leap year) and a time step length of 5 min, so that $k \in [1, ..., 105408]$.

In summary, the tracking process consists of six steps:

- Identify the features p_k^{new} using *goodFeaturesToTrack* (OpenCV library, 2020a) at any time k;
- If there are already features being tracked, p_k^{old}, from $k-1$ to k, we consider only those features p_k^{new} for which the distance to any feature p_k^{old} is greater than 7 km (with this threshold, we enforce consistency with the *minDistance* parameter of the Shi-Tomasi corner detection, see Tab. 4.1). The trackable features p_k are hence the union of p_k^{old} and p_k^{new};
- Track p_k from k to $k+1$ using *calcOpticalFlowPyrLK* (OpenCV library, 2020a);
- Backwards track (from $k+1$ to k) those features p_{k+1} that were obtained in step 3;
- Calculate the distance, d_{back}, from the features p_k to the backward-tracked locations resulting from step 4;
- Keep only those features p_{k+1} where the distance d_{back} is less than 1 km. These features are now p_{k+1}^{old}.

For statistical analysis, each track $T_{i,j,k}$ is characterized by its duration τ (the number of time steps over which the track persists), the overall displacement distance d of the feature along its track, the average feature velocity $v = d/\tau$, and the straightness of the feature's displacement in terms of the sinuosity index SI (which is calculated by dividing d by the Euclidean distance between the feature origin and end locations). The concept of sinuosity is widely used to characterize river curvatures as introduced by Mueller (1968) and was also applied to atmospheric science by Terry and Feng (2010) to quantify the sinuosity of typhoon tracks. In our analysis, we will also use the sinuosity index in order to understand the error of predicted feature locations.

4.2.2 The error of predicted locations

Let p be the true and P be the predicted location of a point feature in a Cartesian coordinate system. At forecast time t, p_t will be equal to P_t. Consider $P_{t+\Delta t} = f(p_t, \Delta t, \Psi_t)$ any function or algorithm that predicts the future location $P_{t+\Delta t}$ of point p_t from any set Ψ_t of predictors that is available at time t or before. In the context of our study, that set of predictors could be, for example, the previous locations p_{t-1}, p_{t-2}, ... of p_t. We then define the error of our prediction – henceforth referred to as *location error* ε – as the Euclidean distance between $P_{t+\Delta t}$ and $p_{t+\Delta t}$.

4.2.3 Extrapolation techniques

In a verification experiment, we can use our collection of tracks T in order to retrieve points p_t for which the location $P_{t+\Delta t}$ at $t + \Delta t$ should be predicted, points that could be used as predictors (Ψ_t), as well as the true location $p_{t+\Delta t}$ of the point at $t + \Delta t$. Assuming that an extrapolation of motion uses feature locations from m time steps before t, the minimum feature track length to produce a forecast would be $m + 1$. In order to retrieve the location error of such a prediction at time $t + \Delta t$, we would need a minimum track length of $m + \Delta t + 1$.

Based on the above terminology, we present in the following the extrapolation models analyzed in the present study. These models are based on the models that were also evaluated in a recent benchmarking study on optical-flow-based precipitation nowcasting (Ayzel et al., 2019a). Table 4.2 gives an overview of model acronyms and their main properties.

Eulerian persistence

As a trivial benchmark, we use the assumption of Eulerian persistence, meaning that the precipitation feature will simply remain at its position at forecast time, i.e., $P_{t+\Delta t} = p_t$.

Linear extrapolation

Linear extrapolation of feature motion assumes that a feature moves, over any lead time, at constant velocity and in the same direction. The displacement vector representing this motion can be obtained in different ways. These ways constitute three different models exemplified in the present study: LK-Lin1, LK-Lin4, and DIS-Lin1. In the case of LK-Lin1 and LK-Lin4, the displacement vector is obtained from "looking back" m time steps from forecast time t to previous feature locations at $t - m$ (tracked by using the Lucas–Kanade method, hence the LK label). For LK-Lin1, m equals 1, so the vector $v(t, p_t)$ to displace feature p_t is the connection from p_{t-1} to p_t; for LK-Lin4, m equals 4, so that the displacement vector results from the connection between p_{t-4} to p_t, where the length of the vector is divided by 4 in order to obtain the displacement velocity. Hence, a forecast at lead time Δt extends the vector $v(t, p_t)$ correspondingly. Please see Fig. 4.3 for an illustration of both the LK-Lin1 and the LK-Lin4 method. Of course, any other look-back time m could be used to obtain a displacement vector. In this study, we arbitrarily used $m \in \{1, 4\}$ in order to examine the effect of m on the forecast performance.

For the DIS-Lin1 model, a complete field of motion vectors $\mathrm{V_{DIS}}$ is obtained from the Dense Inverse Search (DIS) method (OpenCV library, 2020b; the underlying concept and equations of the DIS method have been elaborated by Kroeger et al. 2016), and then used for the extrapolation. A point p_t is linearly extrapolated from t to $t + n$ by n times the velocity vector $\mathrm{v_{DIS}(t, p_t)}$, where $\mathrm{v_{DIS}(t, p_t)}$ is the vector closest to p_t in the $\mathrm{V_{DIS}(t)}$ field (Fig. 4.4). $\mathrm{V_{DIS}(t)}$ is calculated by OpenCV's *cv2.DISOpticalFlow_create* function, which returns velocity vectors for each grid pixel based on the radar frames from $t - 1$ to t. In a recent benchmarking study about optical-flow-based precipitation nowcasting, Ayzel et al. (2019a) showed that the DIS-based model (referred to as the "Dense" model in that paper) is an effective method for radar-based precipitation nowcasting.

Table 4.2: Overview of extrapolation models

Name	Main approach	# time steps looking back
Persist	Eulerian persistence	0
LK-Lin1	Linear extrapolation based on Lukas Kanade	1
LK-Lin4	Linear extrapolation based on Lukas Kanade	4
DIS-Lin1	Linear extrapolation from DIS motion field	1
DIS-Rot1	Semi-Lagrangian extrapolation based on motion field obtained by Dense optical flow	1

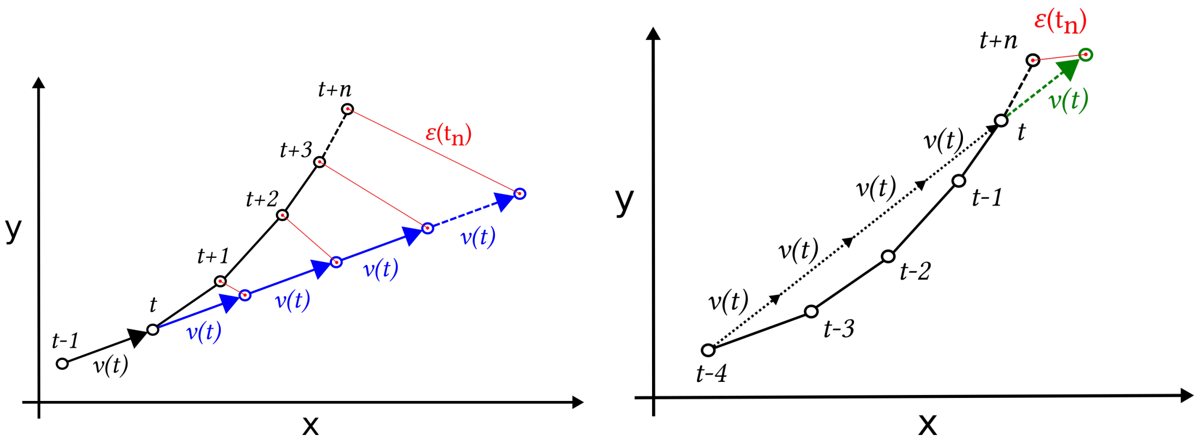

Figure 4.3: Illustration of the linear extrapolation schemes for the LK group: on the left LK-Lin1 and on the right LK-Lin4. The location error is displayed by $\varepsilon(t_n)$.

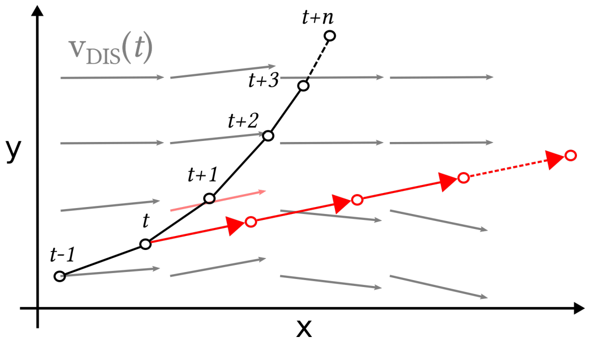

Figure 4.4: In the DIS-Lin1 model, the vector $\mathrm{v_{DIS}(t, p_t)}$ (light red arrow) obtained from $\mathrm{V_{DIS}(t)}$ is transferred to the p_t location and linearly extended to $t+n$.

Semi-Lagrangian approach based on dense optical flow

In a semi-Lagrangian approach, the motion field is typically assumed as constant over the forecast period and the feature trajectory is determined by following the streamlines (Germann and Zawadzki, 2002b). Following this concept, the DIS-Rot1 model (corresponding to ”Dense rotation” in Ayzel et al. 2019a) uses the two most recent radar images, $t-1$ and t, to estimate $\mathrm{V_{DIS}(t)}$ by *cv2.DISOpticalFlow_create* function. Similar to the DIS-Lin1 model, the displacement vector $\mathrm{v_{DIS}(t, p_t)}$ that is closest to p_t is used to extrapolate the motion of p_t from its position at t to $t+1$, providing the location of P_{t+1}. This process is repeated at all lead time steps until the maximum lead time is achieved. Hence, at each lead time step n, we retrieve the vector $\mathrm{v_{DIS}(t, P_{t+n})}$ that is closest to P_{t+n} in order to extrapolate the feature location, P_{t+n+1}. Accordingly, the velocity vector is updated at each lead time step from $\mathrm{V_{DIS}(t)}$, allowing for rotational or curved motion patterns (Fig. 4.5).

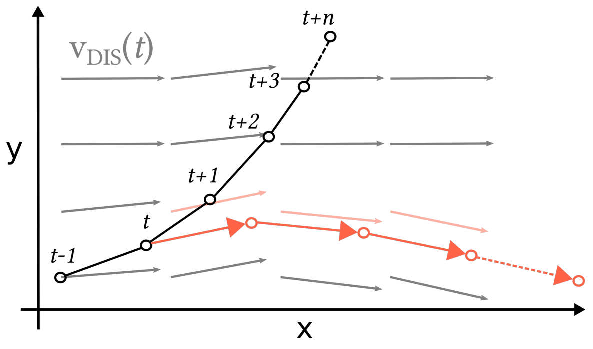

Figure 4.5: Schematic of the DIS-Rot1 model (orange path), where the velocity is updated every time step by transferring the velocity vector $\mathrm{v_{DIS}(t, p)}$ (light orange arrow) closest to p_t (black circles, for $t = 0$) or P_{t+1} (orange circles, for $t > 0$) in $\mathrm{V_{DIS}(t)}$, to the $P_{t+\Delta t}$ location to advect.

4.2.4 Weather radar data and experimental setup

Our benchmarking case study is based on weather radar data from the German Weather Service, namely, the RY product generated as part of the RADKLIM radar reanalysis of the German Weather Service, DWD (Winterrath et al., 2017). The RY product represents a quality-controlled national precipitation intensity composite from 18 C-Band radars covering Germany at 5 min intervals and a spatial resolution of 1 km at an extent of 1100 km $\times$ 900 km. The basis of the composite product is the so-called ”precipitation scans” from each of the 18 radar locations. The precipitation scan is designed to follow the horizon as closely as possible at an azimuth resolution of $1°$ and a radial resolution of 1 km, adjusting the elevation angle for each azimuth depending on the presence of mountains that would interfere with the beam propagation. Quality control includes a wide range of correction methods for e.g. clutter or partial beam blockage (see Winterrath et al. 2017 for details).

The year 2016, selected for this experiment, was characterized by an annual precipitation close to the climatological mean for most regions in Germany, as can be seen in the German Climate Atlas (DWD, 2020). However, the precipitation mean during autumn was below the normal average, and during the winter months slightly above the climatological mean.

As 2016 was a leap year, this experiment was carried out on 105408 radar composite images. Since none of the methods under evaluation requires any kind of training, there was no need to split the data into sets for calibration and validation. Instead, we used all tracks for verification. For each track, we always use, as forecast time t, a time of 20 min after the feature was detected for the first time. That is because our model LK-Lin4 needs to look back four time steps (i.e., 20 min) in order to make a forecast, and we need to make sure, for a fair comparison, to compare all models for the same forecast times.

4.2.5 Computational details

The analysis was carried out in a Python 3.6 environment using the following main open source libraries: NumPy (https://numpy.org, last access: 17 November 2020), NumExpr (https://github.com/pydata/numexpr, last access: 17 November 2020), and SciPy (https://www.scipy.org, last access: 17 November 2020) for general computations; OpenCV (https://opencv.org, last access: 17 November 2020) for feature tracking; as well as Pandas (https://pandas.pydata.org, last access: 17 November 2020), and h5py (https://www.h5py.org, last access: 17 November 2020).

4.3 Results and discussion

4.3.1 Properties of collected tracks

The identification and tracking process detected 376,125 features above the rainfall rate threshold of 0.2 mm h^{-1} and lasting over 20 min, which resulted in 337,776 eligible tracks after applying the extrapolation step. A track was considered as ”eligible” in case all models had a predicted location at all lead times, from t to $t + n$. The loss of 10.2 % that is implied by the above numbers was caused by the DIS group of models which did not generate a valid velocity vector $\mathrm{v_{DIS}(t, p_t)}$ near every p_t point, in the $\mathrm{V_{DIS}(t)}$ field, within a 3.5 km threshold.

Figure 4.6 gives an overview of the properties of the valid tracks. The figure also shows the seasonal dependency of these track properties by summarizing their distribution on a per month basis. We would like to emphasize that this analysis must not be interpreted as a ”climatology” of track properties as it only contains data from a single year. Still, we consider it as illustrative to investigate which properties tend to exhibit a seasonal pattern, and also to discuss whether the observed properties can be considered as representative for the governing rainfall processes in Germany.

In an average month of 2016, we identified and tracked 28,146 features (Fig. 4.6a). The largest number of tracks is found in the months from April to August (all above the average). Yet there is no continuous seasonal pattern in the number of detected tracks because e.g., January and October also show rather large counts.

No pattern at all can be found for the track length (Fig. 4.6b). With an average track length of 128 km, monthly maximum mean and median track lengths occur in January, April and September. A partly similar pattern can be found for the track duration which amounts to 207 min on average (Fig. 4.6c). This is plausible as we would, in general, expect the length of a track to increase with its duration. Yet there are also months – most notably the summer months from May to August – where this expectation is not met. And, of course, the length of a track not only depends on its duration, but also on a feature's velocity. The average feature velocity in 2016 amounted to a value of 42 km h^{-1}; and, in fact, velocity not only shows a clear seasonal pattern (with minimum velocities in the summer months, see Fig. 4.6d), but the seasonal pattern also helps us to understand where the patterns of track length and duration appear to be "inconsistent". For example, the track velocity is at a minimum in May and June, which decreases the length of track despite the rather high duration values for these two months.

The clearest seasonal pattern can be observed for rainfall intensity (Fig. 4.6e). That pattern is very much in line with our expectation as rainfall in the summer months is governed by convective events which tend to be more intense than stratiform event types. However, if we assumed that a higher rainfall intensity along a track is caused by the convective nature of the underlying event, the track duration in the corresponding months (e.g., May and June) is at least surprising: we would expect a convective event not only to be more intense, but also to be rather short (in comparison to wide-spread stratiform rainfall). The apparent inconsistency between the patterns of rainfall intensity and track duration points us to one of the key issues with the presented track inventory: we must not misinterpret a "track" as an "event" in a hydro-meteorological sense. The corner detection algorithm (see Sect. 4.2.1) searches for pronounced features in the sense of strong local gradients and tracks a feature for as long as it stands out. While we define a rainfall event as some coherent process in space and time, the tracking algorithm could "lose" a feature right in the course of an ongoing event, and maybe, at the same time, find another feature to track somewhere else in the field. Obviously, the tracking algorithm was able to track features over a longer duration in May, June and also September of 2016. However, as of now, we do not know which properties of the corresponding rainfall events caused that effect. We should just emphasize that the duration of a track does not necessarily correspond to the duration of an event. In the same way, we cannot expect the tracking algorithm to find features at "representative" locations of a convective cell. It will detect such features anywhere in a rainfall field where local gradients meet the tracking criteria. That could be right in the middle of heavy rainfall, but also at the edges. Hence, the reported precipitation intensities along the tracks will not be representative of the mean precipitation intensities of the corresponding precipitation fields.

Altogether, we have to emphasize at this point, that the seasonal track statistics are indeed plausible. But it must be clear that track statistics are not necessarily representative for "event" statistics. That notion might be irritating for those who have been defining and tracking features in terms of coherent rainfall objects over their lifetime from initiation to dissipation. A new feature track – as we understand it in our analysis – could be found right in the middle of an ongoing event, and it can be lost long before the actual rainfall "object" dissolves. However, that does not at all lessen the value of these tracks for the purpose of our analysis, which is to quantify the forecast location error based on well-defined and scale-invariant features.

Having said that, one final track property shown in Fig. 4.6f has not been discussed yet: the sinuosity index. As pointed out above (Sect. 4.2.1), the sinuosity index illustrates how much the shape of a track deviates from a straight line (which would correspond to a sinuosity index of 1).

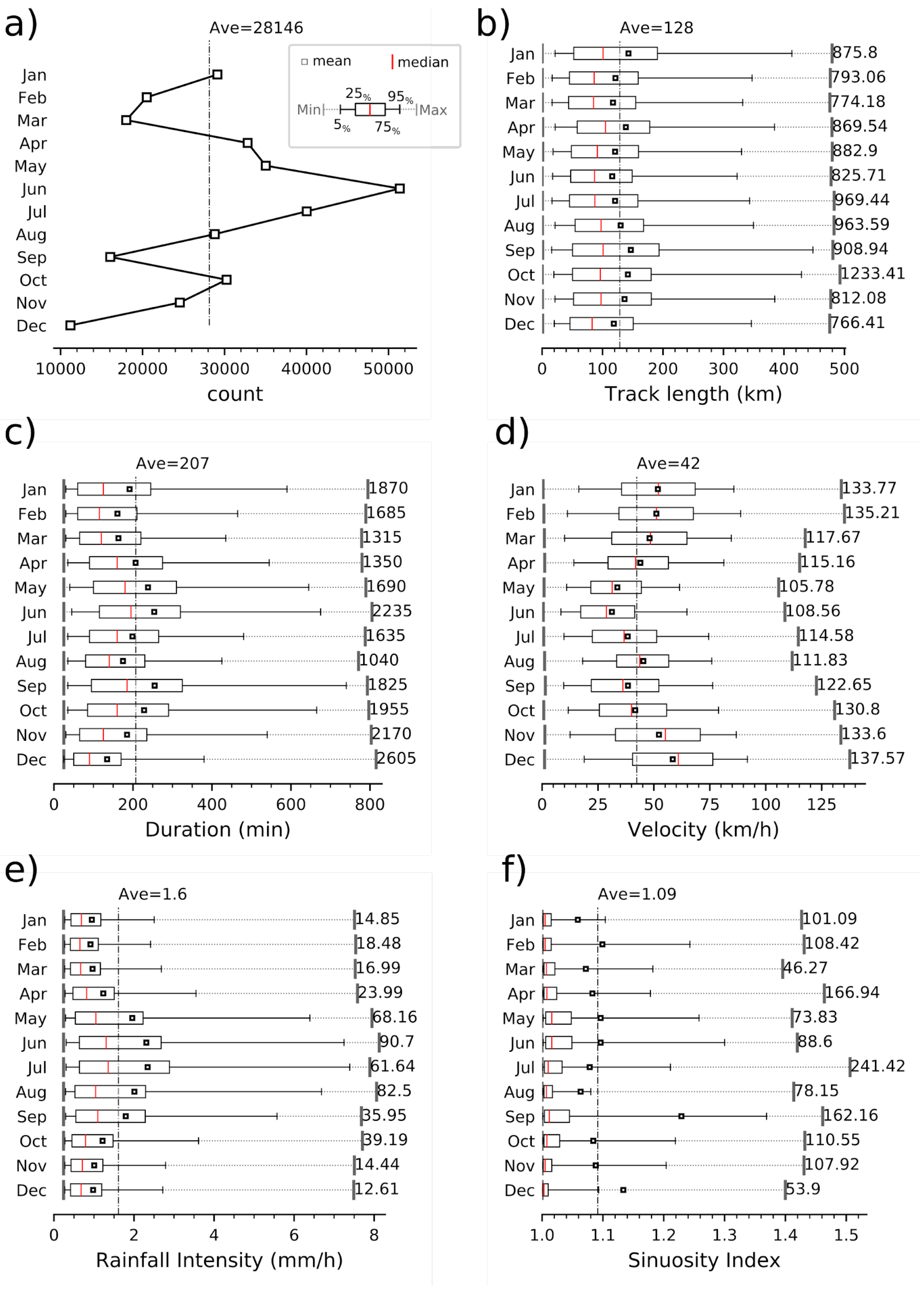

Figure 4.6: Statistical properties of detected tracks, organized by month: a) number of detected tracks, b) track length, c) track duration (time elapsed from detection and loss of a feature), d) feature velocity, e) rainfall intensity of a detected feature, f) sinuosity index of a track.

Figure 4.6f shows rather large sinuosity values for the summer months, May to September, but there is no obvious seasonal pattern. More strikingly, the distribution of the sinuosity index is very heavily tailed. The average value amounts to approx. 1.10 in the year 2016, which is, at the same time, the 90th percentile of the sinuosity values. That means, in turn, that the vast majority of tracks are rather straight, while the remaining tracks show all kinds of curved, meandering, twisted, or just erratic behavior.

Hence, before we systematically show the results of our verification experiment with regard to the location error (see Sect. 4.3.3), we would like to illustrate, in the following paragraph, the behaviour observed tracks in comparison to the forecast tracks under different sinuosity conditions.

4.3.2 Visual examples of observed and predicted tracks

Before we systematically evaluate the performance of different extrapolation techniques, we would like to provide some illustrative examples of observed versus predicted tracks. The selection of tracks for this illustration is arbitrary, and does not intend to be representative of the performance of any of the extrapolation methods. Instead, we aim to exemplify shapes of observed and predicted tracks under different sinuosity conditions in order to convey a better understanding of the various constellations that will finally be condensed into one single location error value.

Figure 4.7 shows a "gallery" of 11 observed tracks in different subplots (a-k). Each subplot also contains the tracks that were predicted by the different extrapolation models. Each dot represents one feature location in a 30 min time step, except the first one that represents the first prediction step at 5 min lead time. LK-Lin1 and LK-Lin4 infer the displacement vector directly from the feature positions at t and $t-1$ or t and $t-4$, respectively. As a reminder, DIS-Lin1 and DIS-Rot1 obtain the displacement vector of a feature from the DIS algorithm, a dense optical flow technique that produces motion fields based on the radar images at t and $t-1$; DIS-Lin1 extrapolates the closest vector linearly over the entire lead time while DIS-Rot1 uses a Semi-Lagrangian scheme in which the displacement vector is updated as the feature moves through the velocity field obtained from the DIS technique. Further details have been provided in Sect. 4.2.3. As in all forecasts of our verification experiment, the forecast time t corresponds to the 5th feature of the observed track. That is because the LK-Lin4 method needs to look four steps back in time ($t-4$) in order to produce a forecast, while the other methods only look back one step in time ($t-1$).

In order to convey a better idea about the rainfall patterns in the examples, the observed rainfall intensity at forecast time t is plotted as a background in grey scale. Furthermore, the sinuosity index and the track duration are printed in the corresponding subplots.

Please note that the duration of the observed tracks in Fig. 4.7 can extend over many hours, very long tracks were capped at a duration of 300 min for the purpose of plotting. Furthermore, the lead time of the predictions in the examples was set to the (capped) track duration minus 20 min (which corresponds to the period $t-4$ until forest time t). As a consequence, the lead times illustrated in Fig. 4.7 are mostly longer than the maximum lead time of 120 min which is used in our verification experiment (see next section). Hence, the first visual impression of Fig. 4.7 is dominated by the considerable errors that can occur for such long lead times. But of course, we should rather be aware of the behavior for shorter lead times up to 120 min. For that reason, the 120 min lead time is highlighted by a larger dot.

Not surprisingly, most of the competing methods appear to remain rather close to the observed track for short lead times of up to 30 min (except for example in subplot **j** in which the DIS-based methods entirely fail to capture the direction of feature movement). After that, the lead time over which the extrapolation models adequately predict the observed feature track varies, depending on the persistence of the motion behavior and the validity of the underlying model assumption. E.g., all models perform quite well for very long times in subplot **f**. In subplot **i**, the Semi-Lagrangian approach (DIS-Rot1) shows a clear advantage, while in subplots **c** and **k**, DIS-Rot1 is outperformed by all other models. Surely, there are several examples (**b**, **d**, **e**, **g**) in which all models entirely fail to anticipate the motion for lead times beyond 120 min.

As this compilation of examples is deliberately arbitrary, it does not provide a basis to infer the general superiority or inferiority of one or the

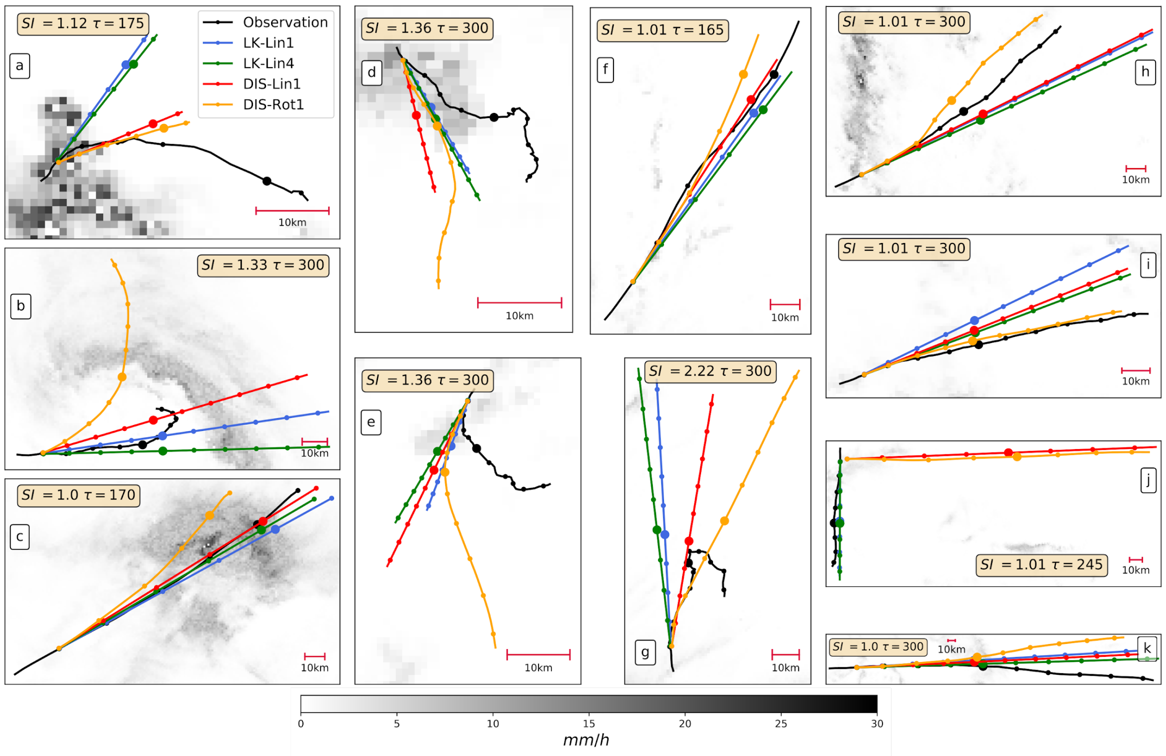

Figure 4.7: Compilation of forecast vs. observed tracks under different sinuosity conditions. Due to the different spatial extents of the windows, the scale of each subplot is different. Hence, a 10 km scale bar is provided for orientation. For each example, the observed track duration τ (in hours) and its sinuosity index SI are shown. The lead time of 120 min is highlighted by a larger dot. Some very long tracks have been capped at a maximum of 300 min for illustrative purposes.

other method. All models appear to struggle with predicting very sinuous tracks (subplots **b**, **d**, **e**, and **g**) which is what we would expect. But while the figure makes it difficult to compare the absolute location error between the examples (due to the different scales), it still appears that the absolute location error does not necessarily depend on the sinuosity. For example, the location error of LK-Lin1 after the maximum lead time (280 min) is higher in subplot **i** (almost straight, SI=1.01) than it is in subplot **d** (SI=1.36). In fact, straight tracks can imply a large error if the initial motion vector of a forecast method fails to represent the average long-term direction (see subplot **j** for a very impressive example). Then again, large errors can occur if a strong sinuosity of the track coincides with a large overestimation of the absolute velocity (e.g., subplots **b** and **g**). In that case, the linear extrapolation quickly departs from the track origin while the actual feature track meanders slowly and remains in the close vicinity of the origin. For such a scenario, the trivial persistence model (the feature just remains at the origin) will be superior even for short lead times.

Altogether, these different examples give us a better idea of how location errors can develop from both inadequate model assumptions (e.g., linear approximation vs. curved or sinuous conditions) and a failure to approximate the average motion from the initial feature locations. It is impossible, though, to diagnose the superiority of one or the other model from these examples. Hence, we will now systematically examine the results of our model verification experiment. We will not only analyze how the location error depends on lead time, but we will also investigate how the model performance relative to the persistence model depends on the sinuosity of the underlying tracks.

4.3.3 Systematic quantification of the location error

After having exemplified different observed and predicted tracks in the previous section, we now present the results of our benchmarking experiment. Figure 4.8 shows the distribution of locations errors for different models and lead times up to 120 min. For each lead time, the box plots specify mean, median, interquartile range, as well as the 5th and 95th percentile of the location error. For all models, the error quantiles increase slightly exponentially, but almost linearly with lead time. The rate at which the location error grows with lead time is, for all models, dramatically lower than for the persistence model; the mean error of persistence is higher than the mean error of any model at any lead time, which means that all models, *on average*, have positive skill at all lead times. For all models, the error distribution is obviously positively skewed, with the mean error being much higher than the median, and thus a heavy tail towards high location errors.

For very short lead times of up to 10 min, the mean error is about 1 km for all competing models except for persistence which is already up at more than 7 km after 10 min. After 60 min, the mean location error of *all* models exceeds a distance of 5 km, and 10 km after 110 min. For all models, at least 25 % of all forecasts exceed an error of 5 km after 50 min, and of 10 km after 90 min. After 75 min, at least 5 % of all forecasts exceed an error of 15 km.

Altogether, the location error can be substantial for a significant proportion of forecasts, while the median location error grows at a more moderate rate.

While this general pattern governs the behavior of all models, there are clear differences between the performance of the competing models. These differences, however, are not always coherent across all error quantiles and lead times, except for the DIS-Rot1 model which has the weakest performance of all models at virtually all lead times and for all quantiles, and the LK-Lin1 model which performs better than DIS-Rot1, but ranks second last. As for the best forecast performance, the LK-Lin4 and the DIS-Lin1 models take turns depending on error quantile and lead time: for the 5th and the 25th percentile, the LK-Lin4 model performs best for lead times up to 100 min, for the median up to 80 min, and for the mean up to 55 min. The DIS-Lin1 model shows the strongest changes of relative performance over lead time: as for the mean error, DIS-Lin1 starts to outperform LK-Lin4 at a lead time of 60 min, and continues this way until the maximum lead time of 120 min. As for the median error, DIS-Lin1 only catches up with LK-Lin4 after 90 min. For the 75th percentile, DIS-Lin1 outperforms LK-Lin4 after 50 min, for the 95th percentile already after 20 min. In summary, LK-Lin4 tends to outperform DIS-Lin1 in the first hour while DIS-Lin1 becomes superior in the second hour, apparently because it tends to avoid very high errors more efficiently than LK-Lin4.

In the following, we would like to better understand how model skill is affected by sinuosity. In Sect. 4.3.2, we have already indicated that the absolute values of location errors do not clearly depend on sinuosity. That was confirmed by the systematic verification experiment (results not shown). Yet the *difference* between an extrapolation model and the (trivial) persistence model might very well depend on sinuosity. In order to formally evaluate that hypothesis, we now examine the *skill* of our models more closely. Skill scores rate the score of a forecast in relation to the score of a reference forecast, in our case persistence. They are particularly useful in benchmark studies such as the present one. Eq. 4.1 shows the general definition of skill as derived from any forecast score, as well as the specific formula if we use the location error ε as the "score" (which becomes zero for a perfect forecast) and persistence as the "reference":

$$\begin{aligned} Skill = \frac{Score_{forecast} - Score_{reference}}{Score_{perfect} - Score_{reference}} = \\ = \frac{\varepsilon_{forecast} - \varepsilon_{persistence}}{-\varepsilon_{persistence}} \end{aligned} \quad (4.1)$$

We examine the forecast skill under different sinuosity conditions. As already pointed out in Sect. 4.3.1, the distribution of sinuosity is highly skewed and 90 % of observed tracks would pass as at least "rather straight" with a sinuosity index equal to or lower as 1.1. Hence, we split the forecasts into three unequal groups, depending on quantiles of the sinuosity index: The first group contains the "straight" 90 % of the forecasts with a sinuosity index below 1.1. We consider the

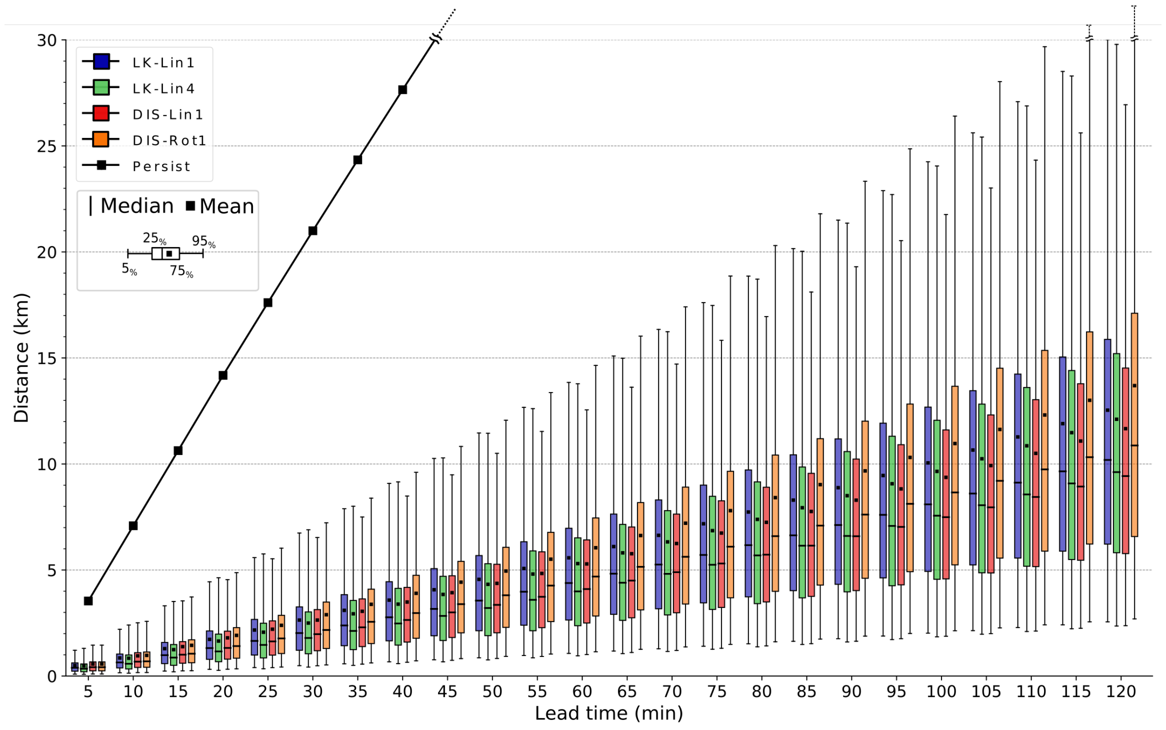

Figure 4.8: The distribution of location errors for different extrapolation models and lead times.

value of 1.1 as an – admittedly – arbitrary threshold between ”rather straight” and ”rather winding” tracks. The remaining 10% of tracks are split in two equally sized groups, again based on sinuosity: The 5 % with the highest sinuosity, exceeding an SI value of 1.2, could be labelled as ”twisted”, and the remaining 5 % with intermediate SI values between 1.1 and 1.2 could be labeled as ”winding”. Figure 4.9 shows the average model’s skill over every lead time for these three sinuosity classes. Clearly, the model skill dramatically varies between these three groups: it ranges between 0.79 and 0.87 for the ”straight” category, mostly between 0.5 and 0.65 for the ”winding” category, and between mostly 0 and 0.5 for the ”twisted” category. This decrease of skill with increasing sinuosity is well in line with our expectation. Furthermore, the ranking of all models based on skill is quite coherent across all categories, and also consistent with our previous analysis of location errors. DIS-Lin1 becomes superior within the second forecast hour, while LK-Lin1 performs better in the first forecast hour. Only in the ”twisted” category, LK-Lin1 and, even more, LK-Lin4 outperform DIS-Lin1 across all lead times. It should be noted, though, that the overall skill in the twisted category is very low for all competing models. In the ”winding” category, LK-Lin1 slightly outperforms LK-Lin4 in the first 20 min. Finally, DIS-Rot1 performs worst at all lead times in all categories.

The change of model skill with lead time should be interpreted with care, as it depends both on the performance of the extrapolation model itself and on the location error of the persistence model. For most models and SI categories, the skill appears to reach an optimum at some lead time which implies that the superiority of the model over persistence reaches a maximum.

4.4 Conclusions

In this paper, we have introduced a framework to isolate and quantify the location error in precipitation nowcasts that are based on field-tracking techniques. While it is often assumed that errors in precipitation nowcasts are dominated by the temporal dynamics of precipitation intensity, the location error of predicted precipitation features has so far not been explicitly and formally quantified.

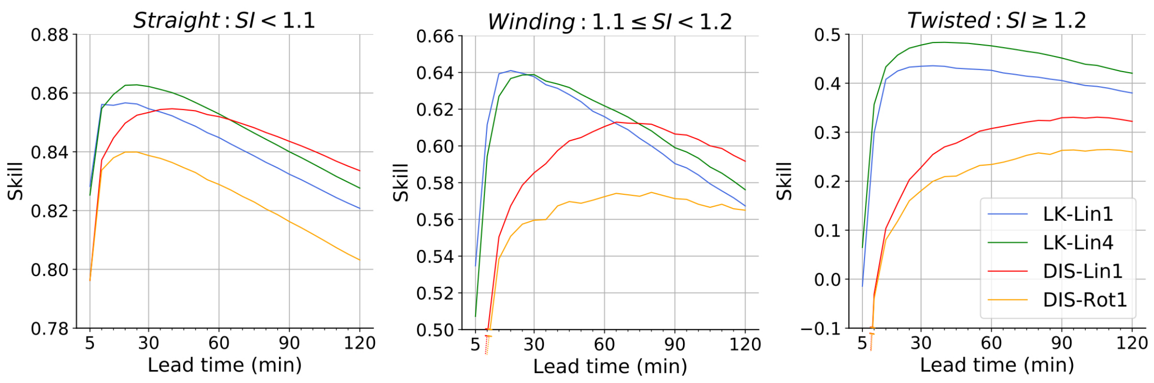

Figure 4.9: The mean model skill over each lead time with regard to location prediction for different extrapolation models and sinuosity conditions. Please note that the very low skill values of the DIS-based models at 5 min lead time (in the winding and twisted groups) are hidden by the scaling of the y-axis. At 5 min lead time, both models only have a skill of about 0.35 (winding) and -0.55 (twisted).

The main idea of our framework is to detect and track scale-invariant precipitation features (corners) in radar images. In our study, we detected features by using the approach of Shi and Tomasi (1994), and tracked these features following the approach of Lucas and Kanade (1981), using both algorithms as implemented in the OpenCV library. We increased the robustness of extracted feature tracks by making sure that the features can be successfully tracked forwards and backwards. That approach, together with a rather strict definition of parameter values for feature detection and tracking, increases our confidence in the reliability of the detected tracks. Still, we have to assume that the feature locations themselves are, as any measurement, uncertain. We expect the main sources of uncertainty to be the grid resolution (which does not allow to resolve errors below 1 km), and complex small scale intensity dynamics that can interfere with motion patterns. For future studies, we suggest a comprehensive sensitivity analysis with regard to the parameters of the feature detection and tracking algorithms in order to better understand the effects on both the number and the robustness of detected tracks in the context of rainfall motion analysis. Still, we assume that the error of extrapolating feature motion is substantially larger than the error of feature tracking itself. In summary, we consider it warranted to use the observed tracks as a reference in order to evaluate the performance (or, inversely, the error) of any model that aims to predict the future locations of such precipitation features. For that purpose, we defined the location error of a forecast at any lead time Δt ahead of the forecast time t as the Euclidean distance between the observed and the predicted feature location at $t + \Delta t$.

One might want to use this approach to comprehensively quantify the location error of any forecast model for the full spatial domain of a forecast grid, e.g., a national radar composite. In such a case, we would need to assume that the average of forecast errors that we have quantified from observed feature locations in a forecast domain is representative for the average error of all location predictions in that domain. We have not yet investigated the validity of that assumption. One might argue that the behavior of locations identified as ”corners” or ”good features to track” might not be representative for the motion behavior of the entire precipitation field; however, it will be difficult to find evidence to either verify or falsify such a hypothesis, as it would require another independent way to quantify the location error. Still, we are convinced that the proposed framework is useful: even without the need of strong assumptions on representativeness, the framework allows us to compare and benchmark the ability of different models to forecast future locations of precipitation features, and thus to specifically focus on improving that ability by future model development.

The hypothesis that such further model developments are urgently required is supported by the results of our benchmarking study. It should be clarified again that this benchmark study does not intend to suggest better extrapolation models, but to demonstrate the ability of our framework to unravel the location errors that are produced by state-of-the-art extrapolation methods. For that purpose, we compared four models: two models use the feature locations before and at forecast time t in order to derive displacement vectors which are then used to linearly extrapolate feature movement over the lead time. Model LK-Lin1 uses the feature locations at t and $t-1$, and LK-Lin4 uses the feature locations at t and $t-4$. The other two models are based on the dense optical flow algorithm DIS that generates a full motion vector field under various smoothness constraints. The model DIS-Lin1 obtains the displacement vector for a feature at t from the nearest motion vector in the field based on the radar images at times t and $t-1$, and uses that vector over the entire lead time. DIS-Rot1, in contrast, uses a Semi-Lagrangian scheme in which the displacement vector is updated as the feature moves through the motion field obtained from the DIS technique. The motivation behind the DIS-Rot1 model is to better represent rotational or curved motion patterns. From these four competing models, LK-Lin4 appears to be the best model in the first forecast hour, and DIS-Lin1 the best in the second. DIS-Rot1 performs consistently the worst. That is not quite in line with our naive expectation in which we would hope that a Semi-Lagrangian approach should be able to better capture at least curved motion patterns. But not even in the winding category does the complexity of the DIS-Rot1 approach pay off. Whether that is due to the implementation of the Semi-Lagrangian approach or due to the lack of validity of the approach should be subject to future research. Comparing LK-Lin1 to LK-Lin4, we see a clear advantage in looking back in time more than one step. It appears that this way, we can retrieve more reliable, more representative, and less noisy displacement vectors which shows in the superiority of LK-Lin4 over LK-Lin1.

For all competing models, the mean location error exceeds a distance of 5 km after 60 min, and 10 km after 110 min. At least 25 % of all forecasts exceed an error of 5 km after 50 min, and of 10 km after 90 min. Even for the best models in our experiment, at least 5 percent of the forecasts will have a location error of more than 10 km after 45 min. When we relate such errors to application scenarios that are typically suggested for precipitation nowcasting – e.g. in the context of early warning systems for pluvial floods in urban environments (see Zanchetta and Coulibaly, 2020) –, it becomes obvious that location errors matter: the order of magnitude of these errors is about the same as the typical extent of a convective cell... or of a medium-sized city. Hence, the uncertainty of precipitation nowcasts at such length scales – just as a result of locational errors – can be substantial already at lead times of less than 1 h.

While similar conclusions have already been drawn by using spatially sensitive verification measures such as the Fractions Skill Score (see, e.g., Ayzel et al., 2020), our framework allows us to isolate the location error for specific models and situations, to better understand the factors that govern these errors and hence to use that knowledge in order to specifically improve the extrapolation of motion patterns in existing nowcasting models. As an example, we have demonstrated how the use of the sinuosity index can help us to better understand the predictive skill and hence the uncertainty of our models in specific situations. We hope that the large number of extracted tracks will help to foster the development of new techniques that use data-driven machine learning models for the extrapolation of feature location. For that purpose, we have made openly available the full set of extracted feature tracks for the year 2016, https://doi.org/10.5281/zenodo.4024272 (de Souza, 2020), to serve as input to future studies. However, such future studies should also use radar data from a longer time period in order to learn more about the seasonal effects related to the properties of feature tracks.

Data Availability

The code and the data of this analysis are available in a repository under https://github.com/arthurcts/loc_error (last access: 17 November 2020). The original source of the radar data as provided by DWD is https://opendata.dwd.de/weather/radar/radolan/ry.

Acknowledgments

We acknowledge the German Weather Service, namely Dr. Tanja Winterrath, for making available the RY data from the latest RADKLIM reanalysis.

Chapter 5

Discussion and Conclusions

Precipitation is an important driver of environmental and hydrological processes, and it has a direct impact on natural hazards, such as flash floods or landslides. Hence, timely and reliable precipitation nowcasts – forecasts with short lead times at high temporal and spatial resolution – are increasingly required for various purposes such as early warning, sewage and reservoir control, or agricultural management. Simply said, nowcasting provides an answer to the question "How much will it rain within the next hour?", typically on the basis of weather radar data and numerical models which extrapolate the observed precipitation field to the imminent future.

This thesis is about *measuring progress* and *making progress* in the development of models for precipitation nowcasting. More specifically, it is about the establishment of open, competitive, and affordable benchmark models, the introduction of a new data-driven model based on deep neural networks, and the quantification of location errors in precipitation nowcasts. In Chapter 2, this thesis introduced a framework that combines the various pieces and fragments that are used in conventional field tracking and extrapolation models in a comprehensive system, the software package rainymotion, and demonstrated that this open system is able to compete with state-of-the-art operational systems. In Chapter 3, we followed a new direction in precipitation nowcasting by exploring the potential of data-driven models. To that end, we introduced RainNet – a model, which is based on a deep convolutional neural network architecture. And, finally, we added a new approach to isolate and quantify the locational error component of precipitation forecasts, which, in orchestration with the error caused by changes in rainfall intensity, contributes to the total error of precipitation nowcasts (Chapter 4). In the following subsections we will recap these chapters in order to summarize and discuss the answers to the research questions outlined in Chapter 1, underline corresponding limitations, and provide an outlook to potential topics for future research.

Benchmarks should (co-)evolve

In the first study (Chapter 2), we developed the open-source rainymotion library and examined the performance of its models for radar-based precipitation nowcasting using a wide range of rainfall events. Our benchmark experiments, including an operational baseline model (the RV product provided by the DWD), showed a firm basis for using openly available methods for precipitation field tracking and extrapolation in radar-based precipitation nowcasting studies. For the majority of the analyzed events, optical-flow-based models from the rainymotion library outperformed the operational baseline.

However, we do not claim that the developed models will generally outperform well-established and excessively tuned operational models for radar-based precipitation nowcasting. Rather, rainymotion models should serve as an essential tool for providing a reliable, fast, and open solution that can serve as a benchmark for further model development and hypothesis testing. Then again, such benchmark models should not be considered as static. Instead, they should evolve, and hence also reflect, in the future, the gradual improvements in the state-of-the-art. The modular structure of rainymotion and its openness explicitly allow to incorporate such improvements, e.g., in field tracking, in the minimization of numerical diffusion for the advection/extrapolation step, or in accounting for the change of precipitation intensity over time.

We are convinced that the open-source community will play the central role as a driver of improvement, and as a channel to continuously transfer scientific progress to open solutions such as rainymotion. Recently, pySTEPS (Pulkkinen et al., 2019) – an open-source and community-driven Python library for probabilistic precipitation nowcasting – was introduced, and rapidly became a popular research tool (Imhoff et al., 2020a,b). Thus, the dynamic evolution and growing competition in the field of quantitative precipitation nowcasting is evident. These positive developments could pave the way for future synergies between competing open-source projects – towards the availability of open, reproducible, and skillful methods in quantitative precipitation nowcasting (see also Heistermann et al. 2015a for a broader discussion of how open-source packages should co-evolve in the weather radar community). For example, rainymotion could be a low-level dependency of PySTEPS, providing the specific methods for optical flow calculation or advection techniques while benefiting from the massive high-level features of PySTEPS such as verification and visualization.

Continue to capitalize on the potential of deep learning

In the second paper (Chapter 3), we have presented RainNet, a deep convolutional neural network architecture for radar-based precipitation nowcasting. Its design was inspired by the U-Net and SegNet families of deep learning models for binary segmentation, and it follows an encoder–decoder architecture in which the encoder progressively downscales the spatial resolution using pooling, followed by convolutional layers, and the decoder progressively upscales the learned patterns to a higher spatial resolution using upsampling, followed by convolutional layers. RainNet was trained to predict precipitation at a lead time of 5 min, using several years of quality-controlled weather radar composites based on the DWD weather radar network. In order to achieve a lead time of 60 min, a recursive approach was implemented by using RainNet predictions at 5 min lead time as model inputs for longer lead times. In the verification experiments, Eulerian persistence, as well as a model from the rainymotion library, served as benchmarks.

RainNet significantly outperformed both benchmark models at all lead times up to 60 min for the routine verification metrics mean absolute error (MAE) and the critical success index (CSI) at intensity thresholds of 0.125, 1, and 5 mm h^{-1}. With regard to these verification metrics, the results correspond to an extension of the effective lead time in the order of 10–20 min by RainNet as compared to rainymotion. However, rainymotion turned out to be clearly superior in predicting the exceedance of higher-intensity thresholds (here 10 and 15 mm h^{-1}) as shown by the corresponding CSI analysis.

RainNet's limited ability to predict high rainfall intensities could be attributed to a remarkable level of spatial smoothing in its predictions. That smoothing becomes increasingly apparent at longer lead times. Yet it is already prominent at a lead time of 5 min. That was confirmed by an analysis of power spectral density which showed, at time $t + 5$ min, a loss of spectral power at length scales of 16 km and below. Obviously, RainNet has learned an optimal level of smoothing to produce a nowcast at 5 min lead times. In that sense, the loss of spectral power at small scales is informative as it reflects the limits of predictability as a function of spatial scale. Beyond the lead time of 5 min, however, the increasing level of smoothing is a mere artifact – an analogue to numerical diffusion – that is not a property of RainNet itself but of its recursive application: as we repeatedly use smoothed nowcasts as model inputs, we cumulate the effect of smoothing over time. That certainly is an undesirable property, and it becomes particularly unfavorable for the prediction of high-intensity precipitation features. As was shown on the basis of the fractions skill score (FSS), rainymotion outperforms RainNet already at an intensity of 5 mm h^{-1} once we start to evaluate the performance in a spatial neighborhood around the native grid pixel of 1 km $\times$ 1 km size. This is because rainymotion preserves distinct precipitation features but tends to misplace them. RainNet, however, tends to lose such features over longer lead times due to cumulative smoothing effects – more so if it is applied recursively.

From an early warning perspective, that property of RainNet clearly limits its usefulness. There are, however, options to address that issue in future research:

- The loss function used in the training could be adjusted in order to penalize the loss of power at small spatial scales. Furthermore, the training procedure may simultaneously attempt to minimize the loss function, as well as preserve ”physical plausibility” – the set of assumptions that have to be explicitly specified by researchers.

- RainNet could be directly trained to predict precipitation at lead times beyond 5 min. While the direct prediction of precipitation at longer lead times should reduce excessive smoothing as a result of numerical diffusion, we would still expect the level of smoothing to increase with lead time as a result of the predictive uncertainty at small scales.
- As an alternative to predicting continuous values of precipitation intensity, RainNet could be trained to predict the exceedance of specific intensity thresholds instead. The additional advantage of training RainNet to predict threshold exceedance is that we could use its output directly as a measure of uncertainty (of that exceedance).

Yet the key issue of precipitation prediction – the anticipation of convective initialization, as well as the growth and dissipation of precipitation in the imminent future – still appears to be unresolved. It is an inherent limitation of nowcasting models purely based on optical flow: they can extrapolate motion fairly well, but they cannot predict intensity dynamics. Deep learning architectures, however, might be able to learn recurrent patterns of growth and dissipation, although it will be challenging to verify if they actually did. In the context of this study, though, we have to assume that RainNet has rather learned the representation of motion patterns instead of rainfall intensity dynamics: for a lead time of 5 min, the effects of motion can generally be expected to dominate over the effects of intensity dynamics, which will propagate to the learning results. The fact that we actually could recursively use RainNet’s predictions at 5 min lead time in order to predict precipitation at 1 h lead time also implies that RainNet, in essence, learned to represent motion patterns and optimal smoothing.

Another limitation in successfully learning patterns of intensity growth and dissipation might be the input data itself. While we do not exclude the possibility that such patterns could be learned from just two-dimensional radar composites, other input variables might add essential information on imminent atmospheric dynamics – the predisposition of the atmosphere to produce or to dissolve precipitation. Such additional data might include three-dimensional radar volume data, dual-polarization radar moments, or the output fields of numerical weather prediction (NWP) models. Formally, the inclusion of NWP fields in a learning framework could be considered as a different way of assimilation, combining – in a data-driven way – the information content of physical models and observations. Hence, we should continue to capitalize on the potential of deep learning in this field, by consequently unlocking further input data which we consider as informative for creating more skillful, reliable, and robust predictive models.

In summary, Chapter 3 provides, after Shi et al. (2015, 2017, 2018), another proof of concept that convolutional neural networks deliver a firm basis to compete with conventional nowcasting models based on optical flow. Moreover, it also reveals the role of evaluation metrics in diagnostics of the properties of model predictions – in a statistical sense but also in how processes of motion and intensity dynamics are reflected. This way, the importance of disentangling sources of model errors has become apparent as another basis for future model development.

Improving models by understanding sources of error

In the third paper (Chapter 4), we have introduced a framework to isolate and quantify the location error in precipitation nowcasts that are based on field-tracking techniques. While it is often assumed that errors in precipitation nowcasts are dominated by the temporal dynamics of precipitation intensity, the location error of predicted precipitation features has so far not been explicitly and formally quantified.

The main idea of our framework is to detect and track scale-invariant precipitation features (corners) in radar images. To this aim, we detected features by using the approach of Shi and Tomasi (1994), and tracked these features following the approach of Lucas and Kanade (1981), using both algorithms as implemented in the OpenCV library. Then, we used these observed tracks as a ”true” reference in order to evaluate the performance (or, inversely, the error) of any model that aims to predict the future locations of such precipitation features. For that purpose, we defined the location error of a forecast at any lead time Δt ahead of the forecast time t as the Euclidean distance between the observed and the predicted feature location at $t + \Delta t$. Hence, the developed framework allows us to compare and benchmark the ability of different models to forecast future locations of precipitation features, and thus to specifically focus on improving that ability by future model development.

The hypothesis that such further model developments are urgently required is supported by the results of our benchmarking study. For all competing models, the mean location error exceeds a distance of 5 km after 60 min, and 10 km after 110 min. At least 25 % of all forecasts exceed an error of 5 km after 50 min, and of 10 km after 90 min. Even for the best models in our experiment, at least 5 percent of the forecasts will have a location error of more than 10 km after 45 min. When we relate such errors to application scenarios that are typically suggested for precipitation nowcasting – e.g., in the context of early warning systems for pluvial floods in urban environments (see Zanchetta and Coulibaly, 2020) –, it becomes obvious that location errors matter: the order of magnitude of these errors is about the same as the typical extent of a convective cell... or of a medium-sized city. Hence, the uncertainty of precipitation nowcasts at such length scales – just as a result of locational errors – can be substantial already at lead times of less than 1 h.

Based on this framework, the next natural step for future research would be to specifically minimize the effect of location errors by a targeted model development beyond conventional linear or semi-Lagrangian extrapolation. In this context, data-driven machine learning models appear particularly promising as they could learn from the large collection of extracted feature tracks. In particular, as track data has a sequential order, modern architectures that benefit from this structure, such as novel deep neural networks based on attention mechanisms (Choromanski et al., 2020; Vaswani et al., 2017), are the most promising.

Towards open science and applications

The idea of open science is at the heart of the presented thesis. In this context, the term ”open” does not just imply open code, but also open computational environments, research data sets, and documentation. Broadly speaking, open science means available, accessible, and reproducible science – an ideal that we, as a community, have to pursue for good. That way, it provides a basis for the continuous development and improvement, by making available (and applicable) the tools that are required to test, or to falsify theories.

For example, in Chapter 2 we provided a benchmark for radar-based precipitation nowcasting based on conventional techniques. To that end, we introduced the open rainymotion software library (https://github.com/hydrogo/rainymotion), and provided corresponding documentation (https://rainymotion.readthedocs.io) with hands-on examples of its use and the sample data. This ensures openness in a broad context and provides a basis for further development and improvement by the research (and application) community.

We tried to pursue the same concept with regard to Chapters 3 and 4, too. Our deep learning model for precipitation nowcasting – RainNet –, as well as its pre-trained weights and data used for the extensive training procedure are freely available in open repositories. Furthermore, we also provide the ”getting started” example that shows how RainNet can be run out-of-the-box in a freely-accessible computational environment (Google Colab) for operational precipitation nowcasting. That way, we provide a basis for scientific reproducibility and dissolve the boundaries between research and application: internet access is all you need to get started in precipitation nowcasting.

To the best of our knowledge, rainymotion models have already been used as a benchmark for a research study that investigates the skill of precipitation nowcasting in the Netherlands (Imhoff et al., 2020b). Moreover, rainymotion models have been utilized as a core for establishing web services and applications for precipitation nowcasting, such as Yandex.Weather (https://yandex.ru/pogoda/potsdam/maps/nowcast), Windy (https://www.windy.com/), and WillyWeather (https:

//www.willyweather.com.au/) – services which have millions of users worldwide.

Altogether, this demonstrates the potential of open science as a guiding principle that drives both in-depth model development and broadens the landscape of practical application of nowcasting technologies.

Bibliography

Agrawal, S., Barrington, L., Bromberg, C., Burge, J., Gazen, C., and Hickey, J.: Machine Learning for Precipitation Nowcasting from Radar Images, URL: https://arxiv.org/abs/1912.12132, last access: 28 January 2020, 2019.

Austin, G. L. and Bellon, A.: The use of digital weather radar records for short-term precipitation forecasting, Quarterly Journal of the Royal Meteorological Society, 100, 658–664, https://doi.org/10.1002/qj.49710042612, URL: http://doi.wiley.com/10.1002/qj.49710042612, 1974.

Ayzel, G.: hydrogo/rainymotion: rainymotion v0.1, https://doi.org/10.5281/zenodo.2561583, URL: https://doi.org/10.5281/zenodo.2561583, last access: 12 November 2020, 2019.

Ayzel, G.: hydrogo/rainnet: RainNet v1.0-gmdd, https://doi.org/10.5281/zenodo.3631038, URL: https://zenodo.org/record/3631038, last access: 17 November 2020, 2020a.

Ayzel, G.: RainNet: pretrained model and weights, https://doi.org/10.5281/zenodo.3630429, URL: https://zenodo.org/record/3630429, last access: 17 November 2020, 2020b.

Ayzel, G.: RYDL: the sample data of the RY product for deep learning applications, https://doi.org/10.5281/zenodo.3629951, URL: https://zenodo.org/record/3629951, last access: 17 November 2020, 2020c.

Ayzel, G.: RainNet: a convolutional neural network for radar-based precipitation nowcasting, https://github.com/hydrogo/rainnet, last access: 10 June 2020, 2020d.

Ayzel, G., Heistermann, M., and Winterrath, T.: Optical flow models as an open benchmark for radar-based precipitation nowcasting (rainymotion v0.1), Geoscientific Model Development, 12, 1387–1402, https://doi.org/10.5194/gmd-12-1387-2019, URL: https://www.geosci-model-dev.net/12/1387/2019/, 2019a.

Ayzel, G., Heistermann, M., and Winterrath, T.: rainymotion:python library for radar-based precipitation nowcasting based onoptical flow techniques, https://github.com/hydrogo/rainymotion, last access: 11 November 2020, 2019b.

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

Badrinarayanan, V., Kendall, A., and Cipolla, R.: SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence, 39, 2481–2495, https://doi.org/10.1109/TPAMI.2016.2644615, 2017.

Bai, S., Kolter, J. Z., and Koltun, V.: An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling, URL: https://arxiv.org/abs/1803.01271, last access: 28 January 2020, 2018.

Baldwin, M. E. and Kain, J. S.: Sensitivity of several performance measures to displacement error, bias, and event frequency, Weather and forecasting, 21, 636–648, 2006.

Bartels, H., Weigl, E., Klink, S., Kohler, O., Reich, T., Rosenow, W., Lang, P., Podlasly, C., Winterrath, T., Adrian, G., Majewski, D., and Lang, J.: Projekt RADVOR-OP Radargestützte, zeitnahe Niederschlagsvorhersage für den operationellen Einsatz (Niederschlag-Nowcasting-System), Teil-I, Final Report 1, Deutscher Wetterdienst (DWD), available at: https://www.dwd.de/DE/leistungen/radvor/radvor_info/abschlussbericht_radvor_op_2005_pdf.pdf, 2005.

Bauer, P., Thorpe, A., and Brunet, G.: The quiet revolution of numerical weather prediction, Nature, 525, 47–55, https://doi.org/10.1038/nature14956, URL: http://www.nature.com/doifinder/10.1038/nature14956, 2015.

Bellerby, T. J.: High-resolution 2-D cloud-top advection from geostationary satellite imagery, IEEE Transactions on Geoscience and Remote Sensing, 44, 3639–3648, https://doi.org/10.1109/TGRS.2006.881117, URL: http://ieeexplore.ieee.org/document/4014303/, 2006.

Berenguer, M., Sempere-Torres, D., and Pegram, G. G. S.: SBMcast - An ensemble nowcasting technique to assess the uncertainty in rainfall forecasts by Lagrangian extrapolation, Journal of Hydrology, 404, 226–240, https://doi.org/10.1016/j.jhydrol.2011.04.033, URL: https://www.sciencedirect.com/science/article/pii/S0022169411002940, 2011.

Berenguer, M., Surcel, M., Zawadzki, I., Xue, M., Kong, F., Berenguer, M., Surcel, M., Zawadzki, I., Xue, M., and Kong, F.: The Diurnal Cycle of Precipitation from Continental Radar Mosaics and Numerical Weather Prediction Models. Part II: Intercomparison among Numerical Models and with Nowcasting, Monthly Weather Review, 140, 2689–2705, https://doi.org/10.1175/MWR-D-11-00181.1, URL: http://journals.ametsoc.org/doi/abs/10.1175/MWR-D-11-00181.1, 2012.

Bouguet, J.-Y.: Pyramidal implementation of the affine Lucas Kanade feature tracker. Description of the algorithm, Tech. rep., Intel corporation, Microprocessor Research Labs, 2000.

Boureau, Y.-L., Ponce, J., and LeCun, Y.: A Theoretical Analysis of Feature Pooling in Visual Recognition, in: Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML'10, pp. 111–118, Omnipress, Madison, WI, USA, 2010.

Bowler, N. E., Pierce, C. E., and Seed, A.: Development of a precipitation nowcasting algorithm based upon optical flow techniques, Journal of Hydrology, 288, 74–91, https://doi.org/10.1016/j.jhydrol.2003.11.011, URL: https://www.sciencedirect.com/science/article/pii/S0022169403004591, 2004.

Bowler, N. E., Pierce, C. E., and Seed, A. W.: STEPS: A probabilistic precipitation forecasting scheme which merges an extrapolation nowcast with downscaled NWP, Quarterly Journal of the Royal Meteorological Society, 132, 2127–2155, https://doi.org/10.1256/qj.04.100, URL: http://dx.doi.org/10.1256/qj.04.100, 2006.

Bradski, G. and Kaehler, A.: Learning OpenCV: Computer vision with the OpenCV library, " O'Reilly Media, Inc.", 2008.

Brox, T., Bruhn, A., Papenberg, N., and Weickert, J.: High accuracy optical flow estimation based on a theory for warping, in: European conference on computer vision, pp. 25–36, Springer, 2004.

Bruhn, A., Weickert, J., Feddern, C., Kohlberger, T., and Schnörr, C.: Variational optical flow computation in real time, IEEE transactions on image processing : a publication of the IEEE Signal Processing Society, 14, 608–15, https://doi.org/10.1109/TIP.2005.846018, URL: https://ieeexplore.ieee.org/document/1420392/, 2005a.

Bruhn, A., Weickert, J., and Schnörr, C.: Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods, International journal of computer vision, 61, 211–231, 2005b.

Chen, P., Chen, G., and Zhang, S.: Log Hyperbolic Cosine Loss Improves Variational Auto-Encoder, URL: https://openreview.net/forum?id=rkglvsC9Ym, last access: 28 January 2020, 2018.

Cheung, P. and Yeung, H. Y.: Application of optical-flow technique to significant convection nowcast for terminal areas in Hong Kong, in: The 3rd WMO International Symposium on Nowcasting and Very Short-Range Forecasting (WSN12), pp. 1–10, URL: http://www.hko.gov.hk/publica/reprint/r1025.pdf, 2012.

Chollet, F. et al.: Keras, https://keras.io, last access: 17 November 2020, 2015.

Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., et al.: Rethinking Attention with Performers, https://arxiv.org/abs/2009.14794, URL: https://arxiv.org/pdf/2009.14794.pdf, last access: 12 November 2020, 2020.

Crisologo, I. and Heistermann, M.: Using ground radar overlaps to verify the retrieval of calibration bias estimates from spaceborne platforms, Atmospheric Measurement Techniques, 13, 645–659, https://doi.org/10.5194/amt-13-645-2020, URL: https://amt.copernicus.org/articles/13/645/2020/, 2020.

Dahl, G. E., Sainath, T. N., and Hinton, G. E.: Improving deep neural networks for LVCSR using rectified linear units and dropout, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 8609–8613, https://doi.org/10.1109/ICASSP.2013.6639346, 2013.

de Souza, A. C. T.: Set of extracted feature tracks for the year 2016., https://doi.org/10.5281/zenodo.4024272, URL: https://doi.org/10.5281/zenodo.4024272, last access: 11 November 2020, 2020.

Dueben, P. D. and Bauer, P.: Challenges and design choices for global weather and climate models based on machine learning, Geoscientific Model Development, 11, 3999–4009, https://doi.org/10.5194/gmd-11-3999-2018, URL: https://www.geosci-model-dev.net/11/3999/2018/, 2018.

DWD: German Climate Atlas, https://www.dwd.de/EN/ourservices/germanclimateatlas/germanclimateatlas.html, last access: 11 November 2020, 2020.

Ebert, E. E.: Fuzzy verification of high-resolution gridded forecasts: a review and proposed framework, Meteorological Applications: A journal of forecasting, practical applications, training techniques and modelling, 15, 51–64, 2008.

Farnebäck, G.: Two-frame motion estimation based on polynomial expansion, Image Analysis, 2003, 363–370, https://doi.org/10.1007/3-540-45103-x_50, URL: http://link.springer.com/10.1007/3-540-45103-X, 2003.

Foresti, L., Reyniers, M., Seed, A., and Delobbe, L.: Development and verification of a real-time stochastic precipitation nowcasting system for urban hydrology in Belgium, Hydrology and Earth System Sciences, 20, 505–527, https://doi.org/10.5194/hess-20-505-2016, URL: https://www.hydrol-earth-syst-sci.net/20/505/2016/, 2016.

Fuhrer, O., Chadha, T., Hoefler, T., Kwasniewski, G., Lapillonne, X., Leutwyler, D., Lüthi, D., Osuna, C., Schär, C., Schulthess, T. C., and Vogt, H.: Near-global climate simulation at 1 km resolution: establishing a performance baseline on 4888 GPUs with COSMO 5.0, Geoscientific Model Development, 11, 1665–1681, https://doi.org/10.5194/gmd-11-1665-2018, URL: https://gmd.copernicus.org/articles/11/1665/2018/, 2018.

Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. N.: Convolutional Sequence to Sequence Learning, in: Proceedings of the 34th International Conference on Machine Learning - Volume 70, ICML'17, pp. 1243–1252, JMLR.org, 2017.

Gentine, P., Pritchard, M., Rasp, S., Reinaudi, G., and Yacalis, G.: Could Machine Learning Break the Convection Parameterization Deadlock?, Geophysical Research Letters, 45, 5742–5751, https://doi.org/10.1029/2018GL078202, URL: http://doi.wiley.com/10.1029/2018GL078202, 2018.

Germann, U. and Zawadzki, I.: Scale-Dependence of the Predictability of Precipitation from Continental Radar Images. Part I: Description of the Methodology, Monthly Weather Review, 130, 2859–2873, https://doi.org/10.1175/1520-0493(2002)130<2859:SDOTPO>2.0.CO;2, URL: https://doi.org/10.1175/1520-0493(2002)130<2859:SDOTPO>2.0.CO;2, 2002a.

Germann, U. and Zawadzki, I.: Scale-Dependence of the Predictability of Precipitation from Continental Radar Images. Part I: Description of the Methodology, Monthly Weather Review, 130, 2859–2873, https://doi.org/10.1175/1520-0493(2002)130<2859:SDOTPO>2.0.CO;2, URL: http://journals.ametsoc.org/doi/abs/10.1175/1520-0493{%}282002{%}29130{%}3C2859{%}3ASDOTPO{%}3E2.0.CO{%}3B2, 2002b.

Germann, U., Zawadzki, I., Turner, B., Germann, U., Zawadzki, I., and Turner, B.: Predictability of Precipitation from Continental Radar Images. Part IV: Limits to Prediction, Journal of the Atmospheric Sciences, 63, 2092–2108, https://doi.org/10.1175/JAS3735.1, URL: http://journals.ametsoc.org/doi/abs/10.1175/JAS3735.1, 2006.

Gilleland, E., Ahijevych, D. A., Brown, B. G., and Ebert, E. E.: Verifying Forecasts Spatially, Bulletin of the American Meteorological Society, 91, 1365–1376, https://doi.org/10.1175/2010BAMS2819.1, URL: https://journals.ametsoc.org/doi/10.1175/2010BAMS2819.1, publisher: American Meteorological Society, 2010.

Golding, B. W.: Nimrod: A system for generating automated very short range forecasts, Meteorological Applications, 5, 1–16, https://doi.org/10.1017/S1350482798000577, URL: http://doi.wiley.com/10.1017/S1350482798000577, 1998.

Grecu, M. and Krajewski, W. F.: A large-sample investigation of statistical procedures for radar-based short-term quantitative precipitation forecasting, Journal of Hydrology, 239, 69–84, https://doi.org/10.1016/S0022-1694(00)00360-7, URL: https://www.sciencedirect.com/science/article/pii/S0022169400003607, 2000.

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J., and Chen, T.: Recent advances in convolutional neural networks, Pattern Recognition, 77, 354–377, https://doi.org/10.1016/j.patcog.2017.10.013, URL: http://www.sciencedirect.com/science/article/pii/S0031320317304120, 2018.

Heistermann, M., Jacobi, S., and Pfaff, T.: Technical Note: An open source library for processing weather radar data (wradlib), Hydrology and Earth System Sciences, 17, 863–871, https://doi.org/10.5194/hess-17-863-2013, URL: https://www.hydrol-earth-syst-sci.net/17/863/2013/, 2013.

Heistermann, M., Collis, S., Dixon, M. J., Giangrande, S., Helmus, J. J., Kelley, B., Koistinen, J., Michelson, D. B., Peura, M., Pfaff, T., and Wolff, D. B.: The emergence of open-source software for the weather radar community, Bulletin of the American Meteorological Society, 96, 117–128, https://doi.org/10.1175/BAMS-D-13-00240.1, URL: http://journals.ametsoc.org/doi/10.1175/BAMS-D-13-00240.1, 2015a.

Heistermann, M., Collis, S., Dixon, M. J., Helmus, J. J., Henja, A., Michelson, D. B., and Pfaff, T.: An open virtual machine for cross-platform weather radar science, Bulletin of the American Meteorological Society, 96, 1641–1645, https://doi.org/10.1175/BAMS-D-14-00220.1, URL: http://journals.ametsoc.org/doi/10.1175/BAMS-D-14-00220.1, 2015b.

Horn, B. K. and Schunck, B. G.: Determining optical flow, Artificial intelligence, 17, 185–203, 1981.

Hunter, J. D.: Matplotlib: A 2D graphics environment, Computing in science & engineering, 9, 90–95, 2007.

Iglovikov, V. and Shvets, A.: TernausNet: U-Net with VGG11 Encoder Pre-Trained on ImageNet for Image Segmentation, URL: https://arxiv.org/abs/1801.05746, last access: 28 January 2020, 2018.

Imhoff, R., Overeem, A., Brauer, C., Leijnse, H., Weerts, A., and Uijlenhoet, R.: Rainfall Nowcasting Using Commercial Microwave Links, Geophysical Research Letters, 47, e2020GL089 365, 2020a.

Imhoff, R. O., Brauer, C. C., Overeem, A., Weerts, A. H., and Uijlenhoet, R.: Spatial and Temporal Evaluation of Radar Rainfall Nowcasting Techniques on 1,533 Events, Water Resources Research, 56, e2019WR026 723, https://doi.org/https://doi.org/10.1029/2019WR026723, URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2019WR026723, e2019WR026723 10.1029/2019WR026723, 2020b.

Jensen, D. G., Petersen, C., and Rasmussen, M. R.: Assimilation of radar-based nowcast into a HIRLAM NWP model, Meteorological Applications, 22, 485–494, https://doi.org/10.1002/met.1479, URL: http://doi.wiley.com/10.1002/met.1479, 2015.

Jones, E., Oliphant, T., and Peterson, P.: SciPy: open source scientific tools for Python, URL: https://scipy.org/, last access: 29 June 2018, 2018.

Kingma, D. P. and Ba, J.: Adam: A Method for Stochastic Optimization, in: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, edited by Bengio, Y. and LeCun, Y., URL: http://arxiv.org/abs/1412.6980, 2015.

Krizhevsky, A., Sutskever, I., and Hinton, G. E.: ImageNet Classification with Deep Convolutional Neural Networks, in: Advances in Neural Information Processing Systems 25, edited by Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., pp. 1097–1105, Curran Associates, Inc., URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf, 2012.

Kroeger, T., Timofte, R., Dai, D., and Van Gool, L.: Fast optical flow using dense inverse search, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 9908 LNCS, pp. 471–488, https://doi.org/10.1007/978-3-319-46493-0_29, URL: http://arxiv.org/abs/1603.03590, 2016.

Lebedev, V., Ivashkin, V., Rudenko, I., Ganshin, A., Molchanov, A., Ovcharenko, S., Grokhovetskiy, R., Bushmarinov, I., and Solomentsev, D.: Precipitation nowcasting with satellite imagery, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2680–2688, 2019.

LeCun, Y., Bengio, Y., and Hinton, G.: Deep learning, Nature, 521, 436–444, https://doi.org/10.1038/nature14539, URL: http://www.nature.com/articles/nature14539, 2015.

Lin, C., Vasić, S., Kilambi, A., Turner, B., and Zawadzki, I.: Precipitation forecast skill of numerical weather prediction models and radar nowcasts, Geophysical Research Letters, 32, https://doi.org/10.1029/2005GL023451, URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2005GL023451, 2005.

Liu, Y., Xi, D. G., Li, Z. L., and Hong, Y.: A new methodology for pixel-quantitative precipitation nowcasting using a pyramid Lucas Kanade optical flow approach, Journal of Hydrology, 529, 354–364, https://doi.org/10.1016/j.jhydrol.2015.07.042, URL: https://www.sciencedirect.com/science/article/pii/S002216941500548X, 2015.

Long, J., Shelhamer, E., and Darrell, T.: Fully Convolutional Networks for Semantic Segmentation, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

Lucas, B. D. and Kanade, T.: An iterative image Registration Technique with an Application to Stereo Vision, in: Proceedings DARPA Image Understanding Workrhop, pp. 674–679, Morgan Kaufmann Publishers Inc., https://doi.org/10.1145/358669.358692, URL: https://ri.cmu.edu/pub_files/pub3/lucas_bruce_d_1981_2/lucas_bruce_d_1981_2.pdf, 1981.

Mecklenburg, S., Joss, J., and Schmid, W.: Improving the nowcasting of precipitation in an Alpine region with an enhanced radar echo tracking algorithm, Journal of Hydrology, 239, 46–68, https://doi.org/10.1016/S0022-1694(00)00352-8, URL: https://www.sciencedirect.com/science/article/pii/S0022169400003528, 2000.

Mittermaier, M. and Roberts, N.: Intercomparison of Spatial Forecast Verification Methods: Identifying Skillful Spatial Scales Using the Fractions Skill Score, Weather and Forecasting, 25, 343–354, https://doi.org/10.1175/2009WAF2222260.1, URL: https://doi.org/10.1175/2009WAF2222260.1, 2010.

Mueller, C. M., Axen, T. S., Oberts, R. R., Ilson, J. W., Etancourt, T. B., Ettling, S. D., and Ien, N. O.: NCAR Auto-Nowcast System, Weather and Forecasting, 18, 545–561, https://doi.org/10.1175/1520-0434(2003)018<0545:NAS>2.0.CO;2, URL: http://journals.ametsoc.org/doi/abs/10.1175/1520-0434{%}282003{%}29018{%}3C0545{%}3ANAS{%}3E2.0.CO{%}3B2, 2003.

Mueller, J. E.: An introduction to the hydraulic and topographic sinuosity indexes, Annals of the association of american geographers, 58, 371–385, 1968.

Nair, V. and Hinton, G. E.: Interpersonal Informatics: Making Social Influence Visible, in: Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML'10, pp. 807–814, Omnipress, Madison, WI, USA, 2010.

Oliphant, T. E.: A guide to NumPy, vol. 1, Trelgol Publishing USA, 2006.

OpenCV library: "OpenCV: Optical flow tutorial", https://docs.opencv.org/4.4.0/d4/dee/tutorial_optical_flow.html, last access: 13 October 2020, 2020a.

OpenCV library: "OpenCV: DISOpticalFlow Class Reference", https://docs.opencv.org/4.4.0/de/d4f/classcv_1_1DISOpticalFlow.html, last access: 13 October 2020, 2020b.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in Python, Journal of machine learning research, 12, 2825–2830, 2011.

Pierce, C., Seed, A., Ballard, S., Simonin, D., and Li, Z.: Nowcasting, in: Doppler Radar Observations, edited by Bech, J. and Chau, J. L., chap. 4, IntechOpen, Rijeka, https://doi.org/10.5772/39054, URL: https://doi.org/10.5772/39054, 2012.

Pulkkinen, S., Nerini, D., Pérez Hortal, A. A., Velasco-Forero, C., Seed, A., Germann, U., and Foresti, L.: Pysteps: an open-source Python library for probabilistic precipitation nowcasting (v1.0), Geoscientific Model Development, 12, 4185–4219, https://doi.org/10.5194/gmd-12-4185-2019, URL: https://gmd.copernicus.org/articles/12/4185/2019/, 2019.

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., and Prabhat: Deep learning and process understanding for data-driven Earth system science, Nature, 566, 195–204, https://doi.org/10.1038/s41586-019-0912-1, URL: https://doi.org/10.1038/s41586-019-0912-1, 2019.

Reyniers, M.: Quantitative precipitation forecasts based on radar observations: Principles, algorithms and operational systems, Institut Royal Météorologique de Belgique, URL: https://www.meteo.be/meteo/

download/fr/3040165/pdf/rmi_scpub-1261.pdf, 2008.

Ronneberger, O., Fischer, P., and Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, edited by Navab, N., Hornegger, J., Wells, W. M., and Frangi, A. F., pp. 234–241, Springer International Publishing, Cham, https://doi.org/10.1007/978-3-319-24574-4_28, 2015.

Rudolf, B., Winterrath, T., Weigl, E., Reich, T., Rosenow, W., and Stephan, K.: Projekt RADVOR-OP Radargestützte, zeitnahe Niederschlagsvorhersage für den operationellen Einsatz (Niederschlag-Nowcasting-System), Teil-II, Final Report 2, Deutscher Wetterdienst (DWD), available at: https://www.dwd.de/DE/leistungen/radvor/radvor_info/abschlussbericht_radvor_op_2012_pdf.pdf, 2012.

Ruzanski, E., Chandrasekar, V., and Wang, Y.: The CASA nowcasting system, Journal of Atmospheric and Oceanic Technology, 28, 640–655, https://doi.org/10.1175/2011JTECHA1496.1, URL: http://journals.ametsoc.org/doi/abs/10.1175/2011JTECHA1496.1, 2011.

Schmid, C., Mohr, R., and Bauckhage, C.: Evaluation of interest point detectors, International Journal of computer vision, 37, 151–172, 2000.

Schneider, P. J. and Eberly, D. H.: Geometric tools for computer graphics, Boston, URL: https://www.sciencedirect.com/science/book/9781558605947, 2003.

Shi, E., Li, Q., Gu, D., and Zhao, Z.: A Method of Weather Radar Echo Extrapolation Based on Convolutional Neural Networks, in: MultiMedia Modeling, edited by Schoeffmann, K., Chalidabhongse, T. H., Ngo, C. W., Aramvith, S., O'Connor, N. E., Ho, Y.-S., Gabbouj, M., and Elgammal, A., pp. 16–28, Springer International Publishing, Cham, https://doi.org/10.1007/978-3-319-73603-7_2, URL: https://link.springer.com/chapter/10.1007%2F978-3-319-73603-7_2, 2018.

Shi, J. and Tomasi, C.: Good features to track, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR-94, pp. 593–600, IEEE Comput. Soc. Press, https://doi.org/10.1109/CVPR.1994.323794, URL: http://ieeexplore.ieee.org/document/323794/, 1994.

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-k., and Woo, W.-c.: Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting, in: Advances in Neural Information Processing Systems 28, edited by Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., pp. 802–810, Curran Associates, Inc., URL: http://papers.nips.cc/paper/5955-convolutional-lstm-network-a-machine-learning-approach-for-precipitation-nowcasting.pdf, 2015.

Shi, X., Gao, Z., Lausen, L., Wang, H., Yeung, D.-Y., Wong, W.-k., and Woo, W.-c.: Deep Learning for Precipitation Nowcasting: A Benchmark and A New Model, in: Advances in Neural Information Processing Systems 30, edited by Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., pp. 5617–5627, Curran Associates, Inc., URL: http://papers.nips.cc/paper/7145-deep-learning-for-precipitation-nowcasting-a-benchmark-and-a-new-model.pdf, 2017.

Singh, S., Sarkar, S., and Mitra, P.: Leveraging Convolutions in Recurrent Neural Networks for Doppler Weather Radar Echo Prediction, in: Advances in Neural Networks - ISNN 2017, edited by Cong, F., Leung, A., and Wei, Q., pp. 310–317, Springer International Publishing, Cham, https://doi.org/10.1007/978-3-319-59081-3_37, URL: https://link.springer.com/chapter/10.1007%2F978-3-319-59081-3_37, 2017.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R.: Dropout: A Simple Way to Prevent Neural Networks from Overfitting, J. Mach. Learn. Res., 15, 1929–1958, 2014.

Srivastava, R. K., Greff, K., and Schmidhuber, J.: Training Very Deep Networks, in: Advances in Neural Information Processing Systems 28, edited by Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., pp. 2377–2385, Curran Associates, Inc., URL: http://papers.nips.cc/paper/5850-training-very-deep-networks.pdf, 2015.

Sun, J., Xue, M., Wilson, J. W., Zawadzki, I., Ballard, S. P., Onvlee-Hooimeyer, J., Joe, P., Barker, D. M., Li, P.-W., Golding, B., Xu, M., and Pinto, J.: Use of NWP for Nowcasting Convective Precipitation: Recent Progress and Challenges, Bulletin of the American Meteorological Society, 95, 409–426, https://doi.org/10.1175/BAMS-D-11-00263.1, URL: https://doi.org/10.1175/BAMS-D-11-00263.

1, 2014.

Sutskever, I., Vinyals, O., and Le, Q. V.: Sequence to Sequence Learning with Neural Networks, in: Advances in Neural Information Processing Systems 27, edited by Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., pp. 3104–3112, Curran Associates, Inc., URL: http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf, 2014.

Terry, J. P. and Feng, C.-C.: On quantifying the sinuosity of typhoon tracks in the western North Pacific basin, Applied Geography, 30, 678–686, 2010.

Thorndahl, S., Einfalt, T., Willems, P., Nielsen, J. E., ten Veldhuis, M.-C., Arnbjerg-Nielsen, K., Rasmussen, M. R., and Molnar, P.: Weather radar rainfall data in urban hydrology, Hydrology and Earth System Sciences, 21, 1359–1380, https://doi.org/10.5194/hess-21-1359-2017, URL: http://www.hydrol-earth-syst-sci.net/21/1359/2017/, 2017.

Van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., and Yu, T.: scikit-image: image processing in Python, PeerJ, 2, e453, https://doi.org/10.7717/peerj.453, URL: https://peerj.com/articles/453/, 2014.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I.: Attention is all you need, in: Advances in neural information processing systems, pp. 5998–6008, 2017.

Weinzaepfel, P., Revaud, J., Harchaoui, Z., and Schmid, C.: DeepFlow: Large Displacement Optical Flow with Deep Matching, in: 2013 IEEE International Conference on Computer Vision, pp. 1385–1392, IEEE, https://doi.org/10.1109/ICCV.2013.175, URL: http://ieeexplore.ieee.org/document/6751282/, 2013.

Welch, P.: The use of fast Fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms, IEEE Transactions on audio and electroacoustics, 15, 70–73, 1967.

Wilson, J. W., Crook, N. A., Mueller, C. K., Sun, J., and Dixon, M.: Nowcasting Thunderstorms: A Status Report, Bulletin of the American Meteorological Society, 79, 2079–2099, https://doi.org/10.1175/1520-0477(1998)079<2079:NTASR>2.0.CO;2, URL: https://journals.ametsoc.org/doi/abs/10.1175/1520-0477%281998%29079%3C2079%3ANTASR%3E2.0.CO%3B2, 1998.

Winterrath, T. and Rosenow, W.: A new module for the tracking of radar-derived precipitation with model-derived winds, Advances in Geosciences, 10, 77–83, https://doi.org/10.5194/adgeo-10-77-2007, URL: http://www.adv-geosci.net/10/77/2007/, 2007.

Winterrath, T., Rosenow, W., and Weigl, E.: On the DWD quantitative precipitation analysis and nowcasting system for real-time application in German flood risk management, in: Weather Radar and Hydrology (Proceedings of a symposium held in Exeter, UK, April 2011) IAHS Publ., vol. 351, pp. 323–329, URL: https://www.dwd.de/DE/leistungen/radolan/radolan_info/Winterrath_German_flood_risk_management_pdf.pdf?__blob=publicationFile&v=4, 2012.

Winterrath, T., Brendel, C., Hafer, M., Junghänel, T., Klameth, A., Walawender, E., Weigl, E., and Becker, A.: Erstellung einer radargestützten Niederschlagsklimatologie, URL: ftp://ftp.dwd.de/pub/data/gpcc/radarklimatologie/Dokumente/Endbericht_Radarklimatologie_final.pdf, last access: 29 June 2018, 2017.

Wolberg, G.: Digital Image Warping, IEEE Computer Society Press, 1990.

Wong, W. K., Yeung, L. H. Y., Wang, Y. C., and Chen, M.: Towards the blending of NWP with nowcast - Operation experience in B08FDP, in: WMO symposium on nowcasting, URL: http://my.hko.gov.hk/publica/reprint/r844.pdf, 2009.

Woo, W.-C. and Wong, W.-K.: Operational Application of Optical Flow Techniques to Radar-Based Rainfall Nowcasting, Atmosphere, 8, 48, https://doi.org/10.3390/atmos8030048, URL: http://www.mdpi.com/2073-4433/8/3/48, 2017.

Wulff, J. and Black, M. J.: Efficient Sparse-to-Dense Optical Flow Estimation Using a Learned Basis and Layers, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

Yeung, L. H. Y., Wong, W. K., Chan, P. K. Y., Lai, E. S. T., Yeung, L. H. Y., Wong, W. K., Chan, P. K. Y., and Lai, E. S. T.: Applications of the Hong Kong Observatory nowcasting system SWIRLS-2 in support of the 2008 Beijing Olympic Games, in: WMO Symposium on Nowcasting, vol. 30, URL: http://my.hko.gov.hk/

publica/reprint/r843.pdf, 2009.

Zahraei, A., lin Hsu, K., Sorooshian, S., Gourley, J. J., Lakshmanan, V., Hong, Y., and Bellerby, T.: Quantitative Precipitation Nowcasting: A Lagrangian Pixel-Based Approach, Atmospheric Research, 118, 418–434, https://doi.org/10.1016/j.atmosres.2012.07.001, URL: https://www.sciencedirect.com/science/article/pii/S0169809512002219, 2012.

Zahraei, A., lin Hsu, K., Sorooshian, S., Gourley, J. J., Hong, Y., and Behrangi, A.: Short-term quantitative precipitation forecasting using an object-based approach, Journal of Hydrology, 483, 1–15, https://doi.org/10.1016/j.jhydrol.2012.09.052, URL: https://www.sciencedirect.com/science/article/pii/S0022169412008694, 2013.

Zanchetta, A. D. and Coulibaly, P.: Recent Advances in Real-Time Pluvial Flash Flood Forecasting, Water, 12, 570, 2020.