Deep Learning for Future Frame Prediction of Weather Maps

Master's Thesis submitted

to

Prof. Dr. Stefan Lessmann

and

PD Dr. Martin Schultz

Humboldt-University of Berlin

School of Business and Economics

Chair of Information Systems

by

Severin Hußmann

(583503)

in partial fulfillment of the requirements

for the degree of

Master of Science (M.Sc.)

in Information Systems

Berlin, July 30, 2019

Abstract

The data available to meteorologists is growing steadily and classical numerical weather prediction is reaching its limits to derive even more knowledge from it. This study focuses on applying data-driven deep learning methodologies to the field of weather forecasting, specifically air temperature over Europe. A future frame prediction model from the computer vision field is trained with the three input variables air temperature, surface pressure, and the 500 hPa geopotential in order to predict the air temperature itself. Future frame prediction models generate the next frame(s) from a given number of preceding frames. This challenge shows striking similarities to weather forecasting as both fields are dealing with a spatio-temporal forecasting problem. The idea is that a suitable deep learning model is able to learn the time-dependent transport patterns of the air temperature near the ground to generate predictions for the next hour(s). The experiments show that the model can make better hourly and even several-hour predictions than the persistence assumption. Contrary to the assumption that a greater variety of data provides better results, the experiments show that fewer variables provide better results. Essential for the application of future frame prediction models to weather forecasting is the integration of prior physical knowledge, especially for the data preparation, the selection and training of a model and its evaluation.

Contents

List of Figures

List of Tables

1 Introduction

Mankind has always striven to better understand and predict the world. This is especially true for weather predictions, which have experienced a remarkable success story over the past decades (Bauer, Thorpe, & Brunet, 2015). The motivation to predict the weather is both intrinsic and extrinsic. On the one hand, there is the human drive to grasp everything and on the other hand, there is the competitive advantage that successful weather forecasts entail, for instance in industry, agriculture, or military.

Weather predictions have greatly improved through (i) the integration of advanced theory in the numerical weather prediction (NWP) centers, (ii) a richer range of data in terms of quantity, variety and quality, (iii) observational systems and their improved and increased assimilation of data and finally, (iv) the rapidly increased computational power (Reichstein et al., 2019). Nonetheless, our capabilities to collect the data exceeds by far our capabilities to process and understand them. This implies that our ability to make predictions has not increased to the same extent that our ability to collect data in recent decades has. However, recent advances in machine learning offer new opportunities to expand our knowledge of the weather system. These promising approaches need to be further developed within the field of geo-scientific analysis, especially within the spatio-temporal context.

In the past, machine learning approaches were divided into spatial and sequence learning, for instance object classification and speech recognition. Currently, there is great interest in combining these two fields, especially for future frame prediction from the field of computer vision (CV). Based on a number of preceding frames, for example from a video of a driving car, a deep learning model predicts the following frames for the sequence in the future (Lotter, Kreiman, & Cox, 2016). This challenge shows striking similarities to many dynamic geo-scientific problems, such as weather forecasting. Both future frame prediction and weather forecasts are dealing with a spatio-temporal forecasting problem. Image sequences of weather maps generally exhibit similar motion changes from one hour to the next. Within the new field of future frame prediction, a small group of researchers has therefore been working on applying this method to weather forecasts. First successful applications already exist with rain and temperature forecasts (Bihlo, 2019; Shi et al., 2017; de Bezenac, Pajot, & Gallinari, 2017).

The aim of this study is to provide a comprehensive insight into deep learning methods of future frame prediction and their application to weather forecasting. In particular, the potential of deep neural networks for the prediction of air temperature over Europe will be investigated by including more information in the input than previous studies have used. In addition to air temperature, surface pressure and the 500 hPa geopotential will be used to predict the air temperature itself. Different configurations will be tested and compared.

The study is structured as follows: Section 2 provides the theoretical background of different neural networks which are required for predicting future frames of videos. Section 3 presents selected literature covering future frame models, especially the ones dealing with weather forecasting. The study setup in Section 4 explains the data and the data extraction process and the model architecture. Section 5 describes the experiments and evaluates the results. Finally, in Section 6 the selected model and the results are critically discussed and Section 7 concludes the study with a summary of the results and an outlook for future research.

2 Background

This section aims to provide the basic concepts behind the models that generate future frame predictions. The focus lies on the machine learning algorithms that are used in the adapted `PredNet` model for this study (Lotter et al., 2016). The intuitive idea behind predicting future frames by a given number of preceding frames is the combination of a convolutional neural network (CNN) in order to capture the spatial information and a recurrent neural network (RNN) to capture the temporal information.

2.1 Recurrent Neural Network (RNN)

Recurrent Neural Networks (RNNs) are used for processing sequential data (Rumelhart, Hinton, Williams, et al., 1988). In comparison to feed-forward neural networks, RNNs are utilizing cyclic information by allowing connections between neurons within the same layer or even backwards. Figure 2 illustrates typical recurrent connections, where V shows a recurrent connection between a hidden unit and W shows a recurrent connection from one output from a hidden unit to another hidden unit at the next time step. This enables information to persist through several time steps and allows sequence learning.

However, a drawback of RNNs is dealing with information over a longer period of time which might lead to the vanishing and exploding gradient problem (Bengio, Simard, Frasconi, et al., 1994). When facing the vanishing gradient problem the weights of the network will not be updated properly as the gradient becomes vanishingly small. The earlier the weights in the network reside, the more terms are going to be included in the product to calculate

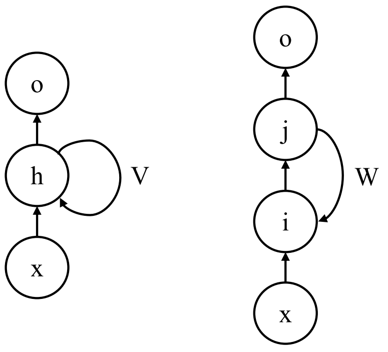

Figure 2: Exemplified recurrent connections in a RNN. Adapted from Goodfellow, Bengio, & Courville (2016)

the gradient. And the more terms that are less than one are multiplied, the quicker the gradient is going to vanish. As the weights get proportionally updated with the gradient and the gradient is vanishingly small, this update is going to be vanishingly small as well. These barely updated or even stuck weights cannot account to reduce the loss and therefore affect the ability of the network to learn. The opposite applies for exploding gradients. Updates of weights move in such huge steps that the optimal value cannot be achieved, because the proportion of by which the weights become updated is just too large (Goodfellow, Bengio, & Courville, 2016). Hochreiter (1997) encountered these problems by extending the RNN, as explained in the subsequent subsection.

2.2 Long Short-Term Memory (LSTM)

In order to overcome the vanishing and exploding gradient problem Hochreiter (1997) introduced long short-term memory (LSTM) or long short-term memory cells (LSTMs). LSTM is a particular type of RNN but with more complex so called memory cells. These memory cells contain gates that can regulate the flow of information and can learn which data in a sequence is important to keep or forget. By doing that it learns to use relevant information to make predictions. LSTM can be found in speech recognition, speech synthesis and text generation.

Figure 2.2 shows a LSTM cell. The core concept of LSTM is the cell state (C_t), which is the upper horizontal line. The cell state contains a current set of information at every time step and information can be added or removed by the gates. Gates consist of a sigmoid neural net layer (σ) and a point-wise multiplication operation (x). An output of the sigmoid neural layer of one means that everything goes through, and zero that nothing goes through.

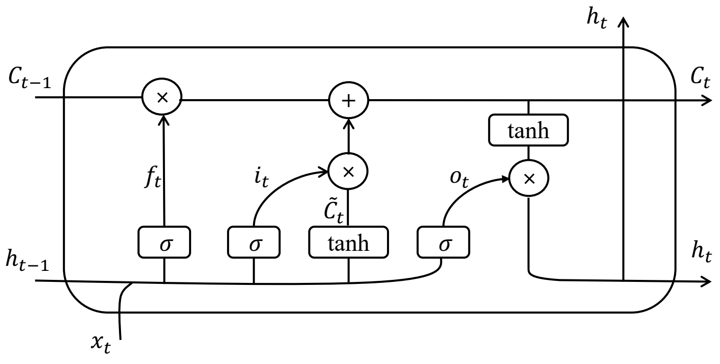

Figure 3: A LSTM cell. Adapted from Olah (2015)

There are three of these gates in a LSTM cell. The first gate works as the forget gate and decides which information can be thrown away (f_t). The next gate, the input gate decides which value to update (i_t). Subsequently, a tanh layer generates new candidate values that can be added to the cell state ($\tilde{C_t}$). The cell state gets updated by multiplying the old state by f_t to forget and adding $i_t \cdot \tilde{C_t}$ for the new candidates scaled by how much the state values should be updated. The output is the filtered state cell. First, a sigmoid layer decides what parts to output (o_t). This output is multiplied by the cell state after tanh to output only the relevant parts (Olah, 2015).

2.3 Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) (LeCun et al., 1989) are typically used for processing data with a spatial structure. The most common processed data is image data which can be thought of as a multidimensional grid of pixels. A CNN is able to successfully convert images into a format that is easier to process while capturing the spatial dependencies in an image. These spatial dependencies are captured by applying relevant filters, so called kernels, onto the data. This process of applying the kernel over the whole image by sliding over it is called a convolution. Figure 2.3 depicts an example of a 3×3 kernel moving nine times over a 5×5 image multiplying each cell with the corresponding value and adding it up. This results in a 3×3 convolved feature vector. A CNN usually applies several filters to an image. These filters are typically quadratic and move over the image with a certain stride value. The stride value sets the distance between each step of the kernel. In the example at hand the stride value equals one. Typically, the first layer of a CNN captures low-level features such as edges (horizontal, vertical, corners, etc.). Subsequent layers capture high-level features such as geometric figures (circles, rectangles etc.), specific forms (eyes, doors etc.), or even

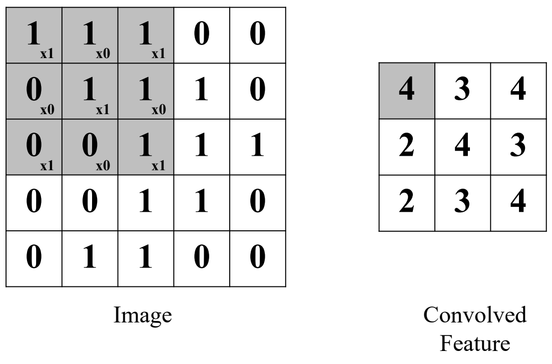

Figure 4: Example of a 5 × 5 convolution with a 3 × 3 kernel

complex forms (humans, cars etc.). With more and more layers, the network gets a more thorough understanding of the images (Olah, 2014; Goodfellow et al., 2016).

Typically a CNN consists of three stages. In the first stage it performs several convolutions to obtain a linear set of activations which are fed into a rectified linear unit (ReLU) activation in the second stage. Finally, in the third stage a pooling function is used to further modify the output. The most prominent pooling functions are max pooling (Zhou & Chellappa, 1988) and average pooling. The former operation returns the maximum output within a rectangular neighborhood, whereas the latter returns the average from the rectangular neighborhood (see Figure 5). Pooling aims at making the representation invariant to minor changes in the input. That means that minor changes in the input do not affect most of the pooled outputs as the max pooling units are only sensitive to the maximum value in the neighborhood. This invariance to minor input changes allows the network to detect features even though they do not appear at the same exact position.

Instead of generating a more abstract and less dimensional representation of the input, CNNs can also be used to generate or up-sample data from abstract representations. This is called transposed convolution or fractionally strided convolution or colloquially "deconvolution". When applying a transposed convolution, the input of the image is stretched by inserting empty rows and columns (zeroes) and then performing a regular convolution (see Figure 6) (Géron, 2017).

2.4 Convolutional LSTM

After introducing LSTMs and CNNs, it is important to highlight the convolutional LSTM (ConvLSTM) network. This is a special LSTM network which is designed to tackle spatio-temporal problems. The ConvLSTM has convolutional structures (see Subsection 2.3) in

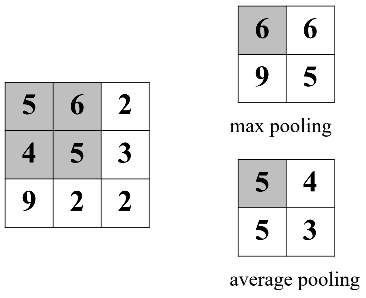

Figure 5: Exemplified max and average pooling

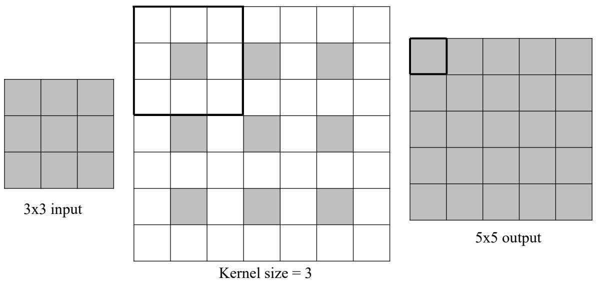

Figure 6: Transposed convolution from a 3x3 input to a 5x5 output

both the input-to-state, and state-to-state transitions. All the inputs, cell outputs, hidden states and gates are 3D tensors with the last two dimensions being spatial dimensions (rows and columns). By these means it is able to simultaneously capture temporal and spatial variations (Shi et al., 2015).

2.5 Loss Functions

Typically, deep learning algorithms are minimizing a loss function which compares the loss of the output of the model with the expected output. In the context of future frame prediction this means the generated next frame ($\hat{y}$) is compared to the next ground truth frame (y), both of size $m \times n$, to produce an error at pixel level. There are numerous different loss functions. Broadly said, two classes of loss functions can be distinguished, regular deterministic loss functions such as L1, or the other one using adversarial loss.

The L1 loss function minimizes the absolute differences (S) between the generated frame and the existing ground truth (Géron, 2017).

$$S = \sum_{i=1}^{m} \sum_{j=1}^{n} |y_{ij} - \hat{y}_{ij}|$$

Adversarial loss is used within a Generative Adversarial Net (GAN) (Goodfellow et al., 2014) setting, where the generative and the discriminative parts are multi-scale CNNs.

2.6 Evaluation

When evaluating the prediction quality of a model, the central idea is to measure the difference from the prediction to the actual ground truth. First, it needs to be clear what a good prediction actually is. But defining and measuring this can be as complicated as the task of generating the prediction itself (see Section 6). In literature there are umpteen metrics to measure the difference between the prediction and the ground truth. A trivial way would be to use the loss function for the evaluation as well. However, the most frequently used metrics are mean squared error (MSE) and peak signal-to-noise ratio (PSNR).

The mean squared error measures the average squared difference between the predicted next frame ($\hat{y}$) and the ground truth (y), both of size $m \times n$ (Fahrmeir, Heumann, Künstler, Pigeot, & Tutz, 2016). The closer the images, the lower the MSE. The MSE of two identical images is 0.

$$\text{MSE}(y, \hat{y}) = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} (y_{ij} - \hat{y}_{ij})^2$$

The peak signal-to-noise ratio is typically used to measure the quality of compressed pictures, whereas mathematically it is just a logarithmic representation of the MSE (Hore & Ziou, 2010; Winkler & Mohandas, 2008). The closer the images, the higher the PSNR.

$$\text{PSNR}(y, \hat{y}) = 20 * \log_{10} \left(\text{pixelMax} \, / \sqrt{\text{MSE}(y, \hat{y})} \right)$$

3 Related Work

Since 2014 more than 40 different architectures of deep models for future frame prediction have been proposed. This section presents selected future frame models. The selection is based on outstanding features such as novel approaches or the combination of different approaches. In addition, the selection includes all models that are dealing with weather forecasting using future frame prediction. Table 1 contains an overview of the compared future frame prediction models. In the following, each model is described with its characteristic feature and the data it was trained on.

Srivastava et al. (2015): Unsupervised Learning of Video Representations using LSTMs

This paper is one of the first approaches to successfully predict future frames given a certain input. The authors developed a LSTM encoder-decoder framework to learn video representations. A sequence of frames is fed into an Encoder LSTM and then this representation is decoded by another LSTM. In their experiments, they trained their model on their self-developed moving MNIST dataset, where two digits are moving within the image. The model manages to predict a persistent motion for several frames into the future. Furthermore, the model was trained on three datasets containing human actions, namely UCF-101 dataset (Soomro, Zamir, & Shah, 2012), HMDB-51 (Kuehne, Jhuang, Garrote, Poggio, & Serre, 2011) and Sports-1M (Karpathy et al., 2014). But instead of directly feeding the pictures into the model at pixel level, they used a CNN pretrained on ImageNet (Deng et al., 2009) to extract the RGB features. The predictions of the model capture some basic representations, but the images are very blurry.

Shi et al. (2015): Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting

In their paper Shi et al. applied future frame prediction methods for the first time to weather forecasting, and precipitation nowcasting in particular. They developed the convolutional LSTM (ConvLSTM) by extending a fully connected LSTM with convolutional structures in both the input-to-state and state-to-state transitions. All the inputs, cell outputs, hidden states, and gates are 3D tensors of which the last two dimensions are the spatial dimensions, i.e. rows and columns. The ConvLSTM model was first trained and tested on the moving MNIST dataset (Srivastava et al., 2015) in order to prove the generalization ability. It is able to predict the overall motion, but the resulting digits are very blurred. The second and main dataset where the model was trained and tested on is the radar echo dataset. It contains three years of weather radar intensities collected in Hong Kong between 2011 and 2013. The radar data is recorded every six minutes, resulting in 240 frames per day. In order to predict precipitation, the top 97 rainy days were selected. The intensity values were normalized and afterwards the radar maps were cropped, re-sized and noise reduced by applying K-means clustering to the monthly pixel average. The authors compared the ConvLSTM model with the Real-time Optical flow by Variational methods for Echoes of Radar algorithm (ROVER) (Woo & Wong, 2014). The algorithm calculates the optical flow of a sequence of radar maps and solves the semi-Lagrangian advection equation on the flow field in order to make predictions (Brox, Bruhn, Papenberg, & Weickert, 2004; Bridson, 2008). The results of the ConvLSTM and ROVER were compared by commonly used precipitation nowcasting metrics: rainfall mean squared error, critical success index, false alarm rate, probability of detection, and correlation. The results show that ConvLSTM outperforms ROVER for precipitation nowcasting. The authors argue that their model is able to handle boundary events better, such as a sudden agglomeration of clouds at the boundary. If the model has seen this type of formations and sudden changes before, it can reasonably predict the precipitation. ROVER, or especially optical flow can hardly achieve that.

Oh et al. (2015): Action-Conditional Video Prediction using Deep Networks in Atari Games

Oh et al. (2015) introduced two models in order to predict future image frames of Atari games. As future image frames in games depend on control variables or actions and previous frames the authors defined it as an action-conditional video modeling problem. The special feature of this approach is that the actions are utilized in order to predict future frames. The authors propose a feedforward model by means of a CNN Autoencoder (AE) and a recurrent model, which consists of a CNN AE and a LSTM. In both approaches they utilize the action-transformed features. For their data they use the Atari games from the Arcade Learning Environment (ALE) (Bellemare, Naddaf, Veness, & Bowling, 2013). The experiments show that the model can even generate realistic frames for over 100 action conditioned time steps.

Klein et al. (2015): A Dynamic Convolutional Layer for Short Range Weather Prediction

Klein et al. (2015) developed the dynamic convolutional layer which generalizes the convolutional layer in order to make short range weather predictions. In contrast to a 'regular' convolutional layer, the dynamic convolutional layer receives two inputs, namely the features maps from the previous layer and the filters. So, instead of constant filters, the dynamic convolutional layer uses filters that will vary from input to input. The filters are calculated by learning a function that maps the input to the filters. The authors use datasets containing radar images taken in Tel Aviv, Israel, Davenport, Iowa and Kansas City, Missouri. The datasets were split into train, validation and test and contain 32,000 images each on the training, 4,800 on the validation and 3,200 on the testing set. On the radar images a constant color transformation function was applied and the authors point out that a simple gray-scale transformation would not produce desirable results as the colorbar of the radar images is not monotonic. The authors compared their model with the last frame, linear regression, a global motion estimator and a regular feedforward CNN. They compare the predictions by means of the squared Euclidean loss. The dynamic CNN outperforms all baseline methods on all three datasets.

Mathieu et al. (2015): Deep multi-scale Video Prediction beyond Mean Square Error

In their paper Mathieu et al. (2015) dealt with the problem of blurry predictions which is caused by the standard mean squared error (MSE) loss function. MSE tends to produce blurry frames because the loss responds to the many possible futures by averaging them. The authors introduced a model utilizing adversarial loss from a GAN setting where the generative and the discriminative parts are multi-scale CNNs. Furthermore, they also introduced an image gradient difference loss (GDL) which penalizes the differences of image gradient predictions in the generative loss function. The authors trained and tested their model with the different loss functions on the Sports-1M (Karpathy et al., 2014) and the UCF-101 (Soomro et al., 2012) datasets. When comparing the PSNR and sharpness of the predictions, the model utilizing adversarial loss combined with gradient difference loss outperforms the other ones.

Patraucean et al. (2015): Spatio-temporal Video Autoencoder with Differentiable Memory

The authors introduced a model which consists of a CNN AE, ConvLSTM and optical flow. At each time step, the model receives the input frame, generates a prediction of the optical flow based on the current input and the LSTM memory state as a dense transformation map and applies it to the input to predict the next frame. They trained their model on the moving MNIST (Srivastava et al., 2015), HMDB-51 (Kuehne et al., 2011), PROST (Santner, Leistner, Saffari, Pock, & Bischof, 2010) and ViSOR (Vezzani & Cucchiara, 2010) datasets, whereas the last three datasets contain videos of human actions. Their model achieved the lowest per-pixel average error on moving MNIST compared to other baseline methods, namely CNN AE, CNN AE + LSTM and CNN AE + ConvLSTM.

Finn et al. (2016): Unsupervised Learning for Physical Interaction through Video Prediction

Finn et al. (2016) developed three action-conditioned models that predict transformations on the input pixels for next frame predictions. Their approach allows to distinguish different objects from each other and the background by merging information of appearance from preceding frames with motion predicted by the model. The appearance and the predicted motion are merged by setting the motion of pixels relative to the previous frame. The next frame is generated by applying this motion to the previous one. In order to test their model the authors introduced the robotic pushing dataset containing 59,000 robot pushing motions, consisting of 1.5 million frames and the corresponding actions at each point in time. Furthermore, they also tested their model on the Humans3.6M (Ionescu, Papava, Olaru, & Sminchisescu, 2013) dataset. Comparing the PSNR and structural similarity (SSIM), the results of the predictions show that their motion-predictive models quantitatively outperform the baseline methods FC LSTM and ConvLSTM, and qualitatively produce plausible future frames for at least ten timesteps. Their results indicate that models that explicitly transform pixels from preceding frames are able to better capture object motion.

Lotter et al. (2016): Deep Predictive Coding Networks for Video Prediction and Unsupervised Learning

Lotter et al. (2016) introduced a predictive neural network (`PredNet`) which is inspired by the neuroscience literature. The model consists of an encoding CNN, a ConvLSTM and a decoding CNN, whereas each layer in the network makes local predictions and only forwards the deviations from those predictions to the subsequent network layers. See Section 4 for a more detailed description of the model, as this study is based on it. The authors trained and tested their model on an artificial rotating faces dataset. The dataset consists of 3D faces randomly rotating at a constant velocity. The authors compared their model to the baseline methods of copying the last frame and a simpler ConvNet which forwards the predictions instead of the deviations. In terms of MSE and SSIM the `PredNet` outperformed the baselines methods. Applied to more complex real-world scenarios the model was trained on the KITTI dateset (Geiger, Lenz, Stiller, & Urtasun, 2013) and tested on the CalTech Pedestrian dataset (Schiele, Dollár, Wojek, & Perona, 2009). Both datasets consist of videos from a dashboard-mounted camera on a vehicle driving around cities. In terms of MSE and

SSIM the model outperformed the two baseline methods. Furthermore, although the model was trained to predict only one frame ahead, the model can predict multiple frames into the future by recursively feeding the predictions back into the model as inputs. The MSE of the model's prediction stayed well below the copying last frame method and increased fairly linearly.

Shi et al. (2017): Deep Learning for Precipitation Nowcasting: A Benchmark and A New Model

In this paper Shi et al. (2017) faced the problem of the previous introduced ConvLSTM model (Shi et al., 2015) which is location-invariant, whereas the weather is location-variant in general. The local correlation structure of motion patterns like rotating or scaling in consecutive frames is different for different positions within the frames and at different time steps. They proposed the Trajectory GRU (TrajGRU) model that is able to learn the local correlation structure for spatio-temporal data in order to provide very short range forecasts of rainfall intensity. Furthermore, as the data for precipitation nowcasting is highly imbalanced, the authors proposed the balanced mean squarred error (B-MSE) and balanced mean absolute error (B-MAE) for training and evaluation. These measures assign more value to heavier rainfalls. The TrajGRU model was trained and tested on the MovingMNIST++ (Shi et al., 2017) dataset where it outperformed other baseline models such as the ConvLSTM or Convolutional GRU (ConvGRU) (Ballas, Yao, Pal, & Courville, 2015). Moreover, they trained and tested the network on the HKO-7 dataset containing radar echo data from 2009 to 2015 collected by the Hong Kong Observatory (HKO). The radar images were taken at a height of 2km covering a $512km \times 512km$ grid with Hong Kong in its center. The radar data is recorded every 6 minutes, resulting in 240 frames per day. The images are linearly transformed to pixel values ranging from 0 to 255 and noise filtered. For training 812 rainy days were selected, for validation 50 and for testing 131. According to the critical success index (CSI) and Heidke Skill Score (HSS)(Hogan, Ferro, Jolliffe, & Stephenson, 2010) the TrajGRU outperformed the ConvLSTM ConvGRU, ROVER and last frame. Additionally, the results show that the models which are trained with balanced loss outperform the optical flow based models.

de Bezenac et al. (2017): Deep Learning for Physical Processes - Incorporating Prior Scientific Knowledge

de Bezenac et al. (2017) developed a model which incorporates physical background knowledge to forecast the sea surface temperature (SST). Their proposed model consists of a CNN that predicts a motion field from the input frames. Based on this motion field, a warping scheme distorts the last input to produce the future frame. The error is calculated using the Charbonnier penalty function between the predicted frame and the ground truth. Under specific parameter settings this function equals the l2 loss, but the authors underline the advantages of the reduction of outlier influence. Multi-frame predictions were obtained by recursively feeding back the predictions into the model as inputs. The special feature of the model is the inclusion of physics. The warping mechanism is adapted to the advection-diffusion principles which are responsible for temperature variation. This fluid transport problem occurs due to the advection and diffusion of fluids. Furthermore, the authors emphasize the possibility to extend the loss function by physical penalty terms. The model was trained on simulated SST data utilizing the NEMO (Nucleus for European Modeling of the ocean) engine (Madec et al., 2015). Daily temperatures from 2006 to 2015 were used for training (80%) and validation (20%). The years 2016 and 2017 were used for testing. The data is divided into several sub-regions and the daily SST of each sub-region was standardised using the mean and standard deviation of all available data for each day in order to remove seasonal effects. By means of MSE, the model was compared to the baseline methods of the state of the art numerical assimilation model by Béréziat and Herlin (2014), an autoregressive CNN, the ConvLSTM model (Shi et al., 2017) and the multi-scale CNN (Mathieu et al., 2015). The proposed model by de Bezenac et al. (2017) outperformed all baselines, especially the NN baselines.

Wichers et al. (2018): Hierarchical Long-term Video Prediction without Supervision

Wichers et al. (2018) introduced an unsupervised approach for long-term video prediction. In comparison to other long-term prediction models, their approach does not require images with annotated high-level structures such as human joint landmarks. First, their model encodes the input frame, discovers high-level features (human pose landmarks). Next, a LSTM generates a high-level encoding prediction of the landmarks into the future. Finally, a visual analogy network (VAN) (Reed, Zhang, Zhang, & Lee, 2015) decodes the final predicted image from the encoded prediction and the last seen frame. The authors trained and tested their model on an artificial bouncing shapes dataset and compared predictions of 1022 frames into the future with the model of Finn et al. (2016). The models were compared whether the shape was still present in the last frames and and if it has the correct color. In comparison to the other model, the shapes of the model of Wichers et al. (2018) did not disappear and had in approximately 97% the right color. The location of the prediction and the ground truth was not compared as the authors state that it is unrealistic to expect the model to accurately predict the location after 1000 time steps because of propagating errors. Furthermore, the model was trained and tested on the Human3.6M dataset and compared to the models of Finn et al. (2016) and Denton and Fergus (2018). According to crowd-sourced human preference, their model performs most of the times better or at least as well as the other ones. Additionally they ran a object detection model pretrained on the MS-COCO dataset (Howard et al., 2017; Lin et al., 2014). They developed a person score indicating the confidence of the detector to detect whether the frame contains a person. For long-term predictions their model outperformed the other two baselines. It is important to point out that their model is able to predict up to 127 frames into the future, which corresponds to a prediction of up to 20 seconds.

Lee et al. (2018): Stochastic Adversarial Video Prediction

In their paper Lee et al. (2018) combine (a) adversarial losses with (b) latent variational variable models that model the underlying stochasticity in order to produce naturalistic high-quality images. Inspired by the work of Denton and Fergus (2018) their model uses stochastic models for future frame prediction. These models are using the framework of variational autoencoders (VAEs) which predict possible futures by sampling latent variables. The predictions of the stochastic adversarial video prediction model were evaluated concerning realism, diversity and accuracy. Realism is measured by human judges according to a real vs. fake two-alternative forced choice (2AFC) test. Diversity is calculated by measuring the average distance between randomly sampled video predictions. This procedure evaluates the ability of the model to represent different possible futures. The accuracy is measured by calculating PSNR and SSIM. The model was trained and tested on the BAIR action-free dataset (Ebert, Finn, Lee, & Levine, 2017) containing a robot arm pushing objects and KTH dataset (Laptev, Caputo, et al., 2004) containing human actions. They compared their model with other stochastic video prediction models (Babaeizadeh, Finn, Erhan, Campbell, & Levine, 2017; Denton & Fergus, 2018), and modifications of their own model containing only the adversarial loss or the VAE. The results show that the model with the adversarial loss only generates realistic but less diverse predictions whereas the model with VAE only generates less realistic but more diverse predictions. Combining the two together leads to more realistic and diverse predictions.

Bihlo (2019): Precipitation nowcasting using a stochastic variational frame predictor with learned prior distribution

In their paper Bihlo (2019) introduced a stochastic variational frame prediction model based on the model proposed by Denton and Fergus (2018). The stochastic variational frame prediction model concatinates the CNN decoded image with stochastic information from the series of previous images obtained from the prior network. This information is then fed into the ConvLSTM to predict the future frame. The prior model is supposed to learn the fundamental physical rules how precipitation cells move and evolve in time and space. The prior model includes a Kullbeck–Leibler (KL) divergence term in its loss. The model was trained to predict one frame into the future and multi-frame predictions were obtained by recursively feeding back the predictions into the model as inputs. The model was trained on radar data obtained on four weather radar stations in Feldkirchen, Patscherkofel, Rauchenwarth and Zirbitzkogel in Austria. The orignally $1km \times 1km$ grid resolution recorded every five minutes was downsampled to a $5km \times 5km$ grid every 15 minutes with an image sizes of 160×110 pixels. The dataset contains radar images from 2014 to 2018, whereas four years were used for training and one year for testing. Only images containing at least 10,000 mm/h precipitation were selected. The authors compared their model with a standard ConvLSTM network by means of SSIM. For the first two frames the ConvLSTM outperforms their model but afterwards the stochatistic variational method significantly outperforms the ConvLSTM.

This study

The underlying study adopts a slight modification of the approach proposed by Lotter et al. (2016). The main difference is the dataset on which the model is trained. The (`PredNet`) is trained on weather data from 2015 and 2016 containing the air temperature, surface pressure and the 500 hPa geopotential. In addition, slight changes were made in the evaluation of the model. Besides the MSE, also the PSNR was calculated. For a detailed description of the study setup see Section 4. In comparison to the other future frame prediction models applied in the context of weather forecasting this study examines how the use of different input variables affects the predictions.

Author(s)	Model	Dataset(s)	Data type	Input frames	Output frames	Resolution	Loss
Srivastava et al. (2015)	LSTM AE	UCF101	HA	16	13	$32 \times 32 \times 3$	CE, L2
		HMDB-51	HA	16	13	$224 \times 224 \times 3$	
		Sports-1M	HA	16	13	$224 \times 224 \times 3$	
		Moving MNIST	A	10	10	$64 \times 64 \times 1$	
Shi et al. (2015)	ConvLSTM	Moving MNIST	A	10	10	$64 \times 64 \times 1$	CE
		Radar echo	W	5	15	$100 \times 100 \times 1$	
Oh et al. (2015)	CNN AE, CNN AE + LSTM	ALE	A	20	10	$210 \times 160 \times 3$	L2
Klein et al. (2015)	Dynamic CNN	Radar images	W	4	1	$70 \times 70 \times 1$	Euclidean
Mathieu et al. (2015)	Multi-scale CNN	Sports-1M	HA	4	1	$32 \times 32 \times 3$	adv.
		UCF-101	HA	4, 8	1, 8	$32 \times 32 \times 3$	
Patraucean et al. (2015)	CNN AE + ConvLSTM + Optical Flow	Moving MNIST	A	1	1	$64 \times 64 \times 1$	Binary CE
		HMDB-51	HA	1	1	$na \times na \times 3$	
		PROST	HA	1	1	$na \times na \times na$	
		ViSOR	HA	1	1	$na \times na \times na$	
Finn et al. (2016)	Dynamic ConvLSTM	Robotic pushing	R	10	10-20	$64 \times 64 \times 3$	L1, L2, GD
		Humans3.6M	HA	2	8-18		
Lotter et al. (2016)	CNN + ConvLSTM	Rotating faces	A	10	1-5	$64 \times 64 \times 1$	L1
		KITTI	D			$128 \times 160 \times 3$	
		CalTech Pedestrian	D			$128 \times 160 \times 3$	
Shi et al. (2017)	Trajectory GRU	Moving MNIST++	A	10	10	$64 \times 64 \times 1$	Balanced MSE, balanced MAE
		HKO-7	W	5	20	$480 \times 480 \times 1$	
de Bezenac et al. (2017)	CNN + Warping	Sea surface temperature	W	4	6	$64 \times 64 \times 1$	L2
Wichers et al. (2018)	AE + VAN + LSTM	Humans3.6M	HA	5	32-126	$64 \times 64 \times 3$	L2, adv.
		Bouncing shapes	A	3	16-1022	$na \times na \times 1$	
Lee et al. (2018)	VAE + ConvLSTM	Robotic Pushing	R	2	10	$64 \times 64 \times 3$	adv., binary CE, L1
		KTH	HA	10	10	$64 \times 64 \times 3$	
Bihlo (2019)	Stochastic VAE + convLST	Radar	W	5	10	$160 \times 110 \times 1$	L2, KL
This study	CNN + ConvLSTM	ERA5 reanalysis	W	10	1-5	$128 \times 160 \times 3$	L1

Table 1: Comparison of selected future frame prediction models. In the model column AE indicates autoencoder whereas VAE indicates variational autoencoder. In the data type column HA, A, D and W indicate human action, artificial, driving and weather. In the loss column CE is cross entropy, adv. is adversarial loss, GD is gradient difference and KL is the Kullback–Leibler divergence.

4 Study Setup

This section provides a detailed description of the processed weather data, the architecture of the applied `PredNet` by Lotter et al. (2016) and how the model was finally trained.

4.1 Data and Data Extraction

As mentioned before, there is a growing amount of data that can be used for weather forecasting. In numerical weather prediction predicting the weather is basically a matter of solving thermodynamic equations, which, however, are always under-determined due to the large scale width. The temperature at a location is determined by local heat sources and sinks (e.g. radiation, evaporation, heat transport in the soil or water) and by advection, i.e. transport of air masses. The transport patterns are complex and very different depending on the height above the ground. Therefore, it is not at all trivial to deduce air movements from satellite images, because one has to know what is transported at which altitude.

The idea now is that a well-built deep learning network will be able to learn the transport patterns contained in the time-dependent, three-dimensional model fields and thus at least predict large-scale temperature patterns near the ground and their temporal change. For a successful temperature prediction, suitable variables have to be selected in order to obtain a representation of the three-dimensional model fields as close as possible. The following parameters were selected for this purpose, which are interdependent in terms of scale in the atmosphere:

1. **Air temperature**: In order to predict the air temperature, this parameter itself is naturally indispensable. More precisely, it is the air temperature at two meters above the surface of land, sea or waters (t2). The temperature is measured in kelvin (K) and by subtracting 273.15 it can be converted to degrees Celcius (°C) [1].

2. **Surface pressure**: This parameter is the pressure of the atmosphere adjusted to sea level. The pressure is the weight of air above the measurement point. More specifically, the surface pressure is the mean level pressure (msl), which is used to identify low and high pressure systems (cyclones and anticyclones). It is measured in pascals (Pa) [2]. According to Gay-Lussac's law an equal volume of gas (such as oxygen, hydrogen or the

[1]ECMFW (2010). 2 metre temperature. https://apps.ecmwf.int/codes/grib/param-db?id=167. [Online; accessed 1-July-2019]

[2]ECMFW (2010). Mean sea level pressure . https://apps.ecmwf.int/codes/grib/param-db?id=151. [Online; accessed 1-July-2019]

atmoshpere of Earth) expands or diminishes the same amount for a given temperature increase or decrease (Benedict, 1984). This means that the pressure of a constant volume of gas is proportional to its temperature. This close correlation of the two parameters qualifies the ground pressure for temperature predictions.

3. **500 hPa geopotential**: This parameter can be derived from the geopotential height, which can be described as the gravity-adjusted height (Mohanakumar, 2008). The 500 hPa geopotential corresponds to the geopotential height that is required to reach the given pressure. This means that it shows how far one has to go up in the atmosphere until the pressure will drop to 500 hPa. Hence, in contrast to ground weather maps, the pressure values are not displayed in a height, but the height of a certain pressure value. The average height above sea level of the 500 hPa geopotential is 5574 m'. The movements of the 500 hPa geopotential level have a regulatory effect on the weather parameters beneath as they roughly move with the winds on the 500 hPa level. High heights indicate anticyclones whereas low heights indicate cyclones. The same applies for ridges and troughs [3].

The data described originates from the European Centre for Medium Range Weather Forecasts (ECMWF). The ECMWF provides hourly reanalyses data which are regularly retrieved and archived in the netCDF format at the Jülich Supercomputing Centre (JSC). For the underlying study ERA5 reanalysis data is used which consists of hourly estimates of various climate variables. The data covers a $31km \times 31km$ grid and by means of data assimiliation systems and highly developed modelling it combines historical observations into global estimates (Guillory, 2017). Plots of the air temperature, surface pressure and 500 hPa geopotential are illustrated in Figures 7, 8 and 9. The plots are created with Panoply, which is a tool developed by NASA to plot geo-referenced arrays in formats like netCDF [4].

The ERA5 reanalysis data is staged within the JSC, whereas the described parameters are selected and extracted. Each original netCDF file for each hour contains around 60 different parameters and is roughly 3GB large. After the staging process by means of a JUBE script [5] each new netCDF file contains only the air temperature, surface pressure, the 500 hPa geopotential and some meta information like date, time, lats and lons and is 5MB large.

[3]ECMFW (2010). Geopotential. https://apps.ecmwf.int/codes/grib/param-db?id=129. [Online; accessed 1-July-2019]

[4]Panoply. https://www.giss.nasa.gov/tools/panoply/. [Online; accessed 1-July-2019]

[5]JUBE. http://www.fz-juelich.de/ias/jsc/EN/Expertise/Support/Software/JUBE/JUBE2/_node.html. [Online; accessed 1-July-2019]

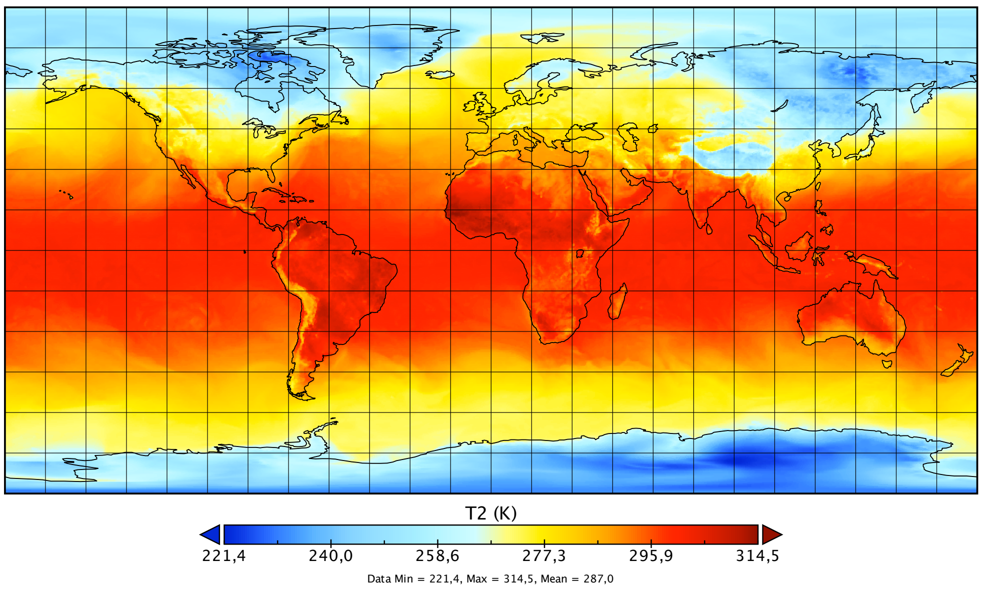

Figure 7: Plot of air temperature on 25.02.2016 at 6pm.

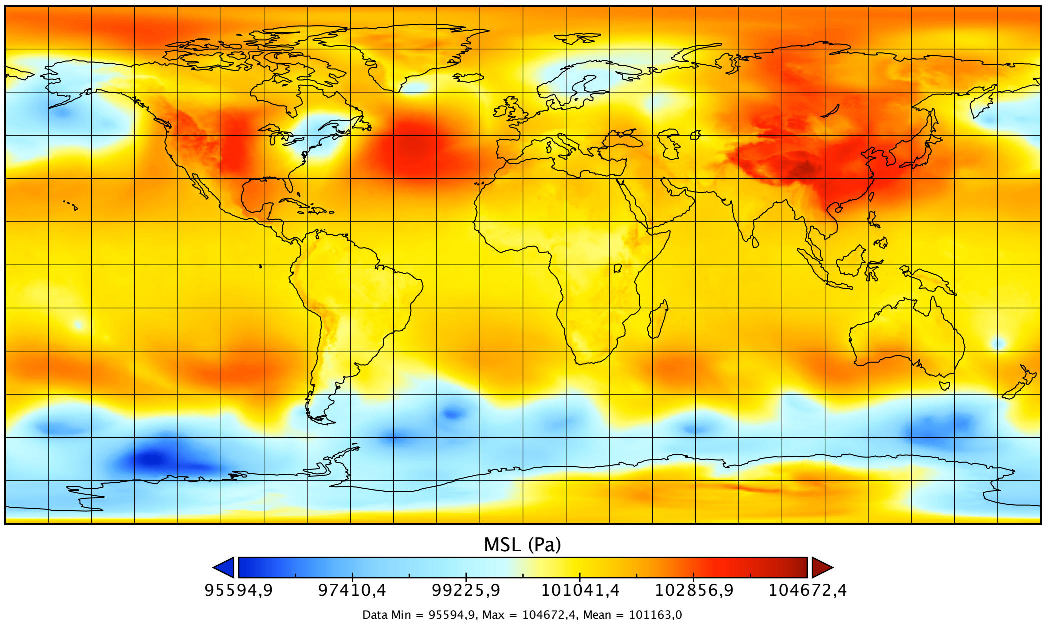

Figure 8: Plot of surface pressure on 25.02.2016 at 6pm.

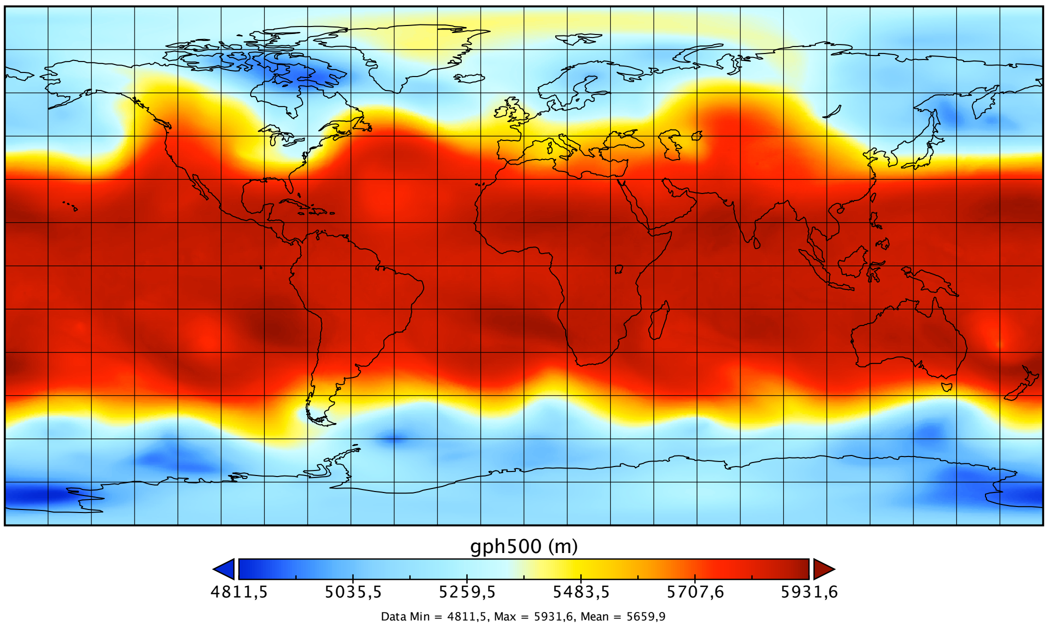

Figure 9: Plot of 500 hPa geopotential on 25.02.2016 at 6pm.

The final model expects three datasets, namely a train, validation and test set. Each set has a shape of $(n, 10, 128, 160, 3)$, where n is the number of sequences, 10 the amount of images in one sequence, 128 and 160 the desired image size in pixels and 3 the number of channels. In the classical context of computer vision the channels would represent the red, green and blue (RGB) channels. In the application of this study these channels are now the selected parameters. Instead of red, green and blue there are the air temperature, surface pressure and the 500 hPa geopotential now stacked on each other. Therefore, the next step is to cut the netCDF data to the desired size, stack the parameters on each other, create according sequences and save them into the train, test and validation set. The resulting data sets were saved as hickle (hkl) files. It is also possible to combine the variables in other compositions. This is done and explained in Section 5. As the desired image size is smaller than the original netCDF files, the weather maps were cut out above Europe. See Figure 10 for three exemplified air temperature plots above Europe at arbitrary points in time. During the training and testing the data was normalized between 0 and 1. In total, hourly weather data from 2015 and 2016 were used, which corresponds to a data basis of 17,544 records. For the training 80% of the data was utilized, 5% for the validation set and 15% for the test set.

The dataflow between the preprocessing, training and evaluation is depicted in Figure 16.

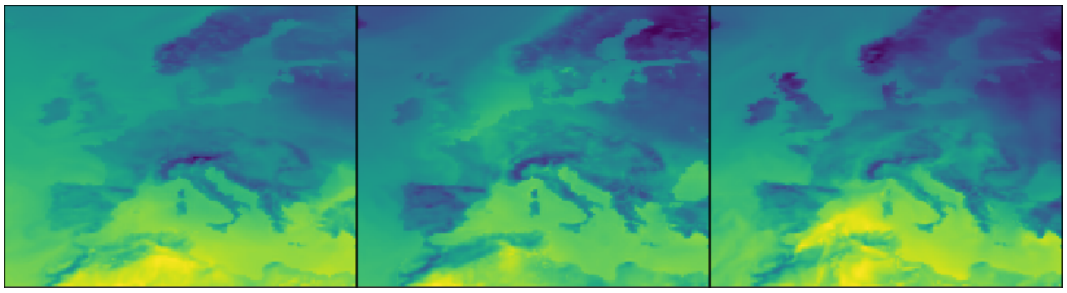

Figure 10: Three plots of cutouts above Europe at arbitrary points in time with a resolution of 128 x 160 pixels.

4.2 Model Architecture

The applied model is the slightly modificated `PredNet` proposed by Lotter et al. (2016). The model is displayed in Figures 11 and 12 and consists of several repeating layers, whereas each layer consists of different modules. Figure 11 depicts the information flow within two layers and Figure 12 reveals a more detailed view of one layer of the model. The network architecture is inspired by the work of Rao and Ballard (1999) about predictive coding in neuroscience and the Deep Predictive Coding Networks of Chalasani and Principe (2013).
Briefly, each layer of the model consists of the following four modules: the representation module R_l, the prediction module $\hat{A}_l$, the input convolutional module A_l, and the error term E_l. The representation layer R_l is a ConvLSTM which generates the t+1 prediction $\hat{A}_l$ of the input A_l. Subsequently, the error E_l is generated by subtracting $\hat{A}_l$ from the target A_l. Then E_l is split into two parts, a rectified positive and a negative term. After passing a convolutional layer, E_l becomes the input $A_l + 1$ for the next layer. Also, R_l receives a copy of E_l and the output of $R_l + 1$. Compared to the intuitive structure of a future frame prediction model consisting of a convolutional network encoding the images and capturing the spatial information, a recurrent network capturing the temporal aspect and a generative network constructing the next frames, A_l and E_l represent the decoding part and E_l the temporal and generative part.
In more detail the model processes a sequence of images, x_t as described in the following. The actual images are fed into the lowest target, i.e. $A_0^t = x_t \, \forall \, t$, where 0 stands for the lowest layer. In higher layers (A_l^t with $l > 0$) E_{l-1}^t becomes the input after applying a convolution over the error units, a rectified linear (ReLU) activation and max-pooling:

$$A_l^t = \text{MaxPool}\left(\text{ReLU}\left(\text{CoNV}\left(E_{l-1}^t\right)\right)\right)$$

For the representation module R_l^t ConvLSTM units are used, introduced by Shi et al. (2015).

The hidden state of the representation module R_l^t is updated according to $R_l^t - 1$, E_l^{t-1} and $R_l^t + 1$. By means of nearest-neighbor, $R_l^t + 1$ is upsampled before, because of the pooling in the forwarding.

$$R_l^t = \text{CONVLSTM}\left(E_l^{t-1}, R_l^{t-1},\ \text{UPSAMPLE}\ \left(R_{l+1}^t\right)\right)$$

The predictions $\hat{A}_l^t$ are generated as follows. First R_l^t is convoluted and then followed by a non-linear ReLU activation. For the final prediction in the lowest layer $\hat{A}_l^t$ is passed through a saturating non-linearity, which ensures that the predictions do not exceed the maximum pixel value of x.

$$\hat{A}_l^t = \text{ReLU}\left(\text{Conv}\left(R_l^t\right)\right)$$

The error term is calculated by subtracting A_l^t from $\hat{A}_l^t$ (and vice versa) from each other. It is split into a positive and negative ReLU-activated error, which in turn are concatenated again.

$$E_l^t = \left[\text{ReLU}\left(A_l^t - \hat{A}_l^t\right); \text{ReLU}\left(\hat{A}_l^t - A_l^t\right)\right]$$

The model states are updated in two phases: First, a top-down pass to compute the R_l^t states and second a forward pass to generate the predictions, error terms and targets. At the beginning R_l and E_l are initialized with zero, which explains the initial blank predictions of the model (see Section 5 for examples). The model is trained to indirectly minimize the L1 error. This is done by minimizing the weighted sum of the error terms. These error terms consist of a subtraction followed by a ReLU activation, which in turn equals the L1 error.

The hyperparameters are set to a 4 layer model with 3×3 kernels for all convolutions and layer channel sizes for both A_l and R_l of (3, 48, 96, 192). The model weights were optimized using the Adam algorithm (Kingma & Ba, 2014) using a loss with a weight of 1 on the lowest layer and 0.1 on the upper layers. The initial learning rate was set to $\alpha = 0.001$ and decreased by a factor of 10 after half of the training. The exponential decay rate for the first and second moment estimates were set to $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

The model is implemented in Python3 using Keras (Chollet et al., 2015) with TensorFlow backend. The training was done using 4 NVIDIA Tesla K80 GPUs on the JURECA supercomputer of the JSC.

5 Experiments

In the following, the experiments and their results are presented. Section 5.1 presents the results of the feature testing, where the selection and combination of different variables was

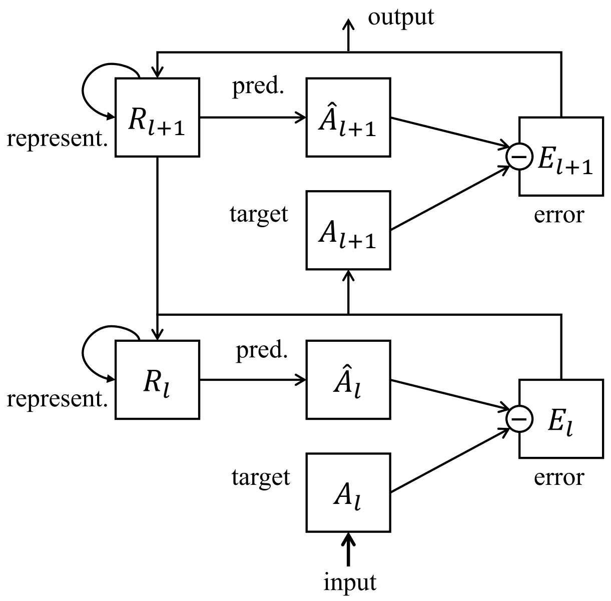

Figure 11: Illustration of the information flow between two layers within the model. R_l is the representation module (ConvLSTM), which outputs a prediction $\hat{A}_l$ at each timestep. The prediction is compared to the target A_l and an error term E_l is calculated serving as the next target and the representation module. Adapted from Lotter et al. (2016)

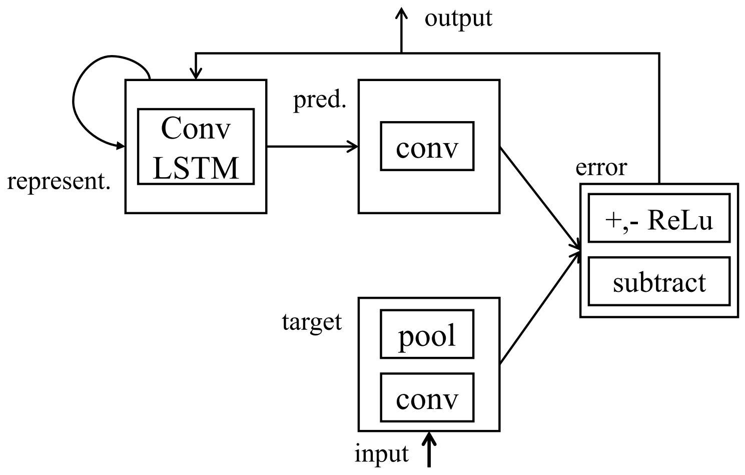

Figure 12: Detailed view of one layer of the model. Adapted from Lotter et al. (2016)

tested. Subsequently, Section 5.2 presents the capabilities of the model to predict several steps into the future.

5.1 Feature Testing

As presented in Subsection 4.1, three variables were selected to train the model, namely the air temperature, surface pressure and 500 hPa geopotential. The idea is that the selected variables provide an appropriate representation of the time-dependent, three-dimensional model fields with that the network can understand and predict the large-scale temperature patterns near the ground and their temporal change.

When creating data sets it is also possible to create any other combination of variables. Table 2 shows the combination of the different variables and their evaluation. The model was trained and tested with each of the variable combinations. In total, hourly weather data from 2015 was used, which corresponds to a data basis of 8,760 records. For the training 80% of the data was utilized, 5% for the validation set and 15% for the test set. Each model was then evaluated using MSE and PSNR and then compared to the MSE and PSNR of the last seen frame compared to the ground truth of $t + 1$. The comparison of the model's prediction with the last frame is the simplest comparison of the performance of the model. The easiest way to predict the weather is to assume that it will be the same in a future point of time as it is right now. This baseline of the persistence assumption is especially a good benchmark for short range forecasts as it is the case here. In an hour the temperature will hardly change. The MSE and PSNR of each model were calculated over the whole sequence of the variables except of the first one, because the representation and error modules are initialized with zero at the beginning, which leads to the initial uniform predictions (see Figure 14 for examples). On the next time steps the prediction is getting better and better after processing more and more motion information. The farther to the end of the sequence, the better the predictions. For this reason, in addition to evaluating the model on the basis of the entire sequence, a sole evaluation of the last prediction of each sequence was also carried out. The results show that the combination of two times the air temperature and one uniform layer (`t2_2`) leads to the best results in terms of the general model MSE and PSNR and especially in terms of the model MSE and PSNR just from the model's prediction of the last frame of the sequences. The model MSE of `t2_2` outperforms the persistence assumption of the last frame MSE by 224%. The model MSE of just the last frame in the sequences outperforms the last frame by even 406%. The model PSNR and the model PSNR of just the last frame outperform the baseline by 8.3% and 14,3%. The combinations `t2_1` and `t2_3` follow in descending order. This means that the best three combinations of variables are only compositions of temperature. Only in fourth place is there a combination that also includes the surface pressure `(t2_2MSL_1)`.

After the feature testing another model was trained for the variable combination `t2_2` with more epochs, more samples per epoch, an increased number of sequences for validation and more data. All in all, hourly weather data from 2015 and 2016 were used, which corresponds to a data basis of 17,544 records (training 80%, validation 5% and test 15%). The resulting `t2_2Max` model MSE and the model MSE of just the last frame outperform the baseline by 295 % and 542% as can be seen in Table 3.

Data	Model MSE	Model MSE last frame	Last frame MSE	Model PSNR	Model PSNR last frame	Last frame PSNR
t2_1MSL_1gph500_1	5.5×10^{-5}	4.0×10^{-5}	6.5×10^{-5}	42.56	43.95	41.89
t2_3	4.4×10^{-5}	3.0×10^{-5}	6.5×10^{-5}	43.56	45.17	41.89
t2_2	$\mathbf{2.9 \times 10^{-5}}$	$\mathbf{1.6 \times 10^{-5}}$	6.5×10^{-5}	**45.35**	**47.86**	41.89
t2_1	3.5×10^{-5}	2.5×10^{-5}	6.5×10^{-5}	44.53	46.07	41.89
t2_2MSL_1	4.8×10^{-5}	3.5×10^{-5}	6.5×10^{-5}	43.17	44.52	41.89
t2_1MSL_2	5.6×10^{-5}	4.1×10^{-5}	6.5×10^{-5}	42.54	43.83	41.89
t2_2gph500_1	7.0×10^{-5}	5.4×10^{-5}	6.5×10^{-5}	41.52	42.71	41.89
t2_1gph500_2	5.7×10^{-5}	4.3×10^{-5}	6.5×10^{-5}	42.41	43.65	41.89

Table 2: Results of the feature testing. `t2_3` stands for three layers of t2 (air temperature). `t2_2MSL_1` stands for two layers of t2 and one layer of MSL (surface pressure). On the contrary `t2_1MSL_2` stands for one layer of t2 and two layers of MSL. The same applies for the combination of t2 and gph500 (500 hPa geopotential).

Data	Model MSE	Model MSE last frame	Last frame MSE	Model PSNR	Model PSNR last frame	Last frame PSNR
t2_2Max	$\mathbf{2.2 \times 10^{-5}}$	$\mathbf{1.2 \times 10^{-5}}$	6.5×10^{-5}	**46.53**	**49.09**	41.89

Table 3: Results of `t2_2` with maxed out parameters for training.

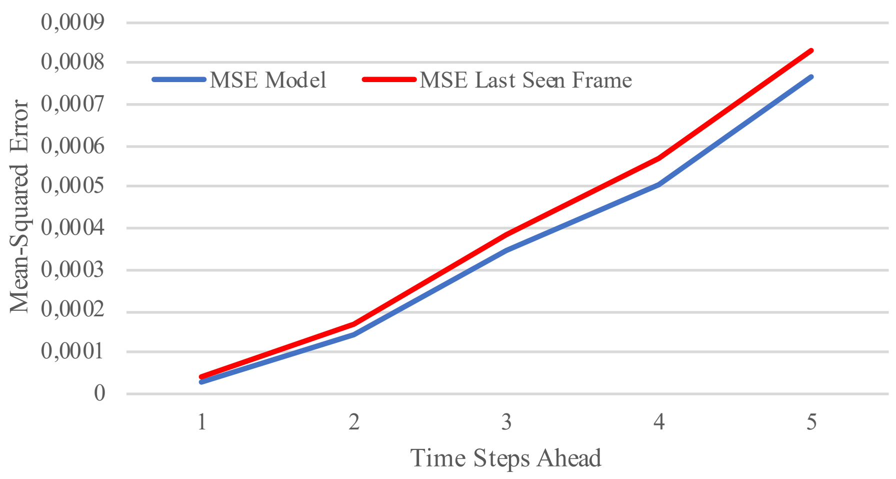

Figure 13: MSE of fine-tuned `t2_2Max` $t+5$ model predictions as a function of number of time steps ahead predicted.

5.2 Multistep Prediction

Although the models were trained to predict only one frame into the future, it is also possible to predict several frames into the future, by recursively feeding back the predictions as inputs. For the training the already trained weights from `t2_2Max` were used. Then the model was trained and fine-tuned with a loss over 15 time steps. For the first ten time steps the actual frames serve as input and for the last five time steps the model's predictions. The training loss was as usual calculated on the activations of the error module E_l for the first ten time steps. For the last five times steps the loss was calculated directly as the mean absolute error between the prediction and the ground truth frames. By fine-tuning the model for extrapolations, the results, although less significant, still exceed the baseline as can be seen in Table 4 and Figure 13. Figure 15 shows that although the results contain slightly blurriness, the model still captures the key structures in its extrapolations.

Model MSE	Model MSE t+5	Last frame MSE	Last frame MSE t+5	Model PSNR	Model PSNR t+5	Last frame PSNR	Last frame PSNR t+5
$\mathbf{2.9 \times 10^{-5}}$	$\mathbf{76.8 \times 10^{-5}}$	6.4×10^{-5}	$\mathbf{83.3 \times 10^{-5}}$	**45.34**	**31.14**	41.91	30.79

Table 4: Results of `t2_2Max` fine-tuned for $t+5$ extrapolations.

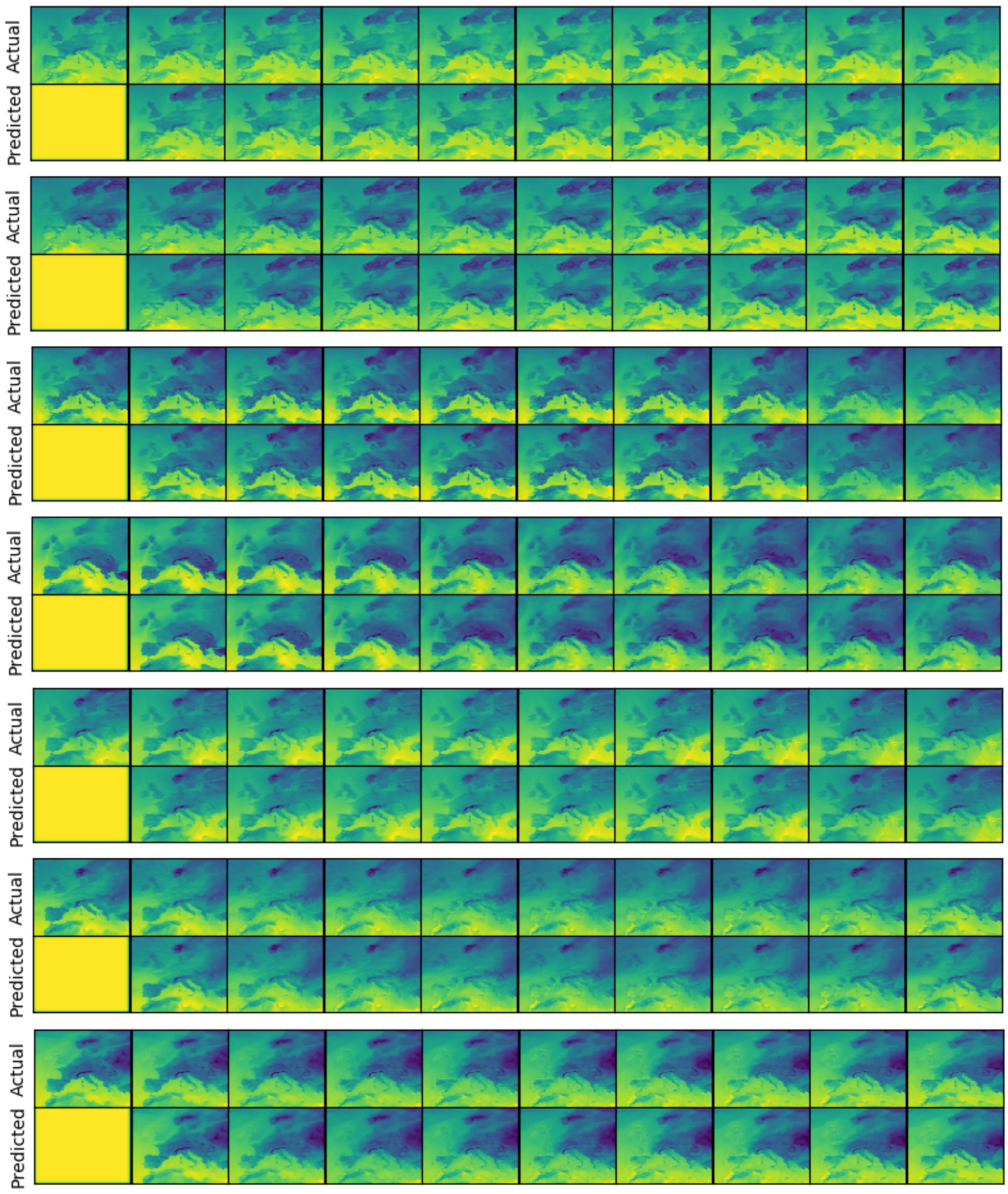

Figure 14: Temperature predictions of the `t2_2Max` network for a cutout above Europe. The first rows display the ground truth (actual) and the second rows display the predictions.

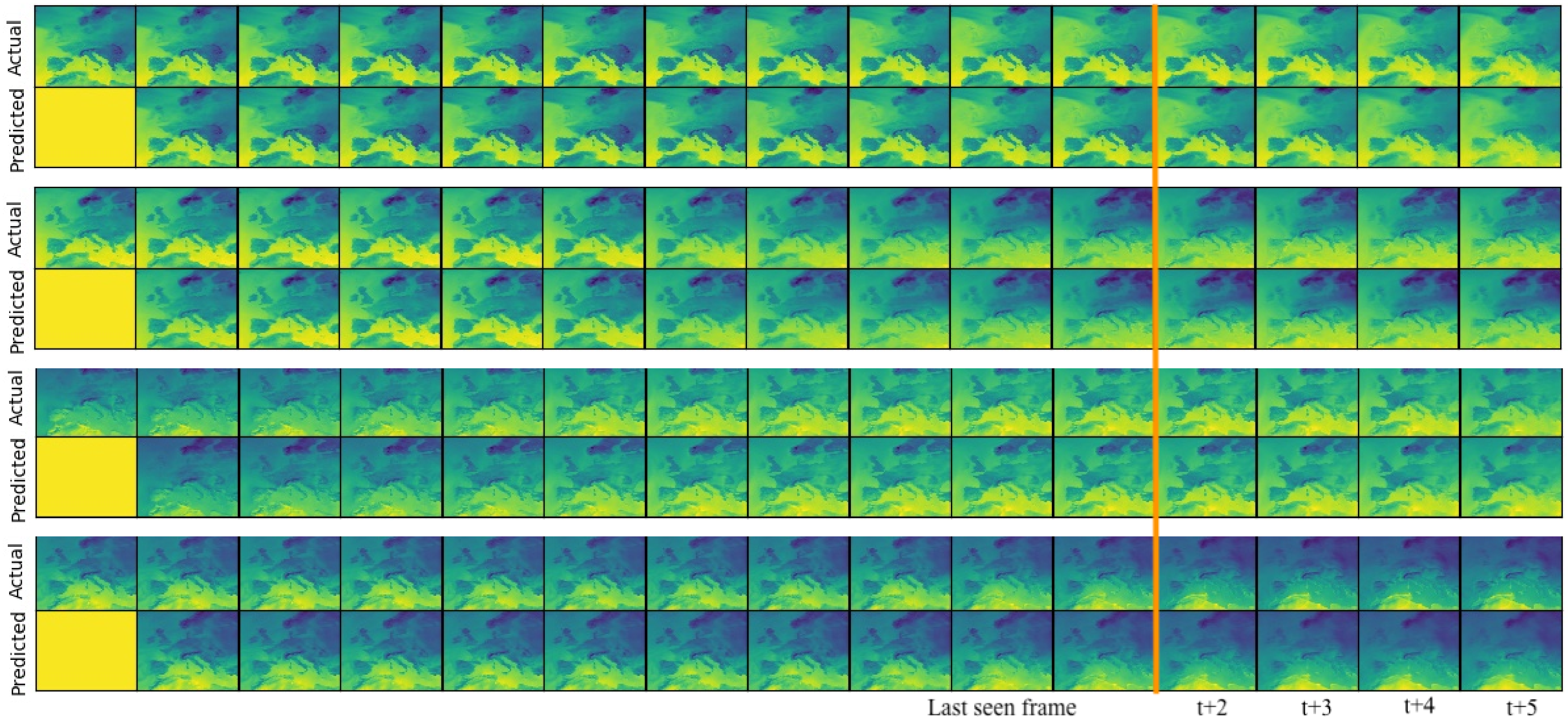

Figure 15: Extrapolation sequences by feeding `t2_2Max` predictions back into the model. Left of the orange line are the normal t+1 predictions. Right of the orange line are the predictions generated by recursively using the predictions as input. The first rows display the ground truth (actual) and the second rows display the predictions.

6 Discussion

This section interprets the results of the experiments, critically examines the chosen approach and points out possible improvements. Finally, by reviewing current literature the integration of physical modeling within machine learning is discussed including its challenges and opportunities.

As the results show, the selected model can make better hourly and even several-hour predictions than the persistence assumption. This finding supports the assumption that the combination of sufficiently available data, current machine learning methods and powerful computers can be an alternative to classical NWP models. An essential finding of this study is that the choice of variables has a big influence on the results. Contrary to the assumption that a greater variety of data provides better results, the experiments show that fewer variables provide better results. The limitation to air temperature as input variables to predict the air temperature itself shows the best results. Interestingly, even the number of the variable air temperature for the input channels has an impact. If all three channels are equipped with air temperature, the performance is worse than with only one channel (and the remaining two channels containing uniform information). But it works best with two channels of air temperature (and one with uniformly distributed information). After that, the combination of two times air temperature and one time surface pressure works best. Only after that comes the combination of all three variables. Compared to the good results for the hourly forecast, the results of the several-hour forecast are more moderate. The performance of the model still exceeds the baseline, captures the key structures in its extrapolations with no parts of the predicted frames disappearing unexpectedly or frames washing out. But the resulting predictions show more blurriness than the hourly predictions and the MSE of the model and the baseline are close together.

In the critical examination of the chosen approach there are some points that could be optimized. In general, more data could be used as this study utilized only two years. All other models presented that deal with weather forecasts have used several years of data for training. Other applications have worked with even more extensive amounts of data. In a next step, typical daily and seasonal temperature variances could be included when training the model. The model could, for example, be trained on hourly time steps of 12 or 24 in order to predict the next frame. If all other parameters are stable, it could be investigated which cycles are best suited. With multistep prediction, considerably more tests could be performed. It would be interesting to see if the inclusion of the surface pressure and 500 hPa geopotential have a positive influence on the longer predictions. Furthermore, there are optimization possibilities for the model itself. The model was trained with the original hyperparameters (number of layers, kernel size, number of channels per layer) and not re-optimized for the weather data. A random hyperparameter search could be performed based on the best performance on the validation set.

However, it is not trivial to measure the performance of a model. This concerns not only the measurement of the results but also the selection of a suitable loss function. There are two important underlying questions: (i) What do we actually want to predict and (ii) what is a good prediction?

The first question is concerned with whether we want to predict the actual future or a possible future and subsequently deals with what loss function to use. Broadly said, two classes of models can be distinguished, one using regular deterministic loss functions such as L1, and the other one using adversarial loss. Models utilizing regular loss functions aim at predicting the actual future. Sometimes this comes with the disadvantage of blurred predictions in order to respond to the many possible futures (Mathieu et al., 2015). Models which are using adversarial loss consistently produce sharper and realistic predictions for the human eye but they might not represent the actual future. Especially, when predicting longer into the future, a setting incorporating adversarial loss produces realistic images. But these long term predictions might have nothing to do with the reality and are not even compared with the ground truth. When for instance making predictions of long term human movement into the future, the results are evaluated by other humans or object detection networks (Wichers et al., 2018). In the context of weather forecasting it is most important to make predictions that are as close as possible to what actually happens, hence regular loss functions are advantageous. But it might still be interesting to examine the application of an adversarial loss or the combination of an adversarial loss with a regular loss. Another promising approach for temperature forecasting is to include physical penalty terms in the calculation of the loss, such as divergence, magnitude and smoothness (de Bezenac et al., 2017). For precipitation nowcasting other authors introduced a balanced loss to deal with the imbalanced challenge of less occuring heavier rainfall which has a high real world impact (Shi et al., 2017).

The second question generally refers to what constitutes a good prediction at all. In general, it is not trivial to evaluate predictions. There are many different scores. The most commonly used scores are MSE and PSNR, which ultimately indicate the pixel wise difference in different ways. But what makes a good prediction? It would be optimal of course, if the prediction was completely equal to the ground truth. But what about a prediction whose objects and contours are correct but slightly shifted? Is that generally bad or is it at least a good start? In weather forecasts, this problem becomes much more concrete. For example, what if the network predicts a rain shower that is 100km off? Even in meteorology there is no consensus on how best to address this problem. Papers dealing with precipitation nowcasting utilize application-specific weather forecasting metrics (Shi et al., 2017; Hogan et al., 2010). The authors evaluate the predictions by means of skill scores with multiple thresholds corresponding to different rainfall levels. Within the field of future frame prediction and especially in the subcategory of weather forecasting, there are no uniform benchmarks such as in object tracking (Wu, Lim, & Yang, 2013) and object segmentation (Perazzi et al., 2016), yet. Therefore, the authors have established their own benchmark against which future papers for precipitation nowcasting can measure themselves. Such a benchmark would also be very useful for temperature forecasting. So far only one other paper has dealt with (sea surface) temperature forecasting (de Bezenac et al., 2017), but for future papers an existing benchmark could be a good guideline. Similar to precipitation forecasting, temperature foreceasting also has unique properties as weather maps are a different type of data that also require an adapted evaulation process. Furthermore, de Bezenac et al. (2017) compare their temperature predictions with state of the art NWP models (Béréziat & Herlin, 2014; Vallis, 2017; Tr'emolet, 2006). For the underlying study and future ones in the field of future frame prediction of air temperature, the results should also be compared to state of the art NWP models.

One of the main challenges in applying deep learning approaches to the complex Earth system science problems is the inherent uncertainty of world dynamics. To some degree all paper dealing with future frame prediction in the weather context include or try to emulate physical rules or phenomena.

Bihlo (2019) applies a future frame prediction model that learns a prior model of uncertainty to the context of precipitation nowcasting (Denton & Fergus, 2018). Denton and Fergus (2018) combine a learned prior with a deterministic estimate of the future frame. They face the challenge of highly uncertain situations like a ball bouncing on the ground that then follows a random trajectory due to a possibly uneven ground. The learned prior is used as a predictive model for uncertainty. Until the ball hits the ground, a deterministic model is sufficient and the learned prior will only predict a low uncertainty but afterwards the prior will predict a high variance event. The model is trained to estimate a latent space prior distribution for each time step based upon previous frames. When applied to precipitation nowcasting the model is expected to learn meaningful stochastic information of the future frame that would not be included in a deterministic prediction model. These stochastic information would correspond to basic physical rules to which precipitation cells develop and move in space and time.

de Bezenac et al. (2017) force a more direct approach to incorporate physical background knowledge in order to forecast the sea surface temperature. The authors point out that changes in temperature are caused by fluid transport, i.e. the advection and diffusion of fluids. During advection, a conserved quantity is transported by a fluid and during a diffusion it spreads out. Their model consists of two parts, first a prediction component which predicts a motion field. The second component is a warping scheme that distorts the last input according to the motion field, in order to produce the future frame. The motion field corresponds to a velocity field advecting the temperatures. Their model seems to conserve temperatures better, as well as capturing the dynamics without predictions blurring out. They also compare their model to the ConvLSTM (Shi et al., 2017) which is used as a part for the applied `PredNet` in this study (Lotter, Kreiman, & Cox, 2015) (see Subsection 4.2). The results of the ConvLSTM seem to not preserve the temperature as the predictions are more blurred. The authors point out that the model partly has functions of an optical flow prediction as their advection equation equals the Brightness Constancy Constraint Equation (BCCE), which is a classical method for calculating optical flow. Additionally, they state that other papers that incorporate optical flow or a similar transformation (Patraucean et al., 2015; Finn et al., 2016) do not allow to introduce prior knowledge. Optical flow also lacks in predicting boundary events (Shi et al., 2015). In precipitation nowcasting this might be a sudden cloudburst and in temperature forecasting it might be a blizzard.

Replicating physical laws and conservation principles by computing their motion seems to be a promising approach in the weather context. Furthermore, with regard to the papers dealing with weather forecasts in the field of future frame prediction, it can be said that theory-driven physics and data-driven application of machine learning could complement each other well. Theory-driven approaches are directly interpretable, whereas data-driven approaches are highly flexible when dealing with a non-linear and multi-scale environment (Reichstein et al., 2019; de Bezenac et al., 2017; Cressie & Wikle, 2015; Gardner & Dorling, 1998).

On the basis of existing data and by means of advancing deep learning models, good predictions can already be made. The data and its preparation play a special role. Like the individual channels of processed RGB images in the field of computer vision, weather maps are also two-dimensional data fields with certain variables. Here, it is air temperature, surface pressure and the 500 hPa geopotential. Several images combined will result in a video. But apart from the similarities there are also fundamental differences. Due to the multimodal problem of the Earth system, much more than three channels could be taken into consideration, for instance variables beyond the visible range or variables with deviating statistical properties. Moreover, the data volume and size is much higher than in typical CV applications. The original netCDF files have a size of 1200×601 pixels (longitude and latitude) while most models typically deal with images of approximately 64×64 to 256×256 pixels. These big datasets require a higher computational demand. Another issue of the multimodality is that the Earth system is often underconstrained, meaning that the unknowns outnumber the equations. A trained model could perform well on the validation and even test set but when applied to more recent data the predictions could deviate strongly (Reichstein et al., 2019).

7 Conclusion

This study was undertaken with the goal to examine the potential of deep learning future frame prediction methods applied to weather forecasting. The main focus of the work was to test which variables or which combination of variables is best suited to predict the air temperature over Europe. The tested variables are the air temperature itself, surface pressure and the 500 hPa geopotential.

The main finding of the experiments is that the selected neural network can make better hourly and even several-hour predictions than the persistence assumption. An important finding of the experiments is that adding more variables does not necessarily lead to better results. The combination of two times air temperature (and one time uniformly distributed information) provides the best results and exceeds the baseline of the persistence assumption by far compared by MSE and PSNR. Furthermore, it can be noted that it is also possible to predict several steps into the future with the chosen approach by recursively using the predictions as input, but the results are more moderate compared to the $t+1$ predictions. The results support the assumption that the combination of sufficiently available data, current machine learning methods and powerful computers can be a supplement or alternative to classical NWP models. A critical look at the chosen approach indicates that by including more data (instead the here used 2 years), the consideration of seasonal and daily variations in training and a random hyperparameter search for the weather data the results could be even improved.

An essential insight for this work and for future work is that it is imperative to include physical background knowledge in deep learning methods applied to weather forecasting. Starting with the data selection, over the data preparation up to the selection of a fitting model with suitable training and evaluation of the weather data. The correct data must be selected and adapted to each other in order to process them together. The range of existing future frame prediction models is wide and when choosing a model one has to pay attention to what exactly should be predicted and how to properly evaluate it. Future work can follow up on promising approaches that incorporate physics by learning a prior model of uncertainty (Bihlo, 2019) or training the model to predict motion fields corresponding to the advection and diffusion of fluids (de Bezenac et al., 2017). Since the research field for future frame predictions for weather forecasts and especially for air temperature is very young, it would be very useful to create a benchmark against which future researchers can measure their results.

References

Babaeizadeh, M., Finn, C., Erhan, D., Campbell, R. H., & Levine, S. (2017). Stochastic variational video prediction. *arXiv preprint arXiv:1710.11252*.

Ballas, N., Yao, L., Pal, C., & Courville, A. (2015). Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*.

Bauer, P., Thorpe, A., & Brunet, G. (2015). The quiet revolution of numerical weather prediction. *Nature*, *525*(7567), 47.

Bellemare, M. G., Naddaf, Y., Veness, J., & Bowling, M. (2013). The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, *47*, 253-279.

Benedict, R. P. (1984). *Fundamentals of temperature, pressure, and flow measurements.* John Wiley & Sons.

Bengio, Y., Simard, P., Frasconi, P., et al. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, *5*(2), 157-166.

Béréziat, D., & Herlin, I. (2014). Coupling dynamic equations and satellite images for modelling ocean surface circulation. In *International Joint Conference on Computer Vision, Imaging and Computer Graphics* (p. 191-205).

Bihlo, A. (2019). Precipitation nowcasting using a stochastic variational frame predictor with learned prior distribution. *arXiv preprint arXiv:1905.05037*.

Bridson, R. (2008). Fluid Simulation for Computer Graphics. Ak Peters Series. *Taylor & Francis*, *3*, 10.

Brox, T., Bruhn, A., Papenberg, N., & Weickert, J. (2004). High accuracy optical flow estimation based on a theory for warping. In *European conference on computer vision* (p. 25-36).

Chalasani, R., & Principe, J. C. (2013). Deep predictive coding networks. *arXiv preprint arXiv:1301.3541*.

Chollet, F., et al. (2015). Keras.

Cressie, N., & Wikle, C. (2015). *Statistics for Spatio-Temporal Data.* Wiley.

de Bezenac, E., Pajot, A., & Gallinari, P. (2017). Deep learning for physical processes: Incorporating prior scientific knowledge. *arXiv preprint arXiv:1711.07970*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (p. 248-255).

Denton, E., & Fergus, R. (2018). Stochastic video generation with a learned prior. *arXiv preprint arXiv:1802.07687*.

Ebert, F., Finn, C., Lee, A. X., & Levine, S. (2017). Self-supervised visual planning with temporal skip connections. *arXiv preprint arXiv:1710.05268*.

Fahrmeir, L., Heumann, C., Künstler, R., Pigeot, I., & Tutz, G. (2016). *Statistik: Der weg zur datenanalyse.* Springer-Verlag.

Finn, C., Goodfellow, I., & Levine, S. (2016). Unsupervised Learning for Physical Interaction through Video Prediction. *Advances in neural information processing systems*, 64-72.

Gardner, M., & Dorling, S. (1998, August). Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric Environment*, *32*(14-15), 2627-2636. doi: 10.1016/S1352-2310(97)00447-0

Geiger, A., Lenz, P., Stiller, C., & Urtasun, R. (2013). Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, *32*(11), 1231-1237.

Géron, A. (2017). *Hands-on machine learning with Scikit-Learn and TensorFlow: Concepts, tools, and techniques to build intelligent systems.* " O'Reilly Media, Inc.".

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning.* MIT press.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y. (2014). Generative Adversarial Nets. *Advances in neural information processing systems*, 2672-2680.

Guillory, A. (2017, November). *ERA5* [Text]. https://www.ecmwf.int/en/forecasts/datasets/reanalysis-datasets/era5.

Hochreiter, S. (1997). Long Short-Term Memory. *Neural computation*, 1735-1780.

Hogan, R. J., Ferro, C. A., Jolliffe, I. T., & Stephenson, D. B. (2010). Equitability revisited: Why the "equitable threat score" is not equitable. *Weather and Forecasting*, *25*(2), 710-726.

Hore, A., & Ziou, D. (2010). Image quality metrics: PSNR vs. SSIM. In *2010 20th International Conference on Pattern Recognition* (p. 2366-2369).

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., . . . Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Ionescu, C., Papava, D., Olaru, V., & Sminchisescu, C. (2013). Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, *36*(7), 1325-1339.

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., & Fei-Fei, L. (2014). Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (p. 1725-1732).

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Klein, B., Wolf, L., & Afek, Y. (2015, June). A Dynamic Convolutional Layer for short rangeweather prediction. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (p. 4840-4848). Boston, MA, USA: IEEE. doi: 10.1109/CVPR.2015.7299117

Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., & Serre, T. (2011). HMDB: A large video database for human motion recognition. In *2011 International Conference on Computer Vision* (p. 2556-2563).

Laptev, I., Caputo, B., et al. (2004). Recognizing human actions: A local SVM approach. In *Null* (p. 32-36).

LeCun, Y., et al. (1989). Generalization and network design strategies. In *Connectionism in perspective* (Vol. 19). Citeseer.

Lee, A. X., Zhang, R., Ebert, F., Abbeel, P., Finn, C., & Levine, S. (2018). Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., … Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision* (p. 740-755).

Lotter, W., Kreiman, G., & Cox, D. (2015, November). Unsupervised Learning of Visual Structure using Predictive Generative Networks. *arXiv:1511.06380 [cs, q-bio]*.

Lotter, W., Kreiman, G., & Cox, D. (2016, May). Deep Predictive Coding Networks for Video Prediction and Unsupervised Learning. *arXiv:1605.08104 [cs, q-bio]*.

Madec, G., et al. (2015). NEMO ocean engine.

Mathieu, M., Couprie, C., & LeCun, Y. (2015, November). Deep multi-scale video prediction beyond mean square error. *arXiv:1511.05440 [cs, stat]*.

Mohanakumar, K. (2008). *Stratosphere troposphere interactions: An introduction.* Springer Science & Business Media.

Oh, J., Guo, X., Lee, H., Lewis, R. L., & Singh, S. (2015). Action-Conditional Video Prediction using Deep Networks in Atari Games. *Advances in neural information processing systems*, 2863-2871.

Olah, C. (2014). *Understanding Convolutions - colah's blog.* https://colah.github.io/posts/2014-07-Understanding-Convolutions/.

Olah, C. (2015). *Understanding LSTM Networks.* http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

Patraucean, V., Handa, A., & Cipolla, R. (2015). Spatio-temporal video autoencoder with differentiable memory. *arXiv preprint arXiv:1511.06309*.

Perazzi, F., Pont-Tuset, J., McWilliams, B., Van Gool, L., Gross, M., & Sorkine-Hornung, A. (2016). A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (p. 724-732).

Rao, R. P., & Ballard, D. H. (1999). Predictive coding in the visual cortex: A functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, *2*(1), 79.

Reed, S. E., Zhang, Y., Zhang, Y., & Lee, H. (2015). Deep visual analogy-making. In *Advances in neural information processing systems* (p. 1252-1260).

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., et al. (2019). Deep learning and process understanding for data-driven Earth system science. *Nature*, *566*(7743), 195.

Rumelhart, D. E., Hinton, G. E., Williams, R. J., et al. (1988). Learning representations by back-propagating errors. *Cognitive modeling*, *5*(3), 1.

Santner, J., Leistner, C., Saffari, A., Pock, T., & Bischof, H. (2010). PROST: Parallel robust online simple tracking. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (p. 723-730).

Schiele, B., Dollár, P., Wojek, C., & Perona, P. (2009). Pedestrian detection: A benchmark. In *Computer Vision and Pattern Recognition (CVPR).*

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-k., & WOO, W.-c. (2015). Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 28* (p. 802-810). Curran Associates, Inc.

Shi, X., Gao, Z., Lausen, L., Wang, H., Yeung, D.-Y., Wong, W.-k., & Woo, W.-c. (2017). Deep Learning for Precipitation Nowcasting: A Benchmark and A New Model. *Advances in Neural Information Processing Systems*, 5617-5627.

Soomro, K., Zamir, A. R., & Shah, M. (2012). UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*.

Srivastava, N., Mansimov, E., & Salakhutdinov, R. (2015). Unsupervised Learning of Video Representations using LSTMs. *International conference on machine learning*, 843-852.

Tr'emolet, Y. (2006). Accounting for an imperfect model in 4D-Var. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, *132*(621), 2483-2504.

Vallis, G. K. (2017). *Atmospheric and oceanic fluid dynamics.* Cambridge University Press.

Vezzani, R., & Cucchiara, R. (2010). Video surveillance online repository (visor): An integrated framework. *Multimedia Tools and Applications*, *50*(2), 359-380.

Wichers, N., Villegas, R., Erhan, D., & Lee, H. (2018). Hierarchical long-term video prediction without supervision. *arXiv preprint arXiv:1806.04768*.

Winkler, S., & Mohandas, P. (2008). The evolution of video quality measurement: From PSNR to hybrid metrics. *IEEE Transactions on Broadcasting*, *54*(3), 660-668.

Woo, W., & Wong, W. (2014). Application of optical flow techniques to rainfall nowcasting. In *The 27th Conference on Severe Local Storms* (Vol. 5, p. 9-16).

Wu, Y., Lim, J., & Yang, M.-H. (2013). Online object tracking: A benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (p. 2411-2418).

Zhou, Y.-T., & Chellappa, R. (1988). Computation of optical flow using a neural network. In *IEEE International Conference on Neural Networks* (Vol. 1998, p. 71-78).

A Code of Related Work

Author(s)	Code
Srivastava et al. (2015)	https://github.com/mansimov/unsupervised-videos
Shi et al. (2015)	https://github.com/wqxu/ConvLSTM
Oh et al. (2015)	https://github.com/junhyukoh/nips2015-action-conditional-video-prediction
Klein et al. (2015)	https://github.com/FredericGodin/DynamicCNN
Mathieu et al. (2015)	https://github.com/coupriec/VideoPredictionICLR2016
Patraucean et al. (2015)	https://github.com/viorik/ConvLSTM
Finn et al. (2016)	https://github.com/tensorflow/models/tree/master/research/video_prediction
Lotter et al. (2016)	https://github.com/coxlab/prednet
Shi et al. (2017)	https://github.com/sxjscience/HKO-7
de Bezenac et al. (2017)	https://github.com/emited/flow
Wichers et al. (2018)	https://github.com/brain-research/long-term-video-prediction-without-supervision
Lee et al. (2018)	https://alexlee-gk.github.io/video_prediction/
Bihlo (2019)	na
This study	https://github.com/severin1992/airtemprednet

Table 5: Links to the code of the selected future frame prediction models.

B Dataflow

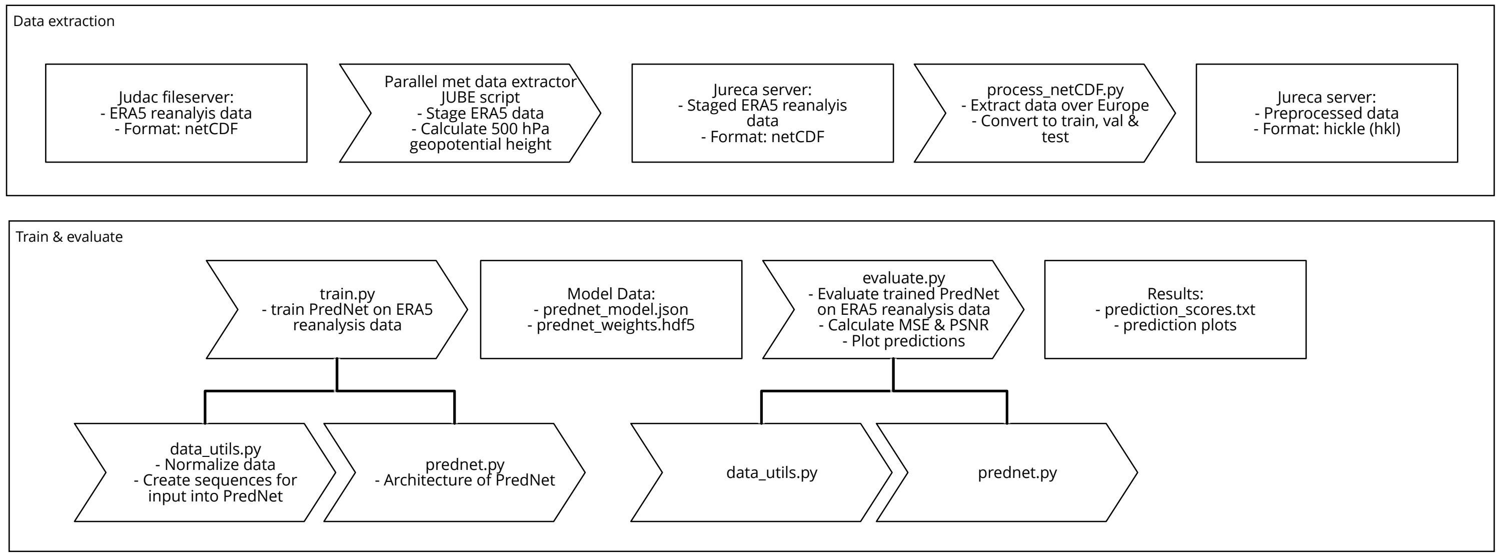

Figure 16: Dataflow of data extraction, training and evaluation of the model

Acknowledgements

I thank Dr. Martin Schultz and all my colleagues from Forschungszentrum Jülich for their confidence in my work, their support, the inspiring discussions and the access to the supercomputers and the data at the Jülich Supercomputing Centre.
I would also like to thank Prof. Lessmann and Johannes Haupt for supporting such exciting research and for the valuable supervision.
Finally, I thank my family and friends – for everything.

Declaration of Authorship

I hereby declare and confirm that this thesis is entirely the result of my own original work. Where other sources of information have been used, they have been indicated as such and properly acknowledged. I further declare that this or similar work has not been submitted for credit elsewhere.

Berlin, July 18, 2019