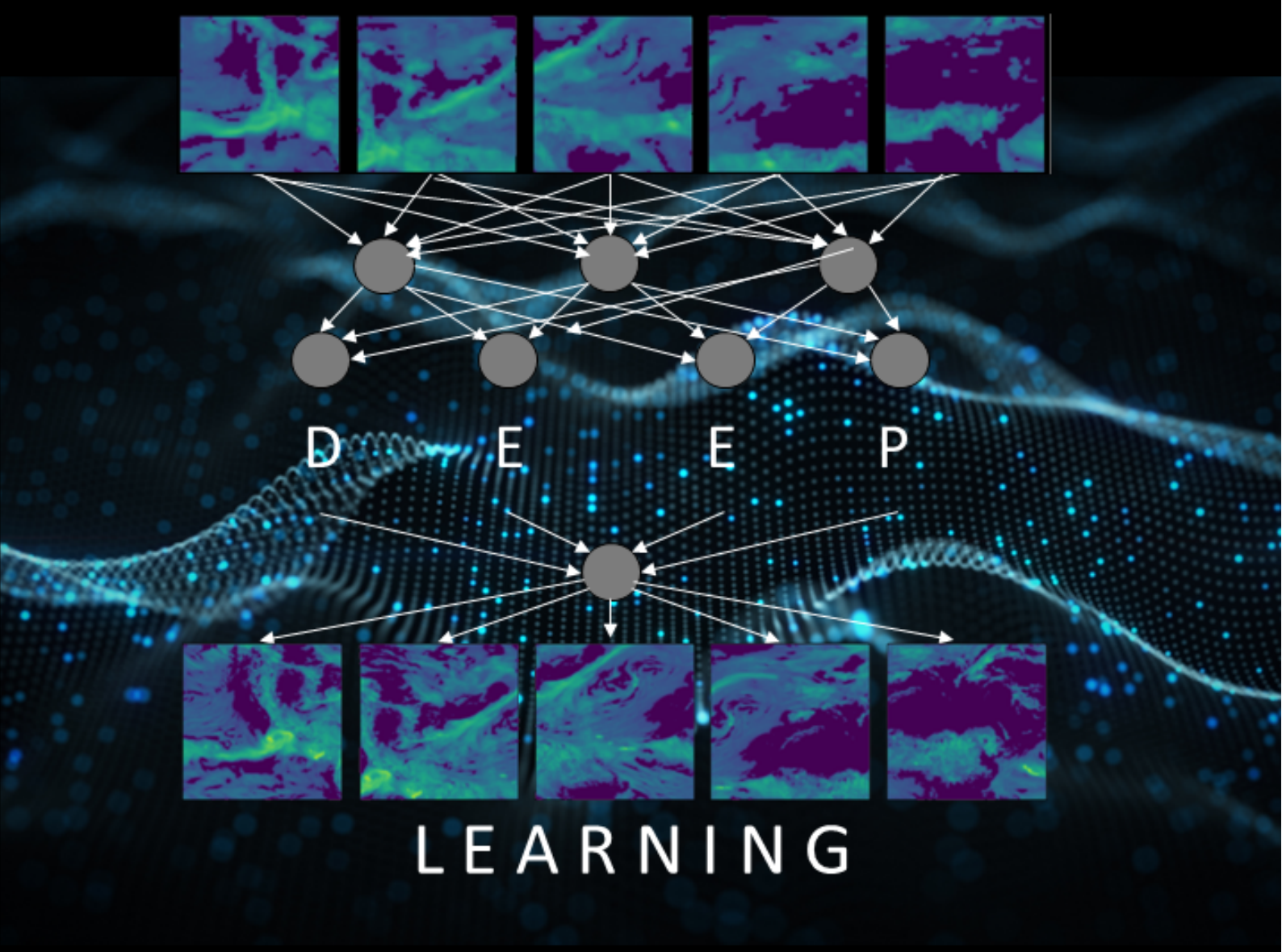
Super-resolution neural networks

A strategy to downscale model output data



Avelon Gerritsma

Super-resolution neural networks:
A strategy to downscale model output data

Avelon Gerritsma

MSc Thesis
Supervisors: Fedor Baart and Albrecht Weerts
Wageningen University

December 2020, Wageningen

Abstract

Running complex geophysical models takes a long computation time. Examples of this are the general circulation models, that are used to study the global climate system. These models run at low horizontal resolution (150-400km), because of their computational demand. Hydrological models require high-resolution meteorological data, and often rely on downscaling techniques to obtain high-resolution data from the low-resolution model output data of the general circulation models. In this research, we test a new statistical downscaling method: downscaling using deep neural networks. This method overcomes many of the problems that the currently used downscaling techniques face, and therefore has great potential in downscaling model output data. We built, trained and tested a deep convolutional auto-encoder to see the potential of this technique to downscale model output data, by downscaling meteorological (reanalysis) data as a case study. As a test, we used the downscaled data as input for a hydrological model of the Rhine. We present a general method outline and focus points for super-resolution neural networks to downscale model output data. Since the downscaling methodology still has some room for improvement, we also suggest improvements for the model structure, optimization and training data.

Keywords
Downscaling, statistical downscaling, deep learning, neural network, auto-encoder, reanalysis data, precipitation, hydrology, model performance

Contents

List of Figures

List of Tables

1 | Introduction

1.1 Problem description

As model complexity increases, the computation time increases. With more and more high-resolution data becoming available, new models arise to make use these sources of high-resolution data. On the other side, when models run at high-resolution, the models behind the model creating input data either have to run on the same high-resolution, or a downscaling step is required. A good example of this are the General Circulation Models (GCMs), that are used to create meteorological input data for hydrological models. GCMs are coupled numerical models that include physical processes of the atmosphere, oceans, cryosphere and land surface (IPCC, 2013). The GCMs can be used to study the response of the global climate system to an increase of greenhouse emissions, for example to study how precipitation patterns change in the future. To study how the climate has changed in the past decades, large reanalysis datasets exist that contain information of multiple meteorological variables. These climate reanalyses datasets are produced by combining past observations with a general circulation model (e.g. Dee et al. (2011); Saha et al. (2010)). The horizontal resolution of GCMs range between 150-400 km, due to their computational demands (Tran Anh et al., 2019). The resolution of reanalysis datasets is higher, but in most cases still not high enough to directly use it as input for hydrological models (Gao, 2013). Therefore, both studies that want to predict the future hydrological regime (climate impact studies) or analyse the past (reanalyses projects), rely on downscaling techniques to obtain sub-grid information.

There are two main downscaling categories, namely dynamical downscaling (DD) and statistical downscaling (SD). Dynamical downscaling methods use regional climate models (RCMs) to produce higher resolution outputs (Maraun et al., 2010; Hwang et al., 2013; Giorgi et al., 2001). Dynamically downscaled reanalysis datasets include non-linear mesoscale features that were absent in the raw large-scale reanalysis datasets. RCMs can be applied to produce reanalysis datasets at 10-50km resolutions (Hwang et al., 2013). The most important drawbacks of this technique is that the model is region specific (cannot be used for other regions) and computationally intensive (Fowler et al., 2007). Statistical downscaling methods establish empirical relationships between large-scale atmospheric variables (predictors) and local or regional variables of interest (predictands) (Baño-Medina et al., 2020; Goodess et al., 2012). Statistical downscaling is computationally efficient and can easily be applied to different regions (Ahmed et al., 2013). The results do however, strongly depend on the choice of predictors (Fowler et al., 2007). Furthermore, statistical downscaled variables can give a poor representation of the observed variance and extremes of the downscaled variable (Wilby et al., 2004; Gao, 2013).

Detailed and reliable precipitation estimates are key for hydrological model performance. Hydrological models are used as a tool for decision making on water management, so higher model performance supports optimal decision making. Several studies have evaluated and/or compared dynamical and statistical downscaling techniques based on the hydrological model performance (Bastola and Misra, 2014; Hay and Clark, 2003; Werner and Cannon, 2016; Hwang et al., 2013; Chen et al., 2018). Bastola and Misra (2014); Wilby et al. (2000) both concluded that the use of a downscaled reanalysis dataset, improved the hydrological model performance compared to the model performance with the raw reanalysis dataset. This highlights the importance of having accurate downscaling techniques to downscale reanalysis (i.a. precipitation) data for hydrological applications.

Deep learning techniques have recently been introduced as a promising new statistical downscaling approach. The deep learning approaches make use of convolutional neural networks (CNN) or generative adversarial networks (GAN). This technique has become quite popular in the computer graphics field, where this technique is referred as "super-resolution" or "upressing". In this field, it is used to generate high-resolution images or videos from low-resolution images and videos (Shi et al., 2016; Dong et al., 2016; Karras et al., 2018; Goodfellow et al., 2014). Next to creating super-resolution images, it can also be used for sharpening patterns in an image. Xie et al. (2018)

used GANs to generate high-resolution images and videos of fluid flow (both smoke and liquid), were the outline of the fluid flows were sharper after the downscaling. For this application, not only spatial but also temporal patterns are resolved by the neural network. The neural network learns from the temporal information, to increase the downscaling performance of sequential data.

Using a neural network to downscale meteorological/climatic data could overcome the limitations of the already excising downscaling techniques. After a neural network is trained to learn the difference between high and low-resolution maps, it takes less computation time for the neural network to downscale large GCM outputs compared to using a RCM. Furthermore, the neural network can be used to downscale the meteorological/climatic variable at all places, instead of a specific region of the RCM. Since the network learns the difference between the high and low-resolution map, the downscaling is independent of predictands (unlike other statistical downscaling techniques). These advantages make this new downscaling technique interesting to use for other applications.

The use of deep learning techniques for downscaling of geophysical data is still rather limited. It is applied in the field of remote sensing, for example to improve land-use classification by generating super-resolution images (Xiong et al., 2020). Furthermore, it has been applied to obtain super-resolution climate and weather forecast images for specific regions (Cheng et al., 2020; Rodrigues et al., 2018). For these specific regions, the downscaling results using a neural network were promising. Using a neural network to downscale climatic/meteorological datasets of the whole world has not been done yet. One of the main limitations of neural networks is that it requires a lot of training data. Reanalysis datasets have a large history, and both coarse and fine resolution reanalysis datasets are currently available. This offers the great opportunity to test the potential of neural networks as downscaling technique for climatic/meteorological data, by trying it out on these reanalysis datasets. Hydrological models can profit from these downscaled datasets. For example, we could generate high-resolution data for periods were only coarse reanalysis data are available. In this way, past hydrological events can be studied at higher resolution using the generated high-resolution inputs.

1.2 Aim

The aim of this research is to set up and train a (deep) neural network to generate high-resolution data from low-resolution model output data. This study focusses on statistical downscaling, dynamical downscaling methods fall outside the scope of this research. To test the potential of neural networks for hydrological applications, we downscaled reanalysis data with a neural network as a case study. Our study focusses on downscaling precipitation data, but the technique could also be used to downscale other meteorological variables. Lastly, downscaled precipitation data are used as input for a high-resolution distributed hydrological model to see if model performance improves compared to model performance using the raw precipitation data (before downscaling) as input.

1.3 Research questions

The main research question is:
What is the potential of neural networks to downscale model output data?

This question will be answered by conducting a case study, of which the goal is to set up a neural network to downscale reanalysis data. The following questions will be answered:

1. How do the downscaled results using a neural network differ from/resemble the downscaled results using interpolation?
2. To what extent does the performance of hydrological models improve using the downscaled data as input compared to using the raw reanalysis data?
3. What are the changes in method needed to apply, when using a super-resolution neural network for downscaling (reanalysis) data instead of pictures?

1.4 Structure report

The structure of this thesis report is as follows. The data used to train and test the neural network is described in chapter 2. Furthermore, the study area (catchment area hydrological model) and input variables of the hydrological model are discussed. In chapter 3, we elaborate the method of data preprocessing, the neural network used (design, tech-

nical details and training) and the method of testing the downscaling performance. In chapter 4 we discuss the results, both the output of the neural network and of the hydrological model. In chapter 5, we discuss the results, compare our research with related work and give recommendations on improving the downscaling methodology. Chapter 6 concludes the findings of this research, were the potential of neural networks to downscale model output data will be discussed.

2 | Data and study area

2.1 Reanalysis data

To answer the research question: What are the (dis)advantages of using a neural network to downscale data for hydrological applications, we need to downscale a relevant forcing dataset. Qualities that needed for this research are:

- Dataset that is currently used for hydrological applications
- Long time span, since a lot of training data are needed for the neural network
- Pre-computed high and low-resolution pairs
- Publicly assessable

Based on these qualities, we decided to use the ERA5 and ERA-20C reanalysis datasets, produced by the European Centre for Medium-Range Weather Forecasts (ECMWF). The ERA5 reanalysis dataset contains high-resolution data and the ERA-20C coarse-resolution data, both datasets are currently used for hydrological applications. The datasets have an overlap in their time span, namely between 1979-2010. These thirty years of data are used for this research. The next two paragraphs contain background information of ERA5 and ERA-20C, in table 2.1, you can find the details of both datasets.

The ERA-20C was the first century-long reanalysis dataset produced by ECMWF. For creating ERA-20C, only surface pressure and marine wind observations were used. The timestep is one day, for each day observations are combined with a background (prior estimate), obtained from the model forecast of the previous day. (Poli et al., 2016)

ERA5 is the fifth generation of reanalysis datasets, produced by the ECMWF. The improvements of ERA5 are the increased resolution, new developments in the model and data assimilation, hourly resolution and more output parameters. Leading to an increase of the global-mean correlation with monthly mean precipitation (GPCP) data from 67% to 77%. In 2020, ERA5 for 1950-present is expected to become available. (Hersbach et al., 2020)

The ERA5 and ERA-20C both have global spatial coverage. We trained and tested the neural network with precipitation data of the whole earth. ERA5 and ERA-20C both contain several atmospheric variables: wind, pressure, temperature, dewpoint temperature and precipitation. Pressure and temperature can be downscaled using standard linear methods (GLM) or interpolation with DEM (Digital elevation model), though for precipitation these methods do not give a good result (Baño-Medina et al., 2020; Tran Anh et al., 2019). Therefore, in this research precipitation fields will be downscaled using neural network.

2.2 Study area

The neural network is trained with precipitation data sampled across the whole globe, to see if a trained neural network can be used to downscale precipitation irrespective of geographical locations. To see the potential of this downscaling method for hydrological applications, precipitation maps of Europe are downscaled and used as input for a distributed hydrological model of the Rhine catchment (Imhoff et al., 2020). A map of the Rhine catchment can be found in figure 2.1. The catchment area of the Rhine equals approximately 160.000 km^2. To evaluate the hydrological model performance, the discharge of the Rhine and its tributaries is studied at several locations: Basel, Andelfingen, Cochem, Plochingen, Frankfurt, Menden, Grolsheim and Lobith. These locations are indicated on the map (next to some other cities), see figure 2.1.

The hydrological model used to model the Rhine discharge is wflow_sbm. Wflow is an open source distributed hydrological model platform developed by Deltares. wflow_sbm is a high-resolution distributed hydrological model, and was parametrized for the Rhine basin by Imhoff et al. (2020). Improvements for the river network, slope and river slope are derived using Eilander et al. (2020). Distributed models have as an advantage over lumped models that they can use high-resolution gridded datasets as their inputs. The model requires three meteorological input variables: temperature, potential evapotranspiration and precipitation. We used ERA5 temperature data, multiple input datasets for precipitation (ERA20C, ERA5 and downscaled data, see section 3.7). The

Table 2.1: Technical details ERA5 and ERA-20C

Properties	ERA-20C	ERA5
Model	Coupled Atmosphere/ Land-surface/Ocean-waves model	Integrated Forecast System
Spatial resolution	125x125 km	30x30 km
Time resolution	3-hourly	Hourly
Period	1900 - 2010	1979 - present

Figure 2.1: Catchment area of the river Rhine (Belz, 2010)

potential evapotranspiration is derived from ERA5 reanalysis radiation fluxes, 2m temperature and surface pressure, using the method of De Bruin et al. (2016).

3 | Methods

3.1 Pre-processing data for neural network

To train and test the neural network, low- and high-resolution images are used. In the previous chapter, we selected the ERA datasets that provide both high and low-resolution precipitation fields. In this section, we describe how these datasets are transformed to images that are suitable for the neural network models. The image pre-processing consists of several steps, an overview of all pre-processing steps can be seen in figure 3.1. For one of the pre-processing steps, we made use of Google Earth Engine (GEE). GEE offers functions needed for pre-processing the data, e.g. creating image tiles, regridding and exporting images as TFRecord files. Furthermore, GEE made a lot of public geospatial datasets available to view/access online, among which ERA5. ERA-20C was not available on GEE. We downloaded the ERA-20C data from the ECWMF site using a web API. ECWMF daily total precipitation data are the accumulated precipitation between 6AM and 6AM. The grid of the downloaded ERA-20C netcdf dataset is $1°/1°$ (111km). This resolution differs from the resolution of the original ERA-20C dataset (125km), the data are interpolated to the grid using MIR (Meteorological Interpolation and Regridding) (Malardel et al., 2016). After a calculation step to determine the total daily precipitation (between 12PM and 12PM), the NetCDF file is converted to GeoTIFF format and then uploaded to a storage bucket (google cloud platform), from where it is uploaded to Google Earth Engine.

In Google Earth Engine, image tiles are cut out of the world precipitation map. The smaller image patches used as input for the neural network are cut out of these bigger image tiles at a later stage. The image tiles overlap, to prevent the neural network from coupling precipitation to a certain location instead of coupling it to the low-resolution image precipitation. The padding dimension is added to the patch dimension to have image tiles that overlap with their neighbours with the padding dimension, see figure 3.2. For this research, the padding dimension is set to be half of the patch dimension, therefore the tile size is two times the number of pixels of the image patches. Ten tiles in total are needed to cover the whole world map,

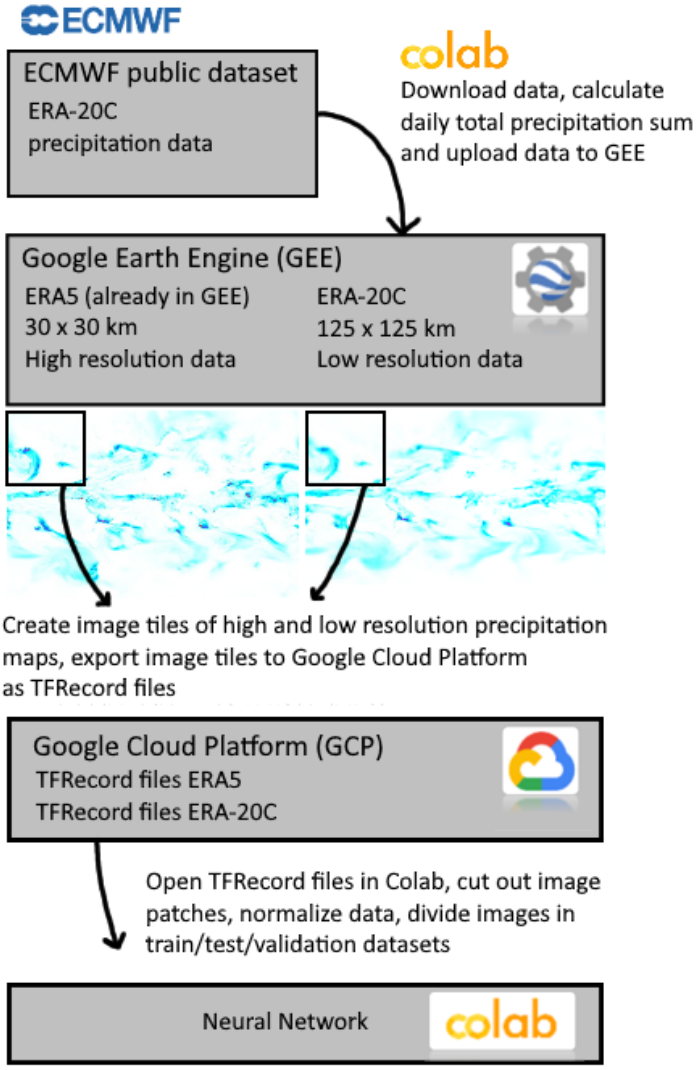


Figure 3.1: Steps of image pre-processing

see figure 3.2. The dimension of image tiles is the patch dimension plus two times the padding dimension. For the ERA-20C this equals 128x128 pixels and for the ERA5 512x512 pixels in total for one image tile. The coverage of the high-resolution image tile should be equal to the coverage of the low-resolution image tile, so the number of pixels times the resolution should give the same tile size. The resolution of the ERA5 is set at 30 km and the resolution of the ERA-20C dataset at 120 km (setting of export function in GEE). The image tiles were saved as TFRecord files, a format used for large datasets for neural networks, and stored in a storage bucket on Google Cloud Platform, see 3.1.

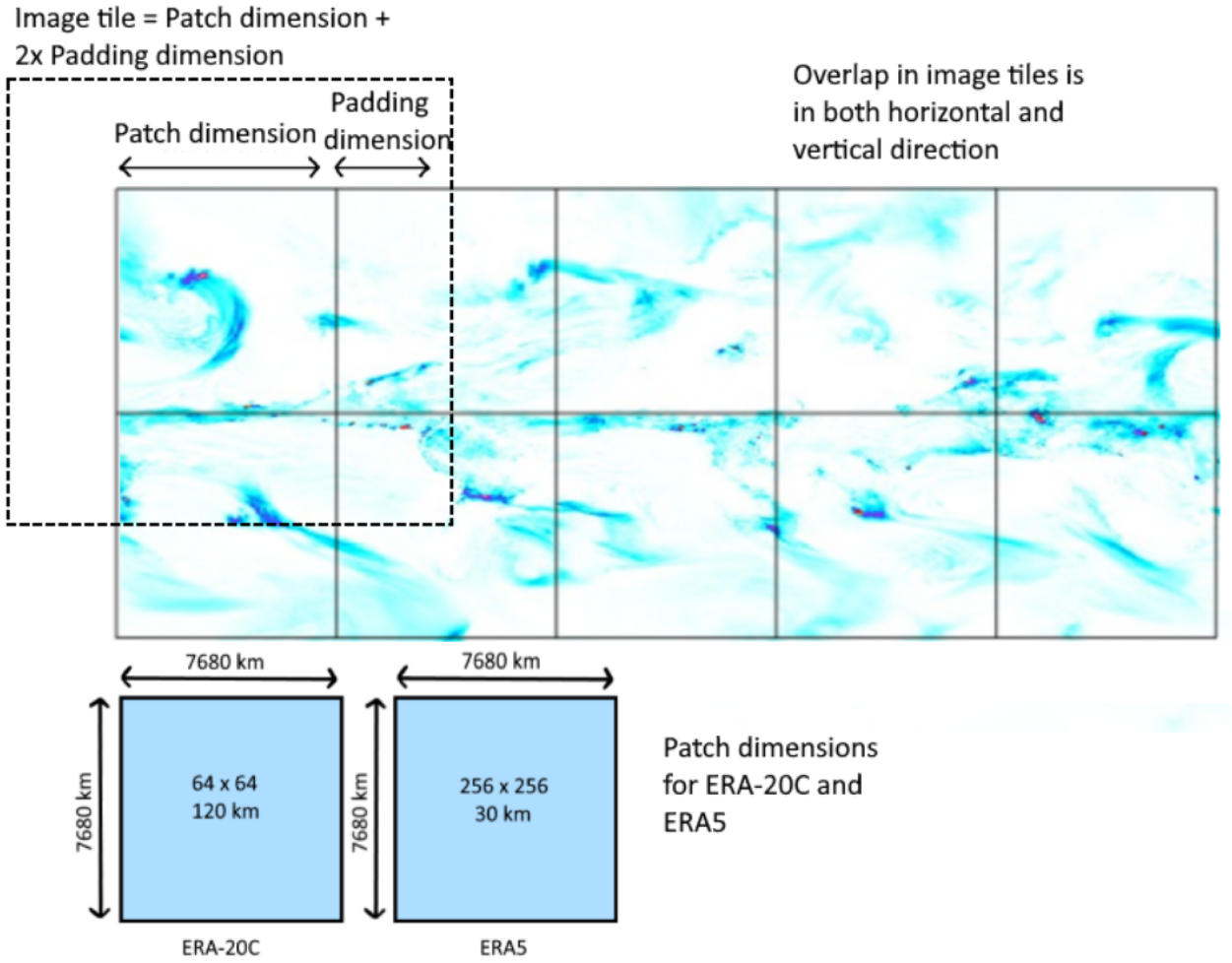


Figure 3.2: Illustration of image tile creation in GEE

The TFRecord files, containing the precipitation tiles, are split into train, test and validation data. To make sure the downscaling performance is independent of the temporal changes of the data quality and characteristics, every first two years of each decade are set apart for validation. We divided the rest of the images in train (80%) and test data (20%), as described in (Bowden et al., 2002; Maier and Dandy, 2000). The training images are used to train the neural network, the test images are used as a measure how well the model learned from the training images. The validation images are downscaled with the trained/tested network, and used as input for the hydrological model.

The TFRecord files are processed in Colab and Jupyterlab. First, calculations were conducted on Colab. For better connection to our storage bucket and faster computations Jupyterlab was used to do most of the calculations. For the training/testing dataset, we cut out 17 patches (64x64 and 256x256) of the 10 image tiles (128x128 and 512x512). The 17 patches cover almost the whole world map and have overlap with each other. For the validation years, we downscale one of the 17 patches that contains Europe. Next to these patches, we cut out an extra patch with Europe in the middle of the patch, so this patch location differs from the training patches. Therefore, we have two datasets with patches containing Europe for the validation years.

Neural networks work best for data ranging between 0 and 1. Therefore, the precipitation patches are normalised. Data normalisation helps accelerating the calculations of algorithms inside the neural network (Thara et al., 2019). A few training sessions showed that rescaling the precipitation with a MinMaxScaler(feature range = [0,1]) was not enough for the model to learn from the training data. Precipitation data has a lot of zeros and values close to zero, and only some larger values (extremes, e.g. thunderstorms in the tropics, monsoon rainfall). Therefore, the

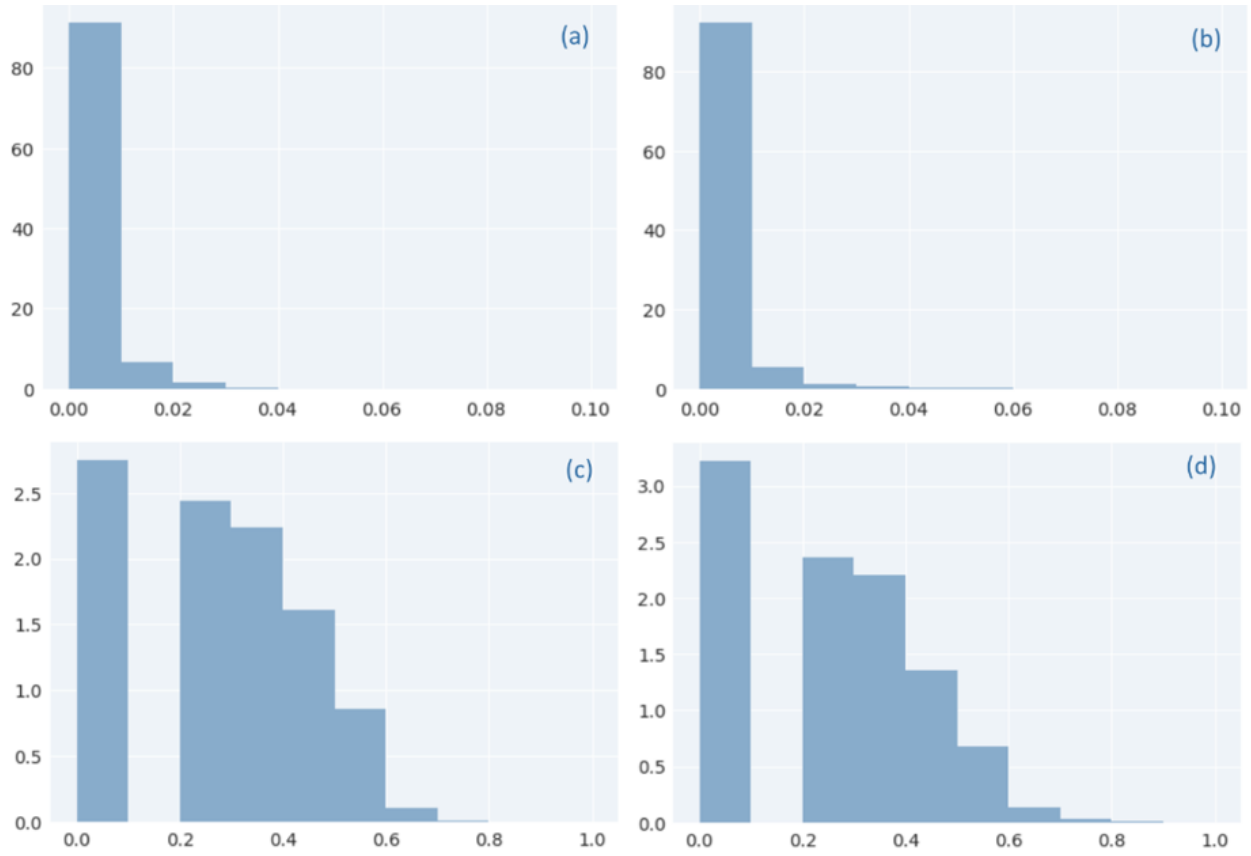


Figure 3.3: Probability density distribution of the precipitation data (using 50 random images of the dataset), before and after rescaling and normalisation. a) Before, ERA-20C b) Before, ERA5 c) After, ERA-20C d) After, ERA5

data rescaling consist of another step, namely by elevating the data to the power of 0.2. This shifted the data, from all data close to zero (a,b in figure 3.3), to a more normal distribution, but still skewed to the right (c,d in figure 3.3). The normalised and rescaled images that are used as input for the neural network can be seen in figure 3.7. In the next section, the neural network is elaborated.

3.2 Neural network: Deep convolutional auto-encoder

For this study, we use a deep convolutional auto-encoder to downscale the precipitation images. To explain the structure of this neural network, first auto-encoders and its layers will be described, thereafter the technical details of the neural network are specified.

Auto-encoders are feed forward neural networks (information is going one-way), trained to reconstruct its input data to its output (Rodrigues et al., 2018). Low-resolution images (precipitation maps) are fed into the auto-encoder, high-resolution images are created from these images and compared with the 'true' high-resolution images, see figure 3.4.

The auto-encoder consists of two parts: an encoder and a decoder. The encoder maps the input data on a latent (hidden) feature, this latent feature represents the most important features of the input data (Hinton and Salakhutdinov, 2006; Yeung, 2017). The decoder is trained to recreate the input using the latent feature (Goodfellow et al., 2014). Three types of layers are included in the auto-encoder: convolutional layers, pooling layers and upsampling layers. These layers will be explained below.

Convolutional layers are used in neural networks to extract features of image. Multiple convolutional layers can be implemented in a neural network (the auto-encoder) to extract image features at multiple

levels (Chen et al., 2016). In this way, the model learns the underlying spatial structure of precipitation maps. Local features are extracted in the convolutional layer, by applying a filter of weights on the input data. The filter 'shifts' over the input images, and computes the sum of the weighted inputs for each patch, see figure 3.5. A bias is added to the weighted sum, and the final value is activated with an activation function σ (to make it non-linear). Al outputs are mapped on an activation map. (Zhang et al., 2020)

$$\sigma(W^T X + b) \tag{1}$$

With W being the weight matrix, T the number of filters, X the matrix values of the patch input and b the bias.

In the pooling layers, the size of the activation map is reduced, by taking either the average or maximum value of a block of values of the activation layer, see figure 3.6. The pooling layers reduce the dimension (size of the image), see the Bottleneck in figure 3.4.

To recreate an image with the same resolution as the high-resolution image, the image needs to be upsampled. This is done with upsampling layers in the decoder. The upsampling layers do the exact opposite of the pooling layers, i.e. increase the size of the output map. The data are upsampled using an interpolation method (nearest or bilinear), see section 3.3. The final upsampled image is compared with the 'true' high-resolution image. The auto-encoder eventually learns, by processing a lot of images, what the most important features are of the low-resolution images and maps these features at a higher resolution.

3.3 Details model structure

We used a modified version of the Deep Denoiseing Super Resolution convolutional network (DDSRCNN), proposed by Mao et al. (2016). An overview of the whole model can be seen in appendix A. The neural network is build in Jupyterlab, using Tensorflow (software for machine learning). The first layer is the input layer, where the low-resolution images (64x64 pixels) are fed into the model. The encoder exist of two blocks. One block consists of two convolutional layers and a pooling layer. The images are grouped in batches, therefore the input (?,64,64,1) contains a question mark, this question mark stands for the batch size. The last number stands for the number of channels, in this case 1 (e.g. RGB would be three channels). A filter of 3x3 is applied in the convolutional layers. When the original image is nxn and the filter fxf, this results in an image of $(n - f + 1) * (n - f + 1)$ pixels. So in this case, the output of the first convolutional layer would be 62x62 pixels $(64 - 3 + 1)$. To have the same output size as the input size, padding (adding a border of zeros) is done.

Rectified linear unit (ReLU) is chosen as activation function, this function is commonly used as activation function in the last few years (Lecun et al., 2015) . The ReLU formula is quite simple, namely $f(x) = max(0, x)$, f(x) is zero when z is less than zero and f(x) is equal to x when x is above or equal to zero. The number of filters shifting over the input layers differs for the convolutional layers, see Appendix A. The number of filters increases when the dimension is reduced, and decreases again for the convolutional layers in the decoder. In the pooling layers, the maximum value of 2x2 blocks are mapped on the output map. This means that with every pooling layer, the dimension is reduced by half the size of the input map. After the two blocks, the dimension of the original input image is reduced four times, so at the botteleneck, the dimension of the map is 16x16. The last layer of the encoder is a convolutional layer with 128 filters shifting over the map to extract features.

The decoder consist of four blocks. In every block, there is one upsampling layer and two convolutional layers. For the upsampling layers, nearest neighbour is used as interpolation technique. After the four blocks, the image is 256x256 pixels, four times doubled in size compared to the output image of the encoder (16x16).

3.4 Experimental setup: defining model structure

To define the optimal model structure of the deep convolutional auto-encoder, several model structures were tested. This includes:

- Changing the number of convolutional blocks
- Adding dense layers (fully connected layers)
- Using convolutional transpose layers instead of the upsampling layers

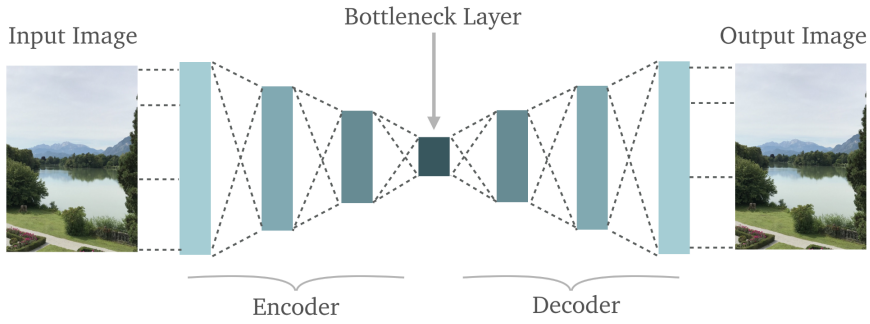


Figure 3.4: Visualization auto-encoder in neural network (Kala, 2020)

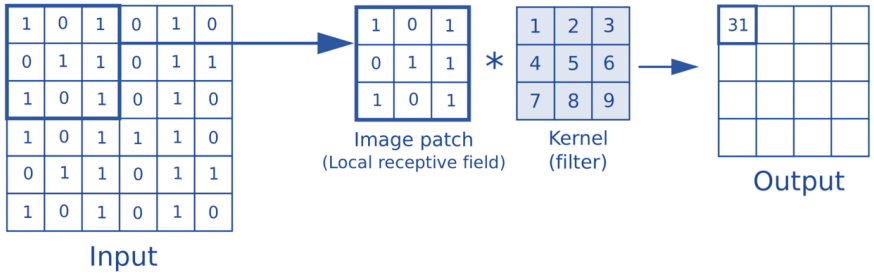


Figure 3.5: Visualization of the multiplication of the filter weights with the patch of input data (Anh H. Reynolds, 2019)

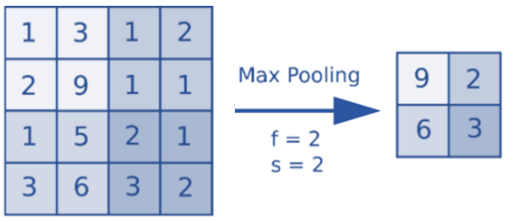


Figure 3.6: Visualization pooling layer (Anh H. Reynolds, 2019)

- Varying the number of convolutional filters (between 32 and 512)

Next to these structural changes, we also tried both bilinear and nearest interpolation for the upsampling layers. The final model structure was chosen based on the validation loss, which will be discussed in the next section.

3.5 Experimental setup: training neural network

The neural network learns by minimizing its loss function. The mean squared error (MSE) is a commonly used loss function for super-resolution networks (Ledig et al., 2017). The MSE is the mean squared difference between the model output and the 'true' high-resolution image (equation 2) . This is visualised in figure 3.7, where the model tries to minimize the difference between the middle row of images (generated high-resolution images) and the last row (true high-resolution images).

$$L(W) = \frac{1}{n} * \sum (y_i - f(x_i, W))^2 \tag{2}$$

With n being the number of images, y_i the true high-resolution image and $f(x_i, W)$ the generated image by the neural network, given its weights W.

The optimal weights to minimise the loss function, are determined by the Adam optimizer proposed by Kingma and Ba (2015). According to Kingma and Ba (2015), the method is computationally efficient, requires little memory and works well for problems that deal with large amounts of data/parameters. Beforehand of training the model, you need to define the number of epochs that you want to train the model. In each epoch, the whole training dataset is used to train the neural network. So the number of epochs says how many times the model iterates through the entire training dataset (Zhang et al., 2020). The model weights are updated after each batch of training images. There are 32 images in one batch. The training loss (loss function) is updated after each batch of images. After each epoch,

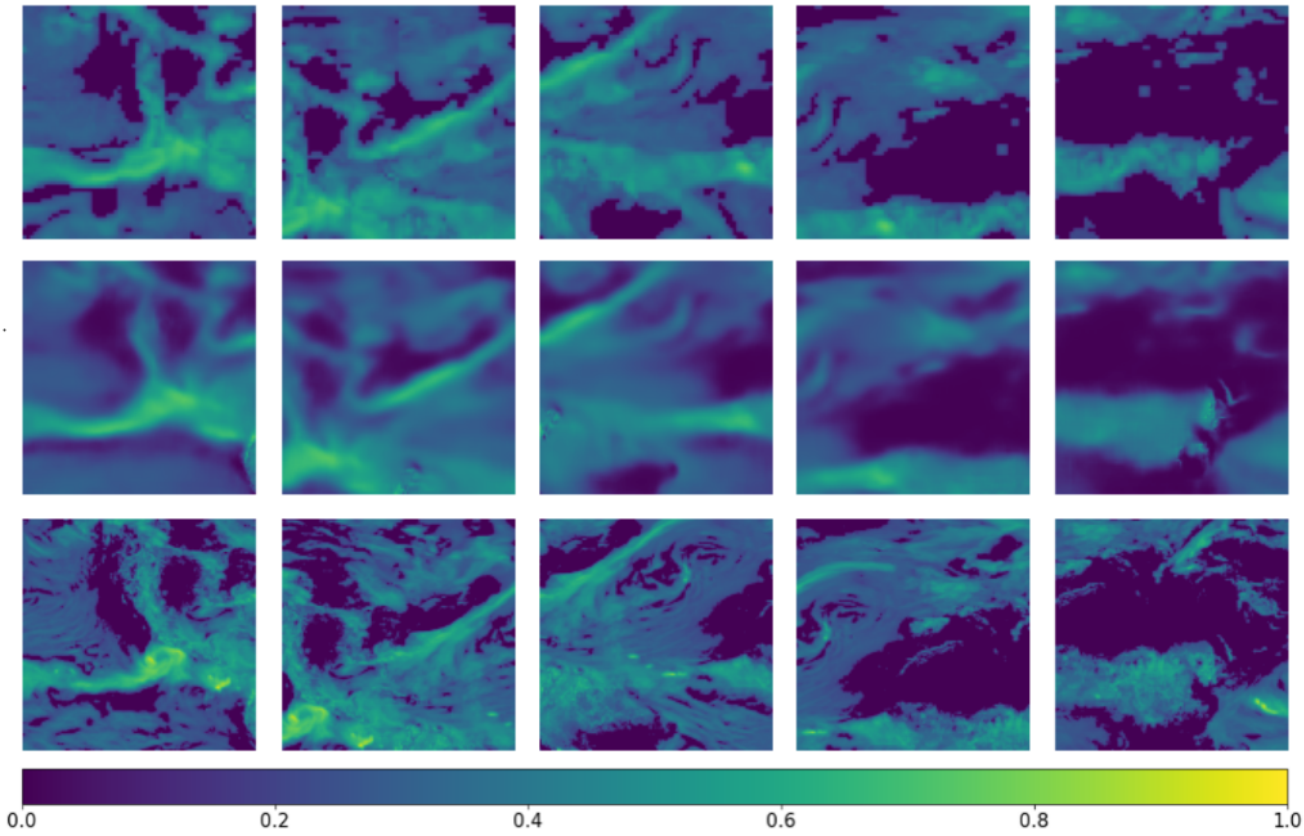


Figure 3.7: First row: Normalised (and rescaled) model input images, second row: model output images, third row: normalised and rescaled 'true' resolution images.

the validation loss is defined, which is the same as the training loss but then for the test dataset. The lower the final validation loss, the better the model performance.

The model performance is dependent on hyper-parameters. Hyper-parameters are parameters that control the learning process like the learning rate and the number of epochs that the model is trained for. The learning rate determines the degree of change of the model when new model weights are defined at the end of each batch. Determining the right value for hyper-parameters is important, training too slow can make the training process take forever, training too many epochs can cause overfitting of the model, training with a too high learning can cause the model to jump over the minimum loss. We chose the right values for the hyper-parameters by trial and error and by looking at the loss function. As long as both the training and testing loss go down, the neural network can train for another epoch. When the testing loss goes up, it is an indication that the model is overfitting. An overfitted model means that the model becomes to specific for the training images and works less well on images that it hasn't seen before, i.e. the testing data (Zhang et al., 2020).
While training the neural network with the MSE as loss function, it appeared that the model predicted values between 0-0.7 (normalised/rescaled precipitation) quite well, while it predicted less well for higher values (>0.7). Therefore, we defined and trained our model on a custom loss function, to make the neural network eager to minimize the error in the prediction of higher values. Our custom function includes the MSE, with a multiplication factor of three for values (>0.7), so errors for values at the higher end have more weight.

$$E_i(W) = \begin{cases} (y_i - f(x_i, W))^2 & \text{if } y_i < 0.7 \\ 3 * (y_i - f(x_i, W))^2 & \text{if } y_i => 0.7 \end{cases} \quad (3)$$

$$L(W) = \frac{1}{n} * \sum E_i(W) \quad (4)$$

In equation 3, y_i refers to the true high-resolution image and $f(x_i, W)$ the generated image by the neural network, given its weights W. In equation 4, the n refers to the number of images and $E_i(W)$ the error of each image retrieved from equation 3. The final neural network used for prediction was trained for 18 epochs with an learning rate of 0.0001. In one epoch, 3840 batches of 32 images were used for training, so in total 122.880 images are used to train the neural network. After each epoch, 960 batches of testing data (30.720 images in total) are used to define the testing loss.

3.6 Downscaling performance: downscaled patches

When the neural network is build, trained and tested, the model can be applied to downscale the two validation datasets. After training the model, we noted that the model could predict better for images that have the same location as one of the training patches, than on a new location. Therefore, we downscaled both options for evaluation and comparison. The downscaled outputs are compared with interpolated coarse ERA-20C images. The performance of both methods is tested by comparing the downscaled output with the 'true' high-resolution output. We calculated the RMSE (root mean squared error) to measure the performance of both techniques.

$$RMSE = \sqrt{\frac{1}{n} * \sum (y_i - f(x_i, W))^2} \qquad (5)$$

With n being the number of images, y_i the true high-resolution image and $f(x_i, W)$ the generated image by the neural network, given its weights W. The RMSE is a quantitative method to describe the quality of the downscaled images. You can also evaluate the downscaling performance on perceptual qualities, i.e. does the downscaled result look good? Baart (2013) listed features that make people have trust in a forecast. Some of these features can also be applied the downscaled results, because they can also be seen as a forecast of the high-resolution precipitation map. Therefore, we will also evaluate the results on the following criteria:

- *Predictive:* does the forecast correlate with measurements?
- *Sharpness:* does the forecast predict uncommon events?
- *Face:* does the forecast appear to predict what it should?
- *Spatial:* does the forecast predict the event at the correct location?
- *Skill:* does the forecast do better than a reference forecast?
- *External:* does the forecast system predict in new situations?

Another commonly used measure for the quality of images and videos is the structural similarity index measure (SSIM). Where the MSE looks at the absolute error of pixel values, the structural error takes into account pixel dependencies and looks at batches of pixels (batch size is defined with the filtersize) of images. Wang et al. (2003) proposed a new variant of the SSIM, the multi-scale structural similarity index measure (MS-SSIM). According to (Wang et al., 2003), the MS-SSIM correlates much better with the perceived quality of images than the MSE. An MS-SSIM of 1 means two identical pictures, a MS-SSIM of 0 means no similarity thus very poor image quality. We tested this measure on part of our results and compared the outcome with the other two measures.

3.7 Downscaling performance: hydrological model performance

To see if neural networks have potential to downscale meteorological forcing data for hydrological applications, the downscaled precipitation maps are used as input for the wflow_sbm model of the Rhine. The downscaled patches that were not in training dataset, had Europe in the middle of the patch. The discharge modelled with this data is further mentioned as Qdown. The patches used for training had Europe on the right side, the discharge modelled with this data is further mentioned as Qtrain. Both discharges (Qdown and Qtrain) are compared with the discharge modelled using the fine ERA5 data (Qhigh) and coarse ERA-20C data (Qlow). wflow_sbm runs at 1 km resolution, so regardless of the inputs' resolution, all maps are further interpolated. Precipitation and ET maps are further interpolated using nearest neighbour. Temperature maps are interpolated with closest distance, making use of the lapse rate. When Qdown/Qtrain resembes Qhigh better than Qlow does, it is a first indication that downscaling (precipitation) maps with a neural network can be

used to improve hydrological model performance. (6)

An overview of all the processes and fluxes included in the wflow_sbm model can be found in figure 3.8. The model includes multiple modules to model snow melt, glaciers, lakes, (interception) evaporation and leakage. Schellekens (2019) documented a detailed description of the model. A kinematic wave module describes flow routing for river, surface, and subsurface lateral flow. The kinematic wave approach assumes that topography is the dominant factor controlling water flow. The model is run with a 24h time step. Both the saturated and unsaturated stores are modelled, and the exchange fluxes between the two. The soil is modelled with four layers. In the Alpine region some larger lakes are included in the model. These lakes are represented in wflow_sbm by a reservoir module and requires the locations, surface areas and initial water levels of all lakes and reservoirs. We evaluate the model performance with the Kling-Gupta efficiency (KGE) and the Nash–Sutcliffe efficiency (NSE), using the discharge modelled with ERA5 as reference (Gupta et al., 2009; Nash and Sutcliffe, 1970).

With r being the correlation coefficient between Qhigh and the discharge that is being compared with ERA5, ERA-20C or downscaled ERA-20C (represented by x). The σ and the μ are the standard deviation and the mean of the discharges that are being compared.

The Nash-Sutcliffe efficiency (NSE) is calculated next to the KGE, to help interpret results where the NSE<0, in that case the model is less predictive than the observed mean, i.e. mean modelled discharge of ERA5 (Gupta et al., 2009). The NSE is obtained by dividing the MSE of the discharge that is being compared with ERA5, ERA-20C or downscaled ERA-20C (represented by x), by the variance of the observed (Qhigh) discharge, and subtracting this ratio from 1:

$$NSE = 1 - \frac{MSE_x}{\sigma^2_{QHigh}} \tag{9}$$

$$KGE = 1 - \sqrt{(r-1)^2 + (\frac{\sigma_x}{\sigma_{QHigh}} - 1)^2 + (\frac{\mu_x}{\mu_{Qhigh}} - 1)^2}$$

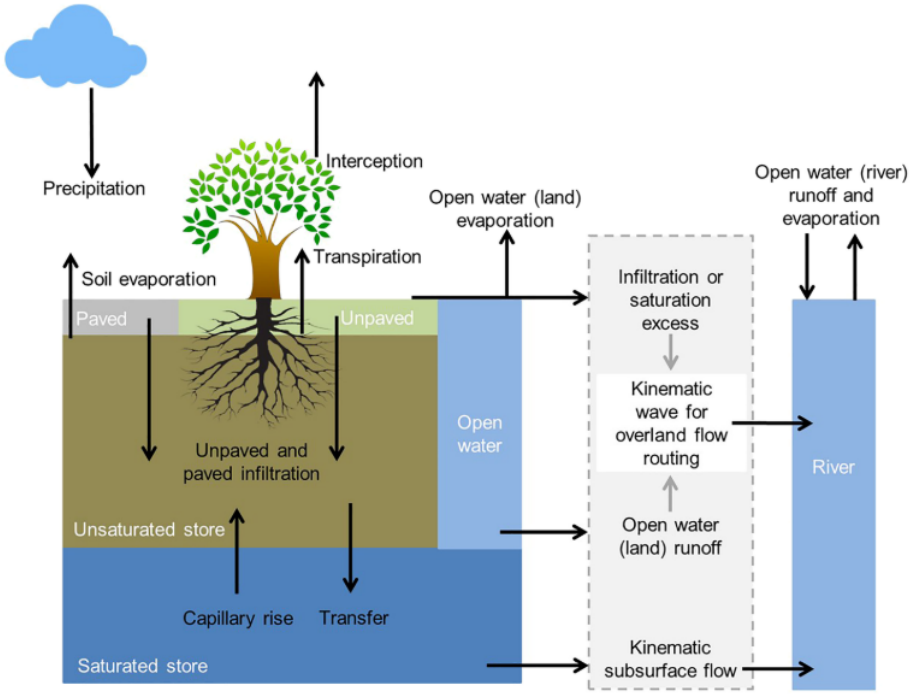


Figure 3.8: Overview of all processes and fluxes included in the wflow model (Schellekens, 2019)

3.8 Scaling factor

The first trial of running the hydrological model using the downscaled data (not presented here), showed us that the modelled discharge did not resemble the reference discharge (ERA5). The total sum of precipitation after downscaling for the catchment area was too low. To account for this, a scaling factor was introduced that is the total sum of precipitation for the low-resolution patch times a factor to account for the difference in scale ($(120/30)^2 = 16$), divided by the total sum of precipitation for the downscaled patch:

$$F = \frac{\sum x_{i,low} * 16}{\sum x_{i,down}} \tag{8}$$

This factor was defined for all patches and multiplied with the downscaled patches, to have a precipitation sum that is closer to the true precipitation.

4 | Results

4.1 Results case study: downscaled precipitation

In this section, we present and interpret our downscaled precipitation maps using a neural network. This regards the first research question about how the downscaled results differ from/ resemble interpolated precipitation. The final training and testing loss of the neural network are 0.0151 and 0.0153, respectively. In figure 4.1, we present the downscaled patches that were on a different location than the training patches. To start, we want to point out that the difference between ERA-20C and ERA5 is not only the resolution, but the reanalysis datasets are produced by different models. Some rainfall features in the fine ERA5 map do not appear in the coarse ERA-20C and the other way around. An example of this is the rain in the yellow squares a and b in figure 4.1, where in both cases the coarse ERA-20C predicts rain in contrast to ERA5. In the case of b, the downscaled maps predicted less rain than the ERA-20C and interpolated map, getting closer to the high-resolution map. The rainfall in c in figure 4.1 is clearly present in the ERA5 map, but not in the coarse ERA-20C, downscaled and interpolated map.

The median RMSE of the downscaled and interpolated patches are in table 4.1. The RMSE of the downscaled images is lower than the RMSE of the interpolated images, so based on the RMSE the downscaled images get closer to the high-resolution map. The RMSE is a measure for how *Predictive* the downscaled precipitation is for the 'true' high-resolution dataset. When we base our judgements on the *Sharpness* of the images, the downscaled result would not do good since it is quite vague compared to the high-resolution image, the rainfall seemed to be smoothed out a bit. In general, the downscaled precipitation maps contain the right rainfall features on the right locations, so *Spatial* features are forecasted well. However, when we base our judgement on *Face* validity, we do miss some important high-resolution details in the downscaled image like rainfall maxima and some detailed rainfall patterns, like for d in figure 4.1. In some cases, the interpolated images even seem to do better based on *Face* validity. The forecast *Skill*, using the interpolated precipitation map as a reference, scores higher for the downscaled image than the reference when using the RMSE as measure. However, the Multi Scale Similarity Index Measure (MS-SSIM) leads to a different conclusion. We calculated the MS-SSIM for the year 2000. The average MS-SSIM of the downscaled and interpolated patches was 0.652 and 0.654, respectively. So according to the MS-SSIM, the quality of the downscaled and interpolated images is almost equal. For 183 out of the 365 days, the downscaled patches had a MS-SSIM equal to or higher than the interpolated patches. Contrasty, for the same year, the RMSE was higher or equal for 358 out of the 365 days.

In figure 4.2, we present the downscaling results of the patches on the same location as one of the training patches. Based on the *Face* validity, these downscaled images look better than the downscaled images in figure 4.1. For example, when we look at the yellow squares b and c in figure 4.1, we see that the downscaled image even learned a high-resolution detail that was not included in the coarse ERA-20C.

Table 4.1: Median RMSE of the downscaled and interpolated patches, for both the downscaling sets for the period 2000-2001

	Interpolated $[mm]$	Downscaled $[mm]$
Patch not in training data	1.59	1.47
Patch in training data	1.48	1.37

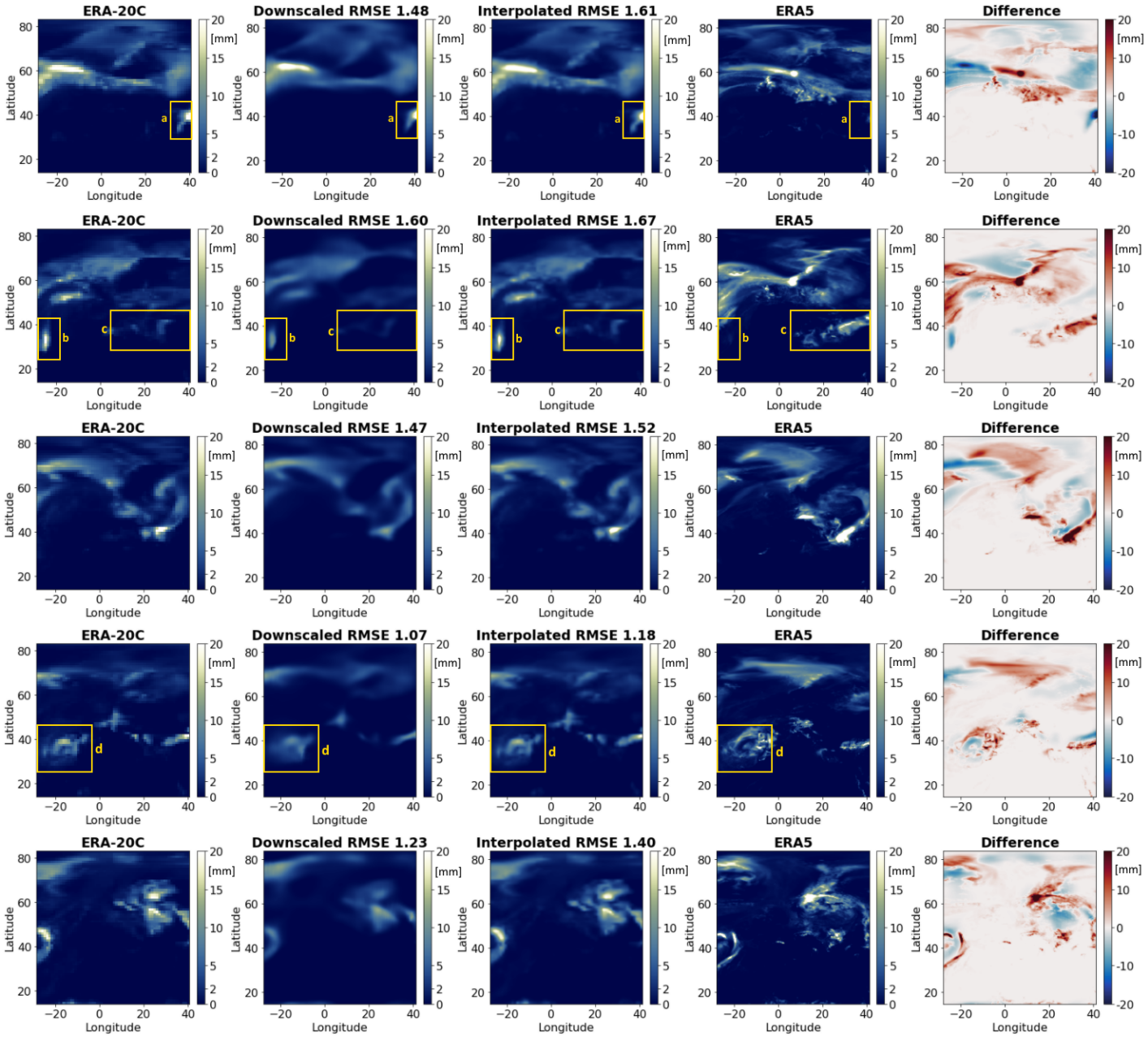


Figure 4.1: The rows show 5 random patches drawn from validation dataset, were the location of the patch was different from the training patches. The columns show the low-resolution map, the downscaled map, the interpolated map, the high-resolution map and a difference map between the high-resolution and downscaled map for each patch. The precipitation is rescaled back to the range of the high-resolution data in [mm].

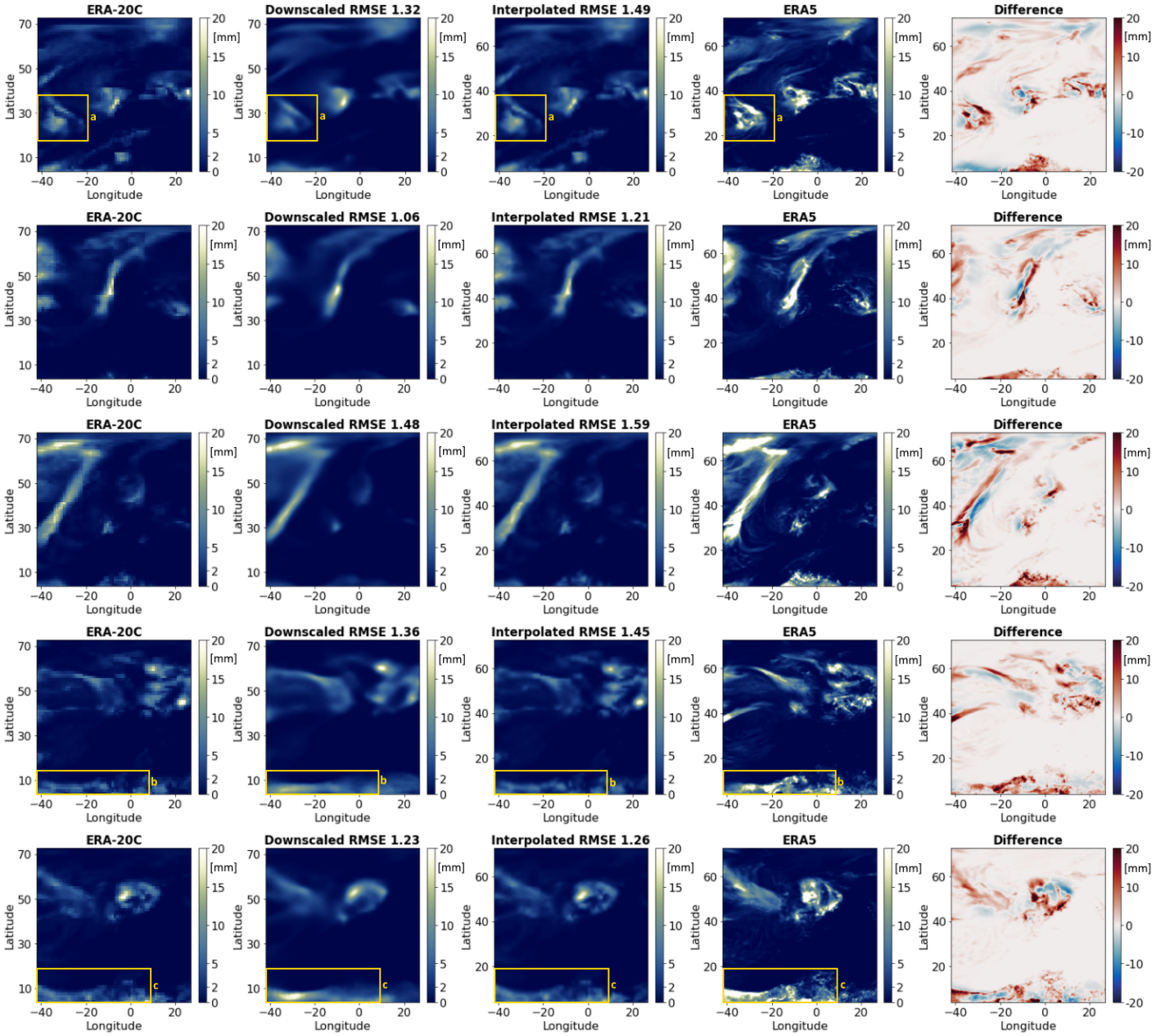


Figure 4.2: The rows show 5 random patches drawn from validation dataset, were the location of the patch matched with one of the training patches. The columns show the low-resolution map, the downscaled map, the interpolated map, the high-resolution map and a difference map between the high-resolution and downscaled map for each patch. The precipitation is rescaled back to the range of the high-resolution data in [mm].

Spatial features in the downscaled images in 4.2 are again on the right locations, noticeably better than the downscaled images in 4.1. This is best visible in the last column of images that show the difference between the high-resolution precipitation map and the downscaled precipitation map, where the area where the prediction is forecasted right (white) is more prone in figure 4.2. The rainfall extremes in ERA5 for the yellow square a in figure 4.2 are not captured in the downscaled image, so on high-resolution details, there is still some room for improvement.

The downscaled images in figure 4.2 are more *Predictive* than the interpolated images for the high-resolution precipitation map, based on the RMSE and *Face* validity. The downscaled images are blurry like the downscaled images figure 4.1, so the *Sharpness* could have been better. The forecast *Skill* of the downscaled images in figure 4.2 is higher than the images in figure 4.1. Furthermore, high-resolution details like rainfall maxima are better predicted. But still, not all high-resolution details are learned by the neural network. Taking into account the forecast criteria mentioned till this point and the RMSE, we can say something about the last forecast criteria *External*. The high-resolution forecast predicts well for new years (validation years not used for training). However, the model predicts less well for new locations, as the downscaled images in figure 4.2 score on most points better than the downscaled images figure 4.1.

4.2 Results case study: hydrological model

The second part of our results are the results of the hydrological model. One of the advantages of downscaling model output could be that it is better suitable to use in a hydrological model simulation. In figure 4.3 and 4.4, you can see the discharge of the Rhine and its tributaries for the year 1981 and 1991. As was described in the method section, the first two years of every decade have been used for validation. We show the discharge of the second year, the first year is used as spin up period, to get the model running with minimal influence of boundary conditions. The discharge are modelled using different precipitation inputs: ERA5, ERA-20C and the two downscaled datasets: Down and DownTrain. In the text these will be described by: Qhigh, Qlow, Qdown and Qtrain. Where Qdown stands for the discharge modelled with the downscaled patches on a different location as the training patches, and Qtrain the discharge modelled with the patches that had the same location as one of the training patches. The model performance is evaluated for two locations of the Rhine river (Basel, Lobith) and for six tributaries (Thur, Mosel, Jagst/Neckar, Main, Ruhr, Nahe). In table 4.2, 4.3 and 4.4, you can find all NSE's and KGE's for the years 1981, 1991 and 2001.

We hypothesised at the start of this research that Qdown would fall in between Qlow and Qhigh. This would mean that the neural network learned high-resolution features, and an increased hydrological model performance as a result. However, even with the scaling factor to correct for the precipitation loss, Qdown and Qtrain still fall below Qhigh and Qlow in most cases. Frankfurt and Grolsheim are an exception, where the discharge is overestimated by Qdown. The discharges of 1981 are more peaky than for 1991. This resulted in lower KGE's and NSE's for Qlow, Qdown and Qtrain since peak discharges of ERA5 are less well represented in the other discharges.

Qlow has a higher KGE and NSE than Qdown/Qtrain for most of the locations, the highest NSE and KGE values for each location are indicated with bold letters in table 4.3, 4.3 and 4.4. Following from this, Qlow resembles Qhigh better than Qdown/Qtrain. Qtrain performs better than Qdown in most cases based on the NSE and KGE. Remarkable is the small difference between Qdown and Qtrain, where a larger difference was expected when looking at the downscaled precipitation patches. For Cochem and Frankfurt, Qdown does well at predicting some of the peaks, were other peaks are clearly overestimated. While the model performance of some locations is quite high, others are very low, there is a large spread in KGE and NSE values.

The modelled discharge of the Rhine at Basel and Lobith have a similar good correlation of Qdown and Qtrain with Qhigh (spearman correlationcoefficient >0.9), which means they follow Qhigh very well and have the peaks and low discharge at the same time. However, the mean Qdown and Qtrain is too low, this is also evident when comparing the NSE and KGE. The NSE becomes negative when the MSE is larger than the squared variance of Qhigh, in that case you are better of using the mean Qhigh as discharge estimate than

Qdown/Qtrain. The decomposed NSE has three components, representing the bias error (difference mean observed and simulated), variance error and correlation of the observed (Qhigh) and simulated discharge. These components don't have equal weight, reason for Gupta et al. (2009) to propose the KGE as better measure for model performance, where in the KGE these three components have equal weight. Accordingly, the KGE's are higher than the NSE's, because in the NSE the bias error (underestimation discharge) has more weight than the correlation coefficient relatively.

For the NSE, the benchmark to distinguish good and bad models is NSE=0. For the KGE this benchmark is less clear, as discussed by Knoben et al. (2019), where negative values do not necessarily mean that the model performs worse than the mean discharge. This is the reason why we present both the NSE and KGE values. For Andelfingen and Basel, the NSE is negative for 1981 and 1991, indicating that you can better use the mean of the Qhigh, which is in line with the figures where it is clearly evident that Qdown and Qtrain underestimate the discharge. Nonetheless, the KGE's are positive because of the good correlation between Qhigh and Qdown/Qtrain.

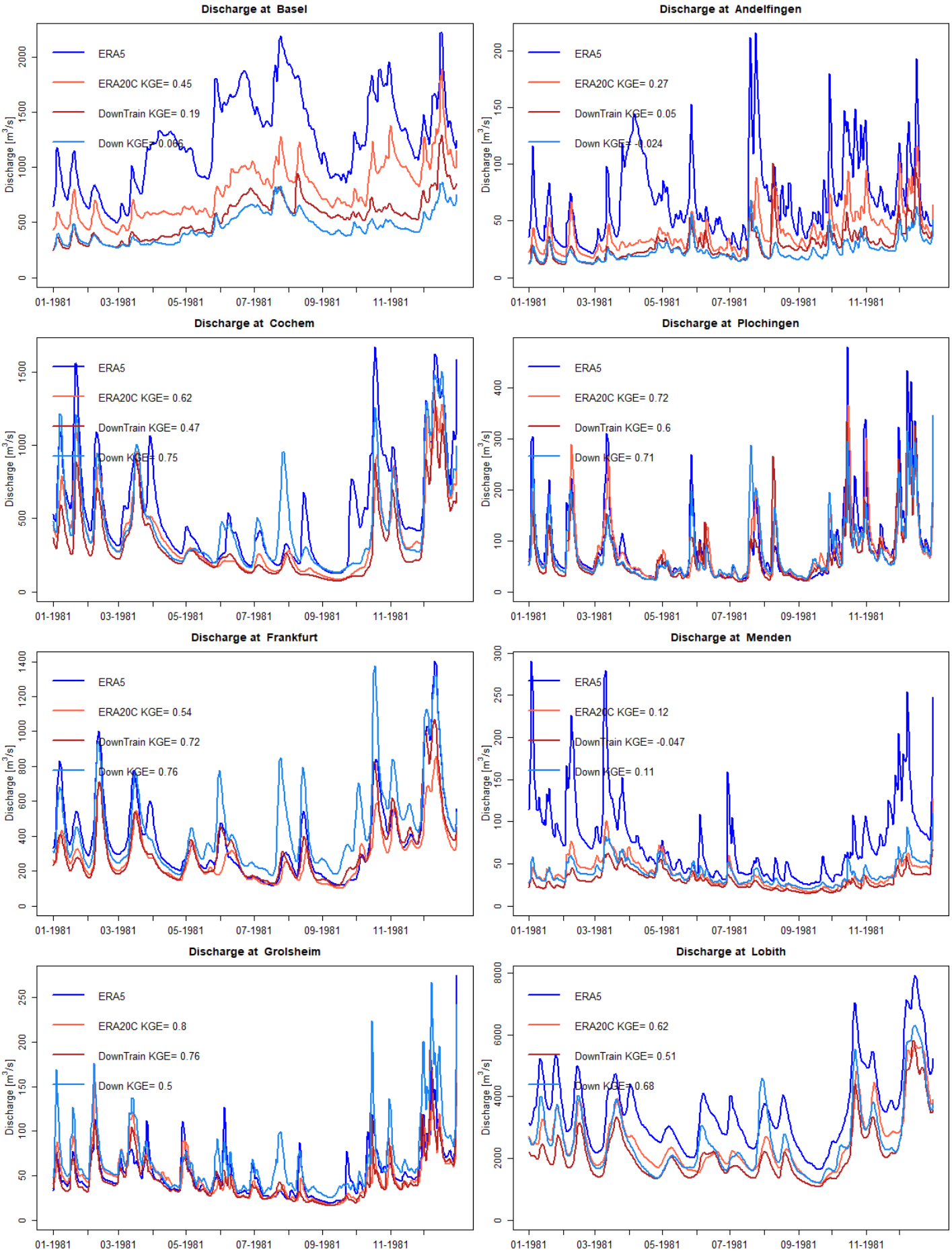


Figure 4.3: Modelled discharges for the year 1981 using the following precipitation datasets: ERA5, ERA-20C, downscaled ERA-20C on a location different from the training patches (Down), and downscaled ERA-20C on the same location as a training patch (DownTrain). Modelled Rhine discharge for Basel, Andelfingen, Cochem, Plochingen, Frankfurt, Menden, Grolsheim and Lobith.

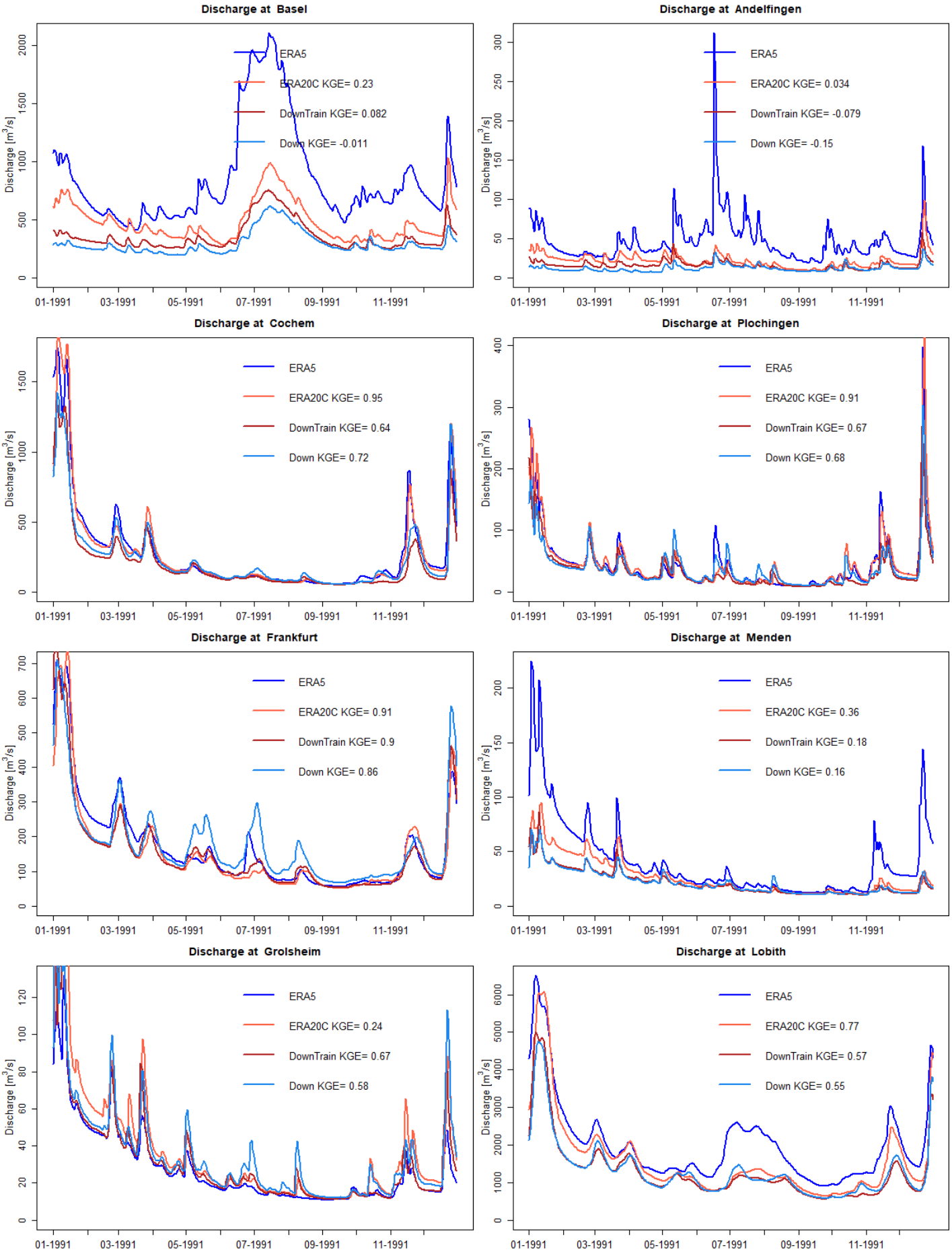


Figure 4.4: Modelled discharges for the year 1991 using the following precipitation datasets: ERA5, ERA-20C, downscaled ERA-20C on a location different from the training patches (Down), and downscaled ERA-20C on the same location as a training patch (DownTrain). Modelled Rhine discharge for Basel, Andelfingen, Cochem, Plochingen, Frankfurt, Menden, Grolsheim and Lobith.

Table 4.2: KGE's and NSE's for the modelled discharge at several locations using the interpolated (Qlow) and downscaled precipitation patches (Qdown and Qtrain) as input, and the discharge modelled with ERA5 as a reference, for the year 1981. Qdown: different location than training patches, Qtrain: same location as training patch.

		Qlow		Qdown		Qtrain	
Location	River	KGE	NSE	KGE	NSE	KGE	NSE
Basel	Rhine	**0,45**	**-0,71**	0,07	-3,48	0,19	-2,60
Andelfingen	Thur	**0,27**	**-0,27**	-0,02	-1,33	0,05	-1,02
Cochem	Mosel	0,62	0,57	**0,75**	**0,66**	0,48	0,34
Plochingen	Jagst, Neckar	**0,73**	**0,74**	0,71	0,73	0,60	0,63
Frankfurt	Main	0,54	0,60	**0,76**	0,60	0,72	**0,74**
Menden	Ruhr	**0,12**	-0,31	0,11	**-0,27**	-0,05	-0,73
Grolsheim	Nahe	**0,82**	**0,65**	0,50	0,17	0,76	0,63
Lobith	Rhine	0,62	0,20	**0,68**	**0,29**	0,51	-0,36

Table 4.3: KGE's and NSE's for the modelled discharge at several locations using the interpolated (Qlow) and downscaled precipitation patches (Qdown and Qtrain) as input, and the discharge modelled with ERA5 as a reference, for the year 1991. Qdown: different location than training patches, Qtrain: same location as training patch.

		Qlow		Qdown		Qtrain	
Location	River	KGE	NSE	KGE	NSE	KGE	NSE
Basel	Rhine	**0,23**	**-0,32**	-0,01	-1,40	0,08	-0,95
Andelfingen	Thur	**0,03**	**-0,52**	-0,15	-1,29	-0,08	-0,95
Cochem	Mosel	**0,95**	**0,93**	0,72	0,86	0,64	0,84
Plochingen	Jagst, Neckar	**0,91**	**0,84**	0,68	0,79	0,67	0,80
Frankfurt	Main	**0,91**	**0,93**	0,86	0,84	0,90	0,92
Menden	Ruhr	**0,36**	**0,45**	0,16	0,18	0,18	0,20
Grolsheim	Nahe	0,24	0,19	0,58	0,62	**0,67**	**0,76**
Lobith	Rhine	**0,77**	**0,71**	0,55	0,43	0,57	0,42

Table 4.4: KGE's and NSE's for the modelled discharge at several locations using the interpolated (Qlow) and downscaled precipitation patches (Qdown) as input, and the discharge modelled with ERA5 as a reference, for the year 2001. Qdown: different location than training patches (no Qtrain available for this period)

		Qlow		Qdown	
Location	River	KGE	NSE	KGE	NSE
Basel	Rhine	**0,47**	**-0,04**	0,22	-1,68
Andelfingen	Thur	**0,29**	**-0,33**	0,12	-1,26
Cochem	Mosel	**0,73**	**0,83**	**0,73**	0,67
Plochingen	Jagst, Neckar	**0,85**	**0,72**	0,71	0,67
Frankfurt	Main	**0,80**	**0,86**	0,58	0,53
Menden	Ruhr	0,58	0,50	**0,74**	0,14
Grolsheim	Nahe	**0,33**	**0,24**	-0,05	-0,84
Lobith	Rhine	**0,67**	**0,77**	0,56	0,42

5 | Discussion

5.1 Downscaling performance

First, we will discuss the downscaling performance measures used. As was mentioned in the results, our downscaling results score well with the RMSE metric. The MS-SSIM metric showed to be more positive about the interpolated images for the patches that were on a new location compared to the RMSE. The MS-SSIM corresponds better with our perceptual assessment, consistent with the conclusion of Wang et al. (2003). This shows the importance of having a good performance metric. An important remark is that the model was trained to minimize the MSE, hence it was not trained to perform good on performance metrics other than the RMSE/MSE. To have a higher similarity index, it should be included in the loss function. In addition, whether a perceptually better results is truly a better result first has to be confirmed by an improvement in the hydrological model performance.

The RMSE is lower for the downscaled patches that were on the same location as the training patch, than for the patches that were on another location. However, the RMSE of the interpolated images is also lower, see table 4.1. This could mean the difference in RMSE of the downscaled images is not only because of a difference in downscaling performance, but that there is another factor. The datasets are exported from GEE, were a bounding box had to be defined. For the training dataset, the bounding box covered the whole Earth, were for the downscaling dataset, only Europe was cut out. To have the right number of pixels (64x64 and 256x256), the bounding box for Europe was a bit different for the high and low-resolution data. This can have caused the datasets to be shifted from each other by one pixel, and associated to this, a larger error for both the interpolated and the downscaled patches.

Vagueness is a side effect of using MSE as loss function to optimize the neural network, so this explains why the downscaled images do not score well on *Sharpness* (Ledig et al., 2017). Even though MSE is commonly used as loss function for super-resolution networks, its ability to capture details that are perceptually relevant is limited (Ledig et al., 2017). The MSE loss function encourages finding pixel averages of plausible solutions, causing a smoothed image as result. This can be explained as follows: when there is a possibility that there is a rainfall feature in the high-resolution image, the MSE will cause just a little bit of rain, since the error is lowest in both cases (yes or no rain). For b in figure 4.1, this resulted in a lower amount of rainfall than the low-resolution image, getting closer to the high-resolution image. In addition, the MSE as loss function can have caused underestimation of high rainfall amounts for the same reason. The model is very good at predicting rain in the middle range, but performs less well on the lower and higher end of the spectrum.

The downscaled images in figure 4.2 score better on *Spatial* features and *Face* validity based on high-resolution details. This proves that the model is trained specific for the locations of the training patches, and coupled rainfall patterns to locations. The 17 patches were not random enough, an irregular pattern of cutting out images of the tiles would have been better to have a model that predicts well irrespective of geographical locations. We used precipitation images of the whole world to have a large training dataset, and to have a model that works for the whole world. We did not expect the model to couple rainfall patterns to locations, but it doesn't sound unreasonable, since rainfall distribution is influenced by topography, land use and land-sea distribution (Lukas, 1992; Smith, 1979; Ter Maat et al., 2013; Saavedra et al., 2020; Fick and Hijmans, 2017). Furthermore, it shows that if you want to create a model to downscale data of a specific region, it is good to train on that area specifically.

Comparing the downscaled ERA-20C with ERA5 is not completely 'fair', since the datasets are produced by different models. Due to this difference, some rainfall features appear in the ERA5 map but not in ERA-20C map and the other way around. These differences result in a volume bias, which affects the downscaling performance. Part of the RMSE can be explained by the difference in models, so this has to be taken into account when evaluating the potential of neural networks to downscale model output data. The case study does show

that the trained neural network could map some low-resolution features at high-resolution, which was exactly to goal of this research. The have a 'fair' evaluation of this downscaling technique, you can use upscaled ERA5 as input data instead of the coarse ERA-20C, and see if the neural network can downscale the upscaled ERA5 back to the original ERA5 precipitation map. Another test would be to use this model to downscale ERA-20C data. In that case, there is no volume bias during training. It would be interesting to see whether a model trained in upscaled ERA5 data, would do good in downscaling ERA-20C.

We compared our results with interpolation, we could also have chosen for a more advanced technique to really 'challenge' our downscaled precipitation. This choice may have influenced our result and part of the conclusion. For the downscaled results, there is still a lot of room for improvement. We started with a simple model and using standard options of Tensorflow like the MSE loss function. Therefore, we compared our results with interpolation as a first indication of the downscaling performance.

5.2 Hydrological model performance

As explained in section 3.8 a scaling factor was introduced to account for precipitation losses in the downscaling procedure. The choice of this scaling factor influences the modelled discharge and therefore also the hydrological model performance.

Now the question is, where did the precipitation got lost? We expect the MSE loss function to have cased the biggest loss of precipitation. The MSE has a smooting effect, which leads to an underestimation of high rainfall amounts. Furthermore, it leads to a underestimation of rain for spatial features that are a possibly in the high-resolution map. We expect that applying a similarity index or perception loss partly solves the problem of losing precipitation. In addition, to ensure no precipitation gets lost a volume loss penalty could be introduced in the loss function. This means that in the loss function, an extra term that equals the difference in total sum of the low-resolution map (x16 to account for different scales) and the output of the neural network times a factor increasing as the difference increases (for a positive difference). This will make the network indirect more keen to conserve the volume of rain. The total volume of rain was larger for ERA5 than for ERA-20C, so you could also include the total high-resolution sum in the loss function. For the scaling factor we deliberately used the low-resolution total, since we want to obtain high-resolution from low-resolution.

Next to the precipitation loss during the downscaling procedure, part of the discharge underestimation of the downscaled precipitation is related to the total precipitation volume bias between ERA5 and ERA-20C. For most locations, the ERA-20C discharge lies below the ERA5 discharge. According to Kim et al. (2018), ERA-20C underestimates rainfall intensity. This is also visible when looking at the precipitation maps in figure 4.1 and 4.2, where in some cases the rainfall intensity is clearly underestimated. This leads to a difference in total precipitation volume, resulting in a lower discharge.

We have chosen one scaling method to rescale/normalize both the high and low-resolution data between 0-1. You could also make two scalers, one for the high and one for low-resolution data. In that case, the rescaled low-resolution data are rescaled with the high-resolution scaler after being downscaled. This choice probably has influenced the results, how the results will change when using two scalers is hard to predict. We fitted the scaler on the high-resolution precipitation, in contrast to the low-resolution data, this dataset included values larger than 0.04. Therefore, the rescaled input image does not include values larger than 0.8, this is different when you fit the scaler on the low-resolution data. There is a possibility that our choice has caused part of the loss of precipitation volume.

Even though we used a scaling factor to correct for the loss, the modelled discharge still underestimates the ERA5 discharge. This again shows the importance of having a mass conservative loss function to optimize you neural network. The large difference in model performance between locations can be related to the scaling factor. All pixels are multiplied with a certain factor to have the same total precipitation sum as the low-resolution patch. This can have caused an overestimation of rain for the locations were rain was predicted, but still an underestimation of rain for areas where no rain was predicted (on locations were ERA5 did have precipitation), because after multi-

plication with the scaling factor there is still no rain.

5.3 Applying super-resolution neural networks on model output data

The case study exposed some of the challenges that are faced when downscaling model output data. To start with, the resolution was not the only difference between the two reanalysis datasets, but they also differ in model physics and data assimilation methods. This highlights the difficulty of having a neural network learning all these differences. Another challenge is to have a good downscaling performance metric and including this metric in the loss function to optimize the model based on your own defined quality measure.

To downscale model output data, changes are needed in the methodology when downscaling data instead of pictures, regarding the last research question. Starting with the input data, two model outputs are needed with resolution difference that is a multiple of two, since neural networks are designed to work with images that have a size of 64,128,256,512 etc. Furthermore, special attention has to be paid on the coordinate references. During the pre-processing there are a lot of steps, important is to make sure that the final training patches spatially match. An extra step in the pre-processing is needed, namely rescaling and normalizing the data between 0-1 (see section 3.1). Our results show that there is still some room for improvement. In section 5.6, we propose some alterations to the method that we used in this research based on the results of the case study.

The case study proves that a neural network can learn high-resolution patterns and preform better than interpolation based on the RMSE, which shows the potential of this technique to downscale all kinds of model output data. Some of the challenges also go together with opportunities. For example, the importance of the performance metric shows that the downscaled results can be steered to a specific direction. This means the model can be optimized for a specific goal, e.g. to predict maxima or minima. This feature can be used as an advantage for many applications. Another advantage is that the model only has to be trained on GPU once, and thereafter can be applied for downscaling model output data. The actual downscaling goes very quick and therefore this method is computationally efficient and cheap. In this research, we chose to use a deep convolutional auto-encoder. In the next section, we discuss two alternative network structures that are commonly applied for super-resolution. Both are originally applied to colour images but can also be applied to downscale model output data with some alterations.

5.4 Alternative model structures

The first alternative to the deep convolutional auto-encoder used in this research is the variational auto-encoder. The concept of the variational auto-encoder was proposed by Kingma and Welling (2014), it is a probabilistic variant of the traditional auto-encoder. The auto-encoder finds the distribution of a latent feature given the input, and the decoder finds a distribution of possible outputs given a latent feature (Yeung, 2017). The VAE network structure has already been applied for creating super-resolution images Ma et al. (2019); Huang et al. (2018). Since the output is a generated image from a distribution, you can explore the variations of the output images of the model. You could therefore create multiple possible high-resolution forecasts and in this way create new (artificial) data. We expect the downscaling *Skill* to be comparable with the *Skill* of the deep-convolutional auto-encoder that we used in this research.

The second alternative would be to use a Generative Adversarial Network (GAN) structure. The GAN framework was introduced by Goodfellow et al. (2014). GANs consist of two convolutional neural networks. The first network is the generator, and is trained to generate high-resolution images that look like the 'true' high-resolution images . The

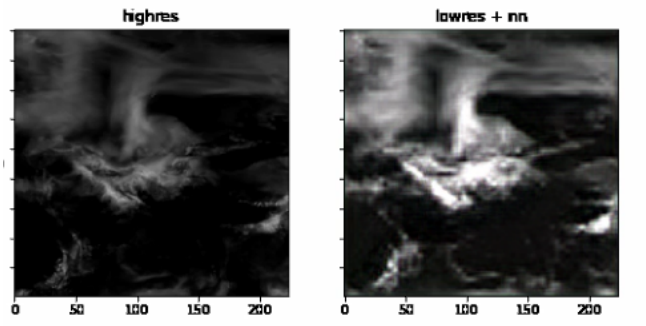


Figure 5.1: Left: 'true' normalised high-resolution map, right: downscaled precipitation map using a GAN.

second, the discriminator, is trained to tell whether an image is 'true' or 'fake' (generated by the generator) (Xie et al., 2018). The disadvantage of GANs is that they are originally made to create super-resolution colour images with three bands. The precipitation data has only one band. The GAN network can still be used, by creating an image with three same bands (of precipitation data). We tested this using our precipitation images and it does look promising, figure 5.1 shows our result. An alternative would be to use a DEM map and pressure/temperature map as the second and third band (all normalised between 0-1). Because GANs are trained to replicate rainfall features and patterns of the high-resolution map, it might result in a better downscaling performance than auto-encoders.

5.5 Related work

Downscaling precipitation data using a neural network has been done before. The most important difference is the input data. Where we used one coarse precipitation map as input, others used multiple coarse precipitation forecasts (Mendes and Marengo, 2010; Rodrigues et al., 2018). Rodrigues et al. (2018) applied a deep convolutional network to downscale precipitation using multiple precipitation forecasts from the Coupled Model Intercomparison project Phase 5 (CMIP5). The downscaling performance was measured by the RMSE and was lower for the downscaled precipitation using the neural network than for linear regression of the coarse precipitation forecasts. Similar to our research, they show the potential of neural networks for generating high-resolution data from low-resolution data. Instead of multiple precipitation maps as input, you can also used multiple other input variables. Vu et al. (2016) trained a neural network to downscale future precipitation estimates of Bankok, using multiple predictors defined with a principal component analysis. They conclude that choosing appropriate predictors is very important, because the downscaling results strongly depend on the choice of predictors. Next to the other meteorological variables (e.g. temperature, pressure), you can also use DEM or landuse as model input. Gerlitz et al. (2015) combined large-scale atmospheric dynamics with local topographic characteristics to downscale ERA interim precipitation. This could be a way to have a global downscaling model, since the neural network is trained to couple rain to elevation, and can therefore predict better for new locations. The goal of this research was to create high-resolution data from low-resolution data, using only the high and low-resolution data and a neural network. Using more predictors could result in a better downscaling performance, yet we started with a 'simple' model to look at the potential of neural networks rather than having a complex model with a lot of predictors.

In this research, we chose the MSE as loss function since it is commonly used for super-resolution, and because it was already included in the Tensorflow software. Ledig et al. (2017) proposed a super-resolution generative adversarial network (SRGAN) with an alternative loss function for the MSE, the perceptual loss function. Their work builds on the work of Johnson et al. (2016), who proposed the perceptual loss function. The perceptual loss function is named as solution to have a perceptually better result compared to the MSE. Where the MSE calculates per pixel similarities and the SSIM the similarity of batches of pixels, the perception loss is based on high-level image feature representations extracted by a pre-trained convolutional neural network (Johnson et al., 2016). Whether a perceptual better result goes together with a better high-resolution precipitation forecast has to be seen. This can be measured by using both downscaled results, trained on MSE and on SSIM/perception loss, as input for wflow_sbm and compare the hydrological model performances.

5.6 Recommendations

The recommendations are split in two lines of thinking: keep the one variable model or create a multi-variable model. In this research, we trained our model solely on high- and low-resolution precipitation data. We think this one-variable model has quite some unexplored potential. Therefore we will first propose several things that could improve the downscaling performance of this model.

We think the largest gain in model performance can be achieved by using a different loss function. Training on pixel similarity results in a blurry image, as we have seen for the downscaling results. We propose to use an image structure similarity measure or perceptual loss as alternative loss function to have a higher *Face* validity. When it appears that precipitation still gets lost using one of the two loss functions, you can include a penalty function (see section 5.2). Including

temporal coherence in the neural network could also improve the forecast, since next to spatial patterns, rain also has temporal patterns. This can be done by including recurrent neurons in a fully connected layer. The recurrent neurons in the fully connected layer give as output a weighted sum of the current input and last output value. Because the last value is used for calculating the new value, the network learns long-term dependencies in the dataset (temporal coherence). Lastly, a GAN model structure can be used as alternative for the deep convolutional auto-encoder. The results show that a location specific model works better when using precipitation as only input to train the model. In case you want to downscale data of the whole globe, you can either cut the world map in several pieces and train a separate model on each region. Another way to go would be to split the training data based on climate zones, and have separate trained models for each climate zone.

For a global model, we expect that a multi-variable model is needed to improve the downscaling performance. This means using multiple meteorological variables as input (e.g. dewpoint temperature, pressure or convective available potential energy (CAPE)) and possibly also land use and DEM (digital elevation model).

After creating a more advanced model, it is better to also evaluate the results with a more advanced precipitation interpolation method like WorldClim. WorldClim is an interpolation technique that uses satellite-derived data, covariables and a smoothing spline algorithm to downscale climate variables (Fick and Hijmans, 2017).

6 | Conclusion

Here we present an approach to use a super resolution neural network to convert coarse model output to higher resolution model output. We have put this approach to the test by evaluating its performance on a real world use case, by downscaling reanalysis precipitation data. First, we evaluated the downscaled precipitation data and compared it with interpolated coarse ERA-20C data. Based on the RMSE, the downscaled map get closer to the fine ERA5 map than the interpolated patches. In contrast to interpolation, the downscaled map showed high-resolution features learned by the neural that were not present in the coarse ERA-20C map. The model could predict well for new years (validation years), but less well for a new location. Using the MSE as loss function showed to have to negative side effects, the downscaled images are smoothed and precipitation maxima are underestimated. The perceptual quality of the downscaled images can still be improved, its therefore important to define a good evaluation metric for the quality of the forecast. Next to optimizing this evaluation metric, the loss function should also include a term to conserve the total volume of precipitation, as the results showed that precipitation got lost in the downscaling process. In addition, you can try including recurrent neurons in your network for temporal coherence or use a GAN network structure to improve the downscaling performance. Part of the error between the downscaled precipitation and the fine ERA5 precipitation was due to differences between ERA5 and ERA-20C (coarse reanalysis data). The models do not only differ in resolution, but also have differences in model physics and data assimilation methods.

The second evaluation of our results was done with a hydrological model of the Rhine, wflow_sbm. The modelled discharges of the Rhine and its tributaries, are compared based on the NSE and KGE for 8 locations. From the results we can conclude that the downscaled precipitation inputs do not improve the model performance of wflow_sbm in comparison with using coarse ERA-20C interpolated to the grid of wflow_sbm. Although a scaling factor was introduced to solve the problem of the volume precipitation loss, the results still show that the modelled discharge underestimates the ERA5 discharge in most cases.

The networks used for generating super-resolution images cannot directly be used to downscale geophysical model output data. For example, GANs are specifically designed for colour images with three bands, yet they can be applied for downscaling data with some modifications. First of all, you need two model outputs with resolution difference that is a multiple of two and having the same coordinate references. Before creating the training dataset from the pair of model output data, decide on making a global or location specific model. To have a model that works well for the whole world, we recommend to spend extra attention on randomizing the training data and using multiple input variables/datasets. The next important difference in method is the data normalisation between 0-1. For colour images this simply means dividing all pixel values by 255. For downscaling geophysical data, you need to look at the data distribution (normal, skewed) and think of a smart way to rescale the data between 0-1. The data distribution after rescaling influences the model output. Lastly, it is better to use a custom loss function instead of a standard loss function like te MSE.

Even though the results show that the quality of downscaled data is not high enough to improve the hydrological model performance, we still think neural networks have potential to downscale geophysical data. The neural network learned high-resolution features. Furthermore, super-resolution neural networks for pictures improve fast, we can benefit from this for our application. A great advantage is the wide-applicability of this downscaling technique, it can be applied to many kinds of model output data, e.g. morphology, hydrogeology, climate forecasts, drought/flood forecasts. Furthermore, neural networks are computationally efficient and can be optimized for a specific goal. Moreover, the model output resolution can be increased without the need of having knowledge on underlying processes. This research contributes to the development of the right method for downscaling geophysical model output data with a neural network by presenting a general method outline and focus points. Based on our suggestions, follow up research can further investigate and apply neural networks for downscaling model output data.

7 | Acknowledgements

I would like to thank my supervisors, Fedor Baart and Albrecht Weerts for their help during my MSc thesis project. Due to this project, I learned how to use deep learning and all the infrastructure that comes with it. I would also like to thank Gennadii Dochyts, for helping out with the neural networks. Furthermore, this project is part of the Enabling Technologies program of Deltares, so I want to thank Deltares and the ET program for giving me the opportunity to do this research.

A | Network Design

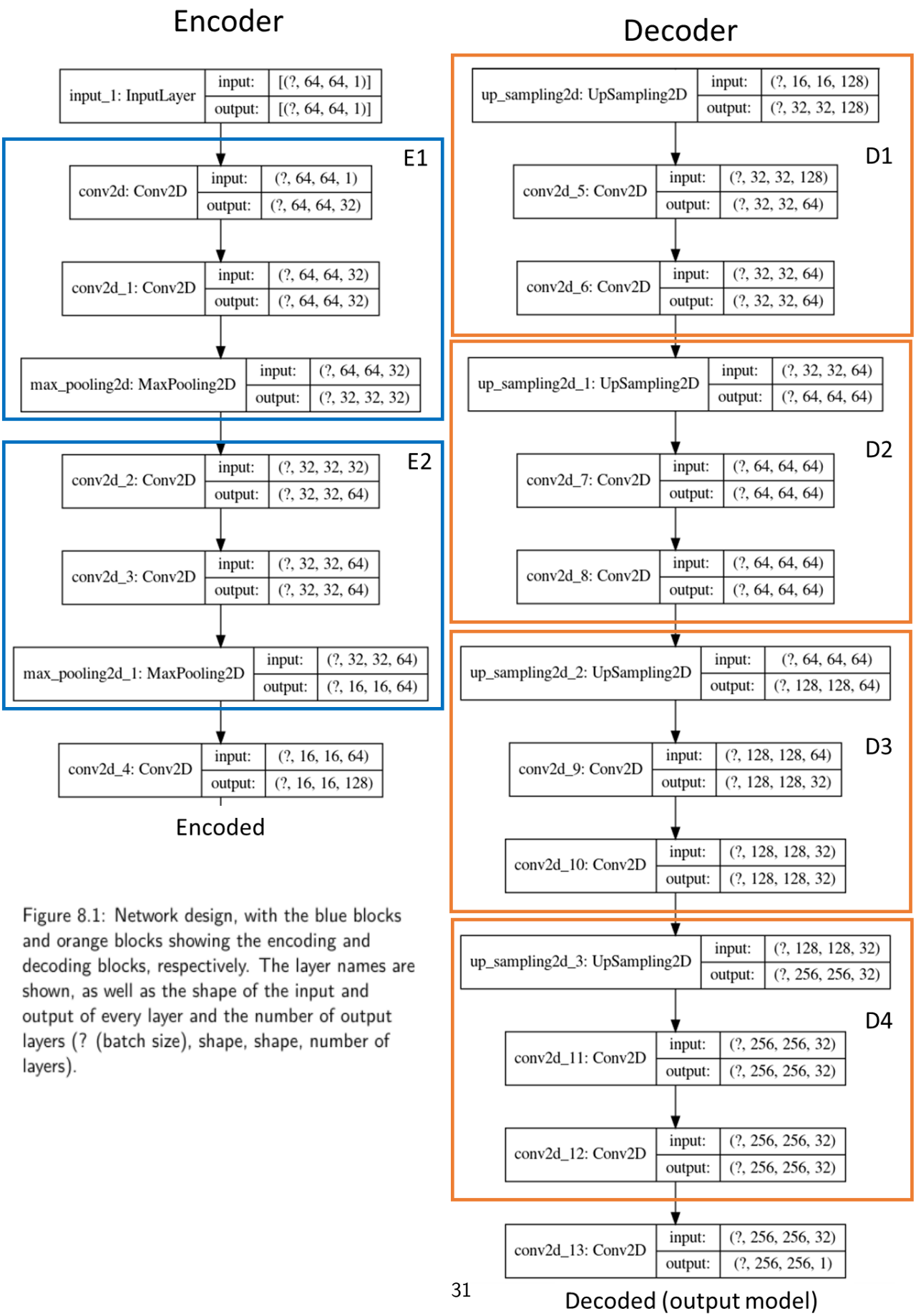


Figure 8.1: Network design, with the blue blocks and orange blocks showing the encoding and decoding blocks, respectively. The layer names are shown, as well as the shape of the input and output of every layer and the number of output layers (? (batch size), shape, shape, number of layers).

Bibliography

Ahmed, K. F., Wang, G., Silander, J., Wilson, A. M., Allen, J. M., Horton, R., Anyah, R., 2013. Statistical downscaling and bias correction of climate model outputs for climate change impact assessment in the U.S. northeast. Global and Planetary Change 100, 320–332.

Anh H. Reynolds, 2019. Convolutional Neural Networks (CNNs).
URL `https://anhreynolds.com/blogs/cnn.html`

Baart, F., 2013. Confidence in coastal forecasts. Ph.D. thesis, TU Delft.

Baño-Medina, J., Manzanas, R., Gutiérrez, J. M., 2020. Configuration and Intercomparison of Deep Learning Neural Models for Statistical Downscaling. Geoscientific Model Development 13 (4).

Bastola, S., Misra, V., 2014. Evaluation of dynamically downscaled reanalysis precipitation data for hydrological application. Hydrological Processes 28 (4), 1989–2002.

Belz, J., 2010. The flow regime of the River Rhine and its tributaries in the 20th century - analysis, changes, trends. Hydrologie und Wasserbewirtschaftung (54), 4–17.

Bowden, G. J., Maier, H. R., Dandy, G. C., 2002. Optimal division of data for neural network models in water resources applications. Water Resources Research 38 (2), 2–1–2–11.

Chen, J., Brissette, F., Chen, H., 2018. Using reanalysis - driven regional climate model outputs for hydrology modelling. Hydrological processes 32, 3019–3031.

Chen, Y., Jiang, H., Li, C., Jia, X., Ghamisi, P., 2016. Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks. ITEEE transactions on geoscience and remote sensing 54 (10), 6232–6251.

Cheng, J., Kuang, Q., Shen, C., Liu, J., Tan, X., Liu, W., 2020. ResLap: Generating High-Resolution Climate Prediction through Image Super-Resolution. IEEE Access 8, 39623–39634.

De Bruin, H. A., Trigo, I. F., Bosveld, F. C., Meirink, J. F., 2016. Thermodynamically based model for actual evapotranspiration of an extensive grass field close to FAO reference, suitable for remote sensing application. Journal of Hydrometeorology 17 (5), 1373–1382.

Dee, D. P., Uppala, S. M., Simmons, A. J., Berrisford, P., Poli, P., Kobayashi, S., Andrae, U., Balmaseda, M. A., Balsamo, G., Bauer, P., Bechtold, P., Beljaars, A. C., van de Berg, L., Bidlot, J., Bormann, N., Delsol, C., Dragani, R., Fuentes, M., Geer, A. J., Haimberger, L., Healy, S. B., Hersbach, H., Hólm, E. V., Isaksen, L., Kållberg, P., Köhler, M., Matricardi, M., Mcnally, A. P., Monge-Sanz, B. M., Morcrette, J. J., Park, B. K., Peubey, C., de Rosnay, P., Tavolato, C., Thépaut, J. N., Vitart, F., 2011. The ERA-Interim reanalysis: Configuration and performance of the data assimilation system. Quarterly Journal of the Royal Meteorological Society 137 (656), 553–597.

Dong, C., Loy, C. C., He, K., 2016. Image Super-Resolution Using Deep Convolutional Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence 38 (2).

Eilander, D., Verseveld, W. V., Yamazaki, D., Weerts, A., Winsemius, H. C., Ward, P. J., 2020. A hydrography upscaling method for scale parametrization of distributed hydrological models. Hydrology and Earth System Sciences (November), 1–34.

Fick, S. E., Hijmans, R. J., 2017. WorldClim 2: new 1-km spatial resolution climate surfaces for global land areas. International Journal of Climatology 37 (12), 4302–4315.

Fowler, H. J., Blenkinsop, S., Tebaldi, C., 2007. Linking climate change modelling to impacts studies : recent advances in downscaling techniques for hydrological modelling. International Journal of Climatology 27, 1547–1578.

Gao, L., 2013. Validation and Statistical Downscaling of ERA-Interim Reanalysis Data for Integrated Applications. Thesis, 167.

Gerlitz, L., Conrad, O., Böhner, J., 2015. Large-scale atmospheric forcing and topographic modification of precipitation rates over High Asia - A neural-network-based approach. Earth System Dynamics 6 (1), 61–81.

Giorgi, F., Hewitson, B., Christensen, J., Hulme, M., on Storch, H., Whetton, P., Jones, R., Mearns, L., Fu, C., 2001. Regional Climate Information – Evaluation and Projections. Tech. rep.

Goodess, C. M., Anagnostopoulou, C., Bárdossy, A., Frei, C., Harpham, C., Haylock, M. R., Hundecha, Y., Maheras, P., Ribalaygua, J., Schmidli, J., Schmith, T., Tolika, K., Tomozeiu, R., Wilby, R. L., 2012. An intercomparison of statistical downscaling methods for Europe and European regions – assessing their performance with respect to extreme temperature and precipitation events 2005 (published as CRU RP11 in 2012) Climatic Research Unit School of Enviro. Tech. rep.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative adversarial nets. Advances in Neural Information Processing Systems 3 (January), 2672–2680.

Gupta, H. V., Kling, H., Yilmaz, K. K., Martinez, G. F., 2009. Decomposition of the mean squared error and NSE performance criteria: Implications for improving hydrological modelling. Journal of Hydrology 377 (1-2), 80–91.

Hay, L. E., Clark, M. P., 2003. Use of statistically and dynamically downscaled atmospheric model output for hydrologic simulations in three mountainous basins in the western United States. Journal of Hydrology 282 (1-4), 56–75.

Hersbach, H., Bell, B., Berrisford, P., Hirahara, S., Horányi, A., Muñoz-Sabater, J., Nicolas, J., Peubey, C., Radu, R., Schepers, D., Simmons, A., Soci, C., Abdalla, S., Abellan, X., Balsamo, G., Bechtold, P., Biavati, G., Bidlot, J., Bonavita, M., De Chiara, G., Dahlgren, P., Dee, D., Diamantakis, M., Dragani, R., Flemming, J., Forbes, R., Fuentes, M., Geer, A., Haimberger, L., Healy, S., Hogan, R. J., Hólm, E., Janisková, M., Keeley, S., Laloyaux, P., Lopez, P., Lupu, C., Radnoti, G., de Rosnay, P., Rozum, I., Vamborg, F., Villaume, S., Thépaut, J. N., 2020. The ERA5 global reanalysis. Quarterly Journal of the Royal Meteorological Society 146 (730), 1999–2049.

Hinton, G., Salakhutdinov, R., 2006. Reducing the dimensionality of data with neural networks. Science 313, 4.

Huang, H., Li, Z., He, R., Sun, Z., Tan, T., 2018. Introvae: Introspective variational autoencoders for photographic image synthesis. Advances in Neural Information Processing Systems 2018-Decem (NeurIPS), 52–63.

Hwang, S., Graham, W. D., Adams, A., Geurink, J., 2013. Assessment of the utility of dynamically-downscaled regional reanalysis data to predict streamflow in west central Florida using an integrated hydrologic model. Regional Environmental Change 13, 69–80.

Imhoff, R. O., van Verseveld, W. J., van Osnabrugge, B., Weerts, A. H., 2020. Scaling Point-Scale (Pedo)transfer Functions to Seamless Large-Domain Parameter Estimates for High-Resolution Distributed Hydrologic Modeling: An Example for the Rhine River. Water Resources Research 56 (4), 1–28.

IPCC, 2013. Climate Change 2013: The Physical Science Basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change. Tech. rep., Intergovernmental Panel on Climate Change, Cambridge.

Johnson, J., Alahi, A., Fei-Fei, L., 2016. Perceptual losses for real-time style transfer and super-resolution. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9906 LNCS, 694–711.

Kala, S., 2020. Autoencoders: Unsupervised Artificial Neural Networks(ANN).
URL `https://medium.com/analytics-vidhya/autoencoders-unsupervised-artificial-neural- networks-ann-a276089f0053`

Karras, T., Aila, T., Laine, S., Lehtinen, J., 2018. Progressive growing of GANs for improved quality, stability, and variation. In: 6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings. pp. 1–26.

Kim, D.-I., Kwon, H.-H., Han, D., 2018. Exploring the Long-Term Reanalysis of Precipitation and the Contribution of Bias Correction to the Reduction of Uncertainty over South Korea: A Composite Gamma-Pareto Distribution Approach to the Bias Correction. Hydrology and Earth System Sciences (February), 1–53.

Kingma, D. P., Ba, J. L., 2015. Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings. pp. 1–15.

Kingma, D. P., Welling, M., 2014. Auto-encoding variational bayes. 2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings (Ml), 1–14.

Knoben, W. J. M., Freer, J. E., Woods, R. A., 2019. Technical note: Inherent benchmark or not? Comparing Nash-Sutcliffe and Kling-Gupta efficiency scores. Hydrology and Earth System Sciences Discussions (July), 1–7.

Lecun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521, 436–444.

Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A. P., Tejani, A., Totz, J., Wang, Z., Others, 2017. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4681–4690.

Lukas, P. J. W. R., 1992. The Coupled Ocean- Atmosphere Response Experiment. Bull. Amer. Meteor. Soc. 73 (9), 1377–1416.

Ma, X., Li, Y., Huang, H., Luo, M., He, R., 2019. Exploiting Style and Attention in Real-World Super-Resolution.
URL http://arxiv.org/abs/1912.10227

Maier, H. R., Dandy, G. C., 2000. Neural networks for the prediction and forecasting of water resources variables: A review of modelling issues and applications. Environmental Modelling and Software 15 (1), 101–124.

Malardel, S., Wedi, N., Deconinck, W., Diamantakis, M., Kühnlein, C., Mozdzynski, G., Hamrud, M., Smolarkiewicz, P., 2016. A new grid for the IFS. Ecmwf (146), 23–28.

Mao, X.-J., Shen, C., Yang, Y.-B., 2016. Image Restoration Using Convolutional Auto-encoders with Symmetric Skip Connections, 1–17.

Maraun, D., Wetterhall, F., Ireson, A., Chandler, R., Kendon, E., Widmann, M., Brienen, S., Rust, H. W., Sauter, T., Themeßl, M., Venema, V. K. C., Chun, K. P., 2010. Precipitation downscaling under climate change: Recent developments to bridge the gap between dynamical models and the end user. Reviews of Geophysics 48 (3).

Mendes, D., Marengo, J. A., 2010. Temporal downscaling: A comparison between artificial neural network and autocorrelation techniques over the Amazon Basin in present and future climate change scenarios. Theoretical and Applied Climatology 100 (3), 413–421.

Nash, J. E., Sutcliffe, J. V., 1970. River flow forecasting through conceptual models part I - A discussion of principles. Journal of Hydrology 10 (3), 282–290.

Poli, P., Hersbach, H., Dee, D. P., Berrisford, P., Simmons, A. J., Vitart, F., Laloyaux, P., Tan, D. G., Peubey, C., Thépaut, J. N., Trémolet, Y., Hólm, E. V., Bonavita, M., Isaksen, L., Fisher, M., 2016. ERA-20C: An atmospheric reanalysis of the twentieth century. Journal of Climate 29 (11), 4083–4097.

Rodrigues, E. R., Oliveira, I., Cunha, R. L. F., Netto, M. A. S., 2018. DeepDownscale : a deep learning strategy for high-resolution weather forecast.

Saavedra, M., Junquas, C., Espinoza, J. C., Silva, Y., 2020. Impacts of topography and land use changes on the air surface temperature and precipitation over the central Peruvian Andes. Atmospheric Research 234 (April 2019), 104711.

Saha, S., Moorthi, S., Pan, H. L., Wu, X., Wang, J., Nadiga, S., Tripp, P., Kistler, R., Woollen, J., Behringer, D., Liu, H., Stokes, D., Grumbine, R., Gayno, G., Wang, J., Hou, Y. T., Chuang, H. Y., Juang, H. M. H., Sela, J., Iredell, M., Treadon, R., Kleist, D., Van Delst, P., Keyser, D., Derber, J., Ek, M., Meng, J., Wei, H., Yang, R., Lord, S., Van Den Dool, H., Kumar, A., Wang, W., Long, C., Chelliah, M., Xue, Y., Huang, B., Schemm, J. K., Ebisuzaki, W., Lin, R., Xie, P., Chen, M., Zhou, S., Higgins, W., Zou, C. Z., Liu, Q., Chen,

Y., Han, Y., Cucurull, L., Reynolds, R. W., Rutledge, G., Goldberg, M., 2010. The NCEP climate forecast system reanalysis. Bulletin of the American Meteorological Society 91 (8), 1015–1057.

Schellekens, J., 2019. The wflow_sbm Model. URL https://wflow.readthedocs.io/en/latest/wflow_sbm_old.html

Shi, W., Caballero, J., Husz, F., Totz, J., Aitken, A. P., Bishop, R., Rueckert, D., Wang, Z., 2016. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. In: IEEE conference on computer vision and pattern recognition. pp. 1874–1883.

Smith, R., 1979. The influence of mountains on the atmosphere. Advances in Geophysics 21, 87–233.

Ter Maat, H. W., Moors, E. J., Hutjes, R. W., Holtslag, A. A., Dolman, A. J., 2013. Exploring the impact of land cover and topography on rainfall maxima in the Netherlands. Journal of Hydrometeorology 14 (2), 524–542.

Thara, T. D., Prema, P. S., Xiong, F., 2019. Auto-detection of epileptic seizure events using deep neural network with different feature scaling techniques. Pattern Recognition Letters 128, 544–550.

Tran Anh, D., Van, S. P., Dang, T. D., Hoang, L. P., 2019. Downscaling rainfall using deep learning long short-term memory and feedforward neural network. International Journal of Climatology 39 (10), 4170–4188.

Vu, M. T., Aribarg, T., Supratid, S., Raghavan, S. V., Liong, S. Y., 2016. Statistical downscaling rainfall using artificial neural network: significantly wetter Bangkok? Theoretical and Applied Climatology 126 (3-4), 453–467.

Wang, Z., Simoncelli, E., Bovik, A. C., 2003. Mssim. New York 2, 9–13.

Werner, A. T., Cannon, A. J., 2016. Hydrologic extremes - An intercomparison of multiple gridded statistical downscaling methods. Hydrology and Earth System Sciences 20 (4), 1483–1508.

Wilby, R. L., Charles, S. P., Zorita, E., Timbal, B., Whetton, P., Mearns, L. O., 2004. Guidelines for Use of Climate Scenarios Developed from Statistical Downscaling Methods. Tech. rep.

Wilby, R. L., Hay, L. E., Gutowski, W. J., Arritt, R. W., Takle, E. S., Pan, Z., Leavesley, G. H., Clark, M. P., 2000. Hydrological responses to dynamically and statistically downscaled climate model output. Geophysical Research Letters 27 (8), 1199–1202.

Xie, Y. O. U., Franz, E., Chu, M., Thuerey, N., 2018. tempoGAN : A Temporally Coherent , Volumetric GAN for Super-resolution Fluid Flow. ACM Transactions on Graphics 37 (4).

Xiong, Y., Guo, S., Chen, J., Deng, X., Sun, L., Zheng, X., Xu, W., 2020. Improved SRGAN for Remote Sensing Image Super-Resolution Across Locations and Sensors. Remote Sensing 12 (8).

Yeung, S., 2017. Lecture 13: Generative models. Stanford University School of Engineering.

Zhang, A., Lipton, Z. C., Li, M., Alexander J. Smola, 2020. Dive Into Deep Learning.