MASTER THESIS



Short term solar irradiance time-series forecasting with machine learning

Computing Science

Supervisors: Prof. dr. A.A. Salah Prof. dr. W. van Sark Lennard Visser

Author: Niels Yannick Hendrikx www.nielshendrikx.nl

May 7, 2020

Contents

Glossary 1							
Ac	ronyı	ms	2				
1	Intro	roduction					
2	Lite	rature Study	9				
	2.1	.1 Solar forecasting					
	2.2	2 Cloud pixel detection					
		2.2.1 Red-Blue Ratio	10				
		2.2.2 Red-Blue Red-Green Ratio	11				
		2.2.3 Normalized Red-Blue Ratio	11				
	2.3 Prediction from onsite sources						
		2.3.1 Regression with features	12				
		2.3.2 Optical flow	13				
		2.3.3 Convolutional neural networks	14				
		2.3.4 Weather pattern classification model	14				
		2.3.5 Perez conversion model	15				
	2.4	2.4 Predicting with external data					
	2.5	Pre-processing	16				
	2.6	Performance evaluation	16				
	2.7	Summary	17				
3	The	oretical Background	19				
	3.1	Time-series forecasting	19				
	3.2	Optical flow	20				
		3.2.1 Sparse optical flow	21				

		3.2.2	Dense optical flow	21			
	3.3	Harris corner detection algorithm					
	3.4	Sun positioning					
	3.5	Neural networks					
		3.5.1	Artificial Neural Networks	25			
		3.5.2	Activation functions	26			
		3.5.3	Back-propagation algorithm	26			
		3.5.4	Convolutional Neural Networks	28			
		3.5.5	Recurrent Neural Networks	29			
		3.5.6	Long short-term memory	29			
	3.6	Rando	m Forests	30			
		3.6.1	Decision tree	30			
4	Proł	olem De	escription	32			
•	4 1	Proble	em definition	32			
	4.2						
	1.2	4.2.1 Sensor setup					
		4.2.2	Data	33			
		423	Hardware experiments	39			
		424	Data-set	39			
		4.25 Train validation and test data					
	4.3						
	1.0	431	Models	40			
		432	Data	41			
		1.5.2		- 11			
5	Metl	hodolog	39	42			
	5.1	Experi	ment setup	42			
	5.2	Pre-processing					
		5.2.1	Numerical data	43			
		5.2.2	Image data	43			
	5.3	Experiments					
	5.4	Model	s	44			
		5.4.1	Model-selection	44			
		5.4.2	Numerical models	44			

		5.4.3	CNN-flow	45
		5.4.4	Persistence Model	45
	5.5	Perfor	mance evaluation	45
		5.5.1	Statistical significance	46
6	Imp	lement	ation	49
	6.1	Data-f	rame	49
	6.2	Nume	rical models	49
		6.2.1	Random Forests	52
		6.2.2	Artificial neural network	52
		6.2.3	Long short term memory	53
	6.3	Flow -	Convolutional neural network	53
7	Resu	ılts		54
	7.1	Model	selection	55
		7.1.1	Random forests	55
		7.1.2	Artificial neural network	55
		7.1.3	Long short-term memory	58
		7.1.4	Flow Model	58
7.2 Test-set				
		7.2.1	Random forests	61
		7.2.2	Artificial neural networks	63
		7.2.3	Long short-term memory	65
		7.2.4	Valuable features	66
		7.2.5	Different weather circumstances	66
		7.2.6	Computation time	67
		7.2.7	Additional training data	67
		7.2.8	Statistical significance	68
		7.2.9	Model comparison	71
8	Con	clusion	S	74
	8.1	Discus	ssion	74
	8.2	usions	75	
8.3 Future work				

A	Appendix						
	A.1	Random forests	77				
	A.2	Artificial neural networks	80				
	A.3	Long short-term memory	82				

Abstract

The intermittent nature of solar irradiance caused by clouds results in rapid short term fluctuations in the power output of photovoltaics(PV)-systems. These fluctuations make this data source difficult to integrate with the grid. This study focuses on short term (0-20 minutes) forecasting of global horizon irradiance (GHI) using all sky imager (ASI) images and numerical data as input. This numerical data is locally obtained (Almeria, Spain) from June to December 2019. Installed sensors measure humidity, temperature and GHI every 15 seconds.

Multiple approaches and models are proposed and compared with each other. We track clouds with optical-flow and generate a future representation of the sky. This future image is the input for a Convolutional neural network (CNN), which is trained to predict GHI. Secondly, extracted information from images and additional numerical data are plugged into different classifier models, including random forests (RF), multi-layer perceptrons (MLP), and long short-term memory networks (LSTM). We tried multiple subsets of features and multiple sequence lengths to find a good model. We categorized the features in 3 subsets and a combination of these are used as input. Models are trained to predict GHI from 0 to 20 minutes ahead, with a step size of 1 minute. We tried improving predictions by making the models predict the clear sky index (CSI), where the GHI can be derived using the Perez conversion model. We implemented this approach into the models, there was no improvement.

In solar forecasting the (smart-)persistence model is a common baseline. Persistence forecasts the last observed GHI. Smart-persistence calculates the future direct normal irradiance (DNI) and combines this with the last observed CSI to predict the expected GHI. We categorize days in cloudy (CSI < 0.25), partially cloudy (CSI > 0.25, CSI < 0.75) and sunny (CSI > 0.75).

We implement multiple models and show that LSTM performs best, which agrees with it's know advantage of dealing with sequence problems. We learn what features are valuable and the amount of minutes prior prediction moment that data has predictive value. The best model(s) are able to predict statistically significantly better than the baseline for sunny and cloudy weather.

Glossary

Azimuth angle The azimuth angle is the compass direction from which the sunlight is coming.. 24, 38

1

- **CSI** The clearness index is a measure of the clearness of the atmosphere. It can be calculated by the ratio of GHI and GHI_{clr} .. 15
- **Declination angle** The Sun declination is the angle between the sun rays and the plane of Earth's equator. It is measured in degrees north and south of the celestial equator. 24
- **Persistence model** According to persistence. The production in the future is equal to current production. 42, 46
- **Smart-persistence model** Equal to persistence but considering change in position of the sun [29]. 42, 46
- **Solar altitude angle** The solar altitude angle is the angle between the sun's rays and a horizontal plane.. 24, 38
- **Solar elevation angle** The solar elevation angle is the angle between the horizon and the sun's disc center. 16, 38

Solar incidence angle The angle between the sun's rays and the normal on a surface.. 15, 38

Solar zenith angle The angle between the zenith and the center of the sun's disc.. 15, 16, 38

Acronyms

- GHI_{clr} Clear sky GHI. 1, 12–14, 38, 54
- ANN Artificial Neural Network. 13, 16, 25, 29, 44, 49, 55, 59, 71
- ASI All Sky Imager. 6, 7, 11–13, 16, 17, 24, 38, 76
- CNN Convolutional neural network. 1, 7, 14, 25, 28, 44, 45, 53
- **CSI** Clearness index. 12, 14, 38, 39, 41, 43, 45, 75
- DHI Diffuse Horizontal Irradiance. 4, 5
- DNI Direct Normal Irradiance. 4, 5, 13
- GHI Global horizon irradiance. 1, 4–7, 9, 10, 12–17, 32, 34, 38, 40–43, 45, 67, 76
- LSTM Long short-term memory. 25, 29, 44, 49, 55, 58, 59
- MAE Mean Absolute Error. 17, 45
- MAPE Mean Absolute Percentage Error. 45
- MLP Multilayer perceptron. 13
- MSE Mean Square Error. 31
- NWP Numerical weather prediction. 6
- **RF** Random Forests. 44, 49, 55, 67, 71
- **RMSE** Root Mean Square Error. 17, 45, 46, 48, 59
- RNN Recurrent neural network. 29
- SS Skill Score. 45, 46

Acronyms

- SVM Support vector machine. 7, 12, 14, 16
- SVR Support vector regression. 16
- **SZA** Solar zenith angle. 5, 6, 15, 45
- **TSI** Total solar irradiance. 4, 15



Chapter 1

Introduction

One of the social challenges of today is an energy transition from mostly fossil fuels to renewable energy. Stated in the Paris agreement, by 2030 there should be a substantial increase in the share of renewable energy in the global energy mix. The goal is that by 2050 100% of the energy will come from a renewable source [43]. One of today's options for an alternative energy resource is solar energy.

The earth receives approximately $1.8 * 10^{11} MW$ power from solar irradiance directly from the sun at an instance [23]. This is more than earths current annual total energy consumption [51].

But solar energy is very dependent on weather conditions. Specifically, the amount of solar irradiance that reaches the surface varies over time. As weather or solar irradiance is not something we can control, the energy provided by a solar energy installation is uncertain. Variation and uncertainty as an energy source make it difficult to integrate this particular resource to the power grid. The problem occurs when keeping balance between generation, loads and losses. Knowing how much PV power will be produced will reduce operational costs of these PV power plants [35].

Solar irradiance is the amount of power received from the sun as electromagnetic radiation, this is measured in W/m^2 . Solar irradiance is categorized in multiple types of irradiance [42] (see fig. 1.1). Total solar irradiance (TSI) is a measure of solar irradiance over all wavelengths incident on the Earth's upper atmosphere.

We define Direct Normal Irradiance (DNI) as the solar radiation measured at the surface of the earth perpendicular to the sun, this is the irradiance directly from the sun [42]. Following, Diffuse Horizontal Irradiance (DHI) is the radiation measured on a horizontal surface on Earth, but this irradiances source is radiance scattered by the atmosphere. Solar irradiance can also reflect by (nearby) objects like the ground or buildings, we call this reflected radiation. Solar panels tend to be tilted away from the reflected irridiance, so it rarely has contribution in the total radiation received by their surface. Global horizon irradiance (GHI) is the total irradiance

4

CHAPTER 1. INTRODUCTION

from the sun (after accounting for the Solar zenith angle (SZA)). It is defined to be the sum of DHI, DNI and reflection:

$$GHI = DHI + DNI * \cos(sza) \tag{1.1}$$

where:

sza = solar zenith angle.



Figure 1.1: Visualization of different types of solar irradiance

The energy output of a solar panel is mostly affected by solar irradiance, which varies predicatively. Unfortunately, the amount of solar irradiance reaching the surface (a solar panel) does not vary in a predictable way. GHI depends a lot on the movement and location of the clouds. When the sky above the solar panel(s) becomes more cloudy the solar irradiance reaching the solar panel drops, which affects the total power output of a photovoltaics(PV)-system [4].

The fluctuation and uncertainty of PV output make accurate GHI prediction of great value. Accurate predictions would directly result in a safer and easier grid management. Which makes GHI predictions important for both economical and sustainable reasons [49].

From this information follows the question "for what future horizon should you predict the GHI?". Literature classifies the following prediction horizons [4]: *Short term forecasting*, also known as *intra-hour* or *now-casting*. These methods cover forecasting from one second up to an hour. This time-span is important to assure grid quality and stability, because in this time frame the operators need to correctly schedule reserves and demand response to the grid [4]. The next forecast horizon is called *intra-day* forecasts, which covers one to six hours. These horizons and are important for grid operators that control different load zones or who trade outside their area [4].

5

The resulting forecast horizons are six hours to one day ahead and two days or more. These horizons are important for planning and trading. For example, this information is valuable when to schedule maintenance efficiently. Preferably, this would be at a moment when the expected production is low [45]. In this study we will set the scope to short term forecasting.

In the past there were multiple methods proposed for short term GHI forecasts. The choice of method depends on the available data and preferred prediction horizon. Figure 1.2 displays which data are used in general for what specific forecast horizon.

It is possible using sky-images to predict from one second up to 30 minutes. This method is used to track clouds and determine the GHI. The limitations are that there is little research compared to other methods and a camera (on location) is needed [4]. A big advantage of using all All Sky Imager (ASI) images is that is able to predict very short term fluctuation due to low clouds and large clouds variability. Capturing these sudden changes is nearly impossible with meteorological records or Numerical weather prediction (NWP) approaches. Where ASI methods are good to capture sudden changes their accuracy immediately decreases over time, after 30 minutes most information from an image is outdated [14]. Quite similar are predictions using satellites, as vectors and information is derived from images. An advantage is that there will be available images for most places in the world, except for the North and South pole. This method provides better short term performance than NWP models, but it is less accurate than ASI models [4, 5].

Another approach is climatology using past weather records. It tries to identify climate trends like seasons and averages, this makes this method static. An example is the persistence model (see section 5.4.4) which predicts the same weather as last observed. An advantage is that these methods are easy to implement. A downside is that they perform bad on sudden fluctuation as the predictions are based on past events [29].

Next, there is Numerical weather prediction (NWP) models using the current weather data as input. An example would be wind, humidity and surface pressure. This is input for mathematical models to simulate processes occurring in the atmosphere. Advantages are that NWP models do not need historical data and they have a very broad prediction horizon from approximately 10 minutes up to 10 days [25, 39]. Although, most often these models are used from six hours to several days [32]. A downside is that these models are hard to implement as one needs physical knowledge to simulate the atmosphere.

Then there are statistical learning models which use historical data to find relations. Where, later direct observations can be translated to predictions. An example, Cristian Crisosto et al. (2017) proposed a method where sky images are combined with external data like SZA and average GHI [3]. An advantage is that statistical models have a broad prediction horizon from several minutes up to a month [4]. In this study is focused on short term GHI predictions using historical and realtime ASI and external data. A disadvantage is that you need access too much accurate historical data.



Figure 1.2: Distribution of different techniques and inputs with respect to their forecast horizon. Image taken from [4].

Multiple studies use feature extraction of ASI. In this case feature extraction is the process of transforming raw pixel values (from the ASI), to more meaningful information. Examples of features are amount of cloud pixels or brightness. These features are suitable input for statistical models like regression or SVM. These models are trained to predict GHI [21, 19].

In 2017 Y. Ai et al. used SVM to predict GHI under clear sky circumstances [2]. While combining this with ASI and optical flow (movement of pixels between frames, see section 3.2) is predicted how much the sun will be blocked. From this a near future GHI is estimated. In 2018 a (Convolutional neural network (CNN)) method has been used. Its estimates the future GHI by taking a generated image is input. This image is constructed with optical flow [6].

In this study we focus on short-term forecast (0 to 20 minutes into the future), which assures grid quality and stability. Current research on this prediction horizon is limited. Additionally, we combine sensor data and image data, which is another contribution. In this study we compare multiple machine learning models which each other. The goal is to find what models and data are suitable for short-term forecasts. We formulate this problem as a time-series forecasting problem (see section 3.1). We use machine learning models random forests (RF), artificial neural networks (ANN) and long short-term memory (LSTM). Additionally, this problem can also be solved by estimating a future image with optical flow and estimating the GHI from this future image. For this we use a convolutional neural network (CNN).

The used data set contains data from July to December, giving not the opportunity to observe how models perform in other months. Additionally, the location Almeria in south Spain has limited days which we categorize as cloudy (CSI < 0.25). A solution would be to conduct similar experiments another region with preferred climate.

This document will first cover relevant literature, followed by a theoretical background. In chapter four we give the problem description and details about used hardware. This chapter also defines the goals and research question of this study. Next, in chapter five we formulate the methodology. First, discussing the experiment setup. Followed by, the experiments, details of models and performance evaluation. Chapter six highlights how the models are implemented and the manner we searched for hyper-parameters. Then in chapter seven we show all the results. This is separated in two sections, in section one we show the results of the model selection. In section two we show the results on the test set. Finally, in the last chapter we discuss the results and conclude.

Chapter 2

Literature Study

In this chapter we describe an overview of existing literature on GHI forecasting. First of all, we highlight what the challenges are in this field. Followed by, the current methods to tackle these problems.

2.1 Solar forecasting

The main question in this field is 'How much solar irradiance will be observed at time *t* at some location?'. Where time *t* is somewhere in the future. The amount of time in the future is also know as the prediction horizon [4]. Solar irradiance is measured in kilowatt-hours per squared meter. An alternative way is forecasting how much power output a photovoltaic(PV)-system delivers some time in the future, this is measured in kilowatt-hour(kWh). The output of a PV-system is determined by the amount of GHI, additionally temperature has a effect on the efficiency of the panels [27]. Panels are less efficient with higher temperatures.

As discussed in the introduction the are multiple models. These models need some input and/or data. Literature classifies this in 3 categories [1]:

- 1. NWP models that use meteorological and geographical parameters.
- 2. Endogenous variables, models that only consider historic solar irradiance.
- 3. Exogenous, models that consider other variables in addition to endogenous variables.

These variables are location bound, so these variables need to be observed by sensors installed at locations. In the last 10 years research has been done all over the world (for example Canada, Greece, China, Australia, USA). However, the historical data is usually not public.

As discussed in the introduction, literature classifies the following prediction horizons [49]:

• Short term, forecasting up to one hour.

9



- Intra-day, one hour to 6 hours.
- Day-ahead, six hours to three days ahead.

In past research most common resolutions are 1 hour or day into the future [49]. A gap in these studies is that most of them have a single prediction horizon, so a question remains how models perform over different prediction horizons.

The fluctuation and uncertainty of PV output make accurate GHI prediction of great value. Accurate predictions would directly result in a safer and easier grid management. Which makes GHI predictions important for both economical and sustainable reasons [49]. In example grid management would include turning on (fossil fuel based) generators to when power output is too low [16]. It is not possible to put a number on the direct financial consequences of forecasting. This is due the different variables per region, for example energy prices differ per country. Additionally, in stable weather forecast would be less beneficial compared to weather with much GHI fluctuations.

2.2 Cloud pixel detection

Clouds are most important when predicting GHI [2, 47]. When forecasting while using images you want to extract information from relevant images. As in this study images of the sky are used as input, we want to know if a pixel in an image is part of a cloud or not.

There are multiple ways to determine if a pixel belongs to a cloud. Chauvin et al. (2015) identified three different categories for the existing cloud detection algorithms [12]: threshold methods, neural networks and dedicated algorithms. In this study the scope is set to only threshold methods. Moreover, a neural network based approach is not possible since the dataset does not include a ground truth on clouded and non-clouded pixels, which is needed to train a neural network. Also, dedicated algorithms will be too computationally expensive and will require dedicated hardware to detect haze, thin clouds and opaque clouds.

Threshold algorithms do a calculation on a pixel, the output is compared with some predefined threshold. As a result, the pixel is classified as cloud or sky. The methods often used in GHI forecasting models are listed below.

2.2.1 Red-Blue Ratio

If The Red-Blue Ratio (RBR) (equation: 2.1) is higher than a fixed threshold it considered a cloud pixel. Usually this threshold is set 0,6-0,8 [26].

$$RBR = \frac{R}{B} \tag{2.1}$$

2.2.2 Red-Blue Red-Green Ratio

The Red-Blue Red-Green Ratio (BRBG) performs better than RBR [46]. The pixels with a higher BRBG (equation: 2.2) then some threshold Th_0 are considered sky. Usually Th_0 is set to a fixed value.

$$BRBG = \frac{B}{R} + \frac{B}{G}$$
(2.2)

2.2.3 Normalized Red-Blue Ratio

Normalized Red-Blue Ratio (NRBR) (equation 2.3) its threshold can be fixed, but QingYong Li et al. (2011) proposed a better performing method with an adaptive threshold using minimum cross-entrophy (MCE) [31]. The MCE thresholding algorithm selects a threshold by minimizing the cross entropy between the original image and its segmented image. The segmented image is calculated by the mean and the standard deviation of the normalized $\frac{B}{P}$ ratio values.

$$NRBR = \frac{R-B}{R+B}$$
(2.3)

Figure 2.1: Image at 21 august 2019 at 12:00:00 PM. In this image no cloud pixel method is applied.



Figure 2.2: Image at 21 august 2019 at 12:00:00 PM. Cloud pixels in green detected with RBR, th = 0.8.

2.3 Prediction from onsite sources

In this study the definition of an onsite (data) source is: all sources that need to be installed locally (on the ground). Examples are sensors for humidity, temperature, irridiance and a camera (ASI). Excluded is the data-source satellites, because they cover a lot of area and are not ground based.



Figure 2.3: Image at 21 august 2019 at 12:00:00 PM. Cloud pixels in green detected with BRBG, th = 2.



Figure 2.4: Image at 21 august 2019 at 12:00:00 PM. Cloud pixels in green detected with NRBR, th = 0.8.

2.3.1 Regression with features

An approach for forecasting GHI is extracting features from a range of consecutive ASI images. Common used features are: The number of cloud pixels in a frame, using a cloud pixel detect method (section 2.2). Amount of cloud edges (section 3.3). The brightness levels, from the normalized pixel values. Historical GHI (4.2), GHI_{clr} (4.2) and CSI derived from GHI and GHI_{clr} prior to the prediction time. Features are used as input for support verctor regression (SVR) or linear regression in past studies [21, 19]. These models can be trained to predict the GHI on some future horizon. The prediction horizon of these models in these particular studies differed from 1 minute to up to 1 hour.

Chia-Lin Fu et al. (2013) proposed a regression model based on feature extraction. Their data-set exist out of 4 weeks, with an image sampling level of 10 seconds. 1 month is a limited dataset, but there are more disadvantages. First of all, the resolution of 640x480 is quite limited (with respect to the setup in this study with resolution 1920 * 1920). Secondly, there is no baseline measurement like persistence, which makes it hard to compare the performance of this model. In this study they predict t = 5, 10, 15 minutes into the future using features of the prior p = 5, 10, 15 minutes. The best results are achieved with t, p = 5 (minutes). Interesting is that when using fewer features better results are achieved [21], unfortunately no explanation is given. This raises a question "What is an ideal time frame prior to prediction for feature extraction?".

Above described method is very similar to a method proposed by Cong Feng et al. (2017). However, Cong Feng et al. (2017) used a SVM as model. The limitations of this study are the static feature time-spam (4 hours) and prediction horizon of 1 hour [19]. What makes this study state-of-the-art is that they achieved better result by splitting the data set in different types of days (see 2.3.4) and trained an individual model for each type of day. Pierrick Bruneau et al. (2018) focused on predictions from 1-hour into the future. As input they had acces to multiple onsite sensors which provided: GHI, Direct Normal Irradiance (DNI), atmospheric air pressure, temperature, humidity and wind direction/speed. The data-set contained data observed over two years, with a sample frequency of 1 minute. Furthermore, the data-set contained data from 5 different locations. Used models are Auto Regressive Integrated Moving Average (ARIMA), Multilayer perceptron (MLP) and XGBoost. ARIMA is a regression model very common in time-series forecasting [49]. As MLP they used an single hidden layer implementation of ANN, XGBoost is an implementation of decision trees with boosted gradients [13]. They concluded that 'correlations between sites shows that data sets for the 5 sites cannot be trivially merged prior to learning prediction models'. Additionally, XGBoost performed best, but the question remains if a more complex ANN would performed better. For future work they propose adapting convolutional neural networks and recurrent neural networks models, because these models perform well on sequential problems [10].

2.3.2 Optical flow

With optical flow (discussed in 3.2) the velocity of clouds can be determined. Yiyang Ai et al. (2017) used the Lukas-Kanade flow method (section 3.2). This method predicts GHI with a future horizon of 1,2,3 minutes ahead. The input is ASI images of 3456×3456 . Their data exists out of a images shot in 1 month between 10 am and 2 pm with a sample time of every 30 seconds. The intuition behind their method is that the ASI is split in blocks of size 400x400 (The smallest possible such that the sun is able to fit in the square). Then with optical flow is approximated what block will be at the same position as the sun. In this block the amount of cloud pixels are calculated, which gives variable *C* the fraction of cloud pixels. They predict the GHI with equation 2.4.

$$G'(t+1) = G(t+1) - 689.9 * C(t+1)$$
(2.4)

where:

G' = the predicted GHI.

 $G = \text{Clear sky GHI} (GHI_{clr}).$

689.9 =output of SVR.

C = fraction of cloud pixels.

Missing in this study is what time between frames is used to determine the velocity of clouds. A skill-score is given (see section5.5), but implemented model and persistence may predict different on weather circumstances, so comparing with this in not reasonable. Additionally, they test on a single day, the question remains how it performs on other types of days.

2.3.3 Convolutional neural networks

S. Tiwari et al. (2018) proposed to use optical flow (see section 3.2) to generate an ASI image 10 minutes into the future [47]. This CNN was trained on the original (resized to 800x800) images, which contain GHI as ground-truth labels.

The most common tool for analysis of visual imagery using deep learning is a CNN. It consists of an input layer, an output layer and multiple hidden layers. The hidden layers perform a convolution operation on the input data (an image) sending the output to the next subsequent layer. The depth of a CNN is very important and it can be crucial for the accuracy convergence of the CNN. This CNN is trained to predict the GHI from an ASI image input. In this particular study ResNet-50 is used. A residual network architecture which has 50 layers, but it contains 3 additional layers before the last layer. Respectively with density of 1000, 500 & 5 neurons. The last layer is replaced with a single neuron with activation "Linear". ResNet-50 was originally used to classify images into 1000 object categories. For background information 3.5.4.

During experiments this CNN was fed an image generated with optical flow. As a result the 10-minute future GHI was predicted. This image was generated with Gunnar Farneback optical flow (see 3.2.2) using two consecutive images with 10 minutes apart. A downside is that this study does not compare to a baseline like persistence, which makes this model hard to compare with others. The study does not answer the questions:

- What is a ideal time span between frames to construct a new image with optical flow?
- What are the limitations of this approach when speaking of prediction horizon?

2.3.4 Weather pattern classification model

C. Feng et al. (2017) achieved better result by training a model to classify what day it by only looking at the first 4 hours. CSI is the ratio of GHI and Clear sky GHI (GHI_{clr}), which can be used as the criteria to classify the weather types.

- Sunny *CSI* > 0.75
- Cloudy CSI < 0.25
- Partially cloudy 0.75 > CSI > 0.25

For each of these day categories a separate model was proposed [19]. A SVM was trained to classify the day by taking the data of first 4 hours. With this method they achieve 84% accuracy, but there is just 1 cloudy day in 201 days. Eventually, they are able to beat persistence with nMAE 6.7% (where persistence has 7.93%) and nRMSE 9.75% (where persistence has 12.94%).

2.3.5 Perez conversion model

Stated by the Perez conversion model [38] you can derive GHI using the clearness index (CSI) and extraterrestrial solar irradiance. In order to improve prediction accuracy, the following is proposed: instead of predicting future GHI directly [21], it is possible to train a model to predict CSI. From this CSI prediction the GHI can be derived using the Perez conversion model (equation 2.5).

$$CSI = GHI/[TSI * (r_0/r)^2 Cos(SZA)$$
(2.5)

where:

- r_0 is the mean earth-sun distance, which is approximately 149597871 km.
- r denotes the earth-sun distance at time t, this comes from a external data source (see 4.2).
- SZA is the solar zenith angle at the time of interest, which can be computed according to a solar position algorithm (see section 3.4) [41, 46].
- Total solar irradiance (TSI), which measures mean extraterrestrial solar irradiance received over a surface perpendicular to the solar beam. This comes from an external data source (section 4.2).

The denominator on the right-hand side of the equation is the extraterrestrial solar irradiance. Therefore, the solar irradiance we would like to predict, which is GHI, can be expressed by the clearness index multiplied by $TSI * (r_0/r)^2 Cos(SZA)$.

In this study we try with this model if we can improve our predictions. Initially, our models predict GHI. But, with the Perez conversion model we predict GHI and calculate the corresponding GHI.

2.4 Predicting with external data

Li et al. (2016) proposed a statistical model for predicting 15 minutes, 1 hour and 1 day ahead. This model was used to predict the output of a PV-system in kilowatt-hour (kWh). The ground truth was taken from a data set of a photovoltaic(PV)-system, where a year of data was available. The following input data came from weather services: temperature, wind speed and wind direction. Additionally, more variables were found usefully:

- Solar zenith angle.
- Cosine of Solar incidence angle

• Solar elevation angle

Statistical models Artificial Neural Network (ANN) and SVR were used, a baseline is missing so the results do not mean much. However, ANN performed better than SVR [32]. The prediction accuracy was worse on rainy and cloudy days. An advantage of this model that is able to predict without image data. It clearly performs worse on rainy and cloudy days which is logical as the models do not have actual data about clouds. The question remains if this model is competitive to models that use ASI images. Since a common baseline lacks, this cannot be verified. Additionally, models might predict different when weather circumstances change. Ideally, models should be compared while predicting on the same dataset.

Chow et al. (2012) built an ANN to obtain real time, 10 and 20 minutes ahead predictions. The following variables where used: dry-bulb temperature (air temperature, not influenced by moisture or irradiance), Solar elevation angle, Solar zenith angle and GHI [15]. These variables were measured on site for 15 days from 06:00 to 19:00 with a sample interval of 10 minutes.

It seems like adding data will improve predictions, but this is not the case for all data. Rana et al. (2016) compared an univariate and a multivariate ANN model. In this study they used PV installation data measured over 2 years, where they sampled every 5 minutes. Other data came from the nearest weather station. It is unclear is how far this weather station is located or how accurate this is. Where the univariate model only has access to PV installation data, the multivariate model has access to solar irradiance, temperature, humidity and wind speed. Two prediction algorithms are used ANN and SVM, whereas a result up to 20 minutes ANN multivariate performs slightly better. From 25-30 minutes ahead the performance is similar. Finally, from 25 to 60 minutes the univariate model outperform the multivariate for both prediction algorithms [40]. This model gives some insight in what input variables are important for prediction. Although, additional research is needed for input variables which were outside the scope of this particular research.

2.5 Pre-processing

In general the first step is elimination, in solar forecasting night data is superfluous. The normalization process depends on the models used. Random forests do not need to normalize data [10]. Neural networks are very sensitive for normalization and common approaches are normalization [0, 1] and [-1, 1].

2.6 Performance evaluation

In order to determine and optimize a model you need to measure how good it performs. In relevant literature the performance evaluation is done by looking at the errors over all forecasted values. The error is defined as:

$$E = |y_{obs} - y_{pred}| \tag{2.6}$$

where:

 y_{obs} = observed value. y_{pred} = predicted value

Common error metrics are Root Mean Square Error (RMSE) and Mean Absolute Error (MAE). RMSE is more sensitive to extreme values with respect to MAE. for a detailed explanation see section 5.5.

2.7 Summary

The past studies in short term GHI prediction using ASI's use two different approaches, where one approach mostly focuses on the extraction of features and uses this as input to a model. The second approach estimates the movement and location of clouds and uses this future prediction image as input for a statistical model. Since all studies have different data-sets, weather circumstances and performance evaluations, performance among models is hard to compare. A general observation is that results are averaged over multiple days. A problem with this is that those days might have easier and more difficult days because of different weather circumstances, this makes models even harder to compare. Additionally, little research is done for short term prediction horizons.

In this thesis, we propose multiple models, using optical-flow and feature extraction. Statistical and meteorological methods are the dominant models in this field [4], important data sources are solar position data, wind direction, wind strength and temperature. A downside is the availability of this data and not being able to capture sudden fluctuations. Present research has gaps in comparing multiple models on the same dataset. Also, using ASI images in combination with data as input variables for machine learning is limited.



Year	Author	Pre-processing	Models	Data	Horizon	Timeframe skyimage	Measures	Notes
2012	Chow et al.	-	ANN	Temperature, SZA, SEA, GHI	10m, 20m		coefficient of correlation	
2013	Chia-Lin Fu, Hsu-Yung Cheng	normalization [0,1]	Regression	#cloudpixels, frame difference, gradient magnitude, intensity level, accumulated intensity, number of corners	5m	15m. 1 per minute.	RMSE, MAE, MBE	Predicting clearness, then calculate GHI. CLI ground truth.
2016	Li et al.	normalization [0, 1]	ANN, SVR	SZA, Temperature, Wind, SEA	15, 1h, 24h		MBE, MAE, RMSE	
2016	Rana et al.	normalization [-1,1]	ANN, SVR	wind, temperature, Humidity, GHI	5,10,,60m		MAE, MRE	
2017	Feng et al.	Elimination, normalization, [0, 1] reconstruction.	SVR	Historical GHI, Clearsky GHI, CSI, Skyimage mean, variance, Renyi entrophy.	lh	4h	nMAE, nRMSE	3 different models for avg weather (CLI) day
2017	Y. Ai et al.	Crop images, 10AM to 2PM	Dense optical flow, SVR	Velocity field, Grid cloud fraction	1,2,3-m	30s	MAE, RMSE, MAPE	
2018	S. Tiwari et al.	Resize to 800*800. Noise removal.	CNN	Predict future image with dense optical flow.	10m	2 images 10m	RMSE, MAE, MAPE	-
2018	Bruneau et al.	Std.	ARIMA, MLP	Onsite sensors	60m		RMSE, MAE	

Chapter 3

Theoretical Background

In this chapter we describe in detail algorithms, machine learning techniques and (statistical) models. These details are necessary to fully understand employed models and related design decisions taken in our work. Later, where necessary it discusses a more extensive description of parts from the literature study.

3.1 Time-series forecasting

Time-series exist out of a sequence of observations. These observations are measured at a specific time [9]. Discrete time-series are time-series where these observations are captured at instances from a discrete set of times. Formally, forecasting discrete time-series using past and present observations are defined as follows. Let y_t denote the value of some variable in period t. Then, a forecast \tilde{y} of its value for time t = 1, made at time t, is formulated as followed [34]:

$$\tilde{y}_{t+1} = f(y_t, y_{t-1}, .., y_{t-l}) \tag{3.1}$$

where:

f = a function taking past and present observations as input.

l = the length/amount of variables used as input.

Traditionally time-series follow a certain behavior [34]:

$$x_t = m_t + s_t + \epsilon_t \tag{3.2}$$

where:

 x_t = the observed values at time *t*.

 m_t = a trend at time t.

 s_t = seasonal bias at time t.

 ϵ_t = an error margin.

In univariate time-series the series is dependent on a single time dependent variable. In this study we have multiple time dependent variables, this is defined as multivariate time series forecasting.

3.2 Optical flow

Optical flow has been a reliable method to predict where future cloud might be positioned [2, 47].

Optical flow is the motion of objects between consecutive frames of sequence, caused by the relative movement between the object and camera. This technique has been used on 1, 2, 3 and 10 minutes future predictions [47]. There are two types of optical flow, sparse and dense optical flow.

Sparse optical flow tracks some distinguished pixels, which represent the boundaries and edges of the object, and is mostly used when the image time frame is in seconds range since there might be much movement.

Dense optical flow, on the other hand, tracks all the pixels in an image and is mostly used when considerable movement is expected within the image frames. Dense optical is more accurate but is more computationally expensive as it takes into account every pixel. The scope is set to the most common method: the Gunnar Farneback method [6].

Optical flow in general: Let I(x, y, t) be a pixel at time t. In the first image the pixel moves by a distance in some time t. Let this distance be dx and dy.

$$I(x, y, t) = I(x + dx, y + dy, t + td)$$
(3.3)

With the Taylor series the movement will be [44]:

$$I(x + \Delta x, y + \Delta y, t + \Delta t) = I(x, y, t) + \frac{\partial I}{\partial x} \Delta x + \frac{\partial I}{\partial y} \Delta y + \frac{\partial I}{\partial t} \Delta t + [H.O.F.].$$
(3.4)

From this equation follows that:

$$\frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t = 0$$
(3.5)

Which results in:

$$\frac{\partial I}{\partial x}V_x + \frac{\partial I}{\partial y}V_y + \frac{\partial I}{\partial t} = 0$$
(3.6)

$$I_x V_x + I_y V_y = -I_t \tag{3.7}$$



3.2.1 Sparse optical flow

The Lucas–Kanade method assumes a neighborhood of pixels has the same velocity. Let q_i be pixel *i* at some time in the neighborhood. $I_x(q_i)$, $I_y(q_i)$ and $I_t(q_i)$ denote the partial derivatives of image *I* with respect to position (x, y) and time *t*, for pixel q_i at the current time. The set of equations is represented in the following matrices where Av = b:

$$A = \begin{bmatrix} I_{x}(q_{1}) & I_{y}(q_{1}) \\ I_{x}(q_{2}) & I_{y}(q_{2}) \\ \vdots & \vdots \\ I_{x}(q_{n}) & I_{y}(q_{n}) \end{bmatrix} \qquad \nu = \begin{bmatrix} V_{x} \\ V_{y} \end{bmatrix} \qquad b = \begin{bmatrix} -I_{t}(q_{1}) \\ -I_{t}(q_{2}) \\ \vdots \\ -I_{t}(q_{n}) \end{bmatrix}$$
(3.8)

Applying least squares fitting to solve for V_x and V_y [37] (page 241-242) will give an approximation for dx and dy.

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i I_x(q_i)^2 & \sum_i I_x(q_i) I_y(q_i) \\ \sum_i I_y(q_i) I_x(q_i) & \sum_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(q_i) I_t(q_i) \\ -\sum_i I_y(q_i) I_t(q_i) \end{bmatrix}$$
(3.9)

3.2.2 Dense optical flow

The Gunnar Farneback method is a two-frame flow estimation algorithm which calculates the flow per pixel. It uses quadratic polynomials to approximate the motion between 2 frames. An example in Figures 3.1-3.5. This can be done efficiently by using the polynomial expansion transform [18].

Approximate each pixel neighborhood by a polynomial:

$$f_1(x) \sim x^T A_1 x + b_1^T x + c_1 \tag{3.10}$$

where:

A = a symmetric matrix.

$$b = a$$
 vector

c = a scalar.

 $X = a \ 1 \times 2$ vector containing running variables *x* and *y*.

Now construct a new signal f_2 with global displacement d.

$$f_{2}(x) = f_{1}(x-d) = (x-d)^{T} A_{1}(x-d) + b_{1}^{T}(x-d) + c_{1}$$

$$= x^{T} A_{1}x + (b_{1} - 2A_{1}d)^{T}x + d^{T} A_{1}d - b_{1}^{T}d + c_{1}$$

$$= x^{T} A_{2}x + b_{2}^{T}x + c_{2}$$
(3.11)

Equating the coefficients in the quadratic polynomials:

$$A_{2} = A1,$$

$$b_{2} = b_{1} - 2A_{1}d,$$

$$c_{2} = d^{T}A_{1}d - b_{1}^{T}d + c_{1}$$

(3.12)

From 3.12 we can solve for translation d, if A_1 is non-singular:

$$2A_1d = -(b_2 - b_1),$$

$$d = -\frac{1}{2}A_1^{-1}(b_2 - b_1)$$
(3.13)



Figure 3.1: Image at 21 august 2019 at 12:00:15 PM.



Figure 3.2: Image at 21 august 2019 at 12:00:30 PM.





Figure 3.3: Pixel intensity visualized with Gunnar-Farneback flow.



Figure 3.4: Image at 21 august 2019 at 12:00:45 PM.



Figure 3.5: Generated image at 21 august 2019 at 12:00:45 PM.

3.3 Harris corner detection algorithm

In the field of pattern recognition, the Harris algorithm is often used to detect corners [30]. Moreover, this method is also used to detect cloud corners in ASI images [21]. The algorithm contains five steps:

- 1. Color to gray-scale.
- 2. Spatial derivative calculation for each pixel, this gives $I_x(x, y)$ and $I_y(x, y)$.
- 3. Structure tensor setup M (see 3.14). This co-variance matrix has the information of how the gradients of local pixels are distributed.
- 4. Harris response calculation with $k \in [0.04, 0.06]$ (see 3.15). The smallest eigenvalue of the tensor is approximated.
- 5. Non-maximum suppression, locate local maxima as corners with a 3 × 3 filter.

$$M = \sum_{(x,y)\in W} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \sum_{(x,y)\in W} I_x^2 & \sum_{(x,y)\in W} I_x I_y \\ \sum_{(x,y)\in W} I_x I_y & \sum_{(x,y)\in W} I_y^2 \end{bmatrix}$$
(3.14)

$$R = det(M) - k(trace(M))^{2} = \lambda_{1}\lambda_{2} - k(\lambda_{1} + \lambda_{2})^{2}$$
(3.15)

3.4 Sun positioning

It is possible to scan for circles or bright areas in an image to locate the sun's position. However it is more precise to use the method proposed by Jiacheng Tang et al. (2018). The orientation of the sun can be represented by Solar altitude angle (H_s) and Azimuth angle (A_s). This is calculated by the local latitude ϕ , Declination angle (δ) and time angle t_a [46].

$$\delta = 23.45 * \sin(360 * \frac{284 + n}{265}) \tag{3.16}$$

Where n is day of the year.

$$t_a = (t_{ts} - 12) * 15^{\circ} \tag{3.17}$$

$$H_s = \sin^1(\sin\phi * \sin\delta + \cos\phi * \cos\delta * \cos t_a) \tag{3.18}$$

$$A_s = \cos^{-1} \frac{\sin H_s * \sin \phi - \sin \delta}{\cos \delta * \cos t_a}$$
(3.19)



From this follows x and y coordinates.

$$x = \frac{R * H_s}{90^\circ} * \cos A_s \tag{3.20}$$

$$y = \frac{R * H_s}{90^\circ} * \sin A_s, \tag{3.21}$$

where *R* is the radius of the (pre-processed) image.

3.5 Neural networks

Neural networks are developed to determine and exploit stochastic dependencies within individual samples of data. In this research we will cover and use Artificial Neural Network (ANN), Convolutional neural network (CNN) and Long short-term memory (LSTM).

3.5.1 Artificial Neural Networks

The greatest advantage of Artificial Neural Network (ANN) over other modeling techniques is their ability to model complex, nonlinear processes without assuming the form of the relationship between input and output variables [15]. ANN have an input layer (L_1) and an output layer (L_{k-1}). The layers in between are called the hidden layers. This is a generic model and it learns a non-linear mapping from an input vector to an output. An Artificial Neural Network (ANN) has k layers, where i = 0..k. Moreover, each layer i has L_i neurons. Where the first layer is activated by the input data $L_1 = input$ the following layers are activated as followed:

$$L_{i+1} = \alpha(W_i * L_i + b_i)$$
(3.22)

where:

 W_i = the weight matrix.

 b_i = the bias matrix.

 α = some activation function (see 3.7).

The routine of pushing data trough this network is called *feed-forward*.

The output of fig. 3.6 is defined as followed:

$$Output = \alpha(W_{i1}x_1 + W_{i2}x_2 + W_{i3}x_3 + b_i)$$
(3.23)

At the final layer the error is calculated with some loss function giving an error. This loss function is dependent on the prediction. As we are dealing with a regression problem mean squared error is an option:

$$E = \frac{1}{N} \sum_{i=1}^{n} \left(|y_i - y'_i| \right)^2$$
(3.24)



where:

 y_i = the ground truth vector.

 y'_i = the predicted vector by the neural network.

Using a learning algorithm like gradient descent, the weights and biases are updated every iteration [22]. As a result the error can get smaller every iteration.



Figure 3.6: A neuron in a Artificial neural network, where z is the result of the sum.

3.5.2 Activation functions

An activation function is a non-linear function applied by a neuron to introduce non-linear properties in the network. Examples are given in fig. 3.7. Common used activation functions are defined as followed [22]:

Sigmoid:
$$f(z) = \frac{1}{1 + e^{-z}}$$
 (3.25)

$$HyperbolicTangent: f(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$
(3.26)

$$ReLU: f(z) = max(0, z) \tag{3.27}$$

3.5.3 Back-propagation algorithm

The weights and biases determine the output of neural networks. These weights and biases have to be updated to improve performance. This is done with a learning algorithm. An example of such a learning algorithm is back-propagation [22]. While pushing data trough the



Figure 3.7: Plotted activation functions, based on image from [50]

network, it has some output. From this output an error E can be calculated while comparing it with the ground truth. This is called a loss function C. Back-propagation updates weights and biases iteratively such that the loss is minimized. Back-propagation begins by applying the chain rule twice to the error function partial derivative 3.28. The amount of samples that is pushed simultaneously trough the network is called batch-size. When all the samples are pushed trough the network it counts as 1 epoch.

$$\frac{\partial E}{\partial w_{i,j}^k} = \frac{\partial E}{\partial a_j^k} \frac{\partial a_j^k}{\partial w_{i,j}^k}$$
(3.28)

where:

E = the error output by the loss function. $w_{i,j}^{k} = \text{the weight for node } j, \text{layer } L_{k} \text{ for incoming node i}$ $a_{j}^{k} = \text{the the output after activation for node } i \text{ in layer } L_{k}.$

Basically, this equation gives the change to the error if you change a particular weight. According to this information the weight is updated, but how much should you change this weight? Here the hyper-parameter learning rate η is introduced. Next, $\Delta w_{i,j}$ is added to $w_{i,j}^k$ as formulated in 3.29.

$$\Delta w_{i,j} = -\eta \frac{\partial E}{\partial w_{i,j}^k} \tag{3.29}$$

Instead of only using current calculated gradient to update the weights, we can also take multiple past steps under consideration. This is defined as *momentum*.

In this study we use 2 learning algorithms stochastic gradient decent (SGD) and adam. SGD uses back-propagation until some defined minimum is found. Adam is an extension of SGD, where it makes use of past seen values to adaptive change the momentum [28].



3.5.4 Convolutional Neural Networks

Convolutional neural network (CNN) are regularized versions of ANN's. The feed-forward architecture stays the same but the input will be an image. The hidden layers perform convolution, activation and pooling (In that order). Stage 1 consists out of receiving input into layer L_{i+1} , then convolution is performed. Convolution is shifting the weight matrix while multiplying both. In the second stage a non-linearity (activation) function is applied. Optionally, afterwards pooling is applied, this can be considered stage 3 [22].

Convolution is performed with a filter, these filters are usually a square of dimension N * N [36]. The filter and bias contain the weights to be updated (while training the model). This filter is 'convolved' over the input (image/matrix), afterwards bias is added. During convolution the usual stride is 1 (shift step size), but this might vary. The result of convolution has a smaller dimension than the previous layer. To counter dimension shrinkage you could add padding. Zero-padding is a margin of zero values which are placed around the image. The difference in output size is given in equation 3.30 (normal output size) and equation 3.31 (output size with zero padding).

$$L_{output} = (N - F)/S - 1$$
(3.30)

where:

N = the size of the current layer.

F = the filter size.

S = the length of the stride.

$$L_{output} = (N + 2P - F)/S - 1$$
(3.31)

where:

P = the padding depth.

Pooling is an operation where the size of the image is reduced. This is done by taking the average or maximum of a group of 'pixels'. This is done the reduce the amount of parameters and computation of a CNN.

When making a neural network 'too deep', the conversion rate will drop. Residual networks are developed as the solution to the problem of training a deep neural network. Normally a neuron on feeds forward to the next neuron. Basically in a residual block it also feeds forward to 2-3 neural forward.

3.5.5 Recurrent Neural Networks

A Recurrent neural network (RNN) is an extension of an ANN. RNN are specially designed to process a sequence of observations. Respectively, the input should be a sequence of observations, mapping it to some output. During this mapping each time step t is processed individually. In every layer h_i of a RNN it takes the output from the previous layer as additional input (see equation 3.32).



Figure 3.8: Example of a RNN, where O_t is the output and x_t the input of hidden layer h_t . Based on [20].

$$h_{i+1,t} = \alpha(W_1 \sigma(h_{i+1,t-1}) + W_2 h_{i,t} + b) \quad for \ t = 1..N$$
(3.32)

where:

 W_i = the weight matrix. b_i = the bias matrix. α = an activation function (see 3.7). $\sigma(h_i, t)$ = the hidden state of layer *i*

The hidden state represents context based information of past input(s). This information can be used in future predictions. Although, a RNN loses this information over time, because it only has information of last output. LSTM's do not have this problem [47].

3.5.6 Long short-term memory

Long short-term memory (LSTM) is a type of RNN, but it is able to 'remember' more of past predictions/input. Basically it decides to keep a hidden state or trow it away. A LSTM has 3 additional 'gates' to do these operations in a 'cell'.

• Forget gate: It decides when information is thrown away. It looks at the previous hidden

state h_{t-1} and input x_t and outputs a number between 0 and 1 for each number in the cell state C_{t-1} . Respectively, 0 is forget and 1 is keep. (Equation 3.33).

- Input gate: It decides what new information i_t is stored. Next, a *tanh* layer creates a vector of new candidate values, \tilde{C}_t , that could be added to the state. (Equation 3.34)
- Output gate: It decides what parts of the cell state are output, giving o_t . Followed by, a multiplication of an activation of the cell state C_t and o_t . Formulated in equation 3.35 [47].

$$f_t = \sigma(W_f * [h_{t-1}, x_t] + b_f)$$
(3.33)

$$i_{t} = \sigma(W_{f} * [h_{t-1}, x_{t}] + b_{f})$$

$$\tilde{C}_{t} = tanh(W_{c} * [h_{t-1}, x_{t}] + b_{c})$$
(3.34)

$$o_t = \sigma(W_o * [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * tanh(C_t)$$
(3.35)

where:

 σ = an activation functions explained earlier this section.

tanh = 'tanh' activation function.

$$W_i$$
 = the weight matrix.

 b_i = the bias matrix.

- h_t = the hidden state at time *t*.
- x_t = the input at time *t*.

3.6 Random Forests

In order to understand random forest you first need to understand the building block 'decision tree'.

3.6.1 Decision tree

A decision tree is basically a tree data structure, but every node has a ratio of the population, where the root node has 100% of the population. This node is split up in multiple nodes [7]. In order to find the best split all possible splits over all possible features are evaluated such that
the split with the lowest MSE is chosen. In literature MSE is common for regression problems [33]. The process of splitting continues until a a limit of samples is reached (minimum of 1) or when the maximum depth of the tree is reached. This trained decision tree can be used to classify other data. But a decision tree tends to overfit the data. Random forests are composed of multiple decision trees. This gives the following hyper-parameters:

- Minimum samples per leaf. If this amount is reached, no more splitting is necessary.
- Maximum depth of a tree *m*. Every time node(s) is/are split at layer *l*. A new layer *l* + 1 is created, which exists out of the children nodes of layer *l*. When *l* == *m*, no more splitting is necessary.

In a random forest each tree is not trained on the same training data. When constructing such a tree, when performing a split operation only a random subset of features is considered. Finally the result is averaged over all decision trees in the forest [7]. The amount of decision trees in a random forest is another hyper-parameter called 'number of estimators'.



Chapter 4

Problem Description

In this chapter we first formulate the problem. Next, we discuss the setup of data source and how this data looks like. Finally, the research questions are formulated.

4.1 Problem definition

On some days GHI fluctuates much, this is clearly visible in figures 5.1 and 4.2. As discussed in the introduction, this causes the grid to be off balanced. Past research delivered some methods for predicting GHI, but all relevant studies use different data-sets (weather conditions) and performance evaluation. It is hard to conclude with certainty what models are most suitable for which circumstances and prediction horizons.

4.2 Data & equipment

4.2.1 Sensor setup

Two cameras are installed in Almeria, Spain. These cameras are owned by EKO whom have a partnership with Copernicus Institute. The model name is "CMS-Schreder ASI-16/50". Its camera hardware features a robust coated quartz glass dome, 180°/360° fish-eye optics, with a 4-Megapixel resolution sensor, and an IR-cut filter avoid degradation of the sensor by long time direct exposure to the sun disk. Additionally, this camera come with built-in sensors for temperature and humidity.

The "ASI-16/50" comes with built-in ventilation and up to 60W airflow heating, to minimize condensation on the glass dome and to quickly remove eventual raindrops and snow. The system has been tested for harsh environments from -40°C up to more than +50°C air temperature. Its double-cover design combined with its forced ventilation will minimize the risk of

32

damages because of long time exposures to very strong direct sun radiation.

These cameras are located in Almeria (see fig. 4.1), Spain. Precisely, at the coordinates $37.091549^{\circ}N$, $-2.363556^{\circ}E$ and $37.095253^{\circ}N$, $-2.354785^{\circ}E$. The distance between these cameras is approximately 880,2 meters. So, this provides data from two locations.



Figure 4.1: Location of equipment, source: Google maps.

4.2.2 Data

We can split up the data in 3 categories: First, image data from the cameras. Secondly, data measure by hardware at location of the cameras. Finally, additional data received from external sources.

These cameras are online since July 23, 2019 and make 180 degrees images, with a sample time of 15 seconds. These images have a usable resolution of 1920x1920 (Fish eye). Additional data is saved which is measured at the same time and location:

• Outdoor temperature, measured in Celsius.



- GHI, measured in W/m^2
- Humidity.

Figures 4.2 to 4.4 are averages and variances of GHI, temperature and humidity over the the months August until December 2019. As can be observed is that the averages follow a bell curve closely. For GHI, September and October have higher variance (see figure 4.2). These months might be harder to predict. This can be explained by moving more clouds, if a cloud starts blocking the sun the GHI will drop, where the temperature is just affected very little. This explains why there is more variance in GHI, making it harder to predict with respect to temperature (see figure 4.3). These images show the differences and trends per season and time of the day.



Figure 4.2: Average and variance GHI (in kWh/m^2) at camera site 1.



Figure 4.3: Average and variance in temperature (in Celsius) at camera site 1.



Figure 4.4: Average and variance humidity at camera site 1.



From the images is information extracted. The intensity *I* as the mean grayscale of an image defined as:

$$I = \frac{\sum_{p \in image} p_r + p_b + p_g}{N} \tag{4.1}$$

where:

N = the amount of pixels.

 $p_r = \text{pixel red.}$

 $p_g = pixel green.$

 p_b = pixel blue.

The amount of cloud pixels is detected with a cloud pixel algorithm RBR 2.1. The amount of edges is detected with the canny edge detect algorithm [11]. The amount of corners is detected with Harris algorithm 3.3. A distribution per month is visible in figures: 4.5-4.8.

An additional feature we implemented we call 'pxl'. We locate the sun by finding the brightest object in the image. Next, we 7 circle areas around the sun with each 20 pixels more distance of the located sun. We count the amount of pixels per area and each of those is a individual feature. An example is given in fig. 4.9.



Figure 4.5: Average and variance number of edges at camera site 1.





Figure 4.6: Average and variance of the intensity at camera site 1.



Figure 4.7: Average and variance of the number of corners at camera site 1.



Figure 4.8: Average and variance for the number of cloud pixels at camera site 1.







Figure 4.9: Pixel location feature at 21 July 2019, 11 AM. The pixels are classified by there distance from the sun (the white/red center). Each different distance is annotated with a different color.

Next to the ASI images, temperature and GHI additional external data is available.

- Clear sky GHI (*GHI*_{clr})
- CSI, calculated by the ratio of GHI and *GHI*_{clr}.
- Solar azimuth angle.
- Solar zenith angle.
- Sun-earth distance
- Apparent sun elevation accounting for atmospheric refraction.
- Actual elevation (not accounting for refraction) of the sun in decimal degrees, 0 = on horizon. The complement of the zenith angle.
- Apparent zenith: apparent sun zenith accounting for atmospheric refraction.
- Solar time in decimal hours (solar noon is 12.00).

Pvlib (version 0.6.3) provides *GHI_{clr}* ('clearsky' module), sun-earth distance at time t ('pyephem' module) and solar position data like Solar zenith angle,Solar incidence angle,Solar elevation angle,Solar altitude angle and Azimuth angle [52].

All experiments and pre-processing will be done in Python (3.5.0). A public GitHub repository is available containing all codes of this project https://github.com/nii3lsh/ASI_playground.

4.2.3 Hardware experiments

All experiments (unless clearly noted) are run on a node that contains an AMD EPYC 7451 24-Core Processos and 256 GB (RAM) memory. Additionally, the node is equipped with a GTX 1080 Ti GPU.

4.2.4 Data-set

We consider data from the 1st of August 2019 to 31 December 2019. These exists out of 121 sunny days, 29 partially cloudy days and three cloudy days (total 153 days). We can not test on all of these because of limited computation power, time in combination with the amount of models. Lets say we predict on a day from 6 AM in the morning to 7PM. Gives 19 - 6 = 13 hours, 13 * 60 = 780 minutes. Per minute we will do 20 predictions for each model. Thus, for as test-set we take a random sample of five sunny days, three partially cloudy days and two cloudy days.

The classes sunny, partially cloudy and cloudy are divided on average CSI per day:

- Sunny *CSI* > 0.75
- Cloudy CSI < 0.25
- Partially cloudy 0.75 > CSI > 0.25

The preliminary data set contains the following days:

- Sunny: 5, 6, 7, 8 October 2019.
- Partially cloudy: 10 October 2019.

We consider the preliminary data-set also as a 'validation-set' to select the best performing models. The selected models will also predict on the test-set. The test set is more balanced in terms of weather circumstances and exists out:

- Sunny: 15 September, 15 October, 15 November and 15 December.
- Partially cloudy: 21 October, 17 November, 16 December.
- Cloudy: 22 October, 3 December.

4.2.5 Train, validation and test data

Let we predict for day t, in general at moment t we do not know anything about t + x, where x > 0. Because of this, to predict for day t we can only use data to train observed before t. An example would be stock pricing. If we want to predict the stock price p for tomorrow, we also

do not know anything about the stock price after tomorrow. Because of this we can only use 'old' data as input or training data.

We consider all data before predicting time t - 3 as training data. Furthermore, the validation set will exist out of t - 3..t (not including t). Time t will be testing data. Therefore, when time passes t will be larger and the training set will grow. Let 11 September 2019 be a day to predict. All data prior to 11 September is taken as training data, excluding 3 days before 11 September (8, 9, 10 September). These 3 days are included as validation set.



Figure 4.10: Example training and test-set. Where p is the day to predict(test) and n the amount of validation days. t_0 is the first data in the dataset.

4.3 Research questions

Regarding discussed available data and problem, the following research question is formulated:

How can we use machine learning to predict (short-time) Global horizon irradiance using all sky images and sensor data in Almeria, Spain?

This research question can be separated in two categories. Questions regarding the model and questions regarding data.

4.3.1 Models

We want to compare multiple models, this raises questions regarding different models:

- What different models are most suitable for what prediction horizon?
- What different models are most suitable for what weather circumstance?
- How does developing a separate model for each individually prediction horizon affect forecast performance?
- To What extend can optical flow models be utilized to improve forecasting?

• How does CSI prediction compare to GHI prediction?

4.3.2 Data

The different models use the same data. This gives questions about how to represent the data and what data is valuable for this problem. Questions about data:

- To what extend is additional training data from another location beneficial for forecast performance?
- What features are important for the models to predict GHI?
- How many minutes prior prediction moment are features relevant for prediction performance?



Chapter 5

Methodology

In this chapter we give an overview of the methodology. First, a global overview is given. Followed by, how the data is represented, pre-processed and used as input. We will describe the specifications of the proposed models. Finally, the performance evaluation is formulated.

5.1 Experiment setup

In this section the general experiments are covered. Later in this chapter we define multiple models, each of these will have access to data of the same days. Accordingly, these models will predict for the same time *t* on a specific day with an identical future horizon *H*. Additionally, these models will have the same limitation/boundaries in terms of training, validation and test-sets. If additional data is used this will be clearly noted. The goal of the experiments is to answer each of the sub-research questions. While building a forecasting model you want to compare the predictions to some baseline. A common baseline in GHI predictions is the Persistence model and Smart-persistence model[29]. Every model (with exception of the CCN-Flow model) will have a variation of input in data:

- Onsite measured data.
- Information/features extracted out images.
- External data from Pvlib 4.2.
- Access to training data from other site.

For every model the GHI ground truth is used, this is measured on location with a pyranometer (see section 4.2).

42

5.2 Pre-processing

The data exists out of numerical and image data (see section 4.2). The pre-processing subroutine is different for these types of data. A part of the data is not use-able, that is when it is dark. Respectively, we only consider data where the measured GHI at that time > 0.

5.2.1 Numerical data

First of all, the numerical data needs to be checked. The temperature contains incorrect 0 (temperature) values and negative data (GHI). The temperature can be interpolated with neighbor data (a measurement 15s before/after). This can be done be taking the average of both. The negative data will be set to 0, as it is a measure error of low GHI (which can not be lower than 0). For all the observed variables the absolute variables and gradient are used as input. For ANN and LSTM all numerical data *y* is normalized to [0, 1], by applying:

$$y = x/z \tag{5.1}$$

where:

$$z = \sqrt{\sum_{i=1}^{n} x_i^2}.$$

This is also know as Euclidean distance.

5.2.2 Image data

For image data, the black border around the raw image data is deleted. Afterwards, the images resolution is still too big and will be resized to 400×400 . Obstacles will be labeled as black pixels. Finally, the image is normalized. This is done by subtracting the mean from each pixel and then dividing the result by the standard deviation. We need the the pixel values to be integer and positive, so the normalization scale is [0,255].

5.3 Experiments

In this section is highlighted what experiments are conducted to answer the research questions. The length of the feature sequences is varied to answer the research question about ideal amount of minutes of features prior prediction moment. Also, to determine what features are important, a model will be trained and predict with a different subset of features (see table 6.2 for details). Additionally, another approach is to take CSI as ground truth and converting this to GHI using the Perez conversion model. We can calculate this ground-truth with observed GHI and GHI_{clr} from Pvlib (see section 4.2).

CHAPTER 5. METHODOLOGY

In order to determine if data from site 2 is values for performance on site 1, the following experiment is proposed. The data from site 2 will be added to the training set. The result of this experiment will give insight if additional training data from the other site improves performance.

First, all models will predict on the preliminary data-set (see section 4.2). Respectively, models that perform well will also predict on the test-set. We say a models performs well if it performs better as the baseline.

5.4 Models

This section formulates proposed models. These models will compete on the same experiments. All models have the same input, with exception of Model 4 - CNN-flow and persistence, this model only has access to the images. Models:

- 1. Random Forests (RF)
- 2. Artificial Neural Network (ANN).
- 3. Long short-term memory (LSTM)
- 4. Convolutional neural network (CNN)-flow
- 5. Persistence, baseline model.

Every numerical model can be adjusted to predict for prediction horizon i. Where i > 0 and $20 \ge i$. Another approach is where a model predicts all values 1..20 together. Respectively, in the results we refer to the models that predict 20 steps at once as 'multi'.

5.4.1 Model-selection

Giving the amount of models multiplied by the amount of settings (amount of features, length of features), first a pre-selection of all the models will be done on the preliminary data-set (see section 4.2). Models that perform well (better than the baseline) will be tested more thoroughly on the test set. This has two reasons, first of all the limit of computational power. Secondly, the more models that predict the bigger the chance one will predict well.

5.4.2 Numerical models

Models RF, ANN and LSTM will work on numerical input.

• These models use subsets of features, the individual elements are onsite measurements, information from images and external data from Pvlib (see section 4.2).

- A model will get additional training data access of site 2. 4.2.
- A model will be trained to predict CSI. This will be converted to GHI using the Perez conversion 2.5.

5.4.3 CNN-flow

This model exists out of two parts. First a CNN will be trained to predict GHI from an image. The dataset has images with ground truth. Next to this a input needs to be generated. With optical flow we can generate an future image from a few consecutive images. This will be the input for the trained CNN. The only data this model uses is the images and the ground-truth GHI to train. Multiple optical flow algorithms exist, the scope is set to Gunnar Farneback method (dense) and sparse optical flow Lucas-Kanade (see section 3.2.2 for details).

5.4.4 Persistence Model

A common baseline model in solar irradiance forecasting is the persistence model [29, 3] (an example in figure 5.4.4). 'Regular' persistence predicts $value_t$ for time t + h, where t is time and h the prediction horizon. An improvement on this is called *smart-persistence*, proposed by Andrew Kumler et al. (2019). It assumes a stable CSI. The current CSI is calculated by the ratio of the current DNI and GHI. While considering the SZA and time the future DNI is calculated. The prediction is the multiplication of the future DNI with the current CSI.

5.5 Performance evaluation

Most often the following evaluation techniques are used in similar studies: Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Skill Score (SS) [21, 2, 3].

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \left(e_i^2\right)}$$
(5.2)

$$MAE = \frac{1}{n} \sum_{i=1}^{n} \left| e_i \right|$$
(5.3)

where:

 e_i = the error factor. The absolute difference between the observed o_i and predicted p_i value. N = number of samples.

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \frac{|o_i - \bar{p_i}|}{o_i} * 100\%$$
(5.4)

$$SS = 1 - \frac{Error_{forecast}}{Error_{baseline}}$$
(5.5)

It is hard to compare results to other studies because of different data-sets, weather and earth position. In forecasting a common metric is the SS (equation 5.5). In our case we compare RMSE, MAE, MAPE and the ramp-score with the baseline. As a baseline method the Persistence model and Smart-persistence model is proposed. All models are build to optimize on 'Mean Squared Error'. Thus, we consider RMSE the most important performance metric.

As an additional error the ramp-score is introduced. This metric aims to measure the ability to forecast relevant GHI fluctuations [48]. Furthermore, in literature these are also called 'ramp events'. Traditional error metrics capture the error at a certain moment. Ramp-score captures fluctuations over a time period, rather than the error per time index. Therefore, this links better to the problem of short term fluctuations as described in the introduction.

First, the prediction and observation time-series sequence is compressed with the swinging door algorithm [8]. The significance of the resulting ramps is defined by a sensitivity parameter ϵ . We can say a new ramp is detected when a linear function is required to approximate the signal with absolute value lower as ϵ . In this study ϵ is set to 0.05.

In the error metric proposed by Vallence et al. (2017) is chosen to average the values by the hour but they have a prediction horizon of 5 hours. Accordingly we average per 5 minutes. Then the ramp-score is calculated with:

$$ramp - score = \frac{1}{t_{max} - t_{min}} \int_{t_{max}}^{t_{min}} |SD(T(t)) - SD(R(t))| dt$$
(5.6)

where:

SD = the output of the swinging doors algorithm, giving the extracted slopes.

 t_{max} = maximum bound of the period with *GHI* > 0.

 t_{min} = minimum bound of the period with GHI > 0.

An example of SD visualization is given in Fig 5.1.

5.5.1 Statistical significance

To compare if some model a is significantly better than model b the *Diabold-Mariano test* [17, 24] is proposed. In this significance test is looked at the prediction errors of both models over a particular prediction horizon. We want to test if the null hypothesis:

$$H_0: E(d_t) = 0, \forall_t \tag{5.7}$$

in comparison with the alternative hypothesis:

$$H_1: E(d_t) \neq 0, \forall_t \tag{5.8}$$



Figure 5.1: Swinging door compression for 'LSTM M 5 all data' in sunny weather circumstances. Respectively, from left to right days 15 September, 15 October, 15 November and 15 December. On 15 November you can see there are many fluctuations in GHI. Calculated ramp-score is 21.06. SD output is averaged by the hour in this figure to give a clear example. In our experiments this is set to 5 minutes.



We define the loss differential between two forecasts as $d_t = e_{1t} - e_{2t}$, meaning the error at time t for model 1 and model 2. In our case the error of some model will be RMSE. We take a common (in forecasting) significance level of $\alpha = 0.05$ [24, 17]. For each horizon this test should be applied differently, since a further prediction horizon leads to more uncertainty. Therefore, errors of models should be compared for a fixed prediction horizon *H*. Accordingly, in case of this study *H* is set to 1..20 minutes.

Chapter 6

Implementation

In this chapter we explain how the models are build in detail. For each model is discussed how the data structure is made and how the hyper-parameters are selected. First, manual tuning is applied. Followed by a search grid of hyper parameters. Additionally, we discuss how the models are trained.

6.1 Data-frame

Let n_f be the number of features (see section 4.2). let l be the length a sequence. We define an instance of a data-frame to be a sequence of length l containing l prior values of each feature f. Let $f_{n,d,t}$ be feature number n at day d and time t in a sequence. Were this sequence predicts at time t + l for some prediction horizon p. Each of these sequences $x_{d,t,l}$ has a observed value $y_{d,t}$.

Table 6.1: Example instance of a data-frame $x_{d,t,l}$, a ground truth value $y_{d,t}$ belongs to it.

These sequences are used as input for each model. Here the size differs for the sequence length l and the number of features n.

6.2 Numerical models

Models RF, ANN and LSTM take the described data-frame as input with a subset from the features. The ANN and LSTM both have training iterations called epochs (3.5.3). To determine

49



Figure 6.1: Flowchart from the start of raw to solar forecast predictions. The 'Machine learning model' in this chart is valid for RF, ANN and LSTM implemented in this research.

how much epochs are optimal, a technique early-stopping is used on the validation set. After each epoch is checked if there is performance improvement in terms of validation-error. A patience parameter is introduced, this parameter is set to 15. Accordingly, If after 15 epochs (3.5) no improvement is observed in the validation-error, the weights are restored to the state with minimum validation-error and training is stopped.

An example is given in figure 6.2. A visible detail is that the training-error is higher than the validation-error. Normally, this would be strange unless the validation-set exist out of easier samples. In our case the validation-set is easier, because it contains days of December. Respectively, the training set contains days from July until November. Summer months have longer days and higher GHI, which makes them harder to predict as Winter days. This can be seen in fig. 4.2.

Next, we implemented subsets of features to determine what data is valuable. The sets are *all-data, img, onsite* and *meteor* (see table 6.2).

CHAPTER 6. IMPLEMENTATION

Features/Subsets	all data	img	onsite	meteor	pxl
Time and date	✓	✓	✓	1	✓
GHI	1	X	\checkmark	X	✓
Temperature	1	X	\checkmark	X	✓
Humidity	1	X	\checkmark	X	✓
Clear sky GHI	1	X	×	1	X
CSI	1	X	×	1	X
Azimuth	1	X	×	1	X
Zenith	1	X	×	1	X
Sun-earth distance	1	X	×	1	X
Number of cloud pixel	1	✓	×	X	X
Brightness	1	✓	×	X	X
Number of edges	1	✓	×	X	X
Number of corners	1	1	×	X	X
Cloud pixels distance sun	X	X	×	X	1

Table 6.2: Feature subsets: what features are in what subset.



Figure 6.2: Loss and validation accuracy (MSE) plotted per epoch (training iteration). Training set: all data but December. Validation set: December.

6.2.1 Random Forests

We chose mean square error (see section 5.5) as loss function for random forest, because we are dealing with a regression problem and we want to penalty extreme values. Our model always considers all features. For the grid search is chosen for a 10-fold-cross-validation on the training set (all days until 15 September). The grid search is done for number of estimators, minimum samples per leaf and the max depth of a tree (see section 3.6 for explanation hyper-parameters).

Hyper-parameters

The search exist out of the following parameters:

- Number of estimators = [50, 100, 150, 200, 300]
- Minimum samples leaf = [1, 2, 4, 12, 24, 64]
- Max depth = [25, 50, 75, 100, 200]

The best results came with: Estimators 200, minimum samples leaf 1 and a max depth of 25.

6.2.2 Artificial neural network

For the ANN is chosen for a 3 layer architecture (see section 3.5). It performed better then 2 layers and 4 performed worse. For this 3 layer network there is a search grid for the amount of nodes per layer, activation functions, optimizers, learning rate and drop out (see section 3.5.3 for explanation hyper-parameters). Then the optimal amount of epochs is taken by taking the minimum validation loss of the model with the minimum validation loss.

Hyper-parameters

The following search grid is used:

- Nodes = [(32, 64, 32), (64, 128, 64), (128, 265, 128)].
- Activation functions = ['relu', 'sigmoid'].
- Learning rates = [0.001, 0.01, 0.1]
- Dropout = [0, 0.1, 0.5]

The best performance was achieved with nodes (64, 128, 64), activation function *ReLu*, learning rate = 0.001 and dropout = 0.

6.2.3 Long short term memory

For the LSTM is chosen for a 3 layer architecture. The first to layers are LSTM layers, were the third one is a dense layer. For this 3 layer network there is a search grid for the amount of nodes per layer, activation functions, optimizers, learning rate (see section 3.5).

Hyper-parameters

The following search grid is used:

- Nodes = [(50, 25, 10), (60, 30, 15), (80, 40, 20)]. Here are the first two layers LSTM layers, the third a dense layer.
- Activation functions = ['relu', 'sigmoid'].
- Learning rates = [0.001, 0.01, 0.1]
- Dropout = [0, 0.1, 0.5]

The best performance was achieved with nodes (50, 25, 10), activation function *ReLu*, learning rate = 0.001 and dropout = 0.

6.3 Flow - Convolutional neural network

For each day a separate CNN is trained, including all the training data until that day. The reason behind this is that you want to evaluate as much data as possible while training. However, you could use transfer learning, but then we have a chance on overfitting on the old data (as we will train on this more) or have some bias towards old data.

Let *t* be the current time and let the prediction horizon be *x* minutes into the future. From the images t - x and *t* the flow will be computed and a new image i will be generated for t + x. Image *i* will be used as input for the trained CNN on that particular day. The output will be the predicted value for t + x.



Chapter 7

Results

In this chapter we give an overview of the results by conducted experiments. Starting with experiments on the preliminary data-set, these are a pre-selection of the proposed models. Thus, models that perform good (see methodology) will be used to predict the test set.

Single models are trained to predict on exactly one prediction horizon, giving 20 different models for 20 different prediction horizons. These 'single' models are annotated with 'S'. Other models predict for all horizon at once making them 20 times more efficient in terms of time, we call these 'Multi' models and are annotated with an 'M'.

All the models have access to data, the number next to the models annotates the amount minutes prior to prediction moment data is access-able. Features are split in different classes (see table 6.2).

- All features available 'all data'.
- Only features extracted out images 'img'.
- Only features from onsite equipment 'onsite'.
- Only features from external data resource 'meteor'.
- The feature subset 'pxl' has access to the onsite features and the amount of cloud pixels per distance of the sun.

Also, models annotated with 'prz' predict GHI_{clr} , which is converted to GHI using the Perez model 2.5. Finally, when training the models usually only training data from the prediction location is used. When additional data is included this is notated as '2CAM'.

This chapters gives the results of the preliminary data set. Next, the results of the test-set (B) are given.

54

7.1 Model selection

As discussed in 4.2, the preliminary data-set exist out of the following days 5-8 October and the 20th of October. To this data the following models are fitted: RF, ANN, LSTM and the flow model. The results are highlighted below. For plots only RMSE is shown, a more detailed overview can be found in the appendix. As discussed, the models that perform relatively well to the baselines will be selected to predict on the test-set.

7.1.1 Random forests

Shown in fig.7.1 and table 7.1 'single' perform worse than 'multi' when the input is equal. A sequence length of 5 has the best predictions, but the difference is too small to conclude anything. The features that do best are the subsets 'all data' and 'onsite'. External data ('meteor') performs bad with respect to the baseline. As selection we take the feature subsets 'all data' and 'onsite' as these perform best. Additionally, sequence length 5, 10, 20, 30 and 60.



Figure 7.1: RMSE per prediction horizon for preliminary data-set

7.1.2 Artificial neural network

Single models tend to be unstable and predict 0's in some occasions, which leads to an extremely high error. Multi models did not have this problem. Models with an relative short sequence length seem to perform better on RMSE, where longer sequence perform better on MAE (see table 7.2). As selection we take feature length 10,20,40,60 with features subsets 'all data' and 'onsite'.



Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	101.62	50.65	31.94	23.96	NA	NA	NA	NA
Smart-persistence	98.02	41.08	24.5	16.2	NA	NA	NA	NA
RF S 10 all data	84.6	44.15	23.72	26.85	0.17	0.13	0.26	-0.12
RF S 10 img	88.5	45.66	25.73	30.76	0.13	0.1	0.19	-0.28
RF S 10 meteor	125.7	76.74	59.47	60.76	-0.24	-0.52	-0.86	-1.54
RF S 10 onsite	86.6	43.9	24.5	25.75	0.15	0.13	0.23	-0.07
RF S 20 all data	85.16	44.92	24.6	26.56	0.16	0.11	0.23	-0.11
RF S 20 img	84.7	43.94	24.92	26.24	0.17	0.13	0.22	-0.1
RF S 20 meteor	125.28	76.4	58.79	58.83	-0.23	-0.51	-0.84	-1.46
RF S 20 onsite	83.98	42.89	24.22	23.86	0.17	0.15	0.24	0.0
RF S 30 all data	85.9	45.84	25.33	26.41	0.15	0.1	0.21	-0.1
RF S 30 img	84.3	43.93	24.95	25.58	0.17	0.13	0.22	-0.07
RF S 30 meteor	129.5	78.46	59.95	61.62	-0.27	-0.55	-0.88	-1.57
RF S 30 onsite	83.99	43.01	24.29	24.31	0.17	0.15	0.24	-0.01
RF M 10 all data	80.22	39.75	16.37	24.19	0.21	0.22	0.49	-0.01
RF M 10 img	84.52	41.88	16.38	27.16	0.17	0.17	0.49	-0.13
RF M 10 meteor	117.88	67.96	23.17	53.56	-0.16	-0.34	0.27	-1.24
RF M 10 onsite	82.52	39.99	16.16	23.68	0.19	0.21	0.49	0.01
RF M 20 all data	80.49	40.58	16.38	24.5	0.21	0.2	0.49	-0.02
RF M 20 img	80.89	39.98	16.26	25.02	0.2	0.21	0.49	-0.04
RF M 20 meteor	119.53	68.96	23.42	54.12	-0.18	-0.36	0.27	-1.26
RF M 20 onsite	81.1	39.84	15.99	23.16	0.2	0.21	0.5	0.03
RF M 5 all data	80.18	39.62	16.13	25.22	0.21	0.22	0.5	-0.05
RF M 5 img	86.28	42.21	16.68	29.73	0.15	0.17	0.48	-0.24
RF M 5 meteor	120.22	69.46	23.13	59.9	-0.18	-0.37	0.28	-1.5
RF M 5 onsite	84.99	41.0	16.65	24.45	0.16	0.19	0.48	-0.02

Table 7.1: RF. Average performance on preliminary data-set.



Figure 7.2: RMSE per prediction horizon for preliminary data-set.



Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score ↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	101.62	50.65	31.94	23.96	NA	NA	NA	NA
Smart-persistence	100.21	42.71	53.17	16.91	NA	NA	NA	NA
ANN M 20 all data	82.67	46.36	133.46	33.14	0.19	0.08	-3.18	-0.38
ANN M 20 all data 2CAM	81.51	47.0	47.6	34.83	0.2	0.07	-0.49	-0.45
ANN M 20 img	90.54	46.78	37.2	27.74	0.11	0.08	-0.16	-0.16
ANN M 20 img 2CAM	91.13	48.15	36.0	28.73	0.1	0.05	-0.13	-0.2
ANN M 20 meteor	145.39	88.66	52.72	78.31	-0.43	-0.75	-0.65	-2.27
ANN M 20 meteor 2CAM	166.78	100.31	63.15	89.16	-0.64	-0.98	-0.98	-2.72
ANN M 20 onsite	92.16	45.7	36.25	25.71	0.09	0.1	-0.13	-0.07
ANN M 20 onsite 2CAM	93.46	46.07	35.25	25.64	0.08	0.09	-0.1	-0.07
ANN M 40 all data	82.33	44.67	86.47	29.01	0.19	0.12	-1.71	-0.21
ANN M 40 all data 2CAM	82.75	47.34	45.54	29.61	0.19	0.07	-0.43	-0.24
ANN M 40 img	100.56	53.02	60.28	35.3	0.01	-0.05	-0.89	-0.47
ANN M 40 img 2CAM	92.55	48.03	66.33	28.4	0.09	0.05	-1.08	-0.19
ANN M 40 meteor	182.12	106.0	46.23	96.85	-0.79	-1.09	-0.45	-3.04
ANN M 40 meteor 2CAM	142.85	95.56	69.09	84.24	-0.41	-0.89	-1.16	-2.52
ANN M 40 onsite	95.87	47.86	32.75	27.52	0.06	0.06	-0.03	-0.15
ANN M 40 onsite 2CAM	95.98	49.12	43.98	24.27	0.06	0.03	-0.38	-0.01
ANN M 60 all data	85.3	48.76	106.5	33.1	0.16	0.04	-2.33	-0.38
ANN M 60 all data 2CAM	83.97	46.58	104.87	29.36	0.17	0.08	-2.28	-0.23
ANN M 60 img	96.15	51.5	59.35	28.85	0.05	-0.02	-0.86	-0.2
ANN M 60 img 2CAM	99.58	55.03	41.59	31.55	0.02	-0.09	-0.3	-0.32
ANN M 60 meteor	153.14	89.42	47.24	78.54	-0.51	-0.77	-0.48	-2.28
ANN M 60 meteor 2CAM	154.98	88.22	113.21	75.43	-0.53	-0.74	-2.54	-2.15

Table 7.2: ANN. Average performance on pre. data-set.

7.1.3 Long short-term memory

LSTM has the same problem as ANN with 'single' models predicting 0's on occasions. Although, it is much more stable as ANN. LSTM seems to be much more sensitive for sequence lengths. Everything above 10 tends not to work in tried experiments. A sequence length of 3 seems to short and has slightly worse prediction performance with respect to lengths 5 and 10 (visible in fig. 7.3 and table 7.3). As selection we pick sequence lengths 5 and 10 with the feature subsets 'all data' and 'onsite'.



Figure 7.3: LSTM. RMSE per prediction horizon for preliminary data-set.

7.1.4 Flow Model

Visible in figure 7.4 is that both optical flow models perform bad with respect to the baseline. An possible explanation is that exists out of two models that both have an error. While combining two models with both an error, the prediction performance only gets worse. The firs error is when generating a future image with optical flow. The seconds error is when classifying this image. While training the CNN there was already an average RMSE 300. These models will not be selected for the test-set.



Figure 7.4: RMSE per prediction horizon for preliminary data-set.

CHAPTER 7. RESULTS

Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	101.62	50.65	31.94	23.96	NA	NA	NA	NA
Smart-persistence	100.21	42.71	53.17	16.91	NA	NA	NA	NA
LSTM S 5 all data	87.09	45.16	39.82	31.48	0.14	0.11	-0.25	-0.31
LSTM M 5 all data Prz	254.07	135.83	193.93	122.66	-1.5	-1.68	-5.07	-4.12
LSTM M 10 all data	232.24	125.1	425.93	111.43	-1.29	-1.47	-12.33	-3.65
LSTM M 10 all data 2CAM	81.12	43.79	53.27	27.03	0.2	0.14	-0.67	-0.13
LSTM M 10 img	87.71	43.09	53.85	25.91	0.14	0.15	-0.69	-0.08
LSTM M 10 img 2CAM	82.27	38.88	33.5	20.91	0.19	0.23	-0.05	0.13
LSTM M 10 meteor	123.55	74.75	98.99	61.03	-0.22	-0.48	-2.1	-1.55
LSTM M 10 meteor 2CAM	136.41	86.27	41.8	71.57	-0.34	-0.7	-0.31	-1.99
LSTM M 10 onsite	80.54	37.93	39.92	22.04	0.21	0.25	-0.25	0.08
LSTM M 10 onsite 2CAM	81.78	39.23	28.26	20.55	0.2	0.23	0.12	0.14
LSTM M 3 all data	80.5	42.86	41.95	28.93	0.21	0.15	-0.31	-0.21
LSTM M 3 all data 2CAM	81.1	42.04	47.31	26.69	0.2	0.17	-0.48	-0.11
LSTM M 3 img	88.62	42.46	44.57	24.26	0.13	0.16	-0.4	-0.01
LSTM M 3 img 2CAM	85.76	42.08	33.62	24.95	0.16	0.17	-0.05	-0.04
LSTM M 3 meteor	110.7	66.62	38.1	51.29	-0.09	-0.32	-0.19	-1.14
LSTM M 3 meteor 2CAM	138.56	81.08	117.08	64.59	-0.36	-0.6	-2.67	-1.7
LSTM M 3 onsite	92.28	45.49	31.84	27.21	0.09	0.1	0.0	-0.14
LSTM M 3 onsite 2CAM	83.8	40.29	50.46	25.58	0.18	0.2	-0.58	-0.07
LSTM M 5 all data	80.0	41.36	50.02	25.67	0.21	0.18	-0.57	-0.07
LSTM M 5 all data 2CAM	82.55	43.21	35.87	31.48	0.19	0.15	-0.12	-0.31
LSTM M 5 img	87.05	43.62	734.07	22.5	0.14	0.14	-21.98	0.06
LSTM M 5 img 2CAM	87.83	42.32	44.96	26.09	0.14	0.16	-0.41	-0.09
LSTM M 5 meteor	134.58	82.13	48.37	72.78	-0.32	-0.62	-0.51	-2.04
LSTM M 5 meteor 2CAM	142.35	81.97	55.08	69.33	-0.4	-0.62	-0.72	-1.89
LSTM M 5 onsite	82.62	38.04	42.77	24.71	0.19	0.25	-0.34	-0.03
LSTM M 5 onsite 2CAM	83.3	39.71	29.39	21.08	0.18	0.22	0.08	0.12

Table 7.3: LSTM. Average performance evaluation preliminary days.

Perez model

The model 'prz' which predicts *CSI* instead of *GHI* performs much worse as both baselines. At prediction horizon 20, the RMSE is 255.28 (visible in table 7.2). This model performs very bad with respect to most others at all prediction horizons. The performance is not visible in figure 7.3 because its above 120. Because of this performance this model will not predict on the test-set. A possible explanation is that predicting CSI is harder and that there is some error in converting CSI to GHI.

Single and Multi models

With the current implementation for ANN and LSTM the performance of the single models is very unstable. It tends to predict 0 many occasions, next to this it takes 20 times more computation time than multi models. On the test-set only multi models will predict. An example of unstable predictions is 'LSTM S 5 all data' (see fig.??), other single models were even more unstable. The RF 'single' models seem not to have this problem, but the computational

59



problem remain. Because of these reasons all 'single' models will be dropped for predictions on the test-set.



7.2 Test-set

In this section we show the results of the test-set. First, the general results of each model. For each model is discussed when they perform best and how they relate to the baselines. Next, we discuss for all models an in-depth performance for:

- Valuable features
- Different weather circumstances
- Times of the day
- Computation time
- Additional training data
- Statistical significance

No model is able to perform well on cloudy weather circumstances, most of these results can be found in the appendix. This bad result is due the fact of limited data, the full dataset contains 3 cloudy days. In total 2 cloudy days are included in this test-set. Respectively, on the prediction of the first cloudy day the models have not encountered a cloudy day in the training-set. The second time, the training-set contains 1 cloudy day, which is still very limited. This is an explanation why current models perform bad on cloudy days with respect to persistence (visible in fig. 7.5). Additionally, tables in this chapter only show averages over all prediction horizons, in the appendix more specific tables and plots are shown (see section A).

7.2.1 Random forests

Random forest is the quickest training of all proposed models (see table 7.2.6). A detail is that this model is trained to minimize MSE and not MAE. For sunny weather 'RF M 5 onsite' is able to beat the baseline on average. However, on average RF predict slightly worse than smart-persistence. But, when considering the biggest prediction horizon of 20 minutes RF prediction performance is (slightly) better as smart-persistence (see table A.1).

For partially cloudy weather 'RF M 60 all data' performs best on average, this model is also able to outperform both persistence and smart-persistence (see tables 7.4-7.5). On cloudy days it performs very bad with respect to persistence. In Fig.7.5 the RMSE is plotted per prediction horizon. More plots and tables are located in the appendix A.1. Finally, short sequences seem to perform best for random forests sunny. Where longer sequences perform better for partially cloudy weather on average. However, when considering the biggest prediction horizon for partially cloudy weather smaller sequences perform better (see table A.2). This break even point is from a prediction horizon of 13 minutes.



Figure 7.5: Average RMSE on test-set for models RF and ANN.

Model	$\mathbf{RMSE}\downarrow$	MAE ↓	MAPE ↓	Ramp-score ↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	59.83	33.65	28.58	26.68	NA	NA	NA	NA
Smart-persistence	55.37	19.77	84.26	21.3	NA	NA	NA	NA
RF M 10 all data	59.31	29.53	19.2	26.73	0.01	0.12	0.33	-0.0
RF M 20 all data	57.88	26.85	20.8	25.24	0.03	0.2	0.27	0.05
RF M 30 all data	57.5	26.78	20.14	25.56	0.04	0.2	0.3	0.04
RF M 5 all data	60.12	30.77	19.85	28.16	-0.0	0.09	0.31	-0.06
RF M 5 onsite	56.73	27.95	18.36	26.17	0.05	0.17	0.36	0.02
RF M 60 all data	58.41	26.75	19.58	26.2	0.02	0.21	0.31	0.02
RF M 5 all data 2CAM	62.25	32.37	22.38	28.9	-0.04	0.04	0.22	-0.08

Table 7.4: RF. Average performance on test-set, with weather circumstance sunny.

Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	84.59	48.08	37.12	29.31	NA	NA	NA	NA
Smart-persistence	87.85	47.82	74.22	28.66	NA	NA	NA	NA
RF M 10 all data	79.98	47.87	44.82	31.17	0.05	0.0	-0.21	-0.06
RF M 20 all data	80.41	47.95	50.19	31.21	0.05	0.0	-0.35	-0.06
RF M 30 all data	78.3	47.26	45.05	30.58	0.07	0.02	-0.21	-0.04
RF M 5 all data	81.11	48.66	46.5	31.59	0.04	-0.01	-0.25	-0.08
RF M 5 onsite	80.26	49.58	45.45	31.36	0.05	-0.03	-0.22	-0.07
RF M 60 all data	77.4	46.43	48.11	29.61	0.08	0.03	-0.3	-0.01
RF M 5 all data 2CAM	81.05	48.16	63.14	31.36	0.04	-0.0	-0.7	-0.07

Table 7.5: RF. Average performance on test-set, with weather circumstance partially cloudy.

7.2.2 Artificial neural networks

ANN models that take a sequence with length 10 perform the best. In general, we observe that when the sequence length increases accuracy drops. In sunny weather circumstances 'ANN M 10 all-data' is able to predict well (with respect to baseline) and the difference is small with 'ANN M 10 onsite'. The downside is that 'ANN M 10 all-data' has a bad ramp-score with respect to the baselines. But, again must be noted that the models are optimized to minimize MSE. Results are shown in 7.6-7.7. In sunny weather circumstances ANN is not able to beat smart-persistence on average. Although, for a prediction horizon of 15-20 minutes, smart persistence is also outperformed. Model 'ANN M 10 onsite' is able to outperform persistence on all prediction horizons for partially cloudy weather (see fig. 7.6). The 'all data' subset performs worse on this weather circumstance. For partially cloudy weather 'ANN M 10 onsite' outperforms the baseline in all error metrics. Furthermore, ANN 10 is not able to beat smart-persistence on the ramp-score metric. Finally, we see that short prediction horizons are more difficult for ANN. The difference with the baselines gets bigger while the prediction horizon grows.



Figure 7.6: Average Root mean squared error, test-set for ANN models.

Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score ↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	59.83	33.65	28.58	26.68	NA	NA	NA	NA
Smart-persistence	55.37	19.77	84.26	21.3	NA	NA	NA	NA
ANN M 40 all data	66.52	41.71	41.63	33.16	-0.11	-0.24	-0.46	-0.24
ANN M 60 all data	71.21	42.79	34.63	34.77	-0.19	-0.27	-0.21	-0.3
ANN M 10 all data	57.2	35.53	38.79	28.4	0.04	-0.06	-0.36	-0.06
ANN M 10 all data 2CAM	59.13	37.26	36.97	30.83	0.01	-0.11	-0.29	-0.16
ANN M 10 onsite	55.51	31.17	59.92	26.93	0.07	0.07	-1.1	-0.01
ANN M 10 onsite 2CAM	55.37	33.53	103.71	27.84	0.07	0.0	-2.63	-0.04
ANN M 20 all data	64.63	40.28	52.96	32.29	-0.08	-0.2	-0.85	-0.21

Table 7.6: ANN. Average performance on test-set, with weather circumstance sunny.



Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	84.59	48.08	37.12	29.31	NA	NA	NA	NA
Smart-persistence	87.85	47.82	74.22	28.66	NA	NA	NA	NA
ANN M 40 all data	81.75	51.35	79.3	36.1	0.03	-0.07	-1.14	-0.23
ANN M 60 all data	85.61	55.55	126.4	39.72	-0.01	-0.16	-2.4	-0.36
ANN M 10 all data	79.29	46.38	73.98	32.21	0.06	0.04	-0.99	-0.1
ANN M 10 all data 2CAM	77.08	47.3	70.14	32.88	0.09	0.02	-0.89	-0.12
ANN M 10 onsite	66.13	39.46	63.26	26.29	0.22	0.18	-0.7	0.1
ANN M 10 onsite 2CAM	66.83	39.56	45.4	26.07	0.21	0.18	-0.22	0.11
ANN M 20 all data	78.08	48.72	72.56	33.93	0.08	-0.01	-0.95	-0.16

Table 7.7: ANN. Average performance on test-set, with weather circumstance partially cloudy.

7.2.3 Long short-term memory

Conducted experiments show that when using LSTM models, the shorter the sequence the better the performance. Actually, everything above a sequence length of 10 did not work reliable. Some og the results were very off. The best results are achieved with a sequence length of 5. On sunny weather 'LSTM M 5 all data' performs best on average (example in fig. 7.9). However, all version of LSTM are not able to beat the baselines on very short prediction horizons. The break even point is after 4 minutes with model 'LSTM M 5 all data'. For partially cloudy, 'LSTM M 5 onsite' performs slightly better with an average RMSE of 67.43. This is shown in tables 7.8, 7.9. LSTM is able to outperform the baselines over all prediction horizons. For short prediction horizons 'LSTM M 5 onsite' also performs better on sunny weather (see Fig.7.7). Additionally, LSTM is the only tested model in this study that had a better ramp-score than (smart-)persistence for sunny and partially cloudy weather.



Figure 7.7: LSTM. Average Root mean squared error per prediction horizon.



Figure 7.8: LSTM. Root mean squared error per prediction horizon for test-set.



Model	$\textbf{RMSE} \downarrow$	MAE ↓	MAPE ↓	Ramp-score ↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	59.83	33.65	28.58	26.68	NA	NA	NA	NA
Smart-persistence	55.37	19.77	84.26	21.3	NA	NA	NA	NA
LSTM M 5 PXL	54.07	30.23	88.99	24.52	0.1	0.1	-2.11	0.08
LSTM M 10 all data	52.25	28.23	38.6	24.3	0.13	0.16	-0.35	0.09
LSTM M 5 all data	48.87	24.51	69.58	21.33	0.18	0.27	-1.43	0.2
LSTM M 5 all data 2CAM	49.29	23.47	48.97	20.91	0.18	0.3	-0.71	0.22
LSTM M 5 onsite	51.84	24.87	38.84	21.21	0.13	0.26	-0.36	0.21

Table 7.8: LSTM. Average performance on test-set, with weather circumstance sunny.

Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	84.59	48.08	37.12	29.31	NA	NA	NA	NA
Smart-persistence	87.85	47.82	74.22	28.66	NA	NA	NA	NA
LSTM M 5 PXL	68.69	38.76	45.62	24.01	0.19	0.19	-0.23	0.18
LSTM M 10 all data	76.13	49.54	622.03	32.88	0.1	-0.03	-15.76	-0.12
LSTM M 5 all data	71.35	41.04	53.63	26.34	0.16	0.15	-0.44	0.1
LSTM M 5 all data 2CAM	76.28	43.58	146.07	28.44	0.1	0.09	-2.93	0.03
LSTM M 5 onsite	67.43	37.34	55.45	23.43	0.2	0.22	-0.49	0.2

Table 7.9: LSTM. Average performance on test-set, with weather circumstance partially cloudy.

7.2.4 Valuable features

As discussed in the experiment setup and implementation, models predict while having access to subset of the features. Clearly visible in table 7.3 is that the subsets 'meteor' and 'img' predict bad with respect to other subsets. An example observation for the LSTM 5 M model on the validation-set is:

- Meteor, RMSE 144.15, MAE 93.68, MAPE 70.38, ramp-score 72.78.
- IMG. RMSE 98.1, MAE 58.67, MAPE 44.96, ramp-score 22.5.
- Onsite, RMSE 92.25, MAE 49.89, MAPE 34.81, ramp-score 24.71.
- All features, RMSE 89.13, MAE 51.81, MAPE 47.83, ramp-score 25.67.

Most likely this is due the fact that the variable to predict GHI is a feature in onsite (and all features), making it much easier to predict. The onsite data-source performs best on implemented models. In the model selection we saw that the subsets all-data and onsite perform best. On the test-set we see that onsite performs best for models RF and ANN. LSTM all data predicts better on sunny weather, where onsite performs better on partially cloudy (while averaging over all prediction horizons).

7.2.5 Different weather circumstances

In this section the difference in prediction performance per weather circumstance is discussed. Again, it should be considered that the full dataset contains 3 cloudy days. In total 2 cloudy
CHAPTER 7. RESULTS



days are included in this test-set. Respectively, on the prediction of the first cloudy day the models have not encountered a cloudy day in the training-set. The second time, the training-set contains 1 cloudy day, which is still very limited. This is an explanation why current models perform bad on cloudy days with respect to persistence (visible in fig. 7.5).

The best performing models of RF, ANN and LSTM outperform the baseline on sunny and partially cloudy weather.

In general for models RF, ANN and LSTM it is clear that easiest day to predict is sunny, because it has the lowest errors. But, while comparing it to the baseline results it seems partially cloudy days are the hardest to predict (but easier for implemented models). Current models are not always able to beat smart-persistence for sunny weather for error metrics RMSE and MAE. But for partially cloudy weather this is easier. The biggest (absolute) difference in performance with respect to the baseline is on partially cloudy weather. A logical explanation is that the baseline assumes constant weather. Additionally, for sunny and partially cloudy weather, when the prediction horizon *h* increases the error difference with respect to the baseline gets bigger.

LSTM seems to be the only model that is able to outperform the baseline quite well on sunny weather circumstances, with SS-RMSE 18% and SS-MAE/SS-Ramp of 22%. Compared with ANN (< 8%) and RF SS-RMSE 5%, SS-MAE 21% and SS-Ramp 5%. The bad average performance of ANN is due to the fact that is not able to perform well on short horizons (< 10). From a prediction horizon of (> 13) it start to predict well with respect to the base line (visible in fig. 7.6). For partially cloudy weather circumstances are mostly able to beat persistence and smart-persistence. ANN performs best with respect to to other models. Respectively, SS-RMSE 22%, SS-MAE 18%, SS-RAMP 11% where LSTM has SS-RMSE 20%, SS-MAE 22%, SS-RAMP 0% and RF SS-RMSE 8%, MAE 3%, SS-RAMP 20%

7.2.6 Computation time

An application of this study would be to predict GHI for an actual solar farm to optimize energy generation. When a model is more computationally intensive it will use more energy. In this section we highlight the resources needed to perform predictions on 1 day. Usually neural networks train on a GPU, but this would not be a fair comparison to RF. To straiten it out training and prediction will be done on only the CPU (details: 4.2.3). 'Single' models would use 20 times more time as showed in table 7.10.

7.2.7 Additional training data

To answer the question if additional training data from the second site (see 4.2) is beneficial we compare the same models. Accordingly, one of these models has additional training data, where the other does not. Over all models this generally not the case, very few instances of

	RF	ANN	LSTM
Training	69.20s	72.59s	722.65s
Prediction	0.014ms	0.0038ms	15,6ms
Parameters	n/a	71452	19280

Table 7.10: Training execution times (in seconds) per model on day 27 September 2019. Training-set contains all available data before 27 September 2019 (excluding 3 days of validation-set). The prediction times are 1 prediction for the next 1 ... 20 minutes in milliseconds. For the neural network the amount of parameters per model.

models trained with data from site 2 predict slightly better. Results are shown tables: 7.4-7.9.



Figure 7.9: Predictions on sunny days of the test-set with prediction horizon 20. Model: LSTM M 5 all data. Respectively, from left to right days 15 September, 15 October, 15 November and 15 December.

7.2.8 Statistical significance

To determine if a some model performs significantly better (see section 5.5.1) than the baseline, the 'Diebold-Mariano' test is applied. The null hypothesis 'competing model performs equal to the baseline persistence' is rejected when p < 0.05. None of the models perform better in cloudy weather. Thus, these p-values are not shown. In the given models below we define 'n/a' if that particular models does not predict better than the baseline for a certain prediction horizon.

Random Forests

In sunny weather, random forests only perform statistically significantly better while using the feature subset 'onsite' with a sequence length of 5 (minutes) on a prediction horizon from 19 to 20 minutes (see table 7.11).

In partially cloudy weather, random forest are only able to predict statistically significantly better on short prediction horizons. The best model 'RF M 60 all data' is able to predict statistically significantly better up to a prediction horizon of 2 minutes (see table 7.11).

	Sunny												
Prediction horizon	RF M 10 all data	RF M 20 all data	RF M 30 all data	RF M 5 all data	RF M 5 onsite	RF M 60 all data	RF M 5 all data 2CAM						
17	0.68641	0.55651	0.49347	0.81773	0.17323	0.51069	0.94135						
18	0.56022	0.45108	0.40583	0.61674	0.13587	0.39119	0.77597						
19	0.5148	0.40375	0.3344	0.53361	0.04567	0.31747	0.66779						
20	0.35767	0.35352	0.27043	0.43588	0.01097	0.23714	0.47557						
	Partially cloudy												
Prediction horizon	RF M 10 all data	RF M 20 all data	RF M 30 all data	RF M 5 all data	RF M 5 onsite	RF M 60 all data	RF M 5 all data 2CAM						
1	0.01943	0.00203	0.00057	0.00602	0.00095	0.00023	0.02562						
2	0.16788	0.02013	0.01895	0.15906	0.01618	0.015	0.19916						
3	n/a	0.34332	0.13963	n/a	0.16606	0.12997	n/a						
4	0.73451	0.32563	0.14682	0.86639	0.22904	0.0731	0.7541						
5	n/a	n/a	0.7329	n/a	0.53724	0.30573	n/a						

Table 7.11: RF: Diabold-Mariono P value per horizon, compared with baseline model Persistence. Weather circumstances: sunny & partially cloudy. (n/a implies competing model is not performing better than the baseline).

Artificial neural network

For sunny weather ANN is able to predict statistically significantly better on short prediction horizons. Model 'ANN M 10 all data' does this best. Respectively, only this model is able to predict statistically significantly better from 16 minutes future horizon (see table 7.12).

The ANN models perform better on partially cloudy weather. Consequently, over all prediction horizons ANN predictions are statistically significantly better. For short horizons model 'ANN 10 onsite 2CAM' performs best. However, it is beaten by 'ANN M 10 onsite' on further prediction horizons. Results can be observed in table 7.12.

Long-short term memory

For sunny weather 'LSTM M 5 all data' predicts significantly better than the other LSTM models with respect to persistence for prediction horizons of 11 minutes to 20. The complete table is given at 7.13. Besides 'LSTM M 5 all data', model 'LSTM M 5 onsite' also starts to predict significantly better from a prediction horizon of 15 minutes.

For partially cloudy weather 'LSTM M 5 onsite' predicts significantly better for all horizons. 'LSTM M 5 onsite' has the smallest p-value for all prediction horizons, which is logical as



Table 7.12: ANN: Diabold-Mariono P value per horizon, compared with baseline model Persistence. Weather circumstances: sunny & partially cloudy. (n/a implies competing model is not performing better than the baseline).

it performs best on these weather circumstances. Other LSTM models predict better (with respect to persistence), but not statistically significantly better.

Sunny										
Prediction horizon	LSTM M 10 all data	LSTM M 5 all data	LSTM M 5 all data 2CAM	LSTM M 5 onsite						
1	n/a	n/a	n/a	n/a						
5	n/a	n/a	0.97638	0.89572						
15	0.00665	0.00106	0.00918	0.08289						
20	0.00031	5e-05	0.0001	0.02221						
		Partially cloudy								
Prediction horizon	LSTM M 10 all data	LSTM M 5 all data	LSTM M 5 all data 2CAM	LSTM M 5 onsite						
1	n/a	0.20505	0.02537	6e-05						
5	n/a	0.1602	0.6053	0.00263						
15	0.38254	0.23984	0.90999	0.00813						
20	0.4408	0.26959	n/a	0.02017						

Table 7.13: LSTM: Diabold-Mariono P value per horizon, compared with baseline model persistence. Weather circumstances: sunny & partially cloudy. (n/a implies competing model is not performing better than the baseline).





7.2.9 Model comparison

In terms of simplicity Random Forests (RF) is good. The training time is relatively low, little tuning of hyper-parameters and normalization is needed. Additionally, it performs reasonable on most sequence lengths. The best model has RMSE 56.73 (sunny) and RMSE 77.4 (partially cloudy) when we average the results over all prediction horizons. The difference get bigger when the prediction horizon grows (see tables in A.1). RF M 60 all data performs best for the random forest models, but with respect to other implemented models it performs bad on every weather circumstance and prediction horizon (see fig 7.10-7.11). Artificial Neural Network (ANN) is able to improve on sunny weather on average with RMSE 55.51, even due the fact it does not perform as well on short horizons as RF. But, on average it performs better on larger horizons. On average, it does predict better on partially cloudy weather with a RMSE 66.13. LSTM performs best on sunny weather with RMSE 48.87. Additionally, it is able to predict significantly better than persistence from a prediction horizon of 11 minutes with respect to ANN which is able to predict significantly better from 15 minutes. For partially cloudy weather, LSTM is slightly worse as ANN with a RMSE 67.43. Additionally, LSTM has the best ramp-score with respect to all tested models and baselines (smart-)persistence for sunny (21.91) and partially cloudy (23.43) weather (see figures 7.10 & 7.11).

All models perform relatively well on partially cloudy weather with respect to the baseline. This is likely due that the baselines assume some continuity. With cloudy and sunny weather circumstances this approach works better, but in partially cloudy weather there is more fluctuation.

When considering the furthest prediction horizon of 20 minutes, the difference with (smart-)persistence is bigger. The results are shown in tables A.7 - A.9. The best obtained results are that RMSE is improved with 32%, MAE with 50% and ramp-score with 48% (with respect to the baseline).

Considering that the full dataset exists out of 121 sunny days, 29 partially cloudy days and 3 cloudy days, on average LSTM will perform best.

Figure 7.10: Average performance on prediction horizon of 20 minutes with cloudy, partially cloudy and sunny weather circumstances. Error in GHI (smaller is equal to better performing).



Average performance on prediction horizon 20 with cloudy circumstances. $_{\text{MAF}}$

Average performance on prediction horizon 20 with partially cloudy circumstances.



Average performance on prediction horizon 20 with sunny circumstances.





Figure 7.11: Average performance over all prediction horizons with cloudy, partially cloudy and sunny weather circumstances. Error in GHI (smaller is equal to better performing).



Average performance on cloudy circumstances.



Chapter 8

Conclusions

8.1 Discussion

In this section we discuss the results of all models. The results give insight in what models work well for what particular prediction horizons and weather circumstances. Additionally, a notion is given of what data is suitable as input for these models. However, these are sets of multiple features (in example set meteor). The elements in a set have a similar data source. But, for every subset of features there is a possibility that it includes a feature which is not relevant for good predictions. It may be that dropping some of these features or partially merging subsets will increase forecast performance.

Models RF and ANN both perform the best only using the feature subset 'onsite', meaning they do not capture value-able information from features in extracted images. However, LSTM performs better while having access to this subset. LSTM is able to capture the complexity of these features, thus being able to predict best over all our tested models. We saw that the feature subset 'onsite' performs best on partially cloudy. The image features do not provide more valuable information to predict when clouds will block the sun. Additionally, feature subset 'meteor' is valuable in sunny weather circumstances. The reason behind this is most likely that the features from subset 'meteor' lose their value in more unstable weather. This is the same reason that smart-persistence performs worse in partially cloudy weather (with respect to persistence and sunny weather) (as discussed before smart-persistence uses identical input as features in the subset 'meteor').

On short prediction horizons (< 5 minutes) with sunny weather our models do not perform better as the baselines. After five minutes or with partially cloudy weather circumstances this does not happen. The baselines assume some continuity. Persistence assumes the same *GHI* and smart-persistence the same *CSI*. Sunny weather is more consistent as cloudy weather, meaning *GHI* fluctuates less in sunny weather. Likely, this is the reason for the difference in performance between sunny and partially cloudy weather.

From all the models that we test, the best ones perform better than (smart-)persistence on partially cloudy and sunny weather circumstances. However, while predicting on cloudy weather the none of the models is able to perform better than (smart-)persistence. Very likely this is due the lack of cloudy days and would not happen in more balanced weather circumstances.

In this research we average the values by 5 minutes while calculating the ramp-score. In relevant literature it was proposed to normalize by an hour but while forecasting 5-hours into the future. 5 minute averages represent fluctuations, but this number is an approximate.

Additionally, all data comes from the source Almeria, Spain. We cannot say with certainty how the models would perform in total different weather circumstances/location while containing the same hardware setup.

8.2 Conclusions

We wanted to answer 'How can we use machine learning to predict (short-time) *GHI* using all sky images and sensor data in Almeria, Spain?'. This study describes the application and implementation of random forest, ANN, CNN and LSTM to forecast *GHI* using different subsets of input. We tested multiple approaches using optical flow and feature extraction and decided to improve on the models that performed best on the preliminary data-set.

Out of our selection of tested models LSTM performed best on all weather circumstances. Partially cloudy weather was easier to predict than sunny weather with respect to the baseline. Although, in both scenarios LSTM performed better than the baselines and other tested models. Furthermore, the predictions on cloudy weather were worse than the baseline. We think this is due the lack of data of cloudy days. Additionally, all machine learning models performed better with respect to (smart-)persistence while prediction horizon increased.

Additionally, we predicted *GHI* using a separate model for each prediction horizon (referred to as 'single' models) and compared this to models that predict 20 prediction horizons at once. From this experiment we learned that 'multi' models (models that predict 20 horizons simultaneously) predict more accurate with respect to 'single' models (models that are trained for one particular prediction horizon).

We wanted to test to what extent we can utilize optical-flow with a CNN. The predictions did not improve on the baselines. This model is not able to predict, likely due limited information. Other models which had access to the onsite sensors performed much better.

We predicted CSI and calculated the *GHI* accordingly, eventually this did not improve our models. Thus, we learned that predicting *GHI* directly is a better approach.

We had access two another identical setup, we tried adding data of a different site as training data. We conclude that it did not improve our models.

In our search to hyper-parameters we tested what subset of features is important. We saw that onsite measure equipment was most important. However, some features did perform better/worse depending on weather circumstances. The subset 'meteor' was beneficial in sunny weather, but it did not have much value in partially cloudy weather.

We observed that a big time-frame as input made performance worse, observed variables from short period before prediction moment performed better. The amount of minutes depended on the model, but we saw in general that the last observed five to 10 minutes is most valuable. Current insights do fill up a gap in current literature. This study shows an approach on how to use a combination of ASI and numerical data in machine learning. Additionally, while forecasting GHI short sequences of features perform much better than long sequences.

8.3 Future work

The data-set is unbalanced due the weather circumstances in south Spain, which we cannot change. As results our models perform bad on cloudy weather. Data augmentation might be a good approach to fix this problem. Moreover, it might even lead to better predictions in general. Next, our models only use observed variables, where for example smart-persistence uses future estimates. The models may predict better when including (more) future estimates. For early stopping we take the 3 prior days before prediction moment. There was manual tuning involved finding this number, but additional research could give another number. Also, while extracting features, we say all the cloud pixels are treated the same. However, you could treat cloud pixels that are close to the sun differently. Additionally, flow could be added to our best performing model LSTM.

In this study we had access to 1 additional site with the same equipment. If you surrounded the location where you want to predict with more sensor sites and additionally wind speed and direction you could predict an actual future while using data from other locations.

Appendix A

Appendix

In this section the additional results are shown: figures for MAE, MAPE and ramp-score, tables average error for prediction horizon of 20 minutes.

A.1 Random forests

Additional RF results.

Model	RMSE ↓	$\mathbf{MAE} \downarrow$	MAPE ↓	Ramp-score ↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	86.01	57.64	56.33	46.56	NA	NA	NA	NA
Smart-persistence	74.21	29.13	95.14	32.86	NA	NA	NA	NA
RF M 10 all data	74.89	38.36	24.08	33.48	0.13	0.33	0.57	0.28
RF M 20 all data	73.84	35.35	26.15	31.77	0.14	0.39	0.54	0.32
RF M 30 all data	73.18	35.07	25.4	33.86	0.15	0.39	0.55	0.27
RF M 5 all data	75.95	39.71	24.55	36.4	0.12	0.31	0.56	0.22
RF M 5 onsite	74.1	38.29	24.43	33.59	0.14	0.34	0.57	0.28
RF M 60 all data	74.2	35.07	25.24	34.2	0.14	0.39	0.55	0.27
RF M 5 all data 2CAM	75.32	39.15	26.48	33.57	0.12	0.32	0.53	0.28

Table A.1: RF. Average performance on test-set with prediction horizon 20, with weather circumstance sunny.







Figure A.2: Ramp-score, MAE per prediction horizon for test. data set



Figure A.3: MAPE, Ramp-score per prediction horizon for test. data set

Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score ↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP †
Persistence	96.92	64.26	59.66	43.59	NA	NA	NA	NA
Smart-persistence	100.43	61.31	82.37	40.46	NA	NA	NA	NA
RF M 10 all data	86.89	56.82	52.87	39.15	0.1	0.12	0.11	0.1
RF M 20 all data	89.55	58.82	54.93	40.88	0.08	0.08	0.08	0.06
RF M 30 all data	88.61	58.63	43.96	40.63	0.09	0.09	0.26	0.07
RF M 5 all data	87.78	56.3	45.19	39.1	0.09	0.12	0.24	0.1
RF M 5 onsite	89.54	61.22	41.24	41.95	0.08	0.05	0.31	0.04
RF M 60 all data	88.43	58.84	44.77	40.67	0.09	0.08	0.25	0.07
RF M 5 all data 2CAM	88.52	56.92	50.07	39.05	0.09	0.11	0.16	0.1

Table A.2: RF. Average performance on test-set with prediction horizon 20, with weather circumstance partially cloudy.

Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score ↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	27.39	18.51	62.57	14.52	NA	NA	NA	NA
Smart-persistence	26.87	18.36	81.59	14.36	NA	NA	NA	NA
RF M 10 all data	86.54	61.66	66.14	49.08	-2.16	-2.33	-0.06	-2.38
RF M 20 all data	87.45	61.34	66.39	48.66	-2.19	-2.31	-0.06	-2.35
RF M 30 all data	103.88	68.78	67.83	54.32	-2.79	-2.72	-0.08	-2.74
RF M 5 all data	96.8	67.8	67.38	53.76	-2.53	-2.66	-0.08	-2.7
RF M 5 onsite	138.12	91.42	76.49	72.83	-4.04	-3.94	-0.22	-4.02
RF M 60 all data	146.77	97.67	69.14	77.68	-4.36	-4.28	-0.11	-4.35
RF M 5 all data 2CAM	86.89	62.46	67.28	49.91	-2.17	-2.37	-0.08	-2.44

Table A.3: RF. Average performance on test-set with prediction horizon 20, with weather circumstance cloudy.



A.2 Artificial neural networks

Additional ANN results.



Figure A.4: MAE, MAPE per prediction horizon for test. data set



Figure A.5: Ramp-score, MAE per prediction horizon for test. data set



Figure A.6: MAPE, Ramp-score per prediction horizon for test. data set



Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score ↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	86.01	57.64	56.33	46.56	NA	NA	NA	NA
Smart-persistence	74.21	29.13	95.14	32.86	NA	NA	NA	NA
ANN M 40 all data	67.84	40.59	37.93	32.07	0.21	0.3	0.33	0.31
ANN M 60 all data	78.01	46.99	32.99	37.46	0.09	0.18	0.41	0.2
ANN M 10 all data	62.96	36.37	34.59	29.36	0.27	0.37	0.39	0.37
ANN M 10 all data 2CAM	66.69	42.44	34.06	33.35	0.22	0.26	0.4	0.28
ANN M 10 onsite	70.26	39.74	42.44	32.67	0.18	0.31	0.25	0.3
ANN M 10 onsite 2CAM	67.64	39.33	42.31	31.53	0.21	0.32	0.25	0.32
ANN M 20 all data	69.84	42.12	37.83	33.67	0.19	0.27	0.33	0.28

Table A.4: ANN. Average performance on test-set with prediction horizon 20, with weather circumstance sunny.

Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score ↓	SS-RMSE †	SS-MAE ↑	SS-MAPE ↑	SS-RAMP †
Persistence	96.92	64.26	59.66	43.59	NA	NA	NA	NA
Smart-persistence	100.43	61.31	82.37	40.46	NA	NA	NA	NA
ANN M 40 all data	87.33	55.45	73.43	40.26	0.1	0.14	-0.23	0.08
ANN M 60 all data	90.91	59.9	78.74	44.17	0.06	0.07	-0.32	-0.01
ANN M 10 all data	89.63	53.82	102.47	39.46	0.08	0.16	-0.72	0.09
ANN M 10 all data 2CAM	88.48	55.81	52.36	40.91	0.09	0.13	0.12	0.06
ANN M 10 onsite	74.72	46.87	160.91	33.19	0.23	0.27	-1.7	0.24
ANN M 10 onsite 2CAM	76.84	48.25	71.97	34.23	0.21	0.25	-0.21	0.21
ANN M 20 all data	87.76	54.88	59.14	40.05	0.09	0.15	0.01	0.08

Table A.5: ANN. Average performance on test-set with prediction horizon 20, with weather circumstance partially cloudy.

Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score ↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE †	SS-RAMP ↑
Persistence	27.39	18.51	62.57	14.52	NA	NA	NA	NA
Smart-persistence	26.87	18.36	81.59	14.36	NA	NA	NA	NA
ANN M 40 all data	56.44	43.15	63.75	34.5	-1.06	-1.33	-0.02	-1.38
ANN M 60 all data	55.99	43.35	63.45	34.47	-1.04	-1.34	-0.01	-1.37
ANN M 10 all data	70.74	54.9	71.35	43.73	-1.58	-1.97	-0.14	-2.01
ANN M 10 all data 2CAM	56.65	40.95	68.39	32.63	-1.07	-1.21	-0.09	-1.25
ANN M 10 onsite	40.74	29.65	103.19	23.4	-0.49	-0.6	-0.65	-0.61
ANN M 10 onsite 2CAM	67.35	43.65	64.62	34.65	-1.46	-1.36	-0.03	-1.39
ANN M 20 all data	59.23	45.3	65.85	35.91	-1.16	-1.45	-0.05	-1.47

Table A.6: ANN. Average performance on test-set with prediction horizon 20, with weather circumstance cloudy.

A.3 Long short-term memory

Additional LSTM results.



Figure A.7: MAE, MAPE per prediction horizon for test. data set



Figure A.8: Ramp-score, MAE per prediction horizon for test. data set



Figure A.9: MAPE, Ramp-score per prediction horizon for test. data set



Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score ↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	86.01	57.64	56.33	46.56	NA	NA	NA	NA
Smart-persistence	74.21	29.13	95.14	32.86	NA	NA	NA	NA
LSTM M 5 PXL	65.79	33.53	46.91	27.32	0.24	0.42	0.17	0.41
LSTM M 10 all data	64.56	35.34	39.47	27.68	0.25	0.39	0.3	0.41
LSTM M 5 all data	58.27	28.79	58.31	24.59	0.32	0.5	-0.04	0.47
LSTM M 5 all data 2CAM	59.72	28.72	32.95	24.01	0.31	0.5	0.42	0.48
LSTM M 5 onsite	68.59	35.31	40.24	27.83	0.2	0.39	0.29	0.4

Table A.7: LSTM. Average performance on test-set with prediction horizon 20, with weather circumstance sunny.

Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	96.92	64.26	59.66	43.59	NA	NA	NA	NA
Smart-persistence	100.43	61.31	82.37	40.46	NA	NA	NA	NA
LSTM M 5 PXL	76.98	46.57	54.8	31.89	0.21	0.28	0.08	0.27
LSTM M 10 all data	83.9	57.55	55.59	40.46	0.13	0.1	0.07	0.07
LSTM M 5 all data	81.25	50.7	68.36	35.41	0.16	0.21	-0.15	0.19
LSTM M 5 all data 2CAM	88.31	54.26	54.41	38.48	0.09	0.16	0.09	0.12
LSTM M 5 onsite	77.03	48.04	50.41	33.01	0.21	0.25	0.16	0.24

Table A.8: LSTM. Average performance on test-set with prediction horizon 20, with weather circumstance partially cloudy.

Model	RMSE ↓	MAE ↓	MAPE ↓	Ramp-score ↓	SS-RMSE ↑	SS-MAE ↑	SS-MAPE ↑	SS-RAMP ↑
Persistence	27.39	18.51	62.57	14.52	NA	NA	NA	NA
Smart-persistence	26.87	18.36	81.59	14.36	NA	NA	NA	NA
LSTM M 5 PXL	52.72	37.61	66.37	29.47	-0.92	-1.03	-0.06	-1.03
LSTM M 10 all data	64.78	49.61	66.32	39.55	-1.37	-1.68	-0.06	-1.72
LSTM M 5 all data	35.85	22.83	63.07	17.59	-0.31	-0.23	-0.01	-0.21
LSTM M 5 all data 2CAM	46.78	32.53	60.27	25.47	-0.71	-0.76	0.04	-0.75
LSTM M 5 onsite	57.98	34.35	65.76	26.94	-1.12	-0.86	-0.05	-0.86

Table A.9: LSTM. Average performance on test-set with prediction horizon 20, with weather circumstance cloudy.



Prediction horizon	LSTM M 10 all data	LSTM M 5 all data	LSTM M 5 all data 2CAM	LSTM M 5 onsite
4	n/a	n/a	n/a	n/a
5	n/a	n/a	0.97638	0.89572
6	n/a	0.61972	0.57584	0.54492
7	0.94459	0.48527	0.59795	0.64945
8	0.70199	0.34392	0.48086	0.58922
9	0.47267	0.17855	0.45788	0.60045
10	0.30276	0.09437	0.35869	0.55231
11	0.10685	0.03963	0.14345	0.30743
12	0.06426	0.01516	0.0755	0.21668
13	0.01873	0.00594	0.03699	0.14147
14	0.00845	0.00167	0.01824	0.06541
15	0.00665	0.00106	0.00918	0.08289
16	0.00448	0.00056	0.00578	0.07158
17	0.0024	0.00033	0.00207	0.06659
18	0.00137	0.00018	0.0009	0.04574
19	0.00066	7e-05	0.00029	0.03836
20	0.00031	5e-05	0.0001	0.02221

Table A.10: LSTM: Diabold-Mariono P value per horizon, compared with baseline model Persistence. Weather circumstance sunny. (n/a implies competing model is not performing better than the baseline).

Bibliography

- S.K. Aggarwal and L.M. Saini. "Solar energy prediction using linear and non-linear regularization models: A study on AMS (American Meteorological Society) 2013–14 Solar Energy Prediction Contest". In: *Energy* 78 (2014), pp. 247–256. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2014.10.012. URL: http://www.sciencedirect.com/science/article/pii/S036054421401161X.
- [2] Yiyang Ai, Yonggang Peng, and Wei Wei. "A model of very short-term solar irradiance forecasting based on low-cost sky images". In: vol. 1839. May 2017, p. 020022. DOI: 10.1063/1.4982387.
- [3] C. Crisosto et al. "One-Hour Prediction of the Global Solar Irradiance from All-Sky Images Using Artificial Neural Networks." In: *Energies* 11 (2018). DOI: 10.3390/en1112906.
- [4] Javier Antonanzas et al. "Review of photovoltaic power forecasting". In: *Solar Energy* 136 (June 2016). DOI: 10.1016/j.solener.2016.06.069.
- Philippe Blanc, Jan Remund, and Loïc Vallance. "6 Short-term solar power forecasting based on satellite images". In: *Renewable Energy Forecasting*. Ed. by George Kariniotakis. Woodhead Publishing Series in Energy. Woodhead Publishing, 2017, pp. 179–198. ISBN: 978-0-08-100504-0. DOI: https://doi.org/10.1016/B978-0-08-100504-0.00006-8. URL: http://www.sciencedirect.com/science/article/pii/B9780081005040000068.
- [6] Gary Bradski. "The openCV library". In: *Dr. Dobb's Journal of Software Tools* 25 (Jan. 2000).
- [7] Leo Breiman. "Random Forests". In: *Mach. Learn.* 45.1 (2001), pp. 5–32. DOI: 10.1023/A: 1010933404324. URL: https://doi.org/10.1023/A:1010933404324.
- [8] E Bristol. "Swinging door trending: Adaptive trend recording?" In: *ISA National Conference Proceedings* (Jan. 1990).
- Peter Brockwell and Richard Davis. An Introduction to Time Series and Forecasting. Vol. 39. Jan. 2002. DOI: 10.1007/978-1-4757-2526-1.
- [10] Pierrick Bruneau, Philippe Pinheiro, and Yoann Didry. "Data-driven forecasting of solar irradiance". In: (Jan. 2018).

- [11] John Canny. "A Computational Approach To Edge Detection". In: Pattern Analysis and Machine Intelligence, IEEE Transactions on PAMI-8 (Dec. 1986), pp. 679–698. DOI: 10.1109/TPAMI.1986.4767851.
- [12] Rémi Chauvin et al. "Cloud Detection Methodology Based on a Sky-imaging System". In: *Energy Procedia* 69 (May 2015), pp. 1970–1980. DOI: 10.1016/j.egypro.2015.03.198.
- [13] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System." In: *KDD*.
 Ed. by Balaji Krishnapuram et al. ACM, 2016, pp. 785–794. ISBN: 978-1-4503-4232-2. URL: http://dblp.uni-trier.de/db/conf/kdd/kdd2016.html#ChenG16.
- [14] Chi Chow et al. "Intra-hour forecasting with a total sky imager at the UC San Diego solar energy testbed". In: *Solar Energy* 85 (Jan. 2011), pp. 2881–2893. DOI: 10.1016/j.solener. 2011.08.025.
- [15] Stanley Chow, E.W.M. Lee, and Danny Li. "Short-term prediction of photovoltaic energy generation by intelligent approach". In: *Energy and Buildings* 55 (Dec. 2012), pp. 660– 667. DOI: 10.1016/j.enbuild.2012.08.011.
- [16] Paul Denholm and Robert M. Margolis. "Evaluating the limits of solar photovoltaics (PV) in traditional electric power systems". In: *Energy Policy* 35.5 (2007), pp. 2852– 2861. ISSN: 0301-4215. DOI: https://doi.org/10.1016/j.enpol.2006.10.014. URL: http://www.sciencedirect.com/science/article/pii/S0301421506003740.
- [17] Francis Diebold and Roberto Mariano. "Comparing Predictive Accuracy". In: *Journal of Business Economic Statistics* 20 (Feb. 1994), pp. 134–44. DOI: 10.1080/07350015.1995. 10524599.
- [18] Gunnar Farnebäck. "Two-Frame Motion Estimation Based on Polynomial Expansion". In: vol. 2749. June 2003, pp. 363–370. DOI: 10.1007/3-540-45103-X_50.
- [19] Cong Feng et al. "Short-term global horizontal irradiance forecasting based on sky imaging and pattern recognition". In: July 2017, pp. 1–5. DOI: 10.1109/PESGM.2017. 8274480.
- [20] Weijiang Feng et al. "Audio visual speech recognition with multimodal recurrent neural networks". In: May 2017, pp. 681–688. DOI: 10.1109/IJCNN.2017.7965918.
- [21] Chia-Lin Fu and Hsu-Yung Cheng. "Predicting solar irradiance with all-sky image features via regression". In: *Solar Energy* 97 (Nov. 2013), pp. 537–550. DOI: 10.1016/j.solener. 2013.09.016.
- [22] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www. deeplearningbook.org. MIT Press, 2016.
- [23] Chris Gueymard. "The Sun's total and spectral irradiance for solar energy applications and solar radiation models". In: *Solar Energy* 76 (Apr. 2004), pp. 423–453. DOI: 10.1016/j. solener.2003.08.039.

- [24] David Harvey, Stephen Leybourne, and Paul Newbold. "Testing the equality of prediction mean squared errors". In: *International Journal of Forecasting* 13.2 (1997), pp. 281–291.
 ISSN: 0169-2070. DOI: https://doi.org/10.1016/S0169-2070(96)00719-4. URL: http://www.sciencedirect.com/science/article/pii/S0169207096007194.
- [25] Detlev Heinemann, Elke Lorenz, and Marco Girodo. "Forecasting of solar radiation". In: Solar Energy Resource Management for Electricity Generation from Local Level to Global Scale (Jan. 2006).
- [26] Evgueni Kassianov, Charles Long, and Mikhail Ovchinnikov. "Cloud Sky Cover versus Cloud Fraction: Whole-Sky Simulations and Observations". In: *Journal of Applied Meteorology* 44 (Jan. 2005). DOI: 10.1175/JAM-2184.1.
- [27] Hussein A Kazem. "Solar Radiation, Temperature and Humidity Measurements in Sohar-Oman". In: *International Journal of Computation and Applied Sciences* 1 (Aug. 2016), pp. 15–20. DOI: 10.24842/1611/0003.
- [28] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. arXiv: 1412.6980 [cs.LG].
- [29] Andrew Kumler, Yu Xie, and Yingchen Zhang. "A Physics-based Smart Persistence model for Intra-hour forecasting of solar radiation (PSPI) using GHI measurements and a cloud retrieval technique". In: *Solar Energy* (Jan. 2019).
- [30] C.S. George L. Shapiro. *Computer Vision*. Prentice Books, Upper Saddle River, 2001.
- [31] Qingyong Li, Weitao Lyu, and Jun Yang. "A Hybrid Thresholding Algorithm for Cloud Detection on Ground-Based Color Images". In: *Journal of Atmospheric and Oceanic Technology* 28 (Oct. 2011), pp. 1286–1296. DOI: 10.1175/JTECH-D-11-00009.1.
- [32] Zhaoxuan Li et al. "A Hierarchical Approach Using Machine Learning Methods in Solar Photovoltaic Energy Production Forecasting". In: *Energies* 9 (Jan. 2016). DOI: 10.3390/ en9010055.
- [33] Andy Liaw and Matthew Wiener. "Classification and Regression by randomForest". In: *R News* 2.3 (2002), pp. 18–22. URL: http://CRAN.R-project.org/doc/Rnews/.
- [34] Helmut Lütkepohl. New introduction to multiple time series analysis. Berlin [u.a.]: Springer, 2005. XXI, 764. ISBN: 3540262393. URL: http://gso.gbv.de/DB=2.1/CMD?ACT= SRCHA&SRT=YOP&IKT=1016&TRM=ppn+366296310&sourceid=fbw_bibsonomy.
- [35] Sakshi Mishra and Praveen Palanisamy. "Multi-time-horizon Solar Forecasting Using Recurrent Neural Network". In: Sept. 2018, pp. 18–24. DOI: 10.1109/ECCE.2018.8558187.
- [36] John Murphy. "An Overview of Convolutional Neural Network Architectures for Deep Learning". In: 2016.
- [37] Nikos Paragios, Yunmei Chen, and Olivier Faugeras. *Handbook of Mathematical Models in Computer Vision*. Jan. 2006. ISBN: 978-0-387-26371-7. DOI: 10.1007/0-387-28831-7.

- [38] Richard Perez et al. "A New Simplified version of the Perez diffuse irradiance model for tilted surfaces". In: *Solar Energy* 39 (Dec. 1987), pp. 221–231. DOI: 10.1016/S0038-092X(87)80031-2.
- [39] Richard Perez et al. "Validation of short and medium term operational solar radiation forecasts in the US". In: *Solar Energy - SOLAR ENERG* 84 (Dec. 2010). DOI: 10.1016/J. SOLENER.2010.08.014.
- [40] Mashud Rana, Irena Koprinska, and Vassilios G. Agelidis. "Univariate and multivariate methods for very short-term solar photovoltaic power forecasting". In: *Energy Conversion and Management* 121 (2016), pp. 380–390. ISSN: 0196-8904. DOI: https://doi.org/10.1016/j.enconman.2016.05.025. URL: http://www.sciencedirect.com/science/article/pii/S0196890416303934.
- [41] Ibrahim Reda and Afshin Andreas. "Solar position algorithm for solar radiation application". In: Solar Energy 76 (May 2004), pp. 577–589. DOI: 10.1016/j.solener.2003.12.003.
- [42] *REL Solar Resource Glossary*. https://www.nrel.gov/grid/solar-resource/solar-glossary. html. Accessed: 12-11-2019.
- [43] REN. Renewables 2016 Global Status Report. Jan. 2016.
- [44] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Vol. 5. Jan. 2011. DOI: 10.1007/978-1-84882-935-0.
- [45] Maxime Taillardat et al. "Calibrated Ensemble Forecasts using Quantile Regression Forests and Ensemble Model Output Statistics." In: *Monthly Weather Review* 144 (Mar. 2016), p. 160301131220006. DOI: 10.1175/MWR-D-15-0260.1.
- [46] Jiacheng Tang et al. "An improved cloud recognition and classification method for photovoltaic power prediction based on total-sky-images". In: *The Journal of Engineering* 2019 (Mar. 2019). DOI: 10.1049/joe.2018.9249.
- [47] Soumya Tiwari, Reza Sabzehgar, and Mohammad Rasouli. "Short Term Solar Irradiance Forecast based on Image Processing and Cloud Motion Detection". In: Feb. 2019, pp. 1–6. DOI: 10.1109/TPEC.2019.8662134.
- [48] Loïc Vallance et al. "Towards a standardized procedure to assess solar forecast accuracy: A new ramp and time alignment metric". In: *Solar Energy* 150 (2017), pp. 408–422.
 ISSN: 0038-092X. DOI: https://doi.org/10.1016/j.solener.2017.04.064. URL: http://www.sciencedirect.com/science/article/pii/S0038092X17303687.
- [49] Cyril Voyant et al. "Machine learning methods for solar radiation forecasting: A review". In: *Renewable Energy* 105 (2017), pp. 569–582. ISSN: 0960-1481. DOI: https://doi.org/10. 1016/j.renene.2016.12.095. URL: http://www.sciencedirect.com/science/article/pii/ S0960148116311648.

- [50] Xin Wang et al. "Robust visual tracking via multiscale deep sparse networks". In: *Optical Engineering* 56 (Apr. 2017), p. 043107. DOI: 10.1117/1.OE.56.4.043107.
- [51] Bührke T. Wengenmayr R. "Sustainable energy concepts for the future." In: *John Wiley & Sons* (2011).
- [52] Clifford W. Hansen William F. Holmgren and Mark A. Mikofski. "pvlib python: a python package for modeling solar energy systems." In: *Journal of Open Source Software* 3(29) (2018). DOI: 10.21105/joss.00884.