CARL VON OSSIETZKY universität OLDENBURG

Fakultät II - Informatik, Wirtschafts- und Rechtswissenschaften
Department für Informatik
Abteilung Computational Intelligence

Masterarbeit

im Fach Umweltmodellierung (M.Sc.)

A Spatio-Temporal Approach for Solar Power Prediction using Recurrent Neural Networks

Vorgelegt von:
Theresa Schöbel
am 10. September 2019

Erstgutachter:	Prof. Dr. Oliver Kramer
Zweitgutachter:	M.Sc. Lars Elend

Zusammenfassung

In den letzten Jahren hat die Integration von erneuerbaren Energien, wie beispielsweise Strom durch Wind- oder Sonnenenergie, deutlich zugenommen. Dies ist unter anderem auf technologische Entwicklungen zurückzuführen, die gleichermaßen dazu beigetragen haben die Technik kostengünstiger als auch effizienter werden zu lassen. Um den Anteil an Sonnenenergie im Energiemix weiter zu erhöhen und auf diese Weise dafür zu sorgen weniger auf fossile Brennstoffe angewiesen zu sein, ist eine gesteigerte und sichere Integration der erneuerbaren Energien notwenig. Dabei stehen die Betreiber von Energiesystemen vor der Herausforderung die Stromversorgung und Netzstabilität trotz der Volatilität erneuerbarer Energien zu erfüllen. Aus diesem Grund ist es zwingend notwendig, vielversprechende Prognosemethoden für die Vorhersage an zu erwartendem Strom aus Quellen wie Sonne oder Wind stetig weiterzuentwickeln und zu analysieren. Methoden des maschinellen Lernens und künstlicher neuronaler Netzwerke bieten hierbei hohes Potential. Insbesondere Netzwerkarchitekturen wie Long Short-Term Memory (LSTM) Netzwerke, die zu den rekurrenten neuronalen Netzen gehören, tragen im Wesentlichen dazu bei, präzise Vorhersagen von Zeitreihen zu generieren, da sie in der Lage sind, längerfristige Abhängigkeiten sequentieller Daten zu erfassen. Solche Netzwerke wurden bereits genutzt um Windgeschwindigkeiten in einem raum-zeitlichen Verfahren zu prognostizieren. Folglich liegt eine Übertragung auf die Prognose von Solarenergie nahe.

Gegenstand der vorliegenden Masterarbeit ist die Entwicklung und Analyse von LSTMs für die kurzfristige Vorhersage von Photovoltaik-Leistung. In einem raum-zeitlichen Ansatz ist PV-Strom für eine einzelne Anlage in einen Zeithorizont von bis zu 5 Stunden vorhergesagt worden. Dafür sind Messwerte der Anlage selbst sowie umgebender PV Anlagen herangezogen worden. Eine Persistenzprognose diente als initiale Referenz, die es zu übertreffen galt. Die LSTM-Architektur, der Einfluss durch die Anzahl der benachbarten Systeme und deren Abstand sowie die Fähigkeit der LSTMs mit anderen maschinellen Lernmethoden zu konkurrieren, wurden sukzessive untersucht. Die erzielten Ergebnisse zeigen, dass bereits eine simple, einschichtige LSTM-Architektur in der Lage ist, kurzfristige Prognosen für eine und auch für mehrere PV-Anlagen gleichzeitig erzeugen zu können. Der Fehler der Prognosen wurde mit dem Root Mean Square Error (RMSE) bestimmt. In einem Vergleich mit anderen maschinellen Verfahren wie der Support Vector Regression (SVR) konnte gezeigt werden, dass LSTMs der SVR für die Prognose von Solarstrom mehrer Anlagen überlegen ist.

Abstract

In recent years, the integration of renewable energy sources like energy from wind power and solar irradiation has significantly increased due to more efficient and affordable techniques. In order to further increase the share of solar energy in the energy mix and thus reduce the reliance on fossil fuels, an extended and secure integration of renewable energies is necessary. The challenge of energy system operators thereby is to meet the requirements of ensured power supply and grid stability by keeping the power production and consumption in balance. Therefore, a development and analysis of most promising forecasting methods becomes inevitable. Deep neural networks and especially recurrent neural networks have shown major contributes to accurately predict time series. Most promising is the use of long short-term memory (LSTM) networks (a type of recurrent neural networks) on this task, because of their ability to capture long-term dependencies in sequential data. Applying those networks for the prediction in a spatio-temporal setup has been done for wind speed predictions so that a transfer to photovoltaic (PV) power forecasts is closely related.

This thesis develops and analyses the capability of LSTMs in a spatio-temporal setup for short-term PV power predictions. On the basis of PV measurements, predictions for mainly one PV system are carried out using additional data from neighboring PV systems. The initial reference to assess model performance is a persistence forecast that had to be outperformed. The LSTM architecture, the impact due to the number of neighboring systems as well as the ability of LSTMs to compete with other machine learning methods has been successively explored. The results show that a simple, one layered LSTM architecture is able to make short-term predictions for one PV system as well as more PV systems at the same time. Error values of the forecasts were determined with the root mean square error (RMSE). In a comparison with other machine learning methods like the support vector regression (SVR) it could be shown that LSTMs are superior to SVR for the prognosis of power of several PV plants.

Contents

1 Introduction

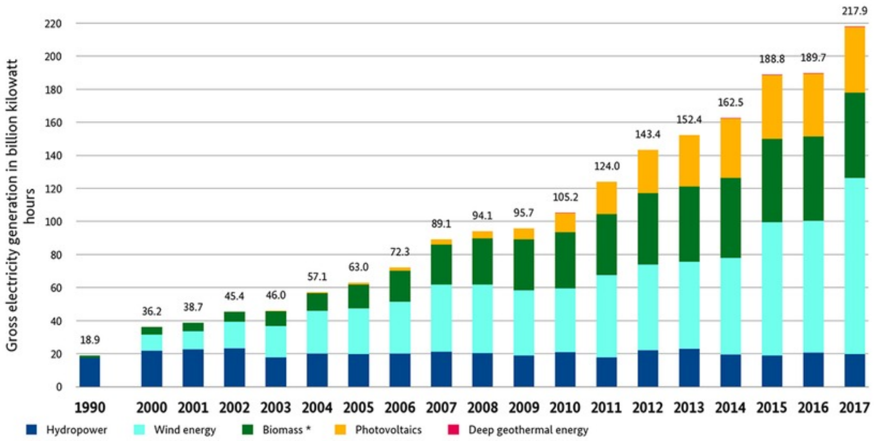
The subject of renewable energy plays an increasingly important role nowadays. Using wind power and solar irradiation to generate power, offers an alternative to nuclear power or fossil fuels. The energy sector is with the use of fossil fuels, such as coal, mineral oil and natural gas, one of the main sources of anthropogenic greenhouse gas emissions. Any development towards an extended use of renewable energy sources is one step further in terms of the energy transition – away from fossil fuels towards a sustainable energy supply.

One promising key player in this shift are photovoltaic (PV) systems. Besides the technical improvement in terms of their efficiency for example, decreasing prices also have contributed to their attractiveness. Nonetheless they depend on environmental conditions and their expected power production remains uncertain to some extent. One way to guarantee safe operation as well as integration into the electricity grid is, by accurately forecasting the expected amount of power using time series. From a methodically point of view the use of machine learning methods and artificial neural network architectures has gained popularity as they provide promising results [Wol17; Sal+17]. The aim of this master thesis therefore is to utilize recurrent neural networks (RNNs) for short-term power predictions. More precisely long short-term memory (LSTM) networks are to be developed, since they have been capable of handling temporal changes in various applications [AM17].

1.1 Motivation

The energy transition in Germany implies a structural shift from large single power plants towards a higher number of distributed, renewable energy generation units. Germany aims at increasing the share of renewables up to 65% of the total energy production by 2030. Currently the energy mix for public power supply splits into approximatly 38% renewables and 62% non-renewables [Bur18]. In order to achieve a rise of renewables and at the same time meet the present demand, it is (amongst others) necessary to increase the installed capacity of PV systems.

Photovoltaics account for the third largest share of electricity generated by renewable energies (see Fig. 1.1). 2017 Germany had a total amount of installed PV capacity of around 43 GW [ISE18; Bur18], producing around 38.4 TWh of electricity [Bur18]. Compared to 2016, this refers to approximately 2.3 GW newly installed PV systems in that year [Bur18]. The growth



* incl. solid and liquid biomass, biogas incl. biomethane, sewage gas and landfill gas as well as the biogenic fraction of waste, from 2010 incl. sewage sludge; BMWi based on Working Group on Renewable Energy-Statistics (AGEE-Stat); as at February 2018; all figures provisional

Figure 1.1: Share of the produced electricity by renewables from 2002 - 2017 in Germany. [Bur18]
.

had been possible due to evolving investments and a decrease in module prices. Overall in 2017 a total of around 16.2 billion euro has been invested in renewables, 10.6% of them into photovoltaics [Bad18]. Additionally, module prices have dropped by around 75% in total since 2006 [ISE18]. Those trends over the last decade show the potential of renewable energies and particularly of photovoltaics.

In order to fully use sources like wind and solar power as a replacement of fossil fuels some challenges remain. Wind power and solar irradiation are inherently volatile resources. Fluctuations due to unforeseen weather changes can lead to over- or under-production of power and thus become a challenge for the safe integration of renewables into the electricity grid. The european electricity network does only tolerate minimal frequency deviations and any exceedance needs to be compensated. This is either at the cost of compensatory energy (mostly of expensive and quickly available resources) or economic losses due to negative energy prices. In order to meet the requirements of ensured power supply and grid stability accurate power generation forecasts become indispensable.

1.2 Related Work

Many methods and models have been developed and tested to address the issue of solar power forecasting. The prediction of the electricity production to be expected from PV systems is a subject that has extensively been researched. The variety of approaches differ in terms of the used data, the type of modeling and the forecasted time horizon. Generally the approaches are classified into three categories: physical, statistical and hybrid methods.

An extensive overview of PV power forecasting methods is given in Chaturvedi and Isha [CI16] and Pelland et al. [Pel+13]. Both provide a comprehensive description of state-of-the-art solar and PV forecasting. The former includes a presentation of certain models and metrics to evaluate model performance, while the latter discusses the importance of measured weather and PV data. Physical methods use parametric PV and solar models, while statistical methods subsumize a number of regressive models and the use of artificial intelligence methods. Machine learning (ML) plays an interesting role thereby, since the use of ML algorithms on several problems has steadily increased. In addition to this artificial neural network (ANN) architectures have evolved and further improved predictions with learning algorithms. One advantage of ML or statistical approaches compared to physical ones is that they purely learn from data. By this researchers can analyze a system (like a PV system in this case) without the need of any additional information and the system can be seen as black box.

In the following some research on artificial intelligence (AI) and PV will be presented. Voayant et al. for example provide an analyzation on several statistical and ML techniques [Voy+17]. They examine 11 techniques and compare them in terms of error metrics. Mellit et al. present an introduction to the operation of AI methods and show several applications (such as forecasting or control) in the field of PV systems [MK08]. Another review on ANNs can be found at Yadav and Chandel [YC14]. They researched on the identification of suitable methods for solar radiation forecast and the accuracy of the presented methods with regard to parameter combinations, training algorithm and architecture configurations.

Further, more specified research on the topic of ANNs and PV also exists, due to the fact that there are different kinds of ANNs and certain optimization possibilities. Xiao et al. for example investigated on the number of hidden neurons for an accurate solar power prediction [Xia+17], whereas Mandal et al. combined wavelet transform and AI techniques to generate a one-hour-ahead power output prediction [Man+12]. One special type of ANN, which is particularly suitable for predicting solar electricity, are so-called RNN. Their advantage compared to feedforward neural network (NN) is the ability to reuse relevant past information and to incorporate them into current predictions. This way they can be developed to make predictions based on time-series and become capable of modeling dynamics of the data. Abdel-Nasser and Mahmoud published a LSTM network (a certain kind of RNN), to accurately forecast the output power of PV systems [AM17].

1.3 Research Question and Thesis Structure

The aim of this thesis is to investigate on solar power prediction using machine learning methods. By this it is part of environmental and machine learning research areas. Previous work by Björn Wolff [Wol17] investigated the usability of ML for solar power prediction and this thesis is a further specification. In contrast to this preliminary work, the present approach aims at utilizing ANNs for this task. The exploration on solar power prediction with LSTMS enables the analysis of a large amount of data and provides an alternative to statistical (like regressive or autoregressive models) and other machine learning methods (like support vector regression (SVR)) for solar power prediction. Analyzing the forecasting capability of LSTMs extends applicable approaches for this task. In most cases the historic data of a targeted system is used. Special to this approach is the additional use of surrounding PV systems power data, leading to a spatio-temporal experimental setting, as proposed in Woon et al. for wind power predictions [WOK17]. The focus will lie on regional short-term PV power forecasts, just like the foundational research by B. Wolff [Wol17]. Improvements in accuracy could lead to more reliable forecasts and thus to a safe, enlarged integration of renewable energy into to electricity grid.

Altogether the work concentrates on the following research questions:

- How are LSTMs capable of predicting short-term solar power generation?
- Is a spatio-temporal approach with LSTMs able to outperform predictions made with a persistence forecast?
- How does spatio-temporal information contribute to PV power forecasts?
- Are PV power predictions made with LSTMs comparable to other supervised machine learning methods such as SVR?

The thesis is structured as follows. First of all the theoretical foundations are covered. Therefore Chapter 2 concentrates on solar power prediction techniques. The following Chapter 3 then covers machine learning and ANNs, including architectures and training algorithms of ANNs. Those basics are followed by a practical application of LSTMs for solar power prediction. Thus in Chapter 4 are three experiments conducted, covering an analysis of LSTM network architecture (Section 4.3), a study on temporal and spatial impacts of the experimental setup (Section 4.4 and Section 4.5), and a comparison of LSTMs capabilities on solar power prediction with SVR (Section 4.6). The thesis closes with an overall summary, general remarks on the performance of LSTMs in the field of PV forecasting, and an outline of possible further research areas in Chapter 5.

2 Solar Power Prediction

The bright variety of PV forecast methods can be seperated from different points of view. First of all they can be devided by their temporal and spatial resolution. In terms of the forecasted time horizon the distinction can be made into intra-hour and intra-day forecasts. Some methods forecast most accurate for several minutes to one to two hours ahead, whereas other methods are more suitable to provide forecasts for several days in advance. The spatial resolution determines whether one single PV system is predicted, compared to a larger number of systems distributed across a geografical area. From another point of view any method can only be used, when the prerequisites are fulfilled. In other words, the available data defines which model is suitable. Some methods require the output from other models (such as PV, solar or atmospheric models), data from weather stations, satellites or PV system data, other rely purely on past PV measurements. Furthermore, models provide different kind of outputs, so the forecast can focus on the PV power output or on the so-called ramp rate, which describes how the power production changes over time [Pel+13]. Table 2.1 summarizes the connection between forecast focus and temporal resolution.

Table 2.1: Relation between forecasting time horizon and application of forecast [CI16].

	Intra-hour	**Intra-day**	**Day ahead**
Forecasted time horizon	15 min to 2 hours	1 to 6 hours	1 to 4 days
Focus of forecast	ramping events	PV power output	

2.1 Physical Methods

Solar power plants convert solar radiation into electricity, thus the main influencing variables in PV power forecasting are the incoming irradiation and the characterization of a PV system [Pel+13; CI16]. PV systems are either concentrating or non-concentrating, which specifies the relevant required irradiation. Concentrating panels concentrate the sunlight from a larger panel area onto a smaller area (the solar cell) with the help of an optical device [Fed18]. Those systems require direct normal irradiation (DNI), which describes the amount of solar radiation that is perpendicularly received by a surface area. Non-concentrating panels rely on global horizontal irradiation (GHI), which is composed of DNI in addition with diffuse irradiation (see Eq. (2.1)).

$$I_{\text{GHI}} = I_{\text{DNI}} + I_{\text{diff}} \tag{2.1}$$

The array plane of the PV panel thereby determines the area of received irradiation (G_i), which is only a fraction of GHI or DNI respectively. Another relevant variable in the forecasting process is the panel temperature T_m at the back of the module [Pel+13]. Both variables can be derived with the help of other forecast models and the knowledge about the PV system specification. Thus physical modeling techniques rely on solar or PV models to forecast PV power.

Irradiation values can be obtained by different approaches. Common sources are either numerical weather prediction (NWP) models or cloud observation models.

2.1.1 Numerical Weather Prediction

Numerical weather prediction models predict how the atmosphere will evolve. On the basis of an initial weather state (captured by measurements) the future condition is calculated with the help of physical equations (such as fluid dynamic or thermodynamic equations). Differences in models arise in terms of their spatial resolution. Generally the approximation of the atmospheric state is done on the basis of a three-dimensional grid that expands vertically from the earth's surface [Pel+13]. Models with larger distanced grid points consider the earth as a whole and are called *Global models*, whereas fewer spaced grid points result in higher resolutions and are used to capture regional areas more precisely (called *Regional models*) [CI16]. Over all, this process is highly computationally expensive, as hundreds of variables are predicted, with GHI being one of them. With GHI at hand, DNI can be derived using post-processing techniques, which also can increase forecasting accuracy [Pel+13]. As Chaturvedi et al. denoted, the focus of NWP models is on atmospheric processes rather than on radiation forecasts, thus "biased forecasts commonly result" [CI16]. Model runs of NWP models are started twice to four times a day[1] and the resulting time horizon of the prediction lies within 3 hours to 15 days ahead [Wol17].

2.1.2 Cloud Observation

Another physical approach for the determination of solar radiation is the identification of clouds and prediction of cloud movement. This can be done by taking pictures of the sky – either from satellites or with ground based so-called total sky imagers. The aim is to identify cloud patterns and structures in an image and compare two successive images with each other. Each pixel is bound to the information of a clouds optical depth (the specification of how thin or

[1] E.g., the ECMWF (European Centre for Medium-Range Weather Forecasts) NWP model is started at 00 and 12 UTC [Wol17].

thick a cloud is), which is referred to as cloud index. The detection of a cloud feature in two sequential images allows to define a cloud motion vector (CMV) that describes the speed and direction of the cloud movement. This method is quite short-termed, as cloud formation and movement can be very spontaneously. Furthermore the distance between clouds and the image taking object (thus the hight of a cloud field) confines the time that is available to capture cloud patterns. Thus CMV methods are best for time horizons from 3 minutes to 30 minutes ahead. The approach is based on the finding that cloud cover accounts for one of the main sources for irradiation extinction. According to Chaturvedi et al. both GHI and DNI can be forecasted through CMV [CI16].

2.2 Statistical Methods

Statistical solar power prediction refers to data driven methods. On the basis of historical data, autoregressive or ANN models are trained to predict PV power. Autoregressive methods are differentiated whether a time series is stationary or non-stationary.

A time series is defined as a (finite) sequence of observations that is either made at discrete or continuous time steps. If a time series does not markedly change within its fluctuations it is called stationary. This is true, when the statistical first and second order characteristics remain unchanged when shifted in time [CI16; BDC02]. Such data can be used for linear predictions, such as autoregressive, moving average or mixed autoregressive moving average models.

The counterpart of stationary are non-stationary time series. Such time series are time dependent, as their characteristics (like mean, variance and/or covariance) change over time. This is most commonly the case for environmental, economical or financial data [Li13]. The modeling of non-stationary time series focuses on homogeneous pattern. Examples might be seasonal reoccurring observations or trends, where some parts of the data are quite similar and thus are able to be treated as stationary data. Forecast models for such time series are ARIMA (auto-regressive integrated moving average) models. Further information for both kind of models can be found at Chaturvedi et al. [CI16].

AI techniques do not need the stationary/non-stationary differentiation in a manner that time series models do. Architectures of ANN that are or can be used to forecast PV power either recognize pattern or perform a regression task [CI16]. The former results in a classification, while the latter maps inputs (such as PV measurements for example) to outputs. The actual explanation of ANN will be discussed in Chapter 3.

2.3 Accuracy Metrics and Performance Comparison

There are several ways to capture the accuracy of a forecast. The necessity of calculating the error between a measurement and a predicted value is given in order to analyze the reliability and capability of the used method. Furthermore an evaluation of the model output will help to estimate the suitability for an application purpose or whether the forecast is useful at all. The statistical metrics for this task can be differentiated by their focus, because each of them expresses different components of the error. The most common ones for evaluating the forecast quality in terms of PV power prediction will shortly be presented.

The mean absolute error (MAE) gives a first insight in the accuracy of a forecast, by expressing the average magnitude of the forecast errors [Pel+13]. Some more information can be gained with the help of the mean bias error (MBE or BIAS). The BIAS is the average forecast error and intends to specify whether the model produces systematic over- or under-predictions. Thus in comparison to MAE, it includes the direction of an error. Both are defined as follows (MAE and BIAS respectively), where N denotes the total number of observations and forecasted values and the symbol ˆ defines predicted values versus observed measurements:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N} |\hat{y}_i - y_i| \tag{2.2}$$

$$\text{BIAS} = \frac{1}{N}\sum_{i=1}^{N} \hat{y}_i - y_i. \tag{2.3}$$

The mean squared error (MSE) and root mean square error (RMSE) are two more metrics for assessing the average deviation between two time series. Both calculate the average of the squared differences between prediction and observation, whereas the RMSE is the square root of the MSE:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N} (\hat{y}_i - y_i)^2 \tag{2.4}$$

$$\text{RMSE} = \sqrt{\text{MSE}}. \tag{2.5}$$

By squaring the deviation, large errors gain more impact. This way the MSE is a useful measure for grid operators, since the amount of deviation contributes disproportionately to the costs for grid stability [Pel+13]. The RMSE on the contrary is more suitable for the assessment of prediction performance, as pulling the root returns the values to their original range [Wol17].

Another way of telling whether a method is more or less capable of providing qualitative forecasts can be done by comparing their accuracy metrics against each other. A comparison can be accomplished by calculating the so-called skill score. The skill score describes the relative improvement of any given accuracy metric with regard to a reference model. The difference between the forecast and the reference is divided by the difference between a perfect forecast and the reference, resulting in a relative improvement:

$$\text{skill score} = \frac{\text{metric}_{\text{forecast}} - \text{metric}_{\text{reference}}}{\text{metric}_{\text{perfect forecast}} - \text{metric}_{\text{reference}}}. \tag{2.6}$$

Skill scores range between $(-\infty, 1]$, where positive values indicate improvement, negative values deterioration and 0 alikeness. E.g., a skill score of 0.32 represents an improvement of 32% of one metric compared to the reference. [Pel+13; Voy+17; Mur88]

Usually any reference model is easy to accomplish. The term easy thereby refers to certain characteristics of a reference model that produces a baseline forecast. Those characteristics are simplicity, reproducibility and calculation rapidity [Pel+13]. A common example is a so-called persistence forecast. The approach of a persistence forecast relies on the assumption that any given condition will remain the same the next time step [Pel+13]. Taking the weather as an example, a persistence forecast would predict sunshine for the next moment, if the sun is shining right now. Thus a persistence forecast can be written as follows:

$$x(t+1) \coloneqq x(t). \tag{2.7}$$

2.4 Concluding Remarks

This part covered an introduction of several PV forecast methods. Physical as well as statistical approaches have been presented and appropriate error measurements have been outlined. Those methods have different strenghts compared to each other. In terms of the forecasted time horizon, NWP predictions are most appropriate for long-term predictions, whereas statistical approaches outperform models that rely on NWP outputs, for shorter periods of time [Pel+13]. Satellite-based CMV models take an intermediate place in between very short-termed and long-termed prediction methods. The shortest temporal horizon can be covered by a persistence forecast, which is one of the most straight-forward, reproducable and accessible methods. Therefore it is often used as a baseline and will serve as such in the present master thesis.

The preceding work by [Wol17] assessed the usability of machine learning for PV power forecasting, by comparing SVR with physical modeling (namely using forecasted irradiance from NWP and CMV models as input in a parametric PV model and PV measurements for a persistence model) [Wol17]. Thus those methods have been introduced more thoroughly. Using the same data like B. Wolff (which will be described in Section 4.1) the usability of another machine learning approach will be investigated in this work. In the following an introduction and explanation for machine learning techniques for PV power forecasting will be given.

3 Artificial Neural Networks

Artificial neural networks (ANNs) are part of artificial intelligence (AI) and by this part of machine learning (ML). The aim of ML is to teach machines to extract information from (mostly huge amounts of) data, in order to gain knowledge [Dey16]. The data thereby is separated into an input space X and an output space Y. Each data point $x \in X$ is called *pattern* and each pattern has certain characteristics, which are called *features.* The amount of features $x_1, \ldots, x_d$ is the dimensionality of X and each pattern holds a corresponding *label* $y \in Y$ in the output space. [Wol17]

The objective of learning algorithms is to find a function that maps input to output: $f : X \rightarrow Y$, in order to predict a label y on a pattern x that the algorithms has not seen before. Therefore any investigated dataset of input-output pairs is split into a training set P and test set R. The first one represents that part of the data the algorithm is shown to adapt, while the second one is used to tune and optimize model and parameter [Wol17]. Sometimes a validation set Q is separated from the training set P, to generate an unbiased evaluation of the model fit during the training process [Gra12]. So P influences the model directly, whereas Q influences the model indirectly.

Learning is usually divided into *supervised* and *unsupervised* learning, referring to the influence on the learning process. The amount of information about the pattern label association available is the decisive factor. Supervised learning algorithms are build upon the information of pattern label relation and thus are commonly used for regression and classification tasks. Unsupervised learning algorithms, on the other hand, miss out on any labeling information and aim at finding similarities in the data, in order to build cluster or reduce the dimension of the data [Wol17].

3.1 Foundations of ANNs

Artificial neural networks refer to nature inspired computing systems. The aim with the development of ANNs has been to use the foundational concept of a biological neuron to solve complex tasks in contrast of replicating the biological system [BH00].

One of the first ANNs is the so-called simple perceptron (SP) (developed by Rosenblatt 1958), which basically is the implementation of the signal transfer from one neuron to another. A biological unit of the nervous system (a neuron) is separated into axon with cell body and dentrites. Several dentrites are connected, resulting in multiple incoming signals of one neuron.

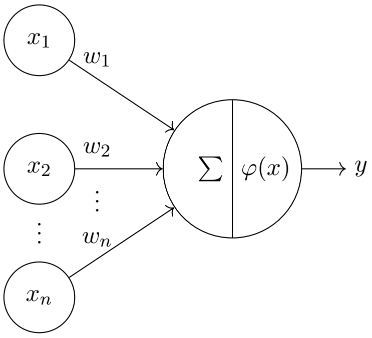


Figure 3.1: Schematic representation of a simple perceptron with $x_1, \ldots, x_n$ inputs and one output.

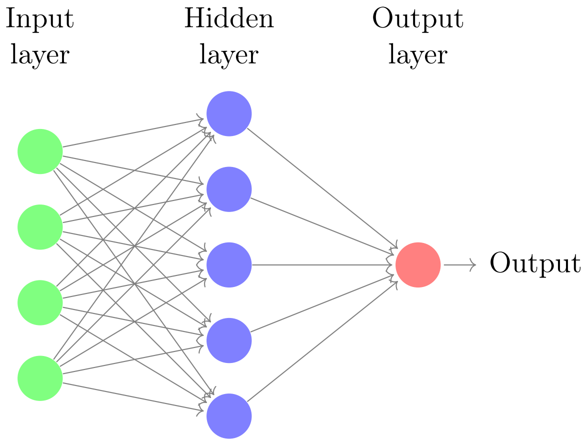


Figure 3.2: Architecture of a multilayer feedforward perceptron with several input and hidden layer nodes. (adapted from [Fau06])

The signal transfer is initiated through an electrical impulse traveling within the dentrites towards (and in form of neurotransmitters into) the synapsis and depending on the magnitude of the impulse, the neuron is forced to transmit a signal itself. Artificially the axon, dentrites and synapsis are represented by nodes (x) and weights (w) respectively. [BH00]

For a neuron to be activated (both the biological and artificial one), a threshold has to be exceeded, otherwise the connection is inhibitory. Mathematically this can be expressed by an binary function, with 1 indicating "on", 0 indicating "off" and b being the threshold (see Eq. (3.1)).

$$\varphi(x) = \begin{cases} 1, & \text{if } \sum_{i=1}^{n} w_i x_i \geq b \\ 0, & \text{else} \end{cases} \tag{3.1}$$

The output of $\varphi(x)$ is dependent on a function (here a weighted sum), which combines the incoming inputs $x_1, \ldots, x_n$ to form one "net input" [BH00].

The SP is also referred as a single layer network, consisting of one neuron with n inputs and a single output (see Fig. 3.1). This limitates the network in its application possibilites. With only one node to compute the activation of the neuron, the SP can only solve linear separable[1] problems.

An extension of the networks' architecture, such as additional nodes in between the input and the output layer, enables to cope with non-linear separable tasks. One example for such networks, with so-called *hidden layers* and *hidden nodes*, is a multilayer perceptron (MLP) (see Fig. 3.2). All connections between nodes are directed to successive layers and the output of

[1] E.g., Data points in a two-dimensional space can be separated by a straight line. Those points have the property of linear separability.[Kra09]

each neuron is dependent on preceding ones. So the overall ANN solution depends on previous hidden layer neuron outputs and the forward directed flow of information leads to the name *feedforward network*.

3.2 The Backpropagation Algorithm

Artificial neural networks are modified during the training process in order to solve a specific task correctly. In supervised learning, adaptation of the network is initiated with the help of the correct solution (cf. Chapter 3). The weights of an ANN are the changeable, adaptable (i.e., learnable) factor. Learning rules determine how these weights should be adjusted, so that the ANN solution comes closer to the correct one.

The so called backpropagation (BP) algorithm is one well known algorithm for (supervised) training of ANNs [Wer74; HRW86]. Backpropagation is a method to change the weights with the goal to find an optimal set of weights for generating the desired ANN solution. In a MLP with BP (called *backpropagation network*) the data is fed forward, whereas the error is propagated backwards, from output over hidden to input layer [BH00; Kra09; Gal15]. This is how the name backpropagation comes about.

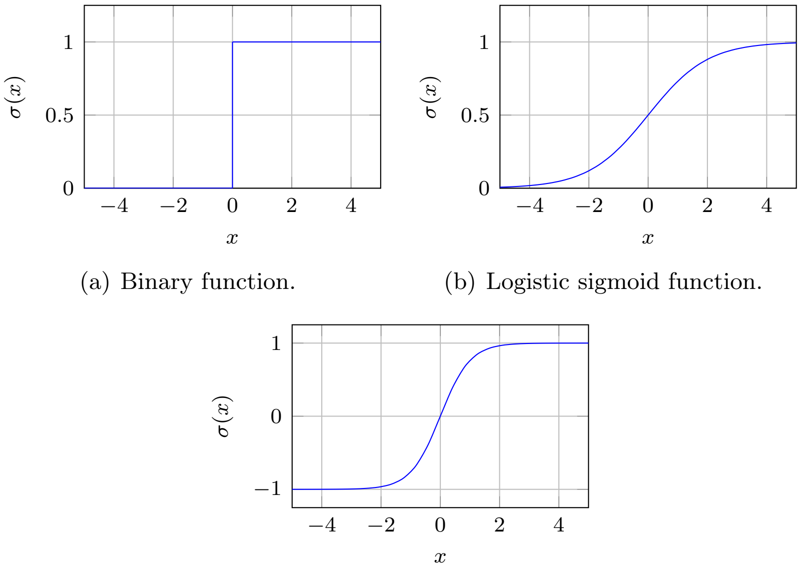
In order for the error backpropagation, the magnitude of the error is determined by comparing the predicted $\hat{y}$ with the correct solution y. The weights are then updated by determining the direction in which the weights have to be altered, so that the error gets minimized. Hence why BP uses an optimization procedure of first order to calculate the direction of change. The gradient descent (GD) is one example of such an optimization procedure. Generally, GD maximizes or minimizes a n-dimensional function [Kri07]. The gradient is the deviation of each dimension of a function, thus the gradient g of a (differentiable) two dimensional function $f(x_1, x_2)$ is defined as follows:

$$g(x_1, x_2) = \nabla f(x_1, x_2). \tag{3.2}$$

The negative norm of g $(-||g||_2)$ refers to the direction of the steepest descent, i.e., the direction of a minimum. Transferred to ANN and BP, the algorithm aims at changing the weights $w \in W$ in the direction of the steepest descent. The error thereby can be considered as a function of all weights $E(W)$ and minimizing the error refers to deviating the error function to w: $\frac{\partial E}{\partial w}$. Formally speaking, the goal is to approximate $\forall p : E_p \approx 0$, with $p \in P$ being a sample of the training set P, whereas P is a subset of the input set X: $P \subset X$.

$$\Delta W = -\eta \nabla E_p(W), \tag{3.3}$$

with η being a proportionality constant called the learning factor. A typical function to determine the error is the sum of squared differences as defined in Eq. (2.4) [Kra09; Bia+17].



(a) Binary function.

(b) Logistic sigmoid function.

(c) Hyperbolic tangent function.

Figure 3.3: Commonly used activation functions $\varphi(x)$. From top to bottom are displayed (a) the binary threshold function, (b) the logarithmic sigmoid function $\sigma(x)$ and (c) the hyperbolic tangent function $\tanh(x)$.

There are three variants of the GD algorithm for optimizing ANNs [Rud16]. The variant explained above is also referred to as batch gradient descent (BGD). In order to perform one weight update, the gradient of the cost function E is calculated for the entire dataset. Thus in each iteration the gradients for each sample are computed and summed. This can be highly, computationally costly for large datasets, as the sum scales linearly with the training dataset size. One alternative to BGD, is the stochastic gradient descent (SGD). In contrast to BGD, this version uses only one sample of the training data set (uniformly at random) at each iteration to update the weights [Ber+15]. The third version is the mini batch SGD that uses a number of samples instead of one, compared to SGD [Rud16].

The activation function $\varphi(x)$ is responsible for the output of each neuron, it can be inserted into the error function. Commonly the sigmoid or logistic function $\sigma(x)$ and the hyperbolic tangent function $\tanh(x)$ are used for the activation (see Fig. 3.3), because of their complaisant first derivation:

$$\sigma'(x) = \frac{1}{1+e^{-x}}, \tag{3.4}$$

$$\tanh'(x) = 1 - \tanh^2(x). \tag{3.5}$$

With that, the adjustment of the weights by the partial derivation from E to w is nothing more than the use of the chain rule to calculate the size of the weight adjustment δ [Guo13]. With the use of $\sigma(x)$ as activation function, a weight update for a hidden layer neuron k and an output layer neuron z is obtained as follows[2]:

$$\delta_z = (y_z - \hat{y}_z)\hat{y}_z(1 - \hat{y}_z), \tag{3.6}$$

$$\delta_k = \hat{y}_k(1 - \hat{y}_k)\sum_z(w_{zk}\delta_z). \tag{3.7}$$

This shows, that exactly like the output of a MLP depends on preceding outputs of each neuron in forward direction, the same is true backwards for the change of weights. Each output neuron contributes to the error of a hidden layer neuron [Guo13].

Usually training ends when one of the following criteria is met: (i) the error has reached a tolerable value $\epsilon \leq \gamma$, (ii) the gradient of error is smaller or equal to a defined value $\nabla E \leq \zeta$. The learning rate η thereby impacts the course of training. A high value for η ensures that the training progresses quickly, as the weights are changed significantly from one cycle to the next. The downside of large values for η is that it may not be possible to find an optimal set of weights. The changes of the weights during training may be too large and as a consequence exceed a minimum. A very small η on the other hand leads to a very slow but continuous convergence towards a minimum. In general, the error decreases with an increasing number of trainings cycles, also called *epochs*. An epoch describes a complete forward and backward pass over the entire training set [Ben12]. [BH00]

3.3 Recurrent Neural Networks

Recurrent neural networks have especially been developed to process sequential data. While feedforward neural networks (such as MLPs) can and have been successfully developed to model time series (see [Bia+17]), they might not be as efficient and accurate as RNNs for that particular task. Multilayer perceptrons are fixed in their number of input and output nodes, while RNNs adapt dynamically to the length of a sequence or time series. In order to accurately model time series with MLPs, a a-priori knowledge of the system is required for the choice of the number of inputs, representing the length of a sequence [Bia+17]. This can be problematic in time series modeling, when the temporal dependencies of interest exceed the chosen number of inputs.

Recurrent neural networks provide a more flexible approach. Through recurrent connections they are capable of creating dependencies from previous results to the next. Those connections

[2] The single steps of the derivation can be looked up at [Kra09].

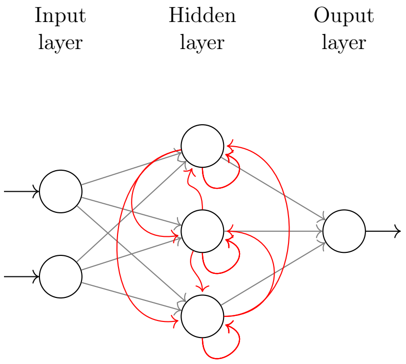


Figure 3.4: Architecture of a recurrent neural network. The hidden layer nodes are connected through recurrent connections (red), allowing the informaion to persist.

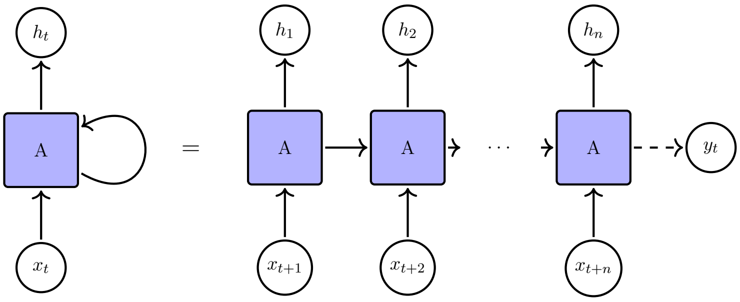


Figure 3.5: A simple *rolled / fold* RNN layer and its corresponding *unrolled / unfold* over time. x_t refers to the current input of time step t, h_t to the output of that time step and A to recurrently connected unit of the network. The output of the previous time step is passed to the next time step and used as additional input. Either the last hidden state h_t forms the output y_t, or in case a dense layer is following h_t is passed and processed to form y_t.

allow the information to loop around and underlying temporal dependencies in time series or other sequential data can be detected. The results of one time step remains by being memorized in a hidden state variable $\boldsymbol{h}_t$. Thus each step t of a sequence or time series of lenght T gets processed and stored in the same way [Bia+17].

Other than in MLPs, the connections between nodes of a hidden layer in RNNs can be to a node itself, to nodes of prior or within the same layer (see Fig. 3.4). Those hidden layer connections can get depicted as one block (a RNN unit) with one self-influencing link (see Fig. 3.5). Such an abstraction allows a less complex illustration of the flow of information over time. This is also known as unrolled or unfold RNN.

A walk-through to get from input to output of a *simple* RNN is composed as follows: In a first step the input of a time step t is used to compute an ouput $\boldsymbol{h}_t$, which then is used to generate results of the succeeding time step $t+1$. So the input of one step assembles of the current input $\boldsymbol{x}_t$ and the hidden state of the previous time step $\boldsymbol{h}_{t-1}$. Those inputs are concatenated and multiplied with a weight matrix $\boldsymbol{W}$. So the update of a hidden state is defined as in (Eq. (3.8)).

The overall output of a RNN depends on the last layer of the network. If it is a RNN layer, the overall output is the last hidden state update: $y_t = h_t$. Typically the last RNN layer (if several RNN layers are used) is followed by a dense layer (a fully connected feed-forward layer) to compute an overall output y_t as defined in Eq. (3.9).

$$h_t = \sigma(\boldsymbol{W}_{hi} \cdot \boldsymbol{x}_t + \boldsymbol{W}_{hh} \cdot \boldsymbol{h}_{t-1}), \tag{3.8}$$

$$y_t = \sigma(\boldsymbol{W}_{oh} \cdot \boldsymbol{h}_t), \tag{3.9}$$

with $\boldsymbol{W}_{hi}$ being the input weight matrix, $\boldsymbol{W}_{hh}$ the hidden state matrix, $\boldsymbol{W}_{oh}$ the output weight matrix and σ a sigmoid function for activation of the neurons.

3.4 Backpropagation Through Time

An unfolded RNN can be considered as a MLP with many layers (also refered to as *deep* neural network) (cf. Section 3.3) [Guo13]. Thus RNNs can exactly like MLPs be trained with BP. While for MLPs the gradient for adjusting the weights is calculated independently for one input (cf. Section 3.2), for RNNs the gradient is computed for each individual time step. For a sequence of length T, this means that the error is the sum of all time steps t:

$$\frac{\partial E}{\partial w} = \sum_{1 \leq t \leq T} \frac{\partial E_t}{\partial w}. \tag{3.10}$$

The weight change of each time step contributes evenly to one update of the weights. This is why BP for RNNs is also called backpropagation through time (BPTT).

One issue that arises during training RNNs is the vanishing or exploding gradient problem[3]. Such event occurs with regard to long term components, where the norm of the gradient exponentially grows or goes to 0 [PMB12]. This happens, due to the characteristics of the used activation function, whether $|\varphi' > 1|$ or $|\varphi' < 1|$. The sum in Eq. (3.10) can be extended to:

$$\frac{\partial E}{\partial w} = \sum_{1 \leq t \leq T} \frac{\partial E_t}{\partial h_t} \cdot \frac{\partial h_t}{\partial h_T} \cdot \frac{\partial h_T}{\partial w}, \tag{3.11}$$

whereas the partial derivates of one state with regard to the previous state can be factorized as follows:

$$\frac{\partial h_t}{\partial h_T} = \frac{\partial h_t}{\partial h_{t-1}} \cdot \ldots \cdot \frac{\partial h_{T+1}}{\partial h_T} = {\varphi'}_t \cdot \ldots \cdot \varphi'_{T+1} \tag{3.12}$$

[Bia+17]. The case $|\varphi' < 0|$ leads to a convergence to 0 with an increase in the difference of $t - T$ and consequently to a vanishing gradient. The opposite is the case for $|\varphi' > 1|$, leading

[3] This problem can equally arise in Deep MLPs.

to an exploding gradient and an unstable model. In either case, the model is unable to capture underlying distanced temporal dependencies [PMB12].

In general some strategies have been developed to handle and reduce the effect of the phenomenon (for example see [Bia+17] for details), whereas in particular the development of LSTMs adressed this issue.

3.5 Long Short-term Memory Networks

Long short-term memory networks refer to a more refined architecture of the beforehand described simple RNNs. Thus they equally belong to RNNs. They were first introduced by Hochreiter and Schmidthuber in 1997 [HS97] and have shown accurate performance in modeling both long and short term dependencies of sequential data [Bia+17]. In principle, they were developed specifically for the purpose of long-term dependencies, in order to overcome the vanishing/exploding gradient problem.

Special to them is their inner implementation for the computation of the hidden state. Instead of using one activation function (as done in the simple RNN), the computation is separated and done with the help of additional layers [Bia+17]. Therefore a cell state $\boldsymbol{C}_t$ is introduced and modified during the computation of the hidden state variable. This leads to the name memory cell for a LSTM unit (see Fig. 3.6). Structurally three so-called *gates* refine the processing of the input, by adding or removing information to the cell state: the input, output and forget gate. Each gate fulfills its own task:

- The *forget gate* f_t decides which information of the previous time step should be omitted.

- The *input gate* i_t decides which parts of the cell state should be updated. In a first step the parts of $\boldsymbol{C}_t$ to be changed are selected and in a second step new candidates ($\tilde{\boldsymbol{C}}_t$) for each part of $\boldsymbol{C}_t$ are calculated. The combination of the first and second step results in an update of $\boldsymbol{C}_t$.

- The *output gate* o_t decides what parts of $\boldsymbol{C}_t$ will form the output, so the output $\boldsymbol{h}_t$ will not be perturbated by irrelevant inputs [HS97].

Consequently, the activation and update of the cell state can formally be described as follows:

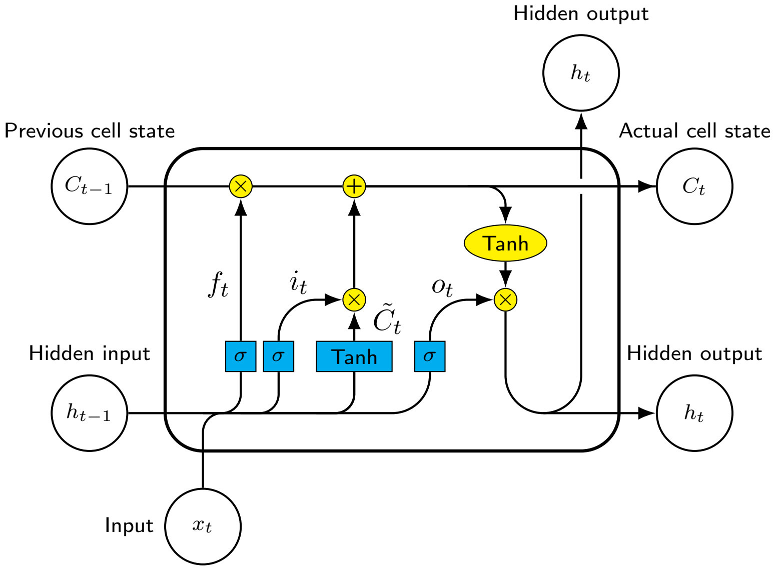


Figure 3.6: A LSTM unit. Blue rectangles indicate a layer operation, while yellow circles / the ellipse represent a pointwise operation to combine outputs of several layers with the cell state. The symbols × and + illustate multiplication, and addtion respectively.
The hidden input (h_{t-1}) and input (x_t) are each time concatenated, before activated by different layers. The activation of the respective gates (f_t, i_t, o_t) are commonly performed by a sigmoid function (σ), whereas tanh symbolizes once the activation to generate new candidates ($\tilde{C}_t$) by a hyperbolic tangent function and another time to update (C_t) [Bia+17]. (adapted from [Leo18])

$$f_t = \sigma(\boldsymbol{W}_f \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t]), \tag{3.13}$$

$$i_t = \sigma(\boldsymbol{W}_i \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t]), \tag{3.14}$$

$$\tilde{\boldsymbol{C}}_t = \tanh(\boldsymbol{W}_C \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t]), \tag{3.15}$$

$$C_t = f_t \cdot \boldsymbol{C}_{t-1} + i_t \cdot \tilde{\boldsymbol{C}}_t, \tag{3.16}$$

$$o_t = \sigma(\boldsymbol{W}_o \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t]), \tag{3.17}$$

$$h_t = o_t \cdot \tanh(\boldsymbol{C}_t), \tag{3.18}$$

whereas for simplicity $[\cdot, \cdot]$ replaces a concatenated multiplication. E.g., Eq. (3.13) can alternatively been written as $f_t = \sigma(\boldsymbol{W}_f \cdot \boldsymbol{h}_{t-1} + \boldsymbol{W}_f \cdot \boldsymbol{x}_t)$. Thus the vectors $\boldsymbol{x}_t$ and $\boldsymbol{h}_{t-1}$ are concatenated and multiplied with the weight matrix $\boldsymbol{W}_f$.

Thus LSTMs overvome the vanishing/exploding gradient problem due to the implementation of gates. The gradients can persist unchanged and flow through the memory, as long as the gates are shut [Sut13].

3.6 Hyperparameter Optimization

There are three possible scenarios for outcomes after the end of training ANNs: the model is *underfitted*, *overfitted* or well fitted. A model is considered to be sufficiently or well fitted if it has learned to generalize from known training data and is able to predict unknown data most accurate. In such a case, both the loss on the training and the test set become minimal. Curves of the losses then converge to zero over the course of epochs. This ability to adequately generalize does not apply to neither an under- nor an overfitted model.

Underfitting is revealed by a downward trend of both training and validation loss, but the performance on the training set is better than on the validation set. Any room for improvements on training and validation loss to approach zero indicates underfitting. Wether or not the accuracy is sufficient in such case highly depends on the data and the purpose of the model.

Overfitting occurs when the model has adapted too strongly on the training data. That is, the model is not able to generalize from what it has learned. Consequently, an overfitted model can produce virtually perfect predictions for the known training data, but fails on predictions with little error on unknown data. This becomes apparent when the loss on the training set becomes minimal, but the loss on the test set begins to increase again after a decline.

Those outcomes (under- and overfitting) indicate the direction in which the researcher needs to adapt the model he or she is building. Each learning algorithm has certain parameters called *hyperparameters*. The choice of these defines the actual learning algorithm [BB12]. For ANNs, for example, this is the number of neurons, epochs, batch size, or when using gradient descent as optimization procedere the learning rate η (cf. Section 3.2). Values derived during training do not count as hyperparameter (e.g., the weights). For LSTMs some hyperparameters seem to be more important than others with regard to the performance to minimize the generalization error [RG17; BB12]. All in all, hyperparameters determine whether a neural network is able to produce "good or even state-of-the-art results" [RG17]. The search for good or even optimal values for any hyperparameter is called *hyperparameter optimization* and can be done manually, as a *grid search* or a *random search*. When the search is done manually, a number of individually determined values is tested. In grid search, a set of values for each hyperparameter is defined and every possible combination is tested. Grid search is an extensive search, as the number of joint values increase exponentially with the number of hyperparameter values. This is why the method suffers from the *course of dimensionality*. In contrast, random search uses randomly drawn examples from a defined set of values for each parameter for a certain number of iterations. Although not all possibilities are examined by this method, it achieves almost as good results as grid search with lower computational effort. [RG17; BB12]

As mentioned before, some hyperparameters are more or less important than others. As so, a short recap corresponding to some hyperparameters and their influence on the neural network will be given.

- *Nodes.* The number of nodes defines how wide a network is [RG17]. More nodes increase the complexity of the network or rather extend the model's capacity [Ben12]. According to [Ben12] and [RG17] the impact of this hyperparameter is low and does not influence the generalization performance of a network much.

- *Batch Size.* The batch size represents that chunk of the training data used during one epoch to update the weights. A large batch size for example requires more memory, which results in an improvement of computational time. A small batch size provides the opposite. Additionally, according to [Ben12] variance in this hyperparameter "does not hurt generalization performance much".

- *Epochs.* The number of epochs represent the number of iterations for training a neural network through the entire training dataset. With regard to computing time and effort, the number of epochs that is desirable is the one that is both as small as possible and as large as necessary to learn the requested function.

- *Optimizer.* The optimization algorithm, as explained in Section 3.2, is responsible for minimizing the error between predicted and correct solution. According to [RG17] the impact on generalization capabilities of the network is high. On the one hand the algorithm itself plays a role, on the other hand settings of the algorithm (i.e., the learning rate) are important.

- *Dropout.* Dropout is a regularization method used to prevent and address overfitting. By randomly dropping out nodes and their connections it simulates training several ANNs with different architectures (e.g., number of nodes per layer) in one ANN. By this less computational effort is required, as one instead of many ANNs is trained. By defining a dropout, the probability of masking a unit is determined. In RNN dropout can be applied to omit the same neurons and connections at each time step. Thus a dropout of 0.2 refers to 20% probability of a neuron being left out. [GG16]

This is only a selection of hyperparameters and this list does not claim to be exhaustive[4]. As a concluding remark, hyperparameter and the choice of their values determine the model performance. Each influences training and output in one or another direction. For example an underfittet model could be addressed with further optimization of the hyperparameters in order to obtain more accurate results, whereas an overfittet model could potentially be compensated by a reduced number of epochs or the use of a dropout rate.

[4] See [RG17] for example, for further information on hyperparameters and their impact.

4 LSTM-based Solar Power Prediction

One goal of this thesis lies within testing the capability of LSTMs for solar power prediction. This puts the prediction in the area of statistical modeling, more precisely in the area of supervised learning, as the LSTMs will be trained in a supervised manner. As described in Section 2.2, PV power can be forecasted regressively, previous measurements or solar irradiation values given. Those prior values are the pattern, whereas the forecasted data is the label. In this thesis only measurements (and no irradiation) of a predefined number of preceding time-steps will be used for forecasting. Thus previous time steps will serve as feature input of LSTMs. In addition to forecasts with LSTMs, a persistence forecast will be generated that serves as a baseline. Results achieved with each forecast will be evaluated using the RMSE (cf. Section 2.3) and the skill score will serve as a metric for comparability of those forecasts against each other.

The present chapter begins with a general assessment of LSTMs for solar power prediction. The first LSTM architecture will be based on [AM17], while the first experimental setup will be similar to [WOK17]. In successive investigations it will be tested whether different architecture desings might lead to improvements in prediction accuracy. The results recieved will be carried along in further experiments. After the structural analysis, temporal and spatial information is examined. By this, the goal is to assess, if additional data might lead to forecast improvements. The chapter closes with a comparison of the forecast capabilites of LSTMs and SVR for solar power predictions. All those experimentes have the purpose to evaluate the extent of LSTMs to make reliable short term PV power predictions.

The implementation is done in Python. The LSTMs are implemented using Keras libary [Cho+15] on top of TensorFlow framework [Mar+15]. Additionally the libraries Pandas [McK10], Numpy [Oli06] and scikit-learn [Ped+11] are utilized for data preparation and preprocessing. In the following, the data, the preprocessing steps, experimental setups and parameters (alongside with their initialization) will be described.

4.1 Data and Preprocessing

The data of investigation is provided by meteocontrol GmbH, a company offering worldwide all-round solutions regarding PV monitoring and control [met19]. The data covers measurements of PV systems all across Germany over a period from 2012 to 2016. The temporal resolution is 15 minutes. For each PV system, the location (coordinates in longitudes and latitudes), an identification number (ID) and information about installed capacity (kW_p[1]) are given. In preprocessional steps, the extent of data has been reduced, in order to base the research on as high-quality data as possible. First it was ensured that measurements and coordinates match, so that for all entries of coordinates measurements were available as well as the other way around. Afterwards all PV systems were removed that had too many missing measured values in their dataset. Finally, the forecasts are build upon daytime values, excluding every measurements outside a time frame from 09:00 am to 4:00 pm. Since the PV systems are of varying installed capacity, covering a broad range of power, the output of each system has been scaled to its installed capacity for better comparability[2]. Furthermore any missing values are either replaced by the preceding value or zero, depending on the lenght of sequence that is missing. The previous value is used if no more than one hour or four time steps are missing, while 0 is used if a longer sequence is missing. This type of treatment is based on the assumption that short periods of missing data may be due to missing measurements, while longer periods may indicate plant failure. In preparation for the LSTMs the data has been (i) normalized into the value range of the used sigmoid activation function, i.e., in between 0 and 1, (ii) split into training and test set, in a ratio of 60% to 40% respectively, whereas 10% of the training set has been used for validation during the training process.

4.2 Spatial Experimental Setup

The spatial experimental setup is designed to predict the future current of a single PV plant τ. The forecast is based on the past measurements of the target as well as measurements of N surrounding PV plants in a distance of radius r to the target (see Fig. 4.1). The selection of the target and its neighboring plants is done using the euclidean distance from a randomly selected PV system to any other system (cf. Eq. (4.1) and Fig. 4.2). If the defined minimal number of N neighbors within the maximal distance r is satisfied, the random plant becomes the target τ and the measurements of the closest neighbors become the additional features for prediction. If the number of surrounding plants cannot be met, the search for a suitable target continues. For reasons of reproducibility, a random seed is used in this process in the frist experiments.

The euclidean distance is calculated using the positional data of the plants. Since all facilities are located within Germany, the spatial area can be assumed to be a flat surface. Any larger distances need the incorporation of the Earth's curvature for accurate distance calculations, since the accuracy of the Euclidean distance decreases with increasing distance between two locations. In such cases, methods of spherical geometry provide more accurate distance mea-

[1] kW_p refers to *kilo watt peak* and denotes the measured nomial power of a PV system under standard test conditions. Those conditions are 25°C temperature and 1000 $\frac{W}{m^2}$ irradiation. [na19]

[2] This scaling for better comparability is the meaning whenever the term "percentage of installed capacity" throughout the whole further thesis is used.

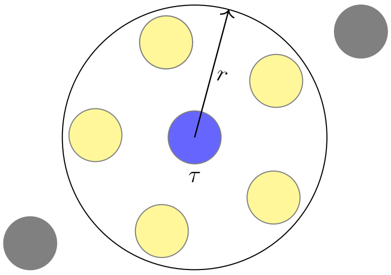


Figure 4.1: Spatio experimental setup. N surrounding PV systems (yellow circles) serve as additional feature to predict the future power output of a targeted PV system τ (blue circle). More distanced PV systems are not considered in the forecast (grey circles).

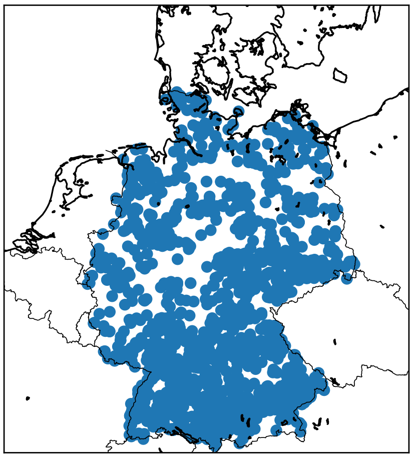


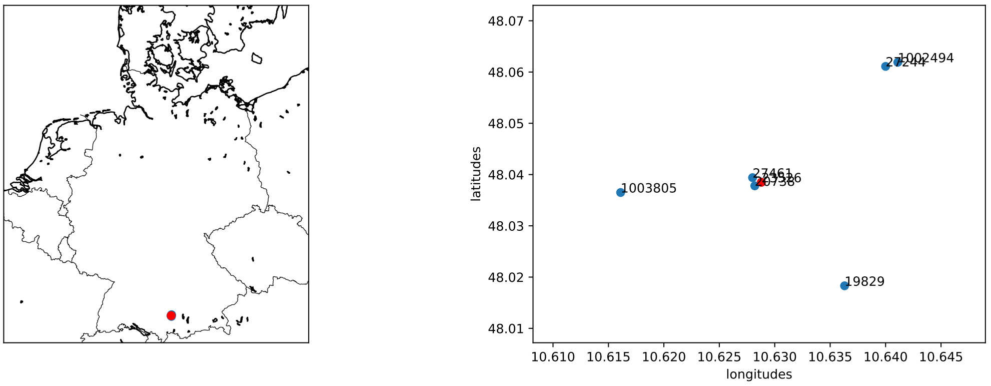
Figure 4.2: Distribution of PV systems across Germany. For the determination of a target plant one has been selected randomly and has been tested to meet certain criteria (minimum distance r to N neighboring systems).

surements [MH11]. The differences between the respective longitudes and latitudes are each multiplied by a factor that represents the distance from one longitude or latitude to another within Germany [SU18; SL41]. The root from the sum of the squared differences then results in the distance of one plant to another:

$$\begin{aligned}
\text{longitude}_{\text{distance}} &= 71.5 \cdot (\text{longitude}_\tau - \text{longitude}_n), \\
\text{latitude}_{\text{distance}} &= 111.3 \cdot (\text{latitude}_\tau - \text{latitude}_n), \\
\text{distance} &= \sqrt{\text{longitude}^2_{\text{distance}} + \text{latitude}^2_{\text{distance}}}. \qquad (4.1)
\end{aligned}$$

One targeted PV system that fulfilled the requirements of $N = 6$ neighbors in a distance of $r = 10$ km has the ID 23526 and is located in “Unterfeld” (South Germany) with 10.6288° longitude and 48.0385° latitude. The neighboring systems are within 0.08 and 2.7 km. Figure 4.3

shows the location and the distribution of the selected subset of PV plants to each other and reveals that almost every compass direction is covered.



(a)

(b)

Figure 4.3: Location of a targeted PV system τ in Germany (red) with $N = 6$ neighboring systems (blue) at a distance of radius $r = 10$ km. The left picture (a) shows a map of Germany (small spots mark some lakes) and the location of the subset. The right picture (b) shows an enlarged view of the location and reveals the distributions of the PV plants to each other. Each PV system is labeld with its own ID.

4.3 Simple LSTM Setup

For the first LSTM configuration a setup as found in [AM17] is chosen. The authors of the referenced work investigated on several LSTMs for solar power prediction. A model with several previous time steps as one input feature achived best results in their work. According to this foundation three previous time steps will serve as one input feature to predict the next time step. Different to [AM17], the granularity in this work is 15 minutes. This means three past steps amount to 45 minutes as feature to predict 15 minutes ahead. In [AM17] one time step accounts to one hour, thus three prior time steps account for three hours in the past to predict one hour in advance[3]. Furthermore a spatio-temporal approach is investigated as described in Section 4.2. This results in an overall dimensionality composed of measurements, per input time steps for each of the $N + 1$ PV systems. The so-called sliding window thereby has a total of 3 time steps, referring to the cohesive input time steps, to predict 1 output (see Fig. 4.4).

The simulation of the first experiment has been as close to [AM17] as possible. Therefore the initial configuration of the LSTM network has been chosen to consist of one LSTM layer with 4 neurons and one dense layer to compute one output (identical to [AM17]). One time, incremental training has been tried (updating the weights after one example, cf. Section 3.2),

[3] In Section 4.4 the discrepancy of the granularity will be adressed, as different time horizons will be investigated.

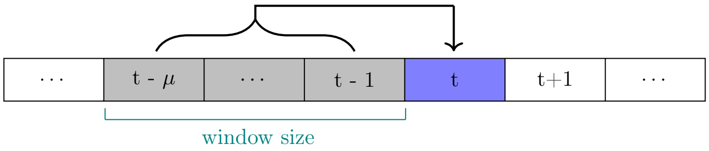


Figure 4.4: General representation of the sliding window approach. The window size is labeled in tourquise and a total of $t - \mu$ steps is marked in gray. The input data for the prediction of the blue marked time step t stays the same for predictions further ahead $(t + 1, t + 2, \ldots)$.

but due to quite unpleasant runtime and model performance the batch size has been set to 72. This setup has been run for several times for different numbers of epochs (see Table 4.1), due to the stochastic initialization of the network weights in the beginning of the learning process.

Table 4.1: Error measurements of a LSTM trained several times for an increasing number of epochs. Displayed are the calculated average RMSE in percentage of installed capacity and the corresponding standard deviation (SD). The average error is compared with a skill score (cf. Section 2.3). The baseline for comparison builds a persistence forecast that achieved a RMSE of 2.33%. A skill score > 0 indicates an improvement of the forecast over the reference.

Epochs	**ø RMSE (± SD)**	**Skill Score (± SD) [%]**
20	2.68 (± 0.41)	-15.27 (± 17.79)
30	2.53 (± 0.14)	-8.64 (± 6.21)
40	2.42 (± 0.14)	-4.19 (± 6.07)
50	2.41 (± 0.08)	-3.69 (± 3.65)
60	2.31 (± 0.10)	0.87 (± 4.36)
70	2.28 (± 0.08)	1.94 (± 3.27)
80	2.25 (± 0.07)	3.37 (± 2.82)
90	2.23 (± 0.06)	4.21 (± 2.46)
100	2.22 (± 0.05)	4.39 (± 2.19)
1000*	2.13	8.53

* The network trained for 1000 epochs has only be simulated once in contrast to each other number of epoch.

Table 4.1 shows the outcomes of the first experiments. Over the course of increasing epochs the average RMSE and its standard deviation decreases. This means, the longer the model is trained, the more accurate the predictions become. Additionaly, the decline in standard deviation along of more epochs denotes more stability in the results. The more narrow the standard deviation, the less impact is in the stochastic element due to randomly initialized network weights.

Overall the experiments produced pleasantly low RMSE values and the results achieved rank in a scale comparable to [AM17]. Converted into percent, the RMSE in their work ranks between 2 and 2.7%. The reference in the present thesis for comparison of the LSTM model performance is a persistence forecast. This baseline resulted in a RMSE of 2.33%. Thus, from a trainings duration of 60 epochs onwards the LSTMs have in average been able to beat the

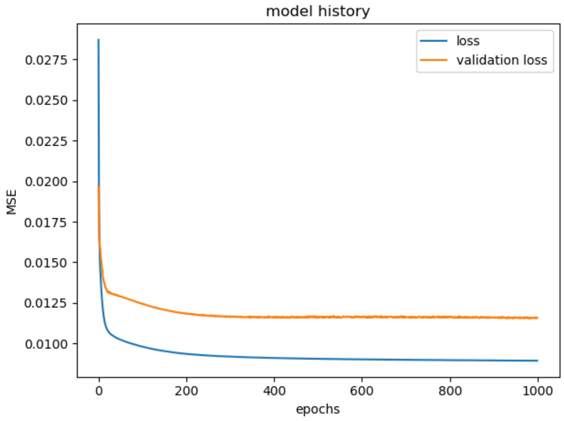


Figure 4.5: Model history of the simple LSTM trained for 1000 epochs. In the picture are the loss on the training set (blue) as well as on the validation set (orange) displayed. The validation loss remains low, so that overfitting has been avoided.

naive approach. An absolute superiority can be confirmed after 80 training epochs. The skill score, including its standard deviation, is always higher than zero at this time.

In difference to the model runs of 20 to 100 epochs, the simulation with 1000 epochs has only been run once. This setup has been chosen whether or not the model would overfit on the trainings data with enough learning time. With 1000 epochs, the trend of a decreasing RMSE could be continued without overfitting (see Fig. 4.5), showing improvements in prediction accuracy with longer trainings periods in this setup.

Differences in model design of the present model and the template model of [AM17] are in (i) the batch size (ii) the used optimizer. The template is one of several LSTM configurations that had been investigated. The batch size has been set to one and *Adam* (short for adaptive moment estimation) has been used as optimizer. The present model in comparison used SGD in its package defaults settings as optimizer and a significantly bigger batch size of 72. Both optimizers are able to converge as [AM17] as well as the present model are able to predict PV power with low RMSE. In general Adam is an optimization algorithm with adaptive learning rate and has been designed in order to improve speed of training [KB14].

[AM17] evaluated training times of 20, 50 and 100 epochs and with the present outcomes it seems that Adam converges more quickly than SGD. Whether or not this is true might be revealed in the following section, where some hyperparameters will be evaluated.

4.3.1 Hyperparameter Investigation

The model has been altered by setting the optimizer to Adam as done in [AM17]. Further the model's hyperparameters have been evaluated by changing the number of neurons, batch size and epochs iteratively. Instead of a grid search or random search, those hyperparameters have been evaluated manually one at a time (cf. Section 3.6). In one model run one parameter is changed, while the others remain in the initial configuration. Due to the small impact by the weight initialization (cf. Section 4.3), those experiments are carried out once.

Table 4.2 displays the differences from the initial model design (1 LSTM layer, 4 neurons, batch size of 72, several epochs), by just changing the optimizer from SGD to Adam. As can been seen each RMSE drops below the ones achieved using SGD as optimizer (cf. Table 4.1). In contrast to the results with SGD, there is no obvious trend that more epochs lead to lower error measurements. Rather a fluctuation can be observed: The RMSE values in Table 4.2 of 20, 50 and 100 epochs are almost identical with $\sim 2.1\%$, but in epochs 30, 40, 60, 70, 80, and 90 slightly lower values could be achieved ($\sim 2.05\%$). The smallest training duration of 20 epochs has a lower RMSE with Adam than with SGD (2.11% compared to 2.68%). Even 1000 epochs cannot undercut this value (2.11% at 20 epochs compared to 2.13% at 1000 epochs).

The skill scores fluctuate to the RMSEs accordingly, but remain always greater than zero. Consequently, the improvement over the reference is still given. These results validate that the change from SGD to Adam represents an improvement in speed of training as well as in prediction accuracy. Or more precisely, an adapted learning rate had been able to improve the results.

Table 4.2: Error measurements and skill scores of a LSTM using Adam as optimizer for an increasing number of epochs. Displayed is the RMSE in percentage of installed capacity. The reference is a persistence forecast that achieved a RMSE of 2.33%. A skill score > 0 indicates an improvement.

Epochs	**RMSE**	**Skill Score [%]**
20	2.11	9.11
30	2.06	11.56
40	2.05	11.96
50	2.13	8.65
60	2.05	11.88
70	2.05	12.05
80	2.04	12.17
90	2.06	11.51
100	2.10	9.53
1000	2.07	10.91

The evaluation of the neurons and batch size was carried out analogously to the evaluation of the epochs. One of the parameters has been changed while the initial configuration of one layer with 4 neurons, a batch size of 72 and a training time of 50 epochs remains. The choice of 50 epochs, despite the outcomes above, has been set, due to the findings in [AM17]. Their best results were obtained by this training time. Furthermore, 50 epochs provide a suitable compromise with regard to the computing time.

A change in neurons resulted in a lowest RMSE of 2.07% (bold marked in Table 4.3). Both 72 and 150 numbers of neurons achieved this due to roundings. The skill scores, being calculated without roundings, reveal that the most performant model occurs having 72 neurons. By this the model had been superior by about almost 11% over the persistence forecast, as the skill score indicates. The batch size could not achieve as much improvements in model skill or RMSE reduction. The minimal reached RMSE value is slightly higher and the model's biggest improvement is observed for a batch size of 32 with a skill score of 10.35%. As can be seen in Table 4.4 a batch size of 1 (incremental learning) also resulted in a minimal RMSE of 2.09%. This is again due to rounding. Differences in the skill scores reveal a marginally better model performance with a batch size of 32.

Table 4.3: Overview of the RMSE values (in % of installed capacity) and the skill scores to a varied number of neurons. Other hyperparameters, such as the number of epochs, batch size and the optimization algorithm are set as follows: 50 epochs, 72 batches, Adam as optimizer. Minimal RMSE and maximal skill score values are marked bold.

Neurons	**RMSE**	**Skill Score [%]**
4	2.13	8.65
8	2.12	9.07
12	2.12	9.06
16	2.10	9.73
24	2.09	10.26
36	2.08	10.51
72	**2.07**	**10.95**
100	2.11	9.38
150	**2.07**	10.89

Table 4.4: Outcomes of error measurements (in percent of installed capacity of the targeted PV system) and skill scores according to different batch sizes. The used optimization algorithm, number of neurons and, number of epochs remain as used before: Adam, 4 and, 50, respectively. Minimal RMSE and maximal skill score values are marked bold.

Batch Size	**RMSE**	**Skill Score [%]**
1	**2.09**	10.28
16	2.10	9.76
32	**2.09**	**10.35**
64	2.11	9.23
128	2.12	8.92
256	2.10	9.88
512	2.14	8.19

Despite the improvements, the influence of both hyperparameters is small. The neurons only reduce the error value by 0.06% compared to the RMSE according to the initial amount of neurons. The batch size has again less impact and achieves a lowest RMSE of 2.09%. Consequently, both the neurons and the batch size did not improve the prediction accuracy much, as no new low in RMSE values was reached. Those findings suit the outcomes in [RG17].

Finally, a combination of hyperparameter values resulting in a lowest RMSE[4] was tested. A runtime of 200 epochs had been chosen, in order to limit runtime but potentially reveal overfitting. Again several model simulations were initiated because of the stochastic initial weight selection to find out about model stability. A combination of each value that produced

[4] See Table 4.3 and Table 4.4, lowest RMSE values have been achieved with 72 neurons and a batch size of 32.

best results with the other parameters in the initial state did not improve the model accuracy. Indeed, the outcomes show an overfitting of the model (see Fig. 4.6).

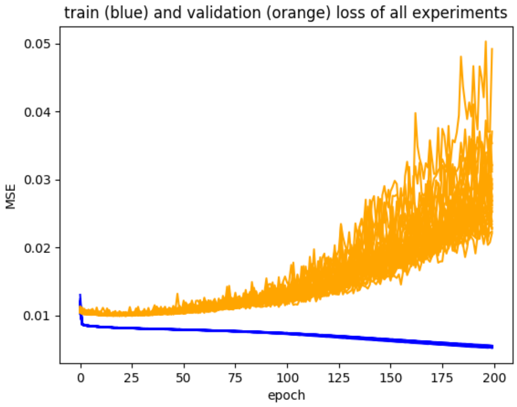


Figure 4.6: Model history of several model simulations with the LSTM layer configuration of 72 neurons, batch size of 32, trained for 200 epochs. In the picture are the loss on the training set (blue) as well as on the validation set (orange) displayed for each simulation run.

Figure 4.6 shows an initial drop in loss on the training set and a slight downturned loss on the validation set until approximately epoch 25. While the trainings loss keeps decreasing, the validation loss of all experiments starts to increase beginning from approximately epoch 25 onwards. This indicates an overfitting of the model on the trainings data. As a result, the hyperparameters can each be optimized individually to improve model performance. However, this is not sufficient to achieve an overall optimal combination. The result shows that single improvements did not add up. Rather the opposite is true, the combination of each hyperparameter that achieved lowest RMSE values yielded a RMSE of 2.59%. Alternatively, the newly found optimum value of each individual hyperparameter has to be set to this value before optimizing the following hyperparameters (i.e. in this case the number of neurons found to be optimal would have to be set to the new value when optimizing the batch size).

The incorporation of a dropout rate to the model architecture has been tested to purposely address the issue of overfitting. The same rates from 20 to 80% probability of units being droped on each time step have been tried for both the inputs as well as the recurrent states seperately. Table 4.5 shows the outcomes.

The middle column “Inputs” clearly shows that adding a dropout rate to the inputs has a worsening effect on the results, as the error values increase the higher the probability of units are dropped. Even the lowest percentage probability rate (20% in the first row) does not improve in the form of a lower RMSE value. Consequently, the combination of the individually optimized values could not be compensated by a dropout of this kind. The input sequences are quite short with 3 time steps. Dropping anything out of those sequences deprives the model of

Table 4.5: Overview of the RMSE values (in % of installed capacity) that had been achieved by adding dropout rates. The model archiecuture is the one derived by evaluation of neurons and batch sizes. Dropped are (i) the units of "the linear transformation of the inputs" (displayed in column "Inputs") and (ii) the units for "the linear transformation of the recurrent state" [Cho+15] (displayed in column "Recurrent States").

Probability of dropped units [%]	**RMSE**	
	Inputs	Recurrent States
20	2.59	2.32
40	4.01	2.19
60	5.52	2.11
80	8.04	2.13

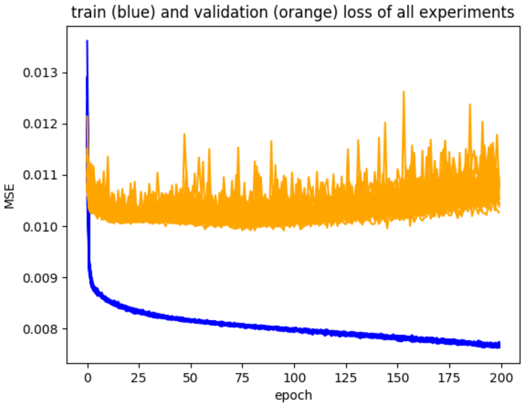


Figure 4.7: Overview of loss on trainings (blue lines) and test set (orange lines) for several simulation runs. The model architecture has been extended by the incorporation of dropout on the recurrent states to adress overfitting.

the basis to approximate the desired function. This could explain the RMSE values rising with the dropout rate.

The situation is different for dropout applied on the recurrent states. A lowest RMSE of 2.11% is reached along a probability of 60% dropped units. This error value is as good as a network configuration trained for 20 epochs of one LSTM layer with 4 neurons, a batch size of 72 and Adam used as optimizer (Table 4.2). However, training that reference for longer achieved lower RMSEs. Consequently, evaluating and optimizing some hyperparameter did not improve the models generalization capacities on this specific task. Changing them only showed an overfitting to the training data. Adding dropout to the recurrent states could regulate this (cf. Fig. 4.7), but not achieve higher prediction accuracy. This statement does not apply on the optimization algorithm. Changing from SGD to Adam helped to reduce the error value. By this, less epochs and thus less computational time were required.

As a conclusion, an investigation of the LSTM architecture showed to be ideal in a configuration as in [AM17] when predicting PV power in a spatial experimental setup as in [WOK17]. A manual change in hyperparameters could not lead to more accurate forecasts. The only exception is the change from SGD to Adam as used optimization algorithm. Each hyperparameter optimized on its own achieved small improvements, but an overall ideal combination for most accurate forecasts had not been retrieved this way. By this it can be concluded that the initial LSTM architecture, with one LSTM layer, 4 neurons, a batch size of 72, trained for in between 50 and 100 epochs, an optimization algorithm with adaptive learning rate, such as Adam, can produce reliable short term PV power predictions in a spatial experimental setup. In the following, the impact of additional data on forecast accuracy will be analyzed. Therefore both, the temporal as well as the spatial variables will be altered in further experiments.

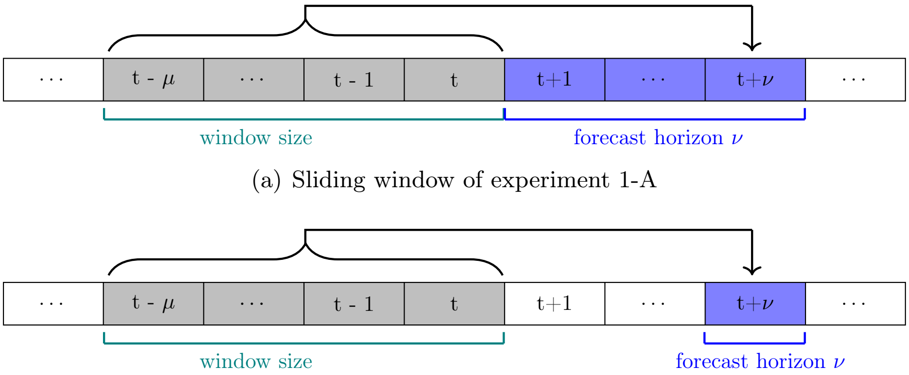
4.4 Time Horizons Experiment

The temporal resolution in this work accounts to 15 minutes for one time step. This granularity is different to the one used in [AM17]. In order to close the gap between different temporal resolutions both, the window size and forecast time horizon will be modified. Three more approaches have been set up to analyze the impact of the granularity and assess the importance of input as well as output sequence length for PV power predictions with LSTMs. The first approach will be referred to as experiment 1-A, the second one will be denoted with experiment 1-B and the third one with experiment 1-C. Once, a window size of 12 time steps has been selected to use 3 hours as feature input for the prediction of 1 hour or 4 corresponding time steps ahead. (1-A)

In another investigation, the window size has been reset to the initial size of 3 time steps, making up 45 minutes as feature again. Other than in the first experiment (cf. Section 4.3), instead of forecasting the directly subsequent time step, the time step corresponding to one hour ahead has been selected. Consequently, the model's task resulted in learning the pattern of time step $t + 4$ on the basis of time steps $t - 2$, $t - 1$ and t (see Fig. 4.8). (1-B)

The last study is also about predicting a distanced time step. It is a compound of experiment 1-A and 1-B, by taking 3 hours or 12 time steps á 15 minutes as feature to predict a single time step in distance $\nu = 1$ hour. (1-C)

The used architecture for all experiments is equal to the initial configuration described in Section 4.3. Again Adam is used as optimizer, due to its improvements on error measurements. The experiments are run for an increasing number of epochs. The number of neurons and batch sizes stay the same, as the evaluation of hyperparameters in Section 4.3.1 did not improve the prediction capability much. Experiment 1-A investigates on the impact of using more feature information in form of longer input sequences. Further the capability of predicting more than one time step is explored. Experiment 1-B and 1-C on the contrary, explore the capability to predict a distanced time step. See Figure 4.8 for visualization. The variation with respect to the sliding window approach that results for the approaches is shown. Overall the selected period of time still remains in the range of short term prediction and does not exceed the investigated time frame of [AM17].



(a) Sliding window of experiment 1-A

(b) Sliding window of experiment 1-B and 1-C

Figure 4.8: Representation of the sliding window approach adapted to predict different time horizons. On time a each time step of a sequence length ν is forecasted (a), whereas another time one single time step in distance ν is forecasted (b). The window size remains of constant amount of time steps.

The RMSE values and skill scores of experiment 1-A, corresponding to the epoch and time steps are listed in Table 4.6. For each individual time horizon the RMSE falls and rises over the course of epochs. This behavior is similar to the ones derived in Section 4.3. Besides the ascents an overall decrease can be observed. The longer the time horizon, the larger the error values. On the contrary, it is noticeable that the increase of the error along the time horizon becomes smaller. Thus the average differences from one time step to another decline.

Comparing the RMSE of time step t of this experiment with the prediction accuracy made in Section 4.3[5] an overall lower RMSE can be seen. On average a reduction of $\sim 0.05\%$ has been achieved by taking a larger number of prior time steps as feature. Different to prior outcomes the lowest RMSE values are attaind after 100 epochs. This means, the model needs longer learning time with more inputs and outputs.

As can be seen, the skill scores reveal an overall improvement against the persistence forecasts for each time horizon. The highest improvements are achieved for predictions of 30 minutes ahead. The values of almost all skill scores are highest for this time step. This great superiority might be caused by the relatively high increase in the RMSE of the persistence forecast from the first to the second forecast time step (time horizon of 15 and 30 minutes). All in all, the skill score first rises to a high and then decreases again. Similarly to the RMSE, the skill score of this experiment for time step t compared to the skill score of the same time step of the previous investigation when Adam was used, the skill score is higher here.

[5] when using Adam as optimizer

Table 4.6: Outcomes of experiment 1-A. Shown are the RMSE values (in % of installed capacity) and skill scores according to each time horizon for several numbers of epochs. The reference is a persistence forecast, which RMSE values according to the time horizon are listed in the last row. Minimal RMSE and maximal skill score values are marked bold.

	RMSE				**Skill Score [%]**			
	Time Horzion [minutes]							
Epochs	15	30	45	60	15	30	45	60
20	2.08	2.64	3.01	3.32	10.45	12.24	11.73	11.20
30	2.05	2.62	3.00	3.28	12.08	12.80	12.12	12.17
40	2.06	2.63	2.98	3.26	11.44	12.69	12.57	12.62
50	2.06	2.63	2.99	3.28	11.39	12.53	12.46	12.25
60	2.07	2.62	2.96	3.25	11.17	13.06	13.15	13.07
70	2.05	2.61	2.99	3.29	12.07	13.12	12.47	12.07
80	2.09	2.64	2.98	3.27	10.12	12.17	12.47	12.45
90	2.10	2.64	3.00	3.30	9.80	12.24	12.14	11.64
100	**2.04**	**2.59**	**2.94**	**3.23**	**12.52**	**13.91**	**13.87**	**13.57**
Persistence	2.33	3.01	3.41	3.74				

Both of these outcomes, RMSE and skill score respectively, indicate the importance of the sequence length for learning. Nevertheless, the influence in this short time frame is still to be considered marginal, since the deviations of the error values due to changes in hyperparameters (especially of the neurons and batch size, see Section 4.3.1) are similarly low.

The RMSE outcomes of experiment 1-B rise and fall similarly to previous RMSE results (cf. Table 4.7). A low of 3.31% is reached at epoch 50 and epoch 80. Due to roundings of the RMSEs, the skill score of epoch 50 is higher than the one of epoch 80. Thus the model has already been sufficiently adapted on the predictions of that time step ($t + 3$) by this number of epochs. All in all the fluctuation over the epochs remains in a small range.

The skill scores behave accordingly to outcomes of the RMSEs. They are all above zero, indicating an improvement of LSTM predictions over persistence predictions. Consequently, the model can predict a more distant time step without the information about the directly preceding time steps.

The skill scores and RMSEs derived in experiment 1-C are presented in Table 4.8. This investigation is done for comparability of the forecast accuracy of time step $t + 4$ with and without the time steps $t + 1$, $t + 2$, and $t + 3$ on the base of an equally long input sequence of 12 time steps. Thus outcomes of the present experiment (1-C) will be compared with outcomes of time horizon according to 60 minutes in experiment 1-A.

Most of the RMSE values in 1-C (Table 4.8) are lower than the corresponding ones in 1-A (Table 4.6). This means the model had been able to approximate the prediction of that certain

Table 4.7: Outcomes of experiment 1-B. Shown are the RMSE values (in % of installed capacity) and skill scores according to a number of epochs. Bold marked values indicate minimal RMSE or maximal skill score respectively.

Epochs	**RMSE**	**Skill Score [%]**
20	3.34	10.72
30	3.32	11.04
40	3.32	11.25
50	**3.31**	**11.49**
60	3.33	10.98
70	3.32	11.13
80	**3.31**	11.33
90	3.34	10.65
100	3.32	11.08

time step more properly, when less time steps had to be learned. As a consequence, the amount of output time steps are as important as the amount of input time steps. The more predictions are requested, the more the model has to reproduce.

The importance of the feature information is supported by the comparison of the RMSE values from 1-B with 1-C. In both experiments the model had to learn the prediction of a single distanced time step. They differ in the used amount of feature information. RMSE results of 1-C are almost overall lower as the ones of 1-B. This leads to the conclusion that a longer input sequence allows the LSTMs to better approximate the function of PV power at that time step.

Table 4.8: Outcomes of experiment 1-C according to different training times. Displayed are the RMSEs in % of installed capacity and the skill scores. The values refer to predictions of a time step in distance $\nu = 4$ on the basis of a window size of 12 steps. The model configuration is set as follows: 1 LSTM layer with 4 neurons, a batch size of 72, and Adam as optimization algorithm. Bold marked values indicate minimal RMSE and maximal skill score.

Epochs	**RMSE**	**Skill Score [%]**
20	3.28	12.33
30	3.23	13.60
40	3.27	12.50
50	3.22	13.95
60	3.34	10.67
70	3.25	12.93
80	**3.18**	**14.87**
90	3.25	12.95
100	3.24	13.41

Summarizing the results of the time horizon experiments, it can be said that the model is able to predict both a longer sequence and a distanced time step. Additionally, both the length of input and output sequence impact the extent of prediction accuracy. Longer input sequences result in more precise predictions as the model is supplied with more information. A single distanced, predicted time step on the other hand, changes the model's task and the predictions become more accurate due to the increased focus.

4.4.1 Extended Time Horizon

The following experiment intends to investigate the limitations of short-term power predictions with LSTMs. Up to now, predictions of maximum one hour ahead have been investigated and this time horizon is to be extended. The goal is to predict as long sequences as possible. The forecast horizon will remain within the scope of short-termed PV power predictions and is therefore between 1 and 5 hours (cf. Chapter 2). The input sequence will remain at 3 hours, corresponding to outcomes of previous investigations. Thus the prediction of 5 hours equals 20 time steps of 15 minutes and is build upon an input or window size of 3 hours or 12 time steps á 15 minutes.

The LSTM configuration as described in Section 4.3 will be used and be trained for 20 to 100 epochs. Further the spatial experimental setup of one target with 6 neighboring systems within a radius of 10 km will remain (cf. Section 4.2). Figure 4.9 displays the distribution of predicted power values of the targeted power plant in percent of installed capacity for each hour ahead. The use of violin plots in contrast to box plots additionally illustrates the values' probability distribution along the vertical axis with regard to the percentage power of installed capacity. The measured power values are displayed for comparison. Thus, the representation for both true and predicted values correspond to the test set, starting from October 2014 to October 2016.

Most noticeable is that the maximum predicted values decrease with increasing time horizon. For example, the maximum measured values exceed all of the predictions. However, these maximum values in the measurement series appear to be outliers. Predicted PV power at forecast horizon of 2 and 3 hours inherit similarly many outliners, but the amplitude of overall percentage installed capacity starts to decrease. At forecast horizon 4 and 5 hours, less outliners and a more narrow range of values can be seen. Thus the variance in the predicted values becomes less and more general values are being predicted. Additionally, the overall low maximum output of the system is noticeable. The targeted PV plant generates only up to $\sim 22\%$ power of its installed capacity. Reasons for this might be the weather conditions in Germany on the one hand or local geografical circumstances, such as mountains causing shade, on the other hand.

The distribution of values also changes along the forecast horizons. While the true measurements have most values in between zero and five percent of installed capacity, the distribution of data points at horizon 2, 3, and 4 hours seem to appear the other way round. A high accumulation at the upper end becomes visible. Only at 5 hours ahead, a slightly more even distribution returned. Consequently, the power outputs have been overestimated. Altogether

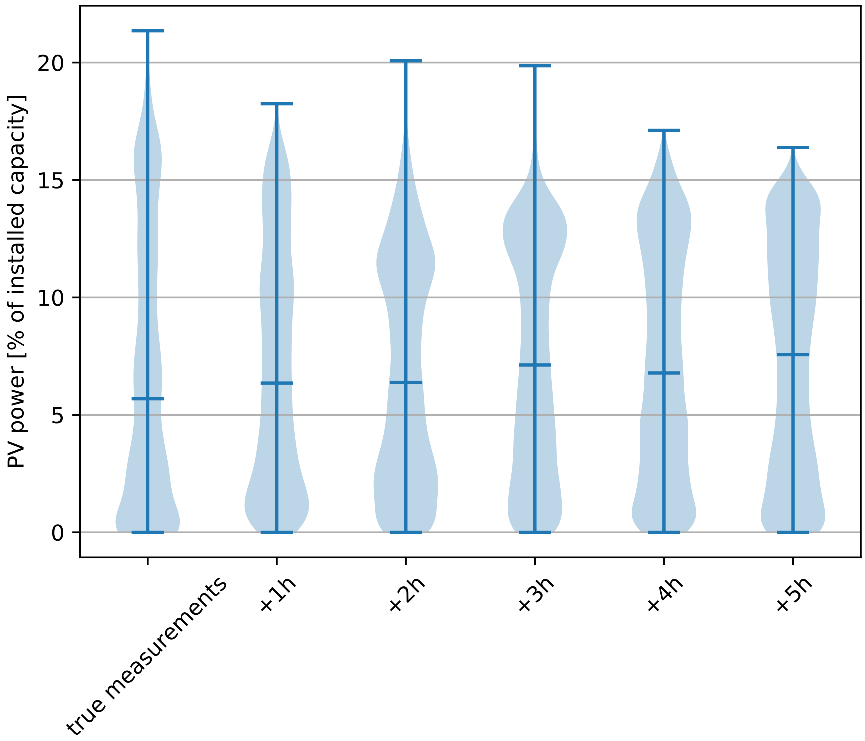


Figure 4.9: Violin plot with displayed medians of different predicted time horizons. Shown is the probability distribution according to all time steps of the test set for the measurements (true measurements) and predictions for several time horizons in percent of installed capacity of the targeted PV plant (cf. Section 4.2). The notation for the predictions can be read as follows: '+1h' remarks all predicted values of the time step at one hour in the future.

more mid-ranged power values have been predicted instead of single peak loads. This can also be seen by the medians, which slightly increase with the time horizons. The predictions become more broad and less precise.

An insight into the annual and daily cycles might provide more information about the distributions. Figure 4.10 shows the annual course for the entire test set for each of the predicted time horizons as well as the measured values of the targeted PV plant. The predictions made with the LSTMs show the same course of the year that can be seen in the measured values. The values begin to rise in January, reach their highest values in June/July and fall again from October. The amplitudes of the predicted values drop along with the increase in time horizons. This matches the outcomes displayed in the violin plots. The shorter the predicted time horizon, the more close the predictions are to the measurements. Whereas, in the other way round the longer the time horizon, the more generic the predictions become.

It is also noticeable that at the beginning of October 2015 no or hardly any electricity was produced. This can be seen in the true measurements and had been approximately reproduced in the predictions. A reason could be that the plant had been out of operation.

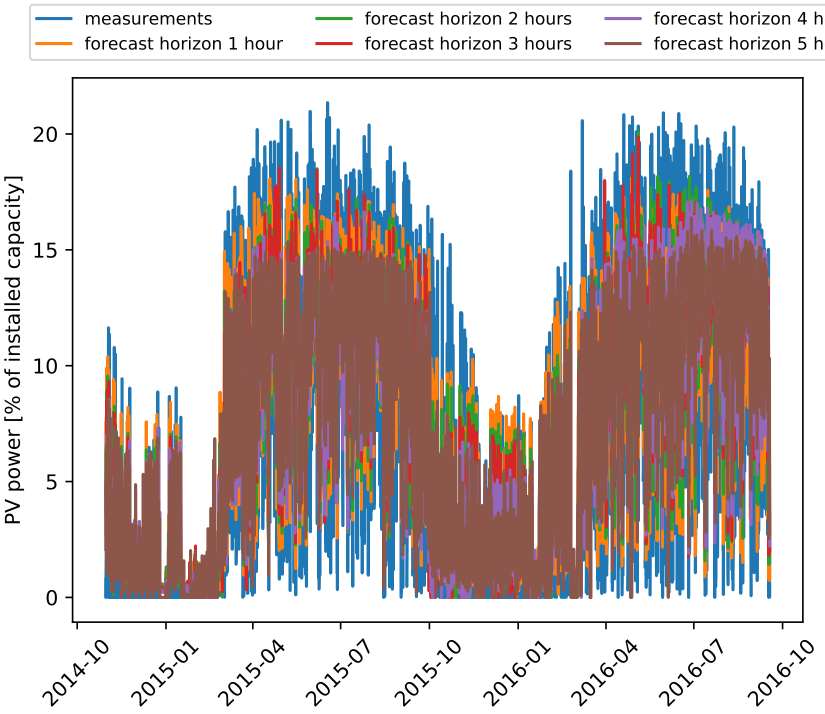


Figure 4.10: Overview of all predictions of different time horizons on the test set. The outcomes are referring to one PV plant and after 100 epochs of training, due to best results in previous experiments.

In Figure 4.11 the power curve of several different days is displayed. Shown are four days out of the test set, one for each season in 2015: winter (**A**), spring (**B**), summer (**C**), and autumn (**D**). Each day begins at 9 am and ends at 4 pm as described in the preprocessing of the data. This is due to the investigation of daytimes only. Those pictures give closer insight whether or not the course over day is met.

A The winter day has overall low power values in both the predictions and measurements. This is due to the overall lower sun position in the middle latitudes. While the true measurements (blue) start to rise in the morning, all predicted values remain low. An increase in the predicted PV power starts at midday, when the measurements decline. Thus the power curves of measurements and predictions are virtually opposite.

B The day in spring shows overall higher measured power values with a lot of fluctuations. The predictions of each forecast horizon in contrast are quite similar. Strong changes in the true power values from large to small values are not imitated by the predictions.

C During the day in summer, the PV system appears to supply power steadily. A slight rise in the values shows a typical course of the sun over the day. In the predictions, however, the power values remain evenly with minor changes at the end of the day.

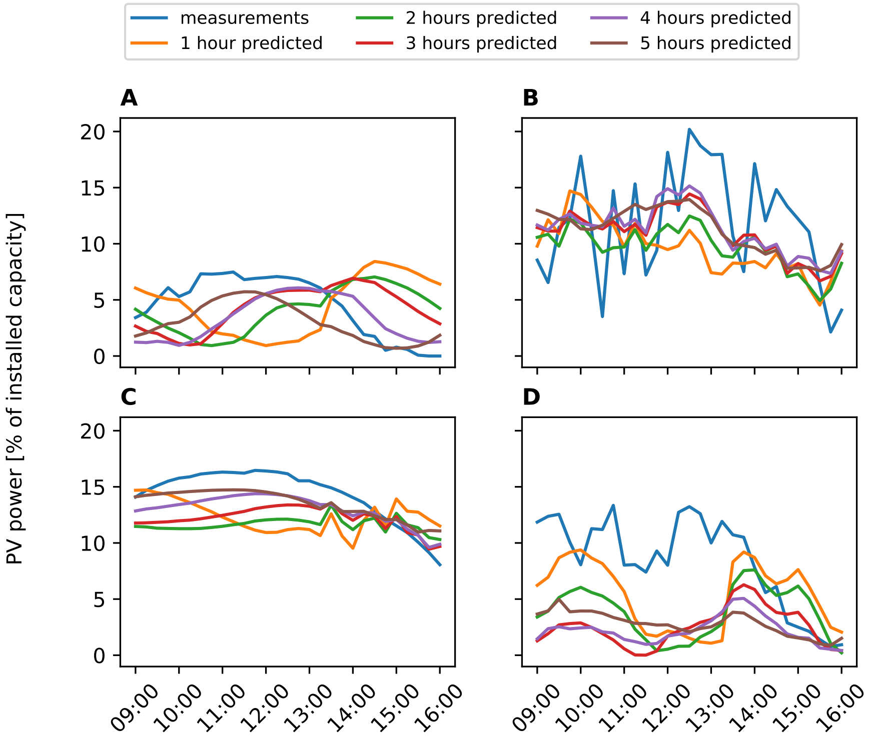


Figure 4.11: Comparison of the prediction accuracy of different time horizons displayed for one day at a time. Each day begins at 9 am and ends at 4 pm. Thus, each day is 7 hours. By this, only daytimes are evaluated (cf. Section 4.1).

D The day in autumn shows a decrease in PV output over the course of the day. The predictions of each time horizon start below the true measurements and approach them at the end of the day. Overall, similar changes in the curves can be seen, comparable to the day in spring (B).

All together, each seasonal daily plot has in common that the curves of the predictions share a similar pattern. The actual power curves change, if other days had been picked, due to different weather and radiation conditions. Still, the order of magnitude of the predicted values equals the one of the measurements. Thus, daily trends have been identified. The selection of daytimes as described in Section 4.1 led to power curves that miss an expected rise in the beginning of a day. The selection had been done to ensure that the sun had been risen for any season equally. As the pictures show this had been achieved, but on the downside inhibits a more comprehensive comparison. Overall there is still room for improvement for the predictions to match the course of the day of the measurements. Visible differences between predictions and true measurements are in the range of the RMSE values, which are presented in the following.

The RMSE values increase with the hours of the time horizon (see Fig. 4.12). The increase, in turn, decreases over time. For example, the increase from the first to the second hour is greater than the increase from the second to the third hour. This suits previous outcomes that with further time more general predictions are carried out. Overall the course over the epochs is quite similar and 100 epochs of training resulted in lowest RMSE values for predictions of time horizons further ahead. The model run has been done once for each epoch, since previous results ensured model stability against model weight initialization.

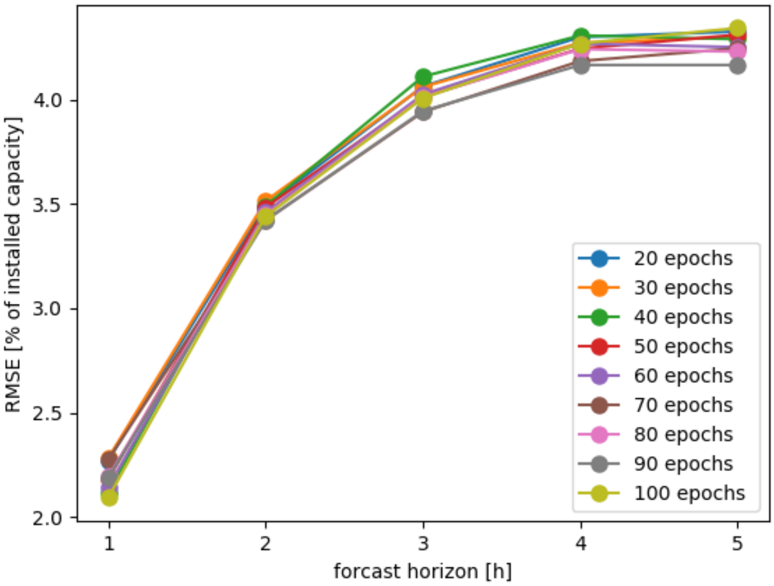
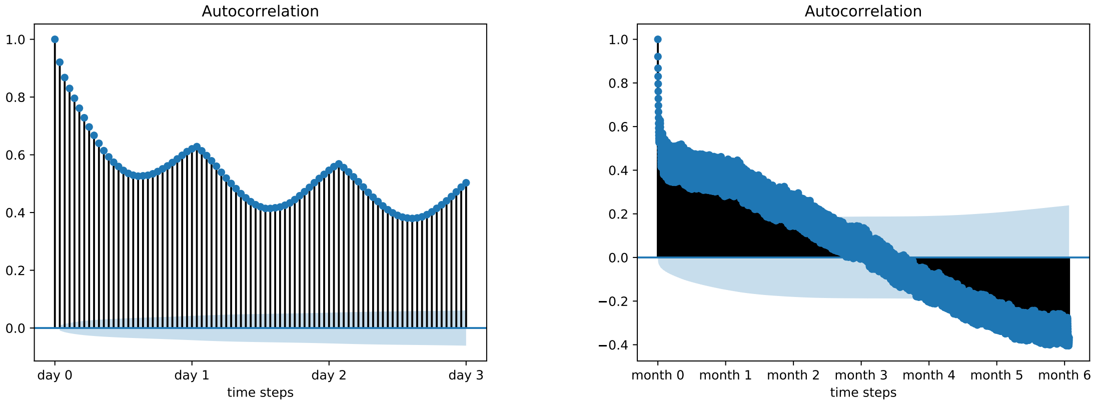


Figure 4.12: Corresponding RMSE values to the extended time horzion experiment. The experimental setup is as described in Section 4.1. The RMSE values are displayed depending on the number of epochs and were calculated for the entire test set.

4.4.2 Autocorrelation

In addition to the exploration of the time horizons that have been investigated, the autocorrelation of the time series of the targeted plant's measurements gives further insight. An autocorrelation describes the extent that a time step is useful for predicting a time step with lag ν. On the x-axis in an autocorrelation plot the lagged time steps are listed. The corresponding y-value displays the strength of correlation between lagged time step and all values of the time series. Large current values correlate with (i) large values at the specified distance, if the correlation is positive or (ii) small values at specified distance, if the correlation is negative [Cen19]. Significant correlations exceed the light blue cone, which represents the confidence interval (95% by default).

As visible in Figure 4.13 (a) the time step with lag $\nu = 1$ correlates highly with the next following time step. The correlation decreases over the course of a day and increases again until the next day. Each day begins with a high, which is less than the high of the previous day. Consequently, a downward trend can be observed with increasing time lags. For three days in advance significant correlations with large values are identified. This is different for an extended perspective. Figure 4.13 (b) shows the correlation for time lags that amount to half a year. The time steps with lags according to the first few days correlate strongly with large values, but the overall correlation strength decreases until 3 month ahead. From that point, a weak correlation with small values begins, which increases up to half a year ahead. After a first decrease of the correlation strength, the measure for the correlation stays within a middle range from approximately 0.4 to -0.4.



(a) Autocorrelation for time steps of 3 days.

(b) Autocorrelation for time steps of half a year.

Figure 4.13: The analyzed time series is the one of the target with ID 23526. Correlations that exceed the blue cone are significant by a confidence interval of 95%.

The significance of the correlations is mostly given for time steps that amount to a total time of one year[6]. After that time the significance vanishes. Additionally, a wave pattern can be identified over the course of several years. This wave pattern indicates an autocorrelation term in the time series. The so-called partial autocorrelation function thereby helps to determine the order of the autoregressive term. The partial autocorrelation reveals the dependencies between a time step and lagged time step by neglecting the influence caused by time steps in between. In case of the time series of the targeted PV plant significant spikes are at lag 1, meaning that this lag causes all higher-order autocorrelations[7].

Those correlations can be able to explain outcomes of the time horizons experiments with LSTM. As solar radiation and thus PV power generation data is non-stationary [AM17], LSTMs are able to model those non-linear dependencies. Despite the low order of the autoregressive term (order=1), predictions for longer short-term periods of time or distanced time steps turned

[6] A visualization of the autocorrelation according to all measurements from January 2012 to September 2016 of the target PV system can be found in the appendix in Figure B.1.

[7] A figure visualizing the partial auto-correlation of the time series can be found in Figure B.2 in the appendix.

out to be closely to the measurements. All predicted values achieved relatively low RMSE values.

4.5 Spatial Experiments

The previous section investigated on LSTM architectures as well as the temporal aspects of solar power predictions. The other part in a spatio-temporal setup for solar power prediction provides an exploration of the spatial dependencies. Therefore, the corresponding variables (distance r and number of surroundings N) of the experimental setup described in Section 4.2 will successively be modified. A total of 3 experiments will be executed. Analogously to the time horizon experiment (cf. Section 4.4) the experiments will be denoted as experiment 2-A, 2-B, and 2-C.

Experiment 2-A scrutinizes the impact on the forecast accuracy due to the number of surrounding PV systems. An alteration of N in a constant distance $r = 10$ km is done. Starting without any neighbors, the number will be incremented by 2 starting from 4 until 12. Equally to the previous investigations, the closest neighbors will be selected.

Experiment 2-B will explore the impact of the distance r, in which the surrounding PV systems are located. In addition to an alteration of the number of neighboring systems N, randomly selected systems within the distances of 5, 10, 15, and 20 km will be evaluated. This setup aims at identifying the influence of the distance in combination with the number of surrounding systems. The selection of randomly drawn systems in contrast to the closest systems within a certain distance increases the probability that the requested distance is really covered. Or in other words, drawing systems randomly enhances the chance that the neighboring systems are truly at a particular distance. E.g., a radius of 20 km is explored and some of the N neighboring systems actualy are within 20 km instead of less kilometers.

In experiment 2-C the distance is investigated once more. A second radius is defined to ensure that the surrounding systems do not fall below a certain distance to the targeted plant. This way a minimal and maximal radius define an area in which additional PV systems can be located. Figure 4.14 displays how the spatial experimental setup described in Section 4.2 is modified. The following pairs of radii (in km) will be investigated for a fixed number of neighbors ($N = 6$): [5, 10], [10, 20], [15, 30], [20, 50]. The first number describes the minimal distance that needs to be hold, whereas the second number defines the maximal distance that has to be adhered. Only measurements of surrounding PV systems that meet those conditions are considered as feature for prediction of PV power of the targeted plant. Or rather, only PV systems that have neighbors in such conditions become selected as target.

The architecture of the LSTMs for all experiments (2-A, 2-B, 2-C) will stay in the configuration that had turned out to perfom well (cf. Section 4.3): 1 LSTM layer with 4 neurons, a batch size of 72, Adam as optimizer and a training time of 100 epochs. Each experiment will be

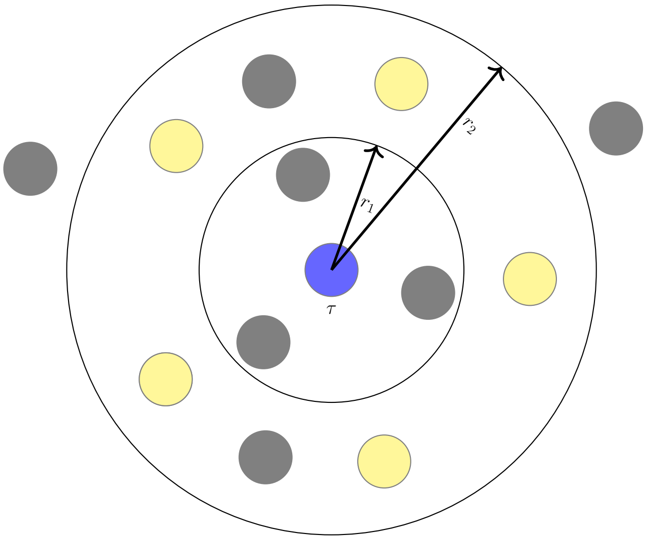


Figure 4.14: Experimental setup of experiment 2-C. Two radii r_1 and r_2 are defined to enclose an area within a certain distance from the targeted PV plant τ. Measurements of PV plants within this area (yellow circles) are used as feature. Other PV plants remain unused (gray circles).

run for 10 samples and the time horizon is selected according to the information obtained from the previous sections. Thus an input sequence of 3 hours assembled by 12 steps á 15 minutes is chosen for the prediction of a single time step at $t+1$. Consequently, the directly succeeding 15 minutes are being predicted (cf. Section 4.3 Fig. 4.4). The model performance as well as

error measurements according to experiment 2-A are listed in Table 4.9. As can be seen, taking surrounding systems as additional feature results in average RMSE values around 2.2 and 3.5 in percent of installed capacity. Those value ranges are similar to the results of prior investigations. More precisely the error measurements are a little greater here, than the ones attained before. Only predictions for the targeted plant made by measurements of the targeted plant ($N = 0$) have minimal error measurements. Hence why, an improvement in prediction accuracy had not been achieved by increasing the number of surrounding system used as feature. This holds almost regardless of the number of nearest neighbors within a distance of $r = 10$ km.

The skill score confirms this, as no clear superiority over the persistence forecast could be achieved for any number of N exceeding 4. The other positive values are too close to zero for this to be seen as an improvement. This is especially true when looking at the standard deviation of the skill scores. The standard deviation of the skill scores show large fluctuations. Consequently, the model performance with regard to the baseline is anything but stable. Reasons for this can possibly be found in the sample size, model architecture, and location as well as distribution of the surrounding systems around the different targeted plants.

- *Sample size.* Only ten samples have been investigated, given the limited time and computational duration of one model run. Ten samples might just be too little to reveal true

Table 4.9: RMSE and skill scores of experiment 2-A and the corresponding standard deviation (SD). The RMSE is listed as a percentage of the installed capacity of the targeted plants respectively. The sample size is 10, thus this setup has been tested for 10 different PV plants. The LSTM has been configured with 4 neurons, a batch size of 72, Adam as optimizer, and trained for 100 epochs. The outcomes are displayed for a varied number of closest neighbors N within a distance of $r = 10$ km.

N	ø RMSE (± SD)	Skill Score (± SD) [%]
0	2.21 (± 0.63)	13.58 (± 13.83)
4	2,62 (± 0,39)	2,47 (± 11,83)
6	2,78 (± 0,76)	-5,02 (± 24,79)
8	2,69 (± 0,53)	0,87 (± 13,89)
10	2,78 (± 0,31)	0,001 (± 13,28)
12	3,46 (± 2,34)	-30,90 (± 104,94)

dependencies between the number of neighbors with regard to prediction accuracy. There is a possibility that both those cases (a superiority of LSTM over persistence as well as the other way round) are rare sinlge cases. A larger sample size is required to ensure whether individual cases bias the results in one direction or the other.[8]

- *Model architecture.* The model architecture might be another reason for the divergent outcomes, despite the analyzation conducted above. As models are designed and trained for a specific task they can only retrieve adequate results for their purpose. The model has been adapted on one PV plant. The arrangement of the systems to each other is different depending on which target system is considered. The architecture might not provide enough complexity to capture multiple or rather some kind of general PV system arrangements.

- *Location and distribution.* The fluctuations might possibly be attributable to the location and distribution of surrounding systems around the targeted PV plant. The impact on prediction accuracy is most likely to be traced back to (i) whether or not the surrounding systems are covering each compass direction, and (ii) whether or not the compass directions are covered by an equally amount of surrounding systems. An example and explanation to this will be given in the following.

In Germany the average direction of the wind is western [Wol17], thus the cloud movement is most likely in accordance to this. The wind direction and cloud movement in turn impact the solar radiation. This means, the prediction accuracy might benefit or be harmed by the location and distribution of surrounding systems. Each of the following scenarios is possible:

- The surrounding systems are equally distributed in each of the compass direction to the target.

[8] The complete table of RMSE and skill scores per sample can be found in Table B.1.

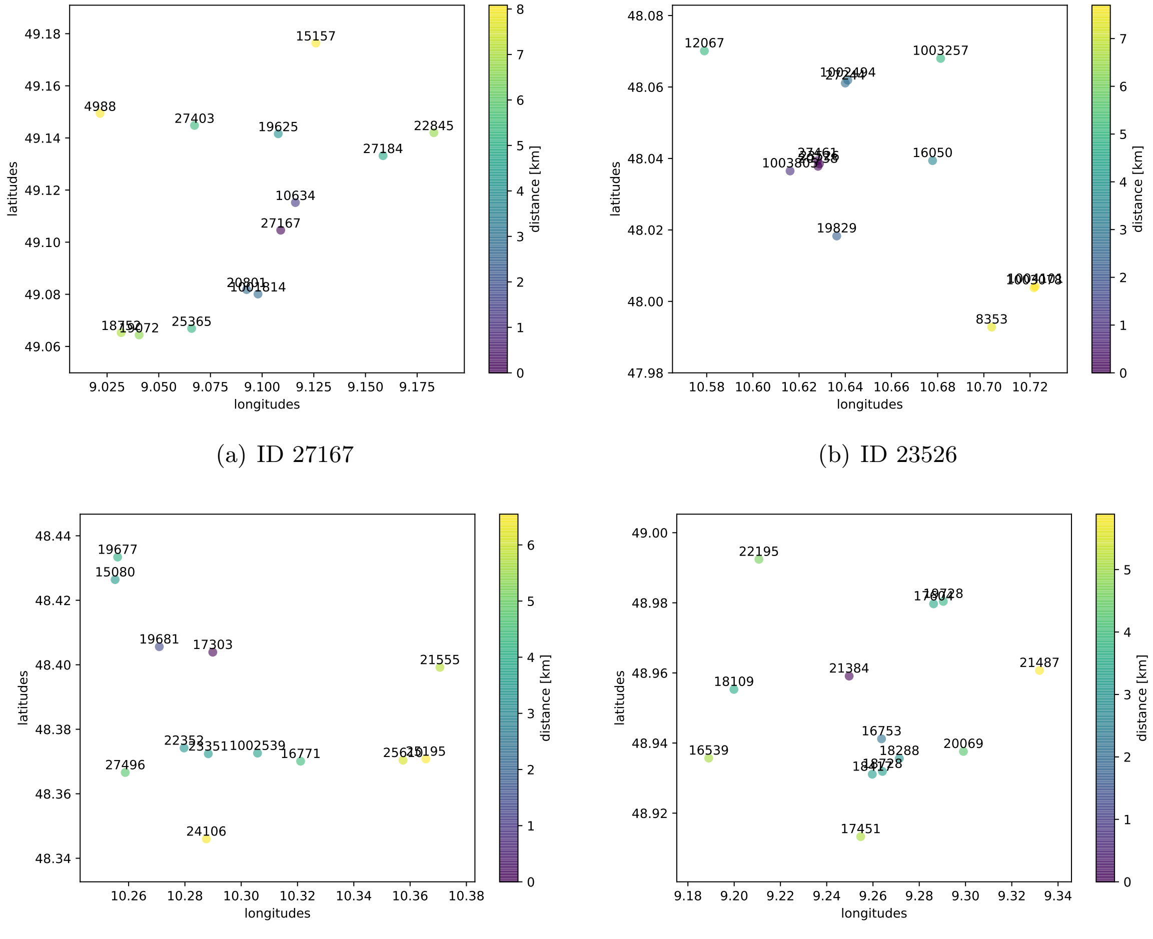
- The surrounding systems are covering each compass direction, but an agglomeration in one (or few) compass direction(s) is given.
- The distances of the surrounding systems with regard to the compass directions are highly divergent (i.e., the north is covered by neighbors within 1 km, whereas other directions are covered by neighbors being 5 and more km away).

In accordance to the wind direction, the predictions of the target system might benefit from surrounding systems that are located in the west. The weather conditions then first reach the surrounding systems before moving forward to the target (in the east). A change in PV power of the surrounding systems could then serve as precurse for the target. On the contrary, an arrangement in which the target is surrounded by systems only in other than western compass directions, the prediction accuracy might be declined. The impact then might rather be the other way round: the target could improve predictions of neighbors in the east. This assumption can only hold for restricted local areas around the target, since cloud movement appears on a short term scale (depending on the wind speed, and the time for clouds to build and decay).

Exemplary the arrangement of four targets of samples in experiment 2-A are shown in Figure 4.15. Some of the samples happen to be the same target for almost any number of surrounding systems. Hence why, the visualization is done for those. Other samples resulted in different targets, corresponding to the required amount of neighbors. Visualized are the distributions in correspondance to the distances for $N = 12$ of the following PV system IDs: 27167, 23526, 17303, 21384.

The examples shown in Figure 4.15 demonstrate the variability of how the PV systems can be locally arranged. It can be seen that some targets have their closest 12 neighbors within more or less than 5 km, while other targets require the use of more space to meet the requirements of 12 closest neighbors (up to 8 km). Further, both scenarios occur in which surrounding systems cover only a few compass directions as well as surrounding systems cover all compass directions but are agglomerated in some. Whether or not the orientation and/or distribution are the reason for the high fluctuation in the skill scores has to be explored in subsequent surveys. A consideration of other additional data, such as wind direction and/or wind speed could be beneficial for this.

Moving on to the outcomes of experiment 2-B. First of all the RMSE and skill scores for 4 up to 8 neighboring systems (N) in all distances are similarly low. For more surrounding systems as feature, the outcomes in varying distances to the targeted plant diverge. The highest RMSE values are achieved by both, $N = 10$ and $N = 12$ in 5 and 10 km distance. Other RMSE values in contrast in greater distances remain rather the same (i.e., all N in 20 km distance have similar RMSE). Thus, the increase of RMSE might probably be traced back to the same reasons denoted as explanation in experiment 2-A, as the standard deviation in accordance to 10 and 12 surrounding systems in 5 and 10 km distance are remarkably high.



(a) ID 27167

(b) ID 23526

(c) ID 17303

(d) ID 21384

Figure 4.15: Exemplary representation of distributions from experiment 2-A. Shown are target systems with 12 neighbors, which were the same for all tested numbers of neighbors. The maximum possible distance is $r = 10$ km. Both the space used for meeting the requirements of the experiment as well as the arrangement of surrounding systems in each compass direction is different per target.

The overall error values are in an order of magnitude comparable to previous results. Further information about the model performance is provided by the skill scores. Most outstanding is the poor performance for any number of neighbors within a radius of 5 km around the target. Not any of these skill scores achieve to beat the persistence forecast (all skill scores are below 0). Chances are high that either one or all of the following key points occur:

- The arrangement of surrounding PV systems around the target is unfortunate (see above).
- The network in its specified configuration is not able to capture and identify the dependencies of too closely arranged PV systems.
- PV systems in immediate vicinity cannot improve prediction accuracy.

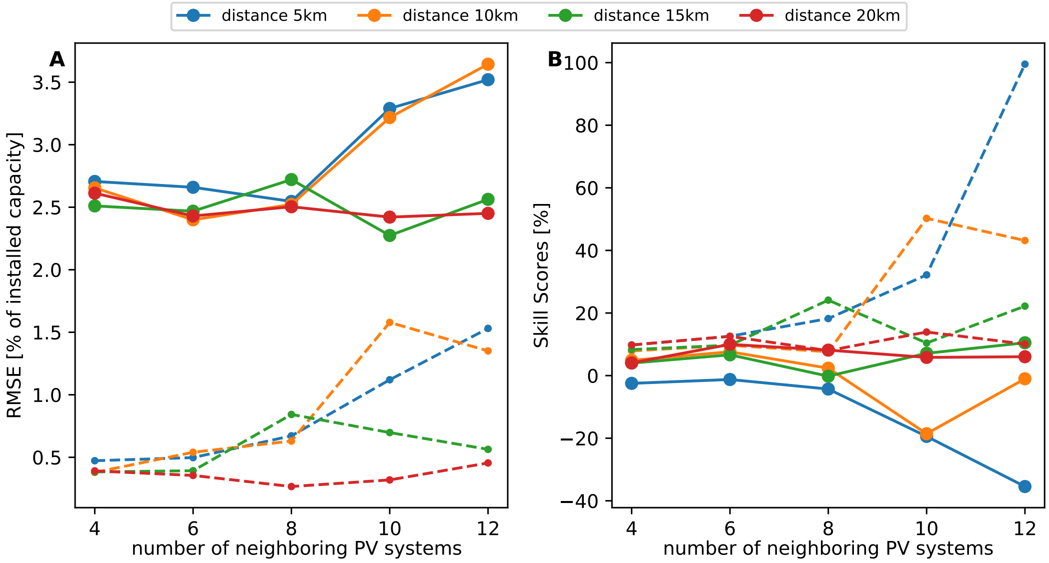


Figure 4.16: The average RMSE and skill scores (solid lines) and standard deviations (dashed lines) are displayed in A and B respectively. The experiment has been run on a sample size of 10. The values are denoted to a corresponding number of neighboring systems. Each number of neighbors can be located at random in different distances to the targeted plant. The distance represents the radius in kilometers around the targeted plant.

Other two distances ($r = 10$ and $r = 15$ km) in contrast do not show a comparable unambiguous outcome. The skill scores corresponding of those two distances are higher as well as lower than zero for any number of neighboring system. Outstanding is the combination of $N = 10$ and $r = 10$ km, which also marks a low point of the skill scores. The negative scores come along with high standard deviations, whereas positive skill scores mostly do not. Thus, maybe just a few targeted PV plants cannot receive accurate predictions in a spatio-temporal approach. Or rather, some predictions for single plants could be outstandingly poor in contrast to other well predicted ones. A complete table of all RMSE values and skill scores per sample can be looked at in the Appendix in Table B.2.

In contrast to outcomes within 5 km, the outcomes for any N within 20 km are the most stable. The skill scores are all higher than zero and the standard deviations are pleasantly low. It is noticeable that the skill score is similar for each of the surrounding systems in that distance. Thus, either there is no difference in the number of surrounding systems used at this distance, or the arrangement of the plants was favorable for samples of PV systems at this distance. A larger sample size would bring this out.

Further, the present representation of the results conceals the actual distance of the neighboring systems to the target. The surrounding PV systems are allowed to be randomly located within radii of 5, 10, 15, and 20 kilometers. In Table 4.10 the actual average distances are listed, in which the surrounding sysmtes are located from the targets. As can be seen, the space used per distance is similar regardless of how many neighbors are considered (similar values per column). Most interestingly, a radius of just under 12 km is not exceeded, even though up to 20 km are allowed as maximum distance. Furthermore a shift in distances can be seen. For example, the allowed radius of 10 km for example is rather an actual radius of about 5 km. Consequently, the distance of 20 km has not been truly investigated and the results obtained refer rather to short distances of about 12 km around the respective target system.

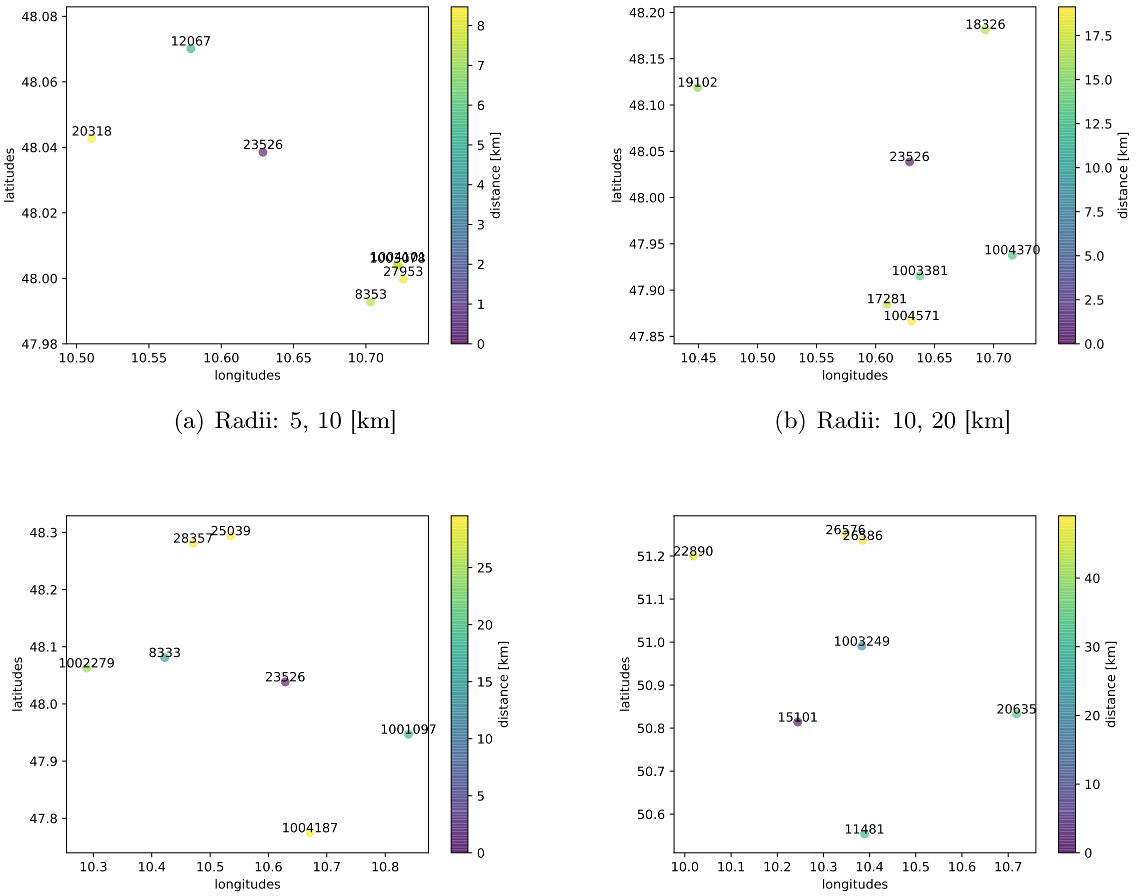
Table 4.10: Table of the actual average distance (and standard deviation (SD)) in which the randomly drawn surrounding PV systems of the target PV system are located. The actual distances are listed in accordance to the number of surrounding systems N. Further the header denotes the radius of maximal distance possible for each neighboring system. Radii of 5, 10, 15, and 20 kilometers have been tested.

N	**Distances (± SD) [km]**			
	5	10	15	20
4	2.61 (± 1.06)	5.19 (±1.94)	7.93 (± 3.83)	10.43 (± 1.91)
6	2.53 (± 0.73)	5.44 (±1.16)	7.47 (± 1.97)	11.71 (± 2.83)
8	2.85 (± 0.66)	5.54 (±1.20)	7.49 (± 1.80)	11.34 (± 2.53)
10	2.40 (± 1.13)	5.48 (±1.79)	8.18 (± 1.90)	11.53 (± 2.85)
12	2.15 (± 1.15)	5.03 (±1.94)	8.45 (± 1.50)	10.23 (± 3.24)

The average actual distances reflect what had been assumed before. The model is well performant for surrounding systems within 10 km and the number of neighbors seem to have little impact. In contrast, the architecture might not provide enough complexity for additional data of larger number of neighboring systems than $N = 8$ and is not able to distinguish measurements of the target from too closely arranged surrounding systems. Those ones are troublesome and cannot beat the persistence forecasts. Uncertainty remains whether the influence on the prediction accuracy is due to the number and distance of the surrounding systems or due to the distribution and arrangement of the neighbors.

The results of the last experiment 2-C are represented in Figure 4.17 and Table 4.11. The figure displays the formation and distribution of surroundings around the target PV system, whereas the table lists the RMSE, skill score, and average distances. Both representations are according to each considered pair of radii. As can be seen in Figure 4.17, the 6 surrounding PV systems of a randomly picked target are mostly located north-western or south-eastern. The distribution varies whether more systems are north-western or south-eastern. Thus, even for one sample that meets the requirements of different radii, the arrangement and distribution of nearby systems is very different.

Each RMSE and skill score shows a superiority for the first three pair of radii above the persistence forecast (skill score > 0). The first pair of radii ([5,10]) matches the results of the previous two experiments (2-A and 2-B). The LSTM superiority is only just given with a small skill score above zero. With further distances of the neighboring systems from the targeted plant the RMSE first declines before rising again. The skill scores in contrast reveal an almost



(a) Radii: 5, 10 [km]

(b) Radii: 10, 20 [km]

(c) Radii: 15, 30 [km]

(d) Radii: 20, 50 [km]

Figure 4.17: Distribution of a target and surrounding PV systems within different radii according to experiment 2-C. The first radius determines the minimal distance that the neighbors have to exceed, whereas the second radius determines the maximal distance in which the neighbors have to be located around the target.

equally well model performance for the middle two pair of radii (radii [10,20] and [15,30]), whilst the furthest tested distance (radii [20,50]) fails to beat the persistence.

Thus it cannot clearly be stated that further distanced surrounding PV systems exclusively enhance or debase predictions of the targeted plant. Rather, the outcomes indicate sufficient forecast accuracy when additional feature measurements are derived from plants in medium distance to the target.

The average distances confirm that a certain distance has been hold. Those distances and the resulting area led to the PV systems being wider distributed, because with an increased area chances are higher for the PV systems being more apart. This circumstance might be the cause for overall low RMSE values in this experiment. Further does it imply that the distribution again plays a crucial role for helping to improve the error measurements by using measurements of surrounding PV systems. Once more, based on the sample size tested, it cannot be entirely excluded that predictions for individual plants may be both very accurate and very poor.

In contrast, the low RMSE values might be due to the determined number of surrounding systems instead of the defined radii. The use of $N = 6$ randomly drawn neighbors showed to result in low RMSE values and a sufficient skill score, as long as the surrounding systems are not too closely located around the target (cf. experiment 2-B). In addition, the model architecture had been analyzed (and optimized) for a spatio-temporal setup using measurements of 6 PV system as additional feature.

Table 4.11: RMSE in percentage of installed capacity, skill scores, and average distances of experiment 2-C. The displayed radii are in km and define an surrounding area around the targeted PV plant.

Radii [min, max]	RMSE	Skill Score [%]	Average Distances [km]
[5, 10]	2.60 (± 0.52)	3.54 (± 14.15)	7.48 (± 0.81)
[10, 20]	2.34 (± 0.32)	9.14 (± 10.05)	15.69 (± 1.23)
[15, 30]	2.26 (± 0.54)	9.02 (± 6.00)	22.88 (± 1.76)
[20, 50]	2.57 (± 0.54)	-0.64 (± 14.76)	37.87 (± 3.41)

As a conclusion, the definition of another radius to set a minimal distance to be hold, specified the investigation of the area around the target PV system. Nonetheless, in order to derive more precise or unambiguous conclusions from the spatial contribution to the LSTM predictions, further investigations need to be taken. Overall the spatial information can partly be used to make sufficiently precise PV power predictions for some distances and number of surrounding systems with LSTMs that are superior to persistence forecasts.

4.6 Comparison to SVR

As a conclusive assessment of the suitability of LSTMs for the prediction of solar power, a methodological comparison will be carried out. In the previous chapters LSTMs are used in a spatio-temporal approach making predictions for one single PV system. The LSTM architecture and temporal as well as spatial aspects have been analyzed. For a more comprehensive impression of the capabilities of LSTMs for solar power prediction, this section focuses on predictions for more than one PV system and compares predictions of LSTMs with predictions made with SVR. As described in Section 1.3, the present thesis is inspired by studies on solar power production using machine learning methods from B. Wolff.

B. Wolff assessed the "general usefulness of machine learning, and especially support vector regression (SVR) as a modeling technique for PV power forecasting." [Wol17]. Therefore his predictions made with SVR will serve as reference for comparison. The data used in this thesis as well as in [Wol17] is the same, thus foundation of the comparability builds the data availability. In the following, a short introduction to SVR will be given as well as a description of the experimental setup and data preprocessing.

4.6.1 Support Vector Regression

The SVR is a machine learning method that evolved from support vector machine (SVM) and has been introduced by Smola and Schölkopf in 2004 [Wol17]. Support vector machines are a class of supervised learning algorithms for solving classification problems and SVR represents an adaption of SVM for regression tasks. The basic idea of SVR is to find a hyperplane in a continuous input space that represents the data points best in compliance with user-defined constrains. [Wol17]

A hyperplane thereby defines a (n-1) dimensional subspace in a n-dimensional space. More precisely, a hyperplane H is a subset of an n-dimensional space $\mathbb{R}^n$, which is shifted by a support vector $s \in \mathbb{R}^n$ from the coordinate origin, with $v \in \mathbb{R}^n\ \{0\}$ as the normal vector of H,

$$H = \{x \in \mathbb{R}^n | \langle v, x - s \rangle = 0\}. \tag{4.2}$$

Examples of hyperplanes are a plane in a space, or a line in a plane. [Dür12]

When SVM is being used to solve a classification problem, the goal is to place a hyperplane in a way that all data points of the input space are assigned to classes whose distances are to be maximal. When reformulating SVM to SVR, the hyperplane supports predicting continuous values. The difficulty thereby is due to infinite possibilities of predictable values. Therefore a margin of tolerance is set in which the predicted values are acceptable. The principal of the method then is to minimize the error (of the loss function) and choose a hyperplane that maximizes the margin with respect to the tolerated error. [Say19]
Futher information can be found in [SS04] and [Wol17].

4.6.2 Experimental Setup

The SVR predictions in [Wol17] are based on measurement data from 921 PV systems. The PV systems had been chosen after an assessment of the data quality. Systems with as high quality data as possible got selected. The training set included the measured values from year 2012, while the test set consists of measurements of year 2013. [Wol17]

Consequently, the present investigation rebuilds the experimental setup as in [Wol17]. Thus, other than preceding experiments in this thesis, the present study trains and predicts for more than one single PV system. The IDs of the PV systems served as attribute to make a congruent selection of the PV systems with the ones in B. Wolff's work. Training and test sets are defined in accordance to [Wol17] as year 2012 and 2013, respectively. Missing values are replaced by either the value of a preceding time step as long as no more than four successive time steps are missing or by zero (cf. Section 4.1). Analogously, the power values of each PV system are scaled to their installed capacity for better comparability of various sized PV systems. Predictions of one time step ($t+1$ and $t+4$) will be generated and compared, as previous results have shown highest accuracy for a single time step. The architecture of the LSTM will remain as described before. Thus, one LSTM layer with 4 neurons, a batch size of 72, Adam as optimizer and 100 epochs of training will be used. As before the predictions will be assessed by calculating the RMSE values between each of the predicted time series with the true measurements. Further the model performances will be evaluated using the skill score. One time the skill score for both methods is calculated with the persistence forecast as reference. Another time, the performance of LSTMs is evaluated using the skill score with regard to SVR forecasts as a reference.

The results are presented in Table 4.12 and Table 4.13. In the first table the average RMSE values and standard deviations are shown in accordance to the two predicted time steps. The average thereby is formed from the RMSE values achieved per PV plant per time series of the entire test set. Only one plant has been excluded from the calculations, as unrealistic results had been obtained for this plant[9].

Table 4.12: RMSEs from the comparison of SVR and LSTM. Shown are the average RMSEs and standard deviations (SD) in % of installed capacity per prediction method and predicted time step. The averages were build with regard to (i) the RMSEs for each system and each predicted time step of the test set and (ii) the averages over all 920 PV systems. P is the abbreviation for persistence forecast here.

Time	**RMSE**		
Horizon	P (± SD)	SVR (± SD)	LSTM (± SD)
15 min ($t+1$)	8.35 (± 1.30)	5.20 (± 0.54)	4.21 (± 1.41)
60 min ($t+4$)	10.28 (± 2.56)	7.35 (± 0.80)	4.66 (± 1.40)

The second table shows the average skill scores and standard deviation analogously. The first two columns (“SVR over P”, “LSTM over P”) represent the skill score with the persistence forecast as reference for SVR and LSTM respectively, whereas the last column represents the skill score for the LSTM performance with SVR as reference.

As can be seen, the RMSE values rise with increased predicted time horizon (vertical comparison). Conspicuously, the increase in the RMSE of the LSTMs from one time step to the other is the lowest in comparison with the other methods. Thus, predictions of the LSTMs hardly deteriorate and are similarly close to the true values within the time horizon of one hour. Further, the LSTMs achieve overall lowest RMSE values. The table shows that SVR beats persistence by lower RMSEs, and that the LSTMs even undercut RMSE values of the SVR.

[9] The outcomes of PV plant with the ID 17738 are excluded. See appendix Table B.3 for those outlined outcomes.

Table 4.13: Skill scores from the comparison of SVR and LSTM and the corresponding standard deviations (SD). Here P is the abbreviation for persistence forecast. The column names are read as follows: method 1 over method 2 represents that method 2 builds the reference for method 1.

Time	**Skill Score [%]**		
Horizon	SVR over P (± SD)	LSTM over P (± SD)	LSTM over SVR (± SD)
15 min ($t+1$)	37.00 (± 6,35)	49.15 (± 16,66)	19.00 (± 27,09)
60 min ($t+4$)	27.19 (± 7.12)	53.78 (± 13.70)	36.48 (± 17.45)

The skill scores show that both, SVR as well as LSTMs, have higher model performance than the naive approach, as their skill scores are positive values. The LSTMs perform even better than the SVR, because the LSTMs achieve higher skill scores than SVR in that comparison (column 1 and 2 in Table 4.13). Most interestingly, the change in skill scores from one time step to the other is different for SVR and LSTMs when persistence is the baseline. The skill score for SVR decreases, as the difference in RMSE increase for SVR is larger than for the persistence from $t + 1$ to $t + 4$. Consequently, the superiority of model performance for SVR over persistence is less for the predictions of one hour ahead compared with predictions of the successive 15 minutes. This is the other way round for skill scores of the LSTMs over persistence. The LSTM skill scores increase from $t + 1$ to $t + 4$ due to the fact that the rise in RMSE of LSTMs is marginal.

The direct comparison of model performance of LSTMs and SVR shows a superiority of LSTMs over SVR (column 3 in Table 4.13). The relative improvement of LSTMs over SVR is given for both time steps, whereas the advantage is even greater for predictions of one hour ahead. An explanation for this is again the alteration of the RMSE from one time step to another. Predictions of both methods lead to higher error values with a larger predicted time horizon, but the increase in RMSE for the LSTMs is considerably smaller. This means, despite a decline in prediction accuracy for both methods, the relative improvement for LSTMs over SVR had been enhanced by this.

As a conclusion, LSTMs had been able to beat persistence forecasts as well as SVR predictions. The comparison shows that LSTMs had been able to improve short-term predictions of solar power, by being more accurate than the referenced methods.

5 Conclusion

A reduction of CO2 emission through an enlarged integration of renewable energies such as solar power requires accurate forecasts. A development of novel methods or improvements in forecast accuracy of proven methods can take major contributions to energy system transformation. By this, the energy grid can be kept stable by energy system operators and the energy supply remains ensured. This thesis focused on improvements of solar power prediction by assessing the use of LSTMs for this task.

5.1 Summary

There are several methods for solar power prediction. The difference of the methods lies within the purpose of the forecasts, whether short-term or long-term predictions are requested. Further the data required and its accessibility limits the possibilities for generating a forecast. Deep learning methods and especially LSTM networks offer high potential to improve predictions of renewable energies and solar power in particular. Their capability to identify long-term dependencies in sequences enables to generate most accurate short-term forecasts.

In this thesis a spatio-temporal approach had been chosen to assess the capability of LSTMs to predict PV power of one single system. First an evaluation of the LSTM architecture had been done by manually experimenting with values of the following hyperparameters: the optimization algorithm, the duration of training, the number of neurons, and the batch size. Almost all configurations had been able to score lower RMSE values than the reference model. The most outstanding improvement had been achieved by changing the optimization algorithm from SGD to Adam. The resulting architecture used to predict most accurately for one PV system is the one with one LSTM layer with 4 neurons, a batch size of 72, a training duration of between 50 to 100 epochs, and Adam as used algorithm. Consequently, LSTMs with a simple architecture are able to outperform a persistence forecast.

Further, an investigation of different predicted time horizons helped to estimate the impact of both, input and output sequence length on the prediction accuracy. The prediction of a single time step achieved most accurate results, even if the time step is distanced from the feature sequence. The results show that with further distanced predicted time steps (a time horizon of several hours), predictions become more general. This can be traced back to the time series of PV measurements themselves as the autocorrelation function as well as partial autocorrelation function revealed very short-term dependencies. Thus, the first two research questions that aim at identifying the configuration of a LSTM architecture and assessing the performance of LSTMs in a spatio-temporal setup have shown that LSTMs are able to generate reliable short-term predictions of solar power.

The analysis of the spatial experimental setup focused on the identification of spatial dependencies and whether or not additional various amount of feature information can help to improve the forecast accuracy for one PV system. This had been adressed by the third reseach question that asked "How does spatio-temporal information contribute to PV power predictions?". Experiments in Section 4.5 compared different spatial setups and evaluated the contribution of the number and distance of surrounding PV systems. The LSTMs were able to process additional data from surrounding PV systems and to make predictions for a targeted PV plant. However, an assignment to specified importance of neighboring systems cannot be extracted. The results achieved revealed spatial dependencies that need further and more detailed studies, such as optimized models for any $N - r$ constellation. The surrounding systems' compass direction as well as distribution around a targeted system seem to have a larger impact on the outcome of forecasts than the amount of surrounding systems and their distance. On that account the contribution due to the spatio-temporal information is dependant on a specifically adapted model architecture (for the $N - r$ combination of interest) as well as the arrangement and distribution of the surrounding systems around a target, since some $N - r$ constellations were superior and other were inferior to the persistence forecasts.

Finally, a methodological comparison was accomplished in order to classify the capability of the LSTMs for solar power prediction in comparison with other machine learning methods. Results of B. Wolff's doctoral thesis [Wol17] served as a reference for a comparison of LSTMs with SVR, since the used data in both is the same. The experimental setup reflected a more general methodical assessment and predictions for more than one PV system had been carried out. The comparison had been assessed by the incorporation of prior results on training and configuring the LSTMs. So, predictions of a single time step have been compared with each other. The results show a superiority of LSTMs over SVR by achieving lower RMSE values and positive skill scores. Consequently, the last research question can be answered by the findings that LSTMs as a prediction method for solar power can keep up with and even surpass other machine learning methods, such as SVR.

5.2 Outlook

The topic of solar power prediction using LSTMs and an analysis of how to improve forecasts even more offers further reaseach possibilities. This section serves as an introdution and inspiration of related research topics, more in-depth analysis possibilities, and subsequent open questions. The aspects and open questions mentioned below remark logical next steps for assessing the use of LSTMs for solar power prediction.

In this thesis, the predictions are exclusively based on historical PV power measurements. A use of further related weather data, such as irradiation, temperature, wind speed and wind direction could be used to investigate whether or not such data can help to increase the prediction accuracy. The mentioned data could be provided by numerical weather prediction and/or cloud movement forecasts. Depending on the source and data accessibility predictions with LSTMs might benefit from this. Results of Section 4.4 suggest an increase in predictions accuracy by providing the model longer input sequences.

A more outstanding role in additional data might be related to wind speed and wind direction data. In Section 4.5 the spatial dependencies have been investigated. The conducted experiments suggest a high impact to the distribution and compass direction location of surrounding PV systems related to the targeted PV system. Additional wind data could help to reveal whether or not such impact is related to wind and cloud movement.

Further spatial correlations could be investigated by separating the experimental steps into a model with two components: one for the spatial information and one for the temporal information. A specified approach that analyzes the spatial dependencies could be the use of convolutional neural networks. Convolutional RNNs have been used in video application to handle both, spatial and temporal relations [Zia+17]. A transfer application on spatio-temporal PV power predictions marks a further research topic. This would bring out more in-depth information about forecast improvements depending on the location of surrounding PV systems whose measurements are used as feature in a spatio-temporal setting.

A consecutive comparison of LSTMs with further reference models extends the analysis of LSTMs more deeply. In this thesis the focus lied on assessing the capability of LSTMs in a spatio-temporal setup. Main comparisons have been against a persistence forecast as that kind of reference is easily and quickly available. Comparing LSTMs with more complex models can help to classify methods of solar power predictions. The comparison with SVR in Section 4.6 served as a first insight. Ongoing comparisons could bring out strengths and weaknesses of each method in relation to each other. The importance of LSTMs and whether or not their predictions are superior over most methods could be assessed by this.

In terms of LSTMs or RNNs in general, the architectures provide elements of adjustment to achieve improvements. The results of Section 4.3 confirmed that some hyperparameters have more impact on prediction accuracy than others. Overall, endless opportunities on tuning LSTMs exist. Beginning by the values tested, moving on to the hyperparameters to be tested and ending with the method used for optimization. Thus, three starting points for further research that have not been considered in this work yet, could be the following:

- The use of other optimization techniques for hyperparameter optimization, such as random search or genetic algorithms.
- The consideration of other and/or more hyperparameters, such as the number of layers or the activation functions used.
- The choice of other modeling approaches like ensemble setups or other RNN architectures such as gated recurrent units (GRUs) or temporal convolutional networks (TCNs).

Altogether, using and analyzing more and different data sources as well as other references and comparative methods could help to find out about the possibilites to improve PV power forecasts. Any improvement in forecast accuracy is helpful for large-scale integration of PV power as a power supply source, scheduling power demands for safe energy system operation when integrating PV power, and last but not least minimize unnecessary costs due to power fluctuations. A comprehensive overview on the methods could then provide the opportunity to select the appropriate method with regard to the forecast purpose and the data availability as well as accessibility. The thesis contributed to such by investigating the use of LSTMs in a spatio-temporal setting and by comparing the capability of LSTMs with SVR.

List of Figures

List of Tables

Acronyms

AI	artificial intelligence
ANN	artificial neural network
BGD	batch gradient descent
BP	backpropagation
BPTT	backpropagation through time
CMV	cloud motion vector
DNI	direct normal irradiation
GD	gradient descent
GHI	global horizontal irradiation
LSTM	long short-term memory
ML	machine learning
MLP	multilayer perceptron
MSE	mean squared error
NN	neural network
NWP	numerical weather prediction
PV	photovoltaic
RMSE	root mean square error
RNN	recurrent neural network

SGD	stochastic gradient descent
SP	simple perceptron
SVM	support vector machine
SVR	support vector regression

List of Symbols

A	Recurrently connected unit of an ANN
b	Threshold of an artificial neuron
$\boldsymbol{C}$	Cell state of a LSTM block
$\tilde{\boldsymbol{C}}_t$	New candidate solutions for cell state of a LSTM block
δ	Adjustment of the weight in a ANN during learning
$E(w)$	Error or loss function
η	Learning rate
f_t	Forget gate of a LSTM block
G_i	Irradiation received by panel plane
γ	Defined tolerable value to end ANN traning
H	Hyperparameter
h_t	Hidden state variable of a RNN, memorizing output of time step t
I_{diff}	Diffuse Irradiation
I_{DNI}	Direct Normal Irradiation
I_{GHI}	Global Horizontal Irradiation
i_t	Input gate of a LSTM block
μ	Previous time steps that serve as feature for solar power prediction

N	Neighboring PV panels (measurements serve as addidtional features for solar power prediction of a targeted PV system)
ν	Forecast horizon, can be a single time step or a sequence of time steps
o_t	Output gate of a LSTM block
P	Training set
$\varphi(x)$	Activation function
Q	Validation set
R	Test set
r	Radius that defines an area in which neighboring PV systems can be located
$\sigma(x)$	Sigmoid or logistic function
t	Time step at current time
$\tanh(x)$	Hyperbolic tangent function
τ	Targeted PV system of which solar power is predicted
T_m	PV panel temperature
$\boldsymbol{W}$	Weight matrix of an ANN
w	Weights of an artificial neuron
X	Input space
Y	Output space
$\hat{y}$	Predicted output
ζ	Defined threshold for the gradient in order to stop training an ANN

Bibliography

[AM17] Mohamed Abdel-Nasser and Karar Mahmoud. “Accurate photovoltaic power forecasting models using deep LSTM-RNN”. In: *Neural Computing and Applications* (2017), pp. 1–14.

[Bad18] ZWS Baden-Württemberg. *Erneuerbare Energien in Deutschland. Daten zur Entwicklung im Jahr 2017.* Tech. rep. Umweltbundesamt, Mar. 2018. URL: https://www.umweltbundesamt.de/publikationen/erneuerbare-energien-in-deutschland-2017.

[BB12] James Bergstra and Yoshua Bengio. “Random search for hyper-parameter optimization”. In: *Journal of Machine Learning Research* 13.Feb (2012), pp. 281–305.

[BDC02] Peter J Brockwell, Richard A Davis, and Matthew V Calder. *Introduction to time series and forecasting.* Vol. 2. Springer, 2002.

[Ben12] Yoshua Bengio. “Practical recommendations for gradient-based training of deep architectures”. In: *CoRR* abs/1206.5533 (2012). arXiv: 1206.5533. URL: http://arxiv.org/abs/1206.5533.

[Ber+15] K. Bergen et al. *Distributed Algorithms and Optimization.* https://stanford.edu/~rezab/classes/cme323/S15/notes/lec11.pdf. Accessed: 2019-02-22. Apr. 2015.

[BH00] Imad A Basheer and Maha Hajmeer. “Artificial neural networks: fundamentals, computing, design, and application”. In: *Journal of microbiological methods* 43.1 (2000), pp. 3–31.

[Bia+17] Filippo Maria Bianchi et al. “Recurrent Neural Network Architectures”. In: *Recurrent Neural Networks for Short-Term Load Forecasting: An Overview and Comparative Analysis.* Cham: Springer International Publishing, 2017, pp. 23–29. DOI: 10.1007/978-3-319-70338-1_3. URL: https://doi.org/10.1007/978-3-319-70338-1_3.

[Bur18] Prof. Dr. Bruno Burger. *FRAUNHOFER INSTITUTE FOR SOLAR ENERGY SYSTEMS ISE. Power generation in Germany – assessment of 2017.* www.energy-charts.de/energy_pie.htm. Accessed: 2018-11-05. 2018.

[Cen19] IBM Knowledge Center. *Autokorrelation und partielle Autokorrelationsfunktionen.* https://www.ibm.com/support/knowledgecenter/de/SS3RA7_sub/modeler_mainhelp_client_ddita/components/dt/timeseries_acf_pacf.html. Accessed: 2019-06-17. 2019.

[Cho+15] François Chollet et al. *Keras*. https://keras.io. 2015.

[CI16] DK Chaturvedi and I Isha. "Solar Power Forecasting: A Review". In: *International Journal of Computer Applications* 145.6 (2016), pp. 0975–8887.

[Dey16] Ayon Dey. "Machine Learning Algorithms: A Review". In: *vol* 7 (2016), pp. 1174–1179.

[Dür12] Klaus Dürrschnabel. "Analytische Geometrie". In: *Mathematik für Ingenieure: Eine Einführung mit Anwendungs- und Alltagsbeispielen.* Wiesbaden: Vieweg+Teubner Verlag, 2012, pp. 143–182. ISBN: 978-3-8348-2559-9. DOI: 10.1007/978-3-8348-2559-9_7. URL: https://doi.org/10.1007/978-3-8348-2559-9_7.

[Fau06] Kjell Magne Fauske. *Neural Network | TikZ example.* http://www.texample.net/tikz/examples/neural-network/. Accessed: 2019-01-29. Dec. 2006.

[Fed18] Mark Fedkin. *Introducing Concentrating Solar Power.* https://www.e-education.psu.edu/eme812/node/646. Accessed: 2018-12-17. 2018.

[Gal15] Crescenzio Gallo. "Artificial Neural Networks: tutorial". In: Jan. 2015. ISBN: 9781466658882.

[GG16] Yarin Gal and Zoubin Ghahramani. "A Theoretically Grounded Application of Dropout in Recurrent Neural Networks". In: *Advances in Neural Information Processing Systems 29*. Ed. by D. D. Lee et al. Curran Associates, Inc., 2016, pp. 1019–1027. URL: http://papers.nips.cc/paper/6241-a-theoretically-grounded-application-of-dropout-in-recurrent-neural-networks.pdf.

[Gra12] Alex Graves. "Supervised sequence labelling". In: *Supervised sequence labelling with recurrent neural networks.* Springer, 2012, pp. 5–13.

[Guo13] Jiang Guo. "Backpropagation through time". In: *Unpubl. ms., Harbin Institute of Technology* (2013).

[HRW86] Geoffrey E Hinton, DE Rumelhart, and Ronald J Williams. "Learning representations by back-propagating errors". In: *Nature* 323.9 (1986), pp. 533–536.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[ISE18] Fraunhofer ISE. *Recent Facts about Photovoltaics in Germany.* https://www.ise.fraunhofer.de/content/dam/ise/en/documents/publications/studies/recent-facts-about-photovoltaics-in-germany.pdf. Accessed: 2018-11-05. July 2018.

[KB14] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[Kra09] Oliver Kramer. *Computational Intelligence: Eine Einführung.* Springer-Verlag, 2009.

[Kri07] David Kriesel. *A Brief Introduction to Neural Networks.* http://www.dkriesel.com/en/science/neural_networks. Accessed: 2018-01-30. 2007.

[Leo18] J. Leon. *w/o.* https://tex.stackexchange.com/questions/432312/how-do-i-draw-an-lstm-cell-in-tikz. Accessed: 2019-02-16. May 2018.

[Li13] Xiaoye Li. “Three Essays on Non-stationary Time Series”. dissertation. The Pennsylvania State University, 2013. URL: https://etda.libraries.psu.edu/files/final_submissions/9136.

[Man+12] Paras Mandal et al. “Forecasting power output of solar photovoltaic system using wavelet transform and artificial intelligence techniques”. In: *Procedia Computer Science* 12 (2012), pp. 332–337.

[Mar+15] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.* Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[McK10] Wes McKinney. “Data Structures for Statistical Computing in Python”. In: *Proceedings of the 9th Python in Science Conference.* Ed. by Stéfan van der Walt and Jarrod Millman. 2010, pp. 51–56.

[met19] meteocontrol. *Über meteocontrol - meteocontrol GmbH.* https://www.meteocontrol.com/unternehmen/ueber-meteocontrol/. Accessed: 2019-03-02. 2019.

[MH11] Jovin J Mwemezi and Youfang Huang. “Optimal facility location on spherical surfaces: algorithm and application”. In: *New York Science Journal* 4.7 (2011), pp. 21–28.

[MK08] Adel Mellit and Soteris A Kalogirou. “Artificial intelligence techniques for photovoltaic applications: A review”. In: *Progress in energy and combustion science* 34.5 (2008), pp. 574–632.

[Mur88] Allan H. Murphy. “Skill Scores Based on the Mean Square Error and Their Relationships to the Correlation Coefficient”. In: *Monthly Weather Review* 116.12 (1988), pp. 2417–2424. DOI: 10.1175/1520-0493(1988)116<2417:SSBOTM>2.0.CO;2. eprint: https://doi.org/10.1175/1520-0493(1988)116<2417:SSBOTM>2.0.CO;2. URL: https://doi.org/10.1175/1520-0493(1988)116%3C2417:SSBOTM%3E2.0.CO;2.

[na19] n.a. *Kilowatt-peak (kWp).* http://www.strom-prinz.de/Strom-Lexikon/details/article/kilowatt-peak-kwp/. Accessed: 2019-03-04. 2019.

[Oli06] Travis Oliphant. *NumPy: A guide to NumPy*. http://www.numpy.org/. Accessed: 2019-04-01. 2006.

[Ped+11] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[Pel+13] Sophie Pelland et al. *Photovoltaic and Solar Forecasting: State of the Art*. Oct. 2013. ISBN: 978-3-906042-13-8.

[PMB12] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. “Understanding the exploding gradient problem”. In: *CoRR, abs/1211.5063* 2 (2012).

[RG17] Nils Reimers and Iryna Gurevych. “Optimal hyperparameters for deep lstm-networks for sequence labeling tasks”. In: *arXiv preprint arXiv:1707.06799* (2017).

[Rud16] Sebastian Ruder. “An overview of gradient descent optimization algorithms”. In: *arXiv preprint arXiv:1609.04747* (2016).

[Sal+17] Hojjat Salehinejad et al. “Recent advances in recurrent neural networks”. In: *arXiv preprint arXiv:1801.01078* (2017).

[Say19] Dr. Saed Sayad. *Support Vector Regression*. https://www.saedsayad.com/support_vector_machine_reg.htm. Accessed: 2019-07-20. 2019.

[SL41] G. Schulte and W. Löhr. “Koordinatensysteme”. In: *Markscheidekunde: für Bergschulen und für den praktischen Gebrauch*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1941, pp. 108–112. ISBN: 978-3-662-22130-3. DOI: 10.1007/978-3-662-22130-3_10. URL: https://doi.org/10.1007/978-3-662-22130-3_10.

[SS04] Alex J. Smola and Bernhard Schölkopf. “A tutorial on support vector regression”. In: *Statistics and Computing* 14.3 (Aug. 2004), pp. 199–222. ISSN: 1573-1375. DOI: 10.1023/B:STCO.0000035301.49549.88. URL: https://doi.org/10.1023/B:STCO.0000035301.49549.88.

[SU18] Sabine Seidel and Kristian Uhlenbrock. *Infoblatt Das Gradnetz der Erde*. https://www.klett.de/alias/1037707. Accessed: 2019-05-09. 2018.

[Sut13] Ilya Sutskever. *Training recurrent neural networks*. University of Toronto Toronto, Ontario, Canada, 2013.

[Voy+17] Cyril Voyant et al. “Machine learning methods for solar radiation forecasting: A review”. In: *Renewable Energy* 105 (2017), pp. 569–582.

[Wer74] Paul Werbos. “Beyond Regression:" New Tools for Prediction and Analysis in the Behavioral Sciences”. In: *Ph. D. dissertation, Harvard University* (1974).

[WOK17] Wei Lee Woon, Stefan Oehmcke, and Oliver Kramer. “Spatio-Temporal Wind Power Prediction Using Recurrent Neural Networks”. In: *International Conference on Neural Information Processing*. Springer. 2017, pp. 556–563.

[Wol17] Björn Wolff. “Support Vector Regression for Solar Power Prediction”. PhD thesis. Carl von Ossietzky Universität, 2017.

[Xia+17] WenBo Xiao et al. “A neural network based computational model to predict the output power of different types of photovoltaic cells”. In: *PloS one* 12.9 (2017), e0184561.

[YC14] Amit Kumar Yadav and SS Chandel. “Solar radiation prediction using Artificial Neural Network techniques: A review”. In: *Renewable and sustainable energy reviews* 33 (2014), pp. 772–781.

[Zia+17] Ali Ziat et al. “Spatio-Temporal Neural Networks for Space-Time Series Forecasting and Relations Discovery”. In: *Data Mining (ICDM), 2017 IEEE International Conference on.* IEEE. 2017, pp. 705–714.

A Appendix of Software

A.1 Software requirements

- Programming Language:
  - Python 3.6
- Dependencies:
  - Tensorflow 1.7.0
  - Keras 2.1.5
  - Numpy 1.14.2
  - Pandas 0.24.2
  - Scikit-learn 0.20.3
  - Scipy 1.0.0
- optional Packages:
  - Matplotlib 2.2.0
  - Basemap 1.2.0
  - Geos 3.6.2
  - Statsmodels 0.9.0
  - h5py 2.9.0

B Appendix of Experiments

B.1 Autocorrelation of whole time series

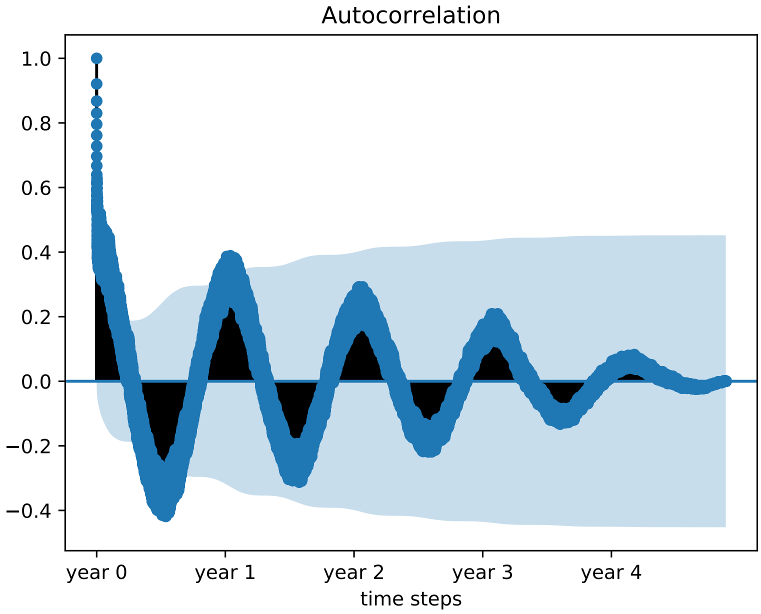


Figure B.1: Autocorrelation of whole time series of a targeted PV plant with ID 23526. A wave pattern indicates an autoregressive term in the time series. The order of that autoregressive term can be found out with the partial autocorrelation function.

B.2 Partial Autocorrelation

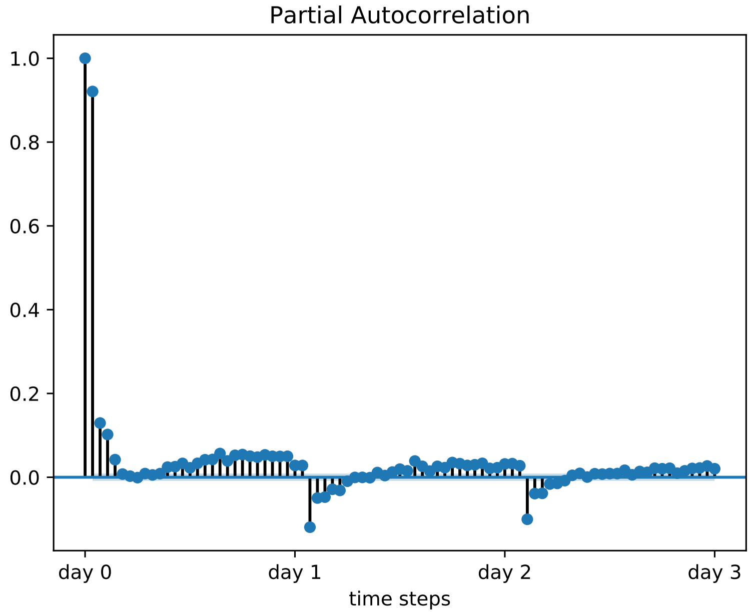


Figure B.2: Partial autocorrelation of a targeted PV plant with ID 23526. The partial autocorrelation function reveals the dependencies between a time step and a lagged time step by neglecting the influence caused by time steps in between.

B.3 RMSEs and Skill Scores per sample of experiment 2-A

Table B.1: Overall outcomes of experiment 2-A. Shown are the RMSE values in percentage of installed capacity and the skill score for each sample. One sample corresponds to one row. The distance of each number of surrounding systems N is a radius of $r = 10$ km.

RMSE						**Skill Score [%]**					
$N = 0$	$N = 4$	$N = 6$	$N = 8$	$N = 10$	$N = 12$	$N = 0$	$N = 4$	$N = 6$	$N = 8$	$N = 10$	$N = 12$
2.78	3.27	3.36	3.31	3.38	3.36	-0.75	-13.66	-16.86	-15.07	-17.47	-16.57
2.83	2.66	2.99	2.64	2.75	3.22	0.77	9.60	8.27	18.96	15.57	1.29
2.35	2.72	2.96	3.09	2.39	2.71	8.32	12.50	4.92	0.71	11.79	15.78
2.78	2.14	2.08	2.71	2.91	2.42	7.28	8.05	10.43	-16.63	-25.06	-3.91
1.08	2.15	1.60	1.60	2.38	9.93	40.26	18.87	15.36	15.71	7.36	-327.35
1.30	2.94	2.72	2.59	2.57	2.67	32.24	-18.14	-9.38	-3.88	-3.26	-7.10
2.01	2.10	2.10	2.19	2.90	2.94	9.43	5.17	4.62	14.69	9.80	8.60
2.61	2.82	2.97	2.76	2.76	1.44	6.49	1.26	-3.97	3.45	3.35	26.81
1.90	2.58	2.67	2.61	2.68	2.74	24.41	7.81	6.36	8.58	5.94	3.81
2.44	2.81	4.32	3.39	3.10	3.17	7.33	-6.76	-69.98	-17.81	-8.01	-10.33

B.4 RMSEs and Skill Scores per sample of experiment 2-B

Table B.2: Overall outcomes of experiment 2-B. Shown are the RMSE values in percentage of installed capacity and the skill score for each sample. One sample corresponds to one row. The distance of each number of surrounding systems N is listed to a corresponding radius r that is denoted in kilometer.

r	RMSE					Skill Score [%]				
	$N = 4$	$N = 6$	$N = 8$	$N = 10$	$N = 12$	$N = 4$	$N = 6$	$N = 8$	$N = 10$	$N = 12$
5 km	2.13	2.86	1.41	2.13	3.37	8.59	-22.72	28.65	14.23	-13.98
	2.50	2.16	2.48	5.39	7.68	3.96	11.20	-24.23	-82.07	-317.05
	3.13	3.36	3.07	3.99	3.43	-3.47	-10.32	-5.83	-34.60	-11.40
	2.59	2.32	2.90	2.65	2.81	8.73	4.86	-29.38	-8.89	2.03
	3.15	3.41	3.36	2.86	3.59	-10.14	-19.33	-17.30	2.02	-24.36
	2.66	2.03	3.16	2.23	2.88	-1.06	12.01	-12.56	1.28	-0.66
	2.55	2.18	2.06	2.77	2.85	3.12	8.34	-2.80	3.08	-4.61
	3.16	2.72	1.76	4.31	2.18	-22.40	-5.36	8.99	-44.54	3.63
	1.90	3.02	2.16	2.28	2.79	-11.21	2.00	17.09	8.30	-1.57
	3.30	2.53	3.11	4.28	3.62	-0.87	6.93	-5.55	-53.01	13.60
10 km	2.04	2.07	2.83	2.95	2.49	12.29	11.18	-2.54	-26.98	-3.36
	2.71	2.54	2.35	2.20	3.23	2.02	8.21	12.84	17.63	2.69
	2.39	2.42	2.61	2.53	4.01	11.95	10.91	-0.07	-18.31	4.01
	2.59	1.51	2.85	7.40	7.10	8.73	22.76	-4.76	-150.4	93.84
	3.02	3.22	3.33	3.60	3.85	-5.99	-12.98	-12.03	-26.23	-37.85
	3.09	2.14	0.93	1.81	2.63	5.30	5.55	3.92	20.73	8.05
	2.74	2.88	2.58	3.69	4.20	-8.38	8.20	12.44	-28.58	-80.09
	2.28	2.48	2.64	2.73	3.21	12.03	7.51	5.21	10.40	4.45
	2.46	2.98	2.82	2.87	3.01	11.97	-1.65	2.99	7.18	-1.75
	3.23	1.76	2.28	2.39	2.70	-1.02	16.47	5.87	8.89	-0.01
15 km	2.07	2.02	2.00	2.00	1.99	11.13	13.16	13.96	13.99	14.37
	2.84	2.97	2.89	2.72	2.66	-1.96	-6.60	-3.95	2.24	4.39
	2.50	2.40	2.46	2.48	2.46	7.72	11.39	9.48	8.39	9.48
	2.55	2.50	2.49	1.60	1.52	10.23	12.19	12.52	18.32	22.35
	3.17	3.28	3.35	3.40	3.34	-10.84	-14.63	-17.18	-18.96	-16.83
	2.48	2.05	2.46	2.16	3.02	2.76	15.52	14.29	12.34	64.2
	1.87	2.25	1.89	2.72	2.21	9.18	7.55	15.76	5.20	-0.57
	2.65	2.49	2.33	2.10	2.57	3.77	7.12	6.03	14.33	6.17
	2.76	2.24	4.81	2.68	2.59	-6.87	13.47	-62.45	5.09	12.45
	2.22	2.48	2.53	0.88	3.27	14.9	7.29	9.75	10.39	-11.09
20 km	2.09	2.09	2.07	2.02	2.03	10.30	10.32	10.97	13.18	12.81
	2.48	2.48	2.52	2.64	2.51	10.46	10.97	9.48	5.18	9.84
	2.51	2.45	2.45	2.36	2.39	7.66	9.55	9.80	13.17	11.82
	2.59	2.59	2.58	2.59	2.58	8.80	8.72	9.32	8.98	9.23
	3.35	3.26	2.07	2.02	2.03	-17.37	-14.06	10.97	13.18	12.81
	3.15	2.47	2.37	3.07	2.05	-0.45	15.66	11.10	-30.62	8.46
	2.28	2.51	2.74	2.40	2.33	11.68	7.03	11.76	-2.09	7.80
	2.29	2.33	2.74	2.60	2.47	12.04	37.46	11.39	6.73	6.14
	2.79	2.12	2.68	2.30	2.53	-7.94	10.67	-14.45	13.92	2.71
	2.60	2.00	2.82	2.21	3.59	5.36	3.52	11.14	16.56	-21.24

B.5 Excluded reults from comparison of LSTM with SVR

Table B.3: RMSEs and skill scores of PV system with ID 17738. Those results have been excluded in the analysis in Section 4.6.2, where machine learning methods for solar power prediction are being compared. The exclusion had been done due to most deviate LSTM outcomes. The RMSEs are in percent of installed capacity. P is the abbreviation of persistence forecast. The 'over' in the column names of the skill scores indicates the reference.

Time	**RMSE**			**Skill Score [%]**		
horizon	P	SVR	LSTM	SVR over P	LSTM over P	LSTM over SVR
$t+1$	9.25	5.56	4224.06	39.96	-45535.79	-75914.63
$t+4$	11.02	7.98	34725628.51	27.59	-315027267.13	-435036162.33

STATUTORY DECLARATION

Hereby I declare that I have written this work independently and have not used any other sources and resources than those mentioned. Furthermore, I assure that I have followed the general principles of scientific work and publication as laid down in the guidelines of good scientific practice of the Carl von Ossietzky University Oldenburg.

German version:

Eidestattliche Erklärung

Hiermit versichere ich, dass ich diese Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Außerdem versichere ich, dass ich die allgemeinen Prinzipien wissenschaftlicher Arbeit und Veröffentlichung, wie sie in den Leitlinien guter wissenschaftlicher Praxis der Carl von Ossietzky Universität Oldenburg festgelegt sind, befolgt habe.

September 10, 2019 Theresa Schöbel