Using Machine Learning to Predict Earth Deformation from InSAR Time Series

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

IGP
Institut für Geodäsie und
Photogrammetrie

Using Machine Learning to Predict Earth Deformation from InSAR Time Series

Bachelor's Thesis

Christoph Baumann

Space Geodesy Group
Institute of Geodesy and Photogrammetry
ETH Zurich

Supervisors
Prof. Dr. Benedikt Soja
Mostafa Kiani Shahvandi
Alexandra Duckstein

Zurich, June 10, 2022

Abstract

Earth deformation can occur as a result of numerous causes and has the potential of far-reaching effects on society, infrastructure, and our natural surroundings. Forecasting its occurrence is crucial for deciding on preventive measures and increasing our understanding of the phenomenon. Using machine learning, or more specifically multilayer perceptrons, to predict earth deformation based on Interferometric Synthetic Aperture Radar (InSAR) time series, could be a promising approach to facilitate these efforts. On a data set focused on a former mining area around the German city of Saarbrücken, the multilayer perceptron, consisting of eight layers, denotes an average improvement of 41% for 12-day and 80% for 60-day predictions compared to a baseline approach using quadratic regression. In addition to the recorded improvements, findings suggest a higher noise resistance and flexibility level than the used baseline approach. While the results are promising, the approach does not tap its full potential, leaving room for further research and improvement.

Contents

List of Figures

List of Tables

1 Introduction

Earth deformation poses an ever present hazard to humans and their environment. Causes are numerous: volcanic activities, tectonic movements, post-glacial rebound, or changes in groundwater levels. Whilst some of them are results of natural processes, others are of anthropogenic nature. As part of hazard management in urban planning and risk management for already existing infrastructure, monitoring and predicting earth deformation is a crucial step towards building safer surroundings. Not only that, it also provides vital information to strengthen and further increase our understanding of the planet and the ecosystem that we live in.

Over the course of the past decades, technological advancements made it possible to measure earth deformation from space using radar technology [1], namely using InSAR. With the launch of the European Space Agency's Sentinel 1A satellite in the spring of 2014 ([2]), monitoring data of the earths surface became available in near real time frequency with a high spatial resolution, allowing for millimeter scale monitoring even in the remotest of areas [3]. This fact provides InSAR with an immense advantage in terms of costs and efficiency compared to traditional geodetic surveying techniques [4]. In the past, research efforts were largely focused on improving the processing method [5] of InSAR data. Expert knowledge and traditional methods are thus largely still required to predict earth deformation based such data. Using machine learning approaches poses an attractive alternative to the already established methods. It is a relatively new field and yet advances are rapid [6]. Research efforts are focused on various types of neural networks, such as: convolutional neural networks (CNN), multi-layer perceptrons (MLP), or recurrent neural networks (RNN) [3], [7]–[10]. This thesis aims to construct a neural network using MLP for the purpose of predicting earth deformation based on InSAR data and assess its applicability for this exact purpose.

The data set, focused around the German town of Saarbrücken, was processed and provided by the Karlsruhe Institute of Technology (KIT) in Germany, as part of a larger data set containing the upper Rhine Graben area, spanning numerous western and central European countries, such as Germany, France, and Switzerland. Over the course of the past 16 years, old mining shafts were continuously flooded, resulting in earth deformations. These deformation trends were recognized in the estimated mean line-of-sight velocities at KIT, thus prompting the interest in predicting future behavior. In this work MLPs are applied as an approach to predict said future behavior specifically for area centered around Saarbrücken, exploring their efficiency and therefore suitability for this application.

This thesis is structured to first present a brief overview of the InSAR technology. This part is followed by a more elaborate insight into the data set, its structure, and the challenges that are thereby imposed on the model. Subsequently, the exact approach and heuristics used are presented in chapter 3. Chapter 4 deals with the model that was developed and its predictions. Followed by a discussion of the results presented in the previous chapter and conclusions that can be drawn from the case at hand.

2 Data Set

2.1 Interferometric Synthetic Aperture Radar

InSAR systems use electromagnetic wave signals to determine the range to a back-scattering object. Said signals consist of two observable parameter: amplitude and phase. Whilst the amplitude largely depends on the distance and back-scattering properties of the object such as surface roughness, the phase changes continuously as the wave travels through the air. A full waveform corresponds to a change in phase from $-\pi$ to π. The difference in phase is linked to the distance to the object via the following equation:

$$2d = \frac{\varphi}{2\pi} \cdot \lambda + N \cdot \lambda + \varepsilon \tag{2.1}$$

where

- φ Phase
- d Range between radar and point on the ground
- λ Wavelength
- N Number of full wave-forms passed
- ε Noise.

The phase difference between two InSAR acquisitions (illustrated in figure 1), usually referred to as master and slave acquisition, can be determined very accurately [12], yielding in cm to mm accuracies for line-of-sight (LOS) displacements measured in the L-, C-, and/or X-band [13].

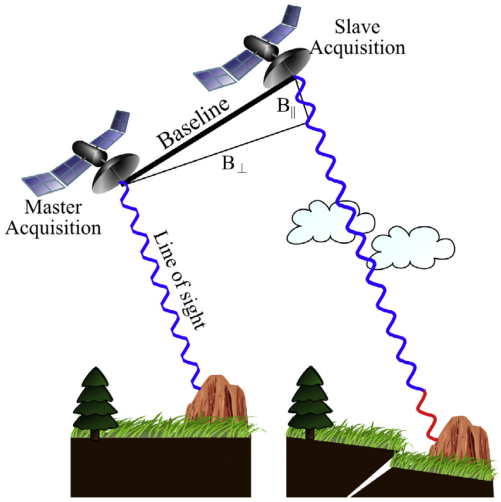

Figure 1: Principle master-slave acquisition in deformation analysis [11]

Attribute	Shape	Unit	Description
lat	218637x1	Degrees east	Latitudes of all points
lon	218637x1	Degrees north	Longitudes of all points
disp	218637x112	Millimeters	Line-of-sight displacements

Table 1: Detailed view of data attributes given in the data set

2.2 Description of the Data Set

The data set at hand was provided by the Karlsruhe Institute of Technology (KIT) in Germany. The radar images were taken by the European Space Agency's (ESA) Sentinel 1A radar satellite, which carries a C-band SAR system [2]. Its repeat cycle spans a time period of 12 days, totaling 113 scenes from November 27th, 2015 to February 22nd, 2020. The images were taken on the descending orbit 139 [14]. The data was processed at KIT using the free software package SNAP (Sentinel Application Platfrom, ESA) to form the interferograms. The Stanford Method for Persistent Scatterers [15] was used to detect permanent scatterers (PS) and the Toolbox for Reducing Atmospheric InSAR Noise [16] to reduce atmospheric effects. A detailed description of how the data was processed can be found in the corresponding publication by Mazroob Semnan et al. [17]. The resulting data thus contains corrected LOS displacement for 218,637 detected PS.

The data set used in this thesis is a subset of what is shown in figure 2 with focus on the Saarland area. Said region is notably visible in figure 2 with its high mean velocity in line-of-sight direction just to the east of the French city of Metz. Said subset contains 113 scenes for each of the 218,637 PS. These 113 scenes result in 112 line-of-sight displacements as seen in table 1. This is due to the fact that one scene is used as the master day. All displacement values are relative to what was

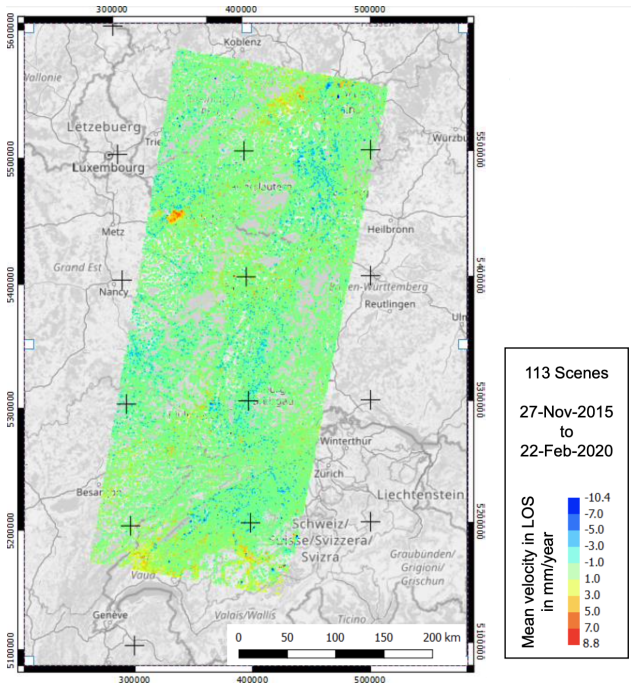

Figure 2: Mean LOS velocities for orbit dsc139 of the Sentinel 1A satellite from November 27, 2015 to February 22, 2020 [14]

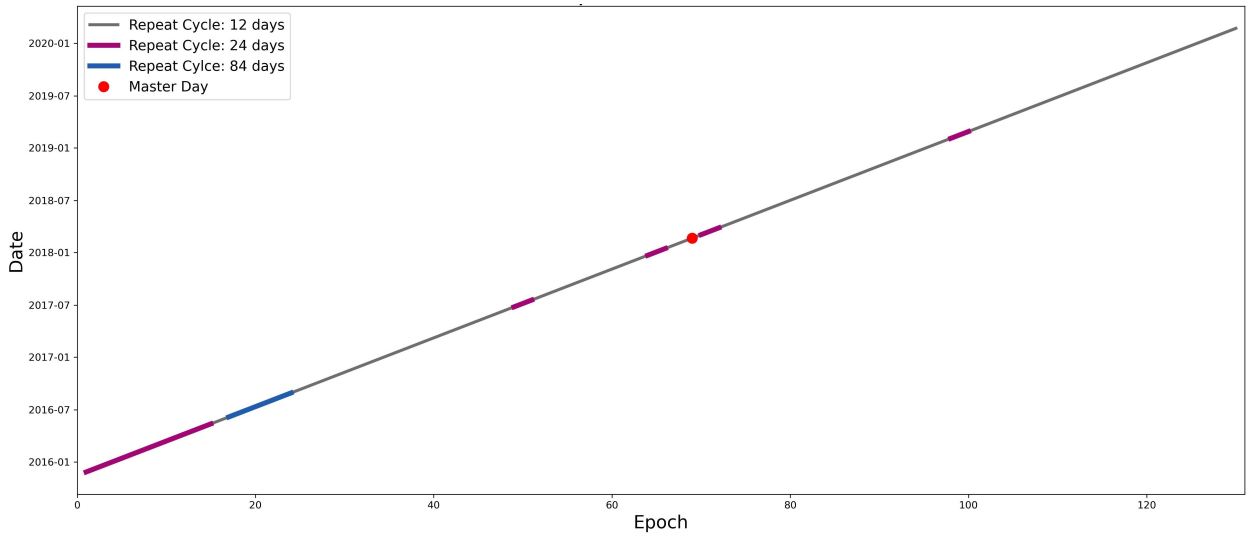

Figure 3: Visual in which period of time data was acquired and what the repeat cycle was at that time

measured for the master scene. Though, it could technically be chosen arbitrarily, the choice of master scene fell on the 69th epoch, which corresponds to an approximate center position for all epochs. Figure 3 shows the course of all acquisitions over time. Notably, there is a period of 84 days in the second half of 2016 where no data was acquired. Apart from that, the data was largely acquired with a repeat cycle of 12 days. There could be a number of reasons for the occasional change in the repeat cycles, such as: the data could not be read by SNAP, problems during the coregistration process, the orbit data was not available, or the ERA5 data was not available.

2.3 Focus Region

As mentioned in section 2.2, the data set used in this thesis consists of a subset of the data shown in figure 2, focusing on the area around the German city of Saarbrücken, covering territory in both Germany and France, called the Lorraine-Saar basin. The area was famously known to be a hotspot for coal mining, with mining dating back to the nineteenth century. On the French side, mining works were stopped in 2004. By 2006, the process of flooding the mines was initiated. Mining shafts were progressively flooded, creating mine water reservoirs [18].

Comparing figures 4 and 5 using the border between France and Germany for alignment, it is clearly visible that the locations of the flooded mines match with parts of the extreme mean line-of-sight velocities observed around latitude $49°10'N$. Groundwater influx and outflow are known causes for earth deformations, due to changing soil moisture, resulting in a shrinkage or expansion of the grounds volume. In the case of flooding old mining shafts, deformations can be amplified by potentially collapsing tunnels.

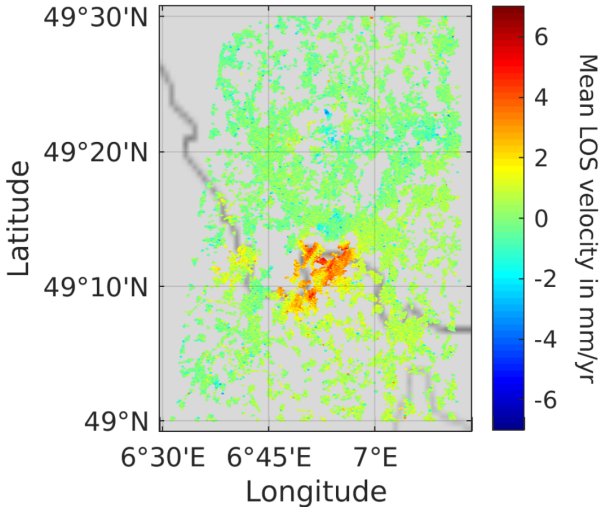

Figure 4: Mean line-of-sight (LOS) velocity for the data set at hand [14]

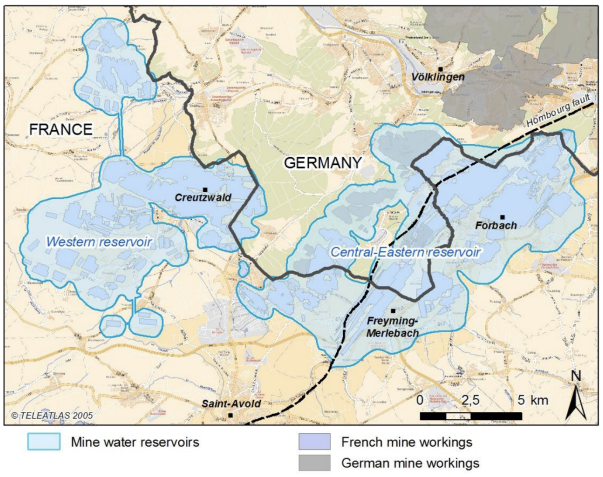

Figure 5: Mining in the Lorraine-Saar basin [18]

3 Methodology

In this chapter the methodology applied to the prediction problem is discussed in further detail.

3.1 Multilayer Perceptron

Neural networks are a highly effective approach in forecasting [19]. A type of neural network is the feed-forward neural network, also known as MLP, where its name is actually a misnomer, since multilayer perceptrons consist of multiple layers containing perceptrons and not, as the name suggests, of perceptrons made up of multiple layers. The basic structure of MPLs can be seen as a series of functional transformation of M linear combinations:

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)} \tag{3.1}$$

where $j = 1, \ldots, M$. D denotes the total number of input variables. Parameters w_{ji} describe weights whereas w_{j0} describes the bias for the j-th linear combination. These parameters are adaptive and will be changed during the training process to minimize the prediction error, or more specifically the validation loss. The superscript (1) in equation 3.1 refers to the first layer of the MLP. Each quantity a_j, called activations, is then being transformed using an activation function, resulting in the transformed output quantity z_j:

$$z_j = h\left(a_j\right) \tag{3.2}$$

Finding the ideal activation function is difficult and often done by the means of trail and error processes. Activation functions $h(\cdot)$ can differ from layer to layer [21]. Important to note is that they have to be differentiable. Additionally, when adding multiple hidden layers, all but the last one should be chosen to be nonlinear. Commonly used for this purpose is the hyperbolic tangent function ($tanh$). Other popular activation functions include for example the Rectified Linear Unit ($ReLU$), which is limited to positive values [20]. This fact makes it unsuitable for this purpose, since deformations are expected to be positive and negative. Therefore, $tanh$ is used in all models. The

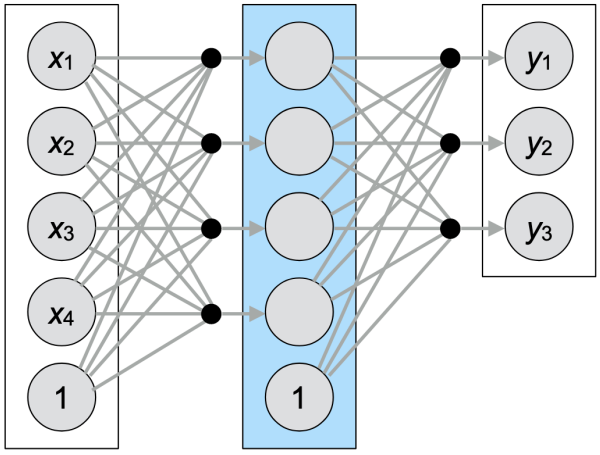

Figure 6: MLP with 4 inputs $x_1, \ldots, x_4$, 3 outputs $y_1, \ldots, y_3$, and a single hidden layer (blue) [20]

output layer uses a linear activation. The reason for this choice is that at this stage the weighted sums should not be changed anymore, since the output is only to be projected to the right length and no further adjusted are to be made [22].

In this work, all networks are implemented in Python using the TensorFlow API [23]. Based on the underlying assumption that the entire input influences its output, Keras Dense Layers are used in a sequential build. This provides full connectivity between two consecutive layers, allowing for all inputs to influence all linear combinations of subsequent layers, since each neuron of a previous layer is then connected to all neurons of said subsequent layer [24]. Figure 6 is an exemplary setup of an MLP with four inputs and three outputs using a single hidden layer. The arrows represent the contribution of each input to a neuron (black dots). For fully connected layers, each input has an arrow leading to all subsequent neurons, where they then are combined linearly (a_j in equation 3.1) and eventually transformed as a whole using the corresponding activation function, resulting in z_j as given in equation3.2. Adding the number one to each layer serves the purpose of incorporating biases. These bias features are multiplied with their respective weights, which are trained to optimize the bias' influence on the output.

3.2 Prediction Problem and Feature Structure

To use the MLP-based approach on the given data, it has to be uniformly spaced in time. For this purpose a linear interpolation is performed on the time domain for each individual time series in the data set. This underlies the assumption that a linear interpolation has the least skewing effect on the overall trend of each time series. Visually analyzing the raw data allows for the assumption that for a significant amount of time series linear behavior is in fact accurate, therefore least influencing the time series inherent course. Naturally, for time series where this assumption does not hold up, linear interpolation disturbs the underlying behavior. Though, the effects are likely to be minimal, since only short periods of time have to be interpolated.

3.2.1 Types of Prediction Problems

There are various types of prediction problems, each of them requiring a different feature structure, the simplest of them being the one-to-one prediction. For this type, a single observation is used to predict a single step ahead. This concept can also be applied with an arbitrary number of observations used to make an arbitrary number of predictions, which is the equivalent of the most general case: the many-to-many prediction. Common terms used for the various types of prediction problems are:

- one-to-one
- one-to-many
- many-to-one
- many-to-many

The many-to-many problem, being the most general one, is primarily pursued in this thesis. Allowing to use the maximum level of information to predict multiple steps into the future. The one of the most pressing aspects to consider is the exact number of observations going into the model. This number depends on two factors: the nature of the physical phenomenon to be modeled and the maximal length of a single sample to still result in a reasonable total number of samples per PS for the model to be trained on. The latter is most likely to be the limiting factor in this study. This is due to the fact that with a total number of 130 epochs, there is only a limited number of samples to be constructed to have reasonable sizes for the train, validation, and test set. The goal

is to have as many and as accurate predictions as possible. The exact setup will have to be tested empirically.

3.2.2 Feature Structure

The data set contains a total number of $218,637$ time series of length 130. Each of them is then split into smaller sequences. Assuming the sequences are split for a model which uses 3 inputs to predict 2 steps ahead, each time series is split as shown in figure 7, where x corresponds to the input feature vector and y to the output feature vector. The numbers in the boxes in figure 7 represent indices. Windows of sizes 2 and 3 are pushed along the time series cropping out samples of the respective lengths. This is done for all time series stacking their input and output features to form the final input and output vector. The total number of features is thus dependent on the prediction type. Due to the fact that the time series are rather short, the limiting factor for the prediction model is the number of samples per time series and not the total number of samples. Particularly challenging is the balancing act between having long input features for more accurate predictions and still having enough samples to properly train, validate, and test the model. This will be discussed in section 3.3.2 in more detail.

3.3 Training Process

As for any prediction problem, first the number of prediction steps has to be determined. Naturally, one is interested in the most accurate prediction. For long-term phenomena such as earth deformation, this requires to consider a larger number of empirical data to be fed into the prediction model. Choosing too high of an output length, would therefore result in a small number of samples per time series, hindering the model from learning a specific time series' typical behavior. Since a long-term phenomenon is investigated, the focus lies on prioritizing the input length over the output length. Therefore, a reasonable choice of output length is 5, which corresponds to 60 days.

Before the network can be trained, a few assumptions have to be made. Using MPLs, features cannot be linked to one another based on their locations and still be trained on all data points at once. Though, one could train for each point individually, due to the time series' short length, the data sample would not be large enough to result in a robust output. To account for this, all points are treated equally regardless of their geographic locations. This in turn means that a prediction

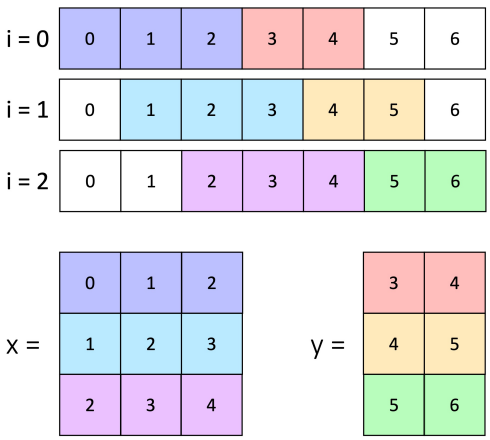

Figure 7: Splitting a time series into input and output vectors of length 3 and 2 respectively

made by the model is based on the average behavior shown by the time series in the data set. Naturally, this is not ideal, especially for regions where larger rates of change are expected. For those regions, the prediction is expected to not follow the observations to their maxima, seeming like a smoothed version of the observations.

3.3.1 Loss Function

The mean absolute error (MAE in 3.3) is used as loss function. The choice here is between a typical regression loss such as the MAE and a loss function that also considers the error relative to the observed changes such as the symmetric Mean Absolute Percentage Error (sMAPE in 3.4). Incorporating linkages to changes can be beneficial, since a predictions accuracy of 1mm sounds might sound convenient, if the actual overall trend is less than that, the prediction is meaningless. On the other hand, the sMAPE has the big shortcoming that it becomes unstable when both the observations and prediction are close to zero. Plus, the sMAPE will automatically hit its upper-bound whenever the input feature is equal to zero [25]. Conclusively, for this application, the disadvantages of the sMAPE outweigh its advantages and therefore leaving the MAE to be the measure of choice.

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n} \tag{3.3}$$

$$\text{sMAPE} = \frac{100\%}{n} \sum_{i=1}^{n} \frac{|y_i - x_i|}{(|y_i| + |x_i|)/2} \tag{3.4}$$

3.3.2 Training, Validation, and Testing

The given data totals 130 epochs after interpolation. 26 of them are left aside for independent testing, corresponding to 20% of all epochs. The other 104 time steps are used to create samples to be fed into the model. To determine the validation set, a built in feature from Keras' fit function is used, namely the validation split. This parameter is set to 0.1, meaning 10% of samples are used for validation. Samples are shuffled randomly to select a validation set, allowing to circumvent potential effects due to spatial correlations between points and over-fitting. Additional over-fitting protection is provided by the "EarlyStopping" callback function provided by Keras. This callback function monitors the validation loss. If it does not decrease for a set patience level, the training process is stopped and the best weights are restored. The mentioned patience level is set to 30 epochs for this application [24].

Due to the high number of samples, in the case of a 7-to-5 prediction the number of input samples totals around 21 million samples, other validation techniques, such as cross-validation, would be very time-consuming. As a rule of thump: cross-validation increases the computation time by a factor of 5 to 10 [20]. Therefore, cross-validation is not applied in the approach pursued in this thesis.

3.3.3 Hyperparameter Optimization

At the core of building a neural network lies its hyperparameter optimization. Various strategies can be applied. A well-performing setup is strongly dependent on the data set's unique properties and therefore has to be adapted in each individual case. For this thesis, a structured trial and error approach was designed to proceed effectively. To increase efficiency, a batch size of $65,536$ is chosen. This means that $65,536$ samples are run through the model before the parameters are changed.

This rather large number speeds up the training process and allows for a higher number of different network configurations to be tested. At the end of the hyperparameter-training phase the batch size is to be decreased as long as it increases the model performance, resulting in a decorrelation of the input samples.

In a first step the focus lies on the network architecture, meaning the input length, the number of layers, as well as the numbers of neurons per layer. Each setup is to be tested for different input lengths, trying to find the best performing construct. The starting point is chosen to be very simple. From there, the model complexity is increased continuously as long as it increases the model's performance. As it can be concluded from various publications ([26]–[28]), a promising guideline, which is also used in this thesis, is to focus on powers of two and multiples of 15 as numbers of neurons per layer. These layers are then added sequentially and in symmetrical order. E.g., a model is to be chosen using a three-layer setup including the following sizes: 4, 8, and 16 neurons. The network setup would then result to be: 4-8-16-8-4-5, with 5 being the size of the output layer. All of these layers use the hyperbolic tangent function as an activation function. Lastly, a layer with the same size as the output it added. This layer uses a linear activation function, since at this point the weighted sums should not be changed anymore.

After running each setup for various input lengths, the performance of each individual model is assessed using two separate methods. The first one is the minimum validation loss, which is to be minimized. The second influencing factor is the plotted learning curve. It shows the training and validation loss. Using a visual comparison, as shown in figures 8, its course is contrasted to exemplary plots of phenomena, such as over- and underfitting, that can typically occur during the process of optimizing hyperparameters. Combining the strengths of both methods gives ground to a good premise to select the best-performing model at each level of complexity. After concluding the network architecture optimizing process, the batch size is decreased to decorrelate the samples and optimize the performance.

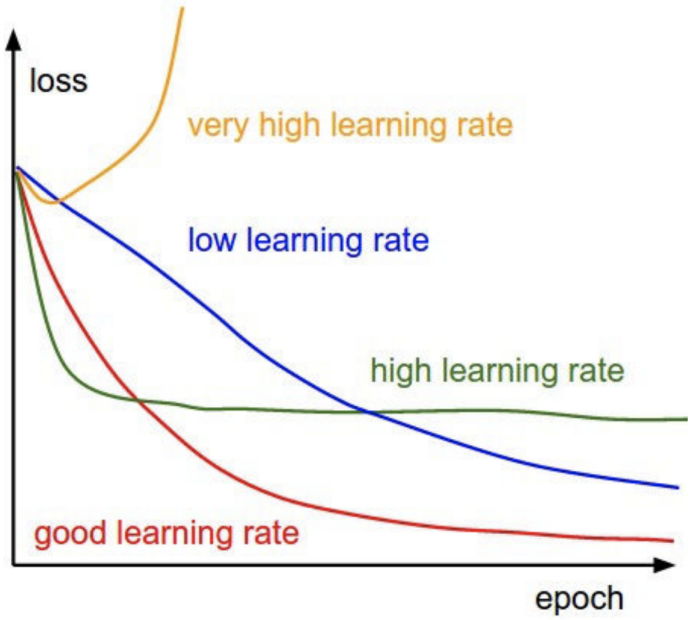

Figure 8: Examples of positive and negative learning behavior's results reflected in the learning curve [20]

The optimizer Adam used in all model setups. The Adam algorithm using stochastic gradient descent was first introduced by Diederik P. Kingma and Jimmy Ba. Their definition is the following: "a method for efficient stochastic optimization that only requires first-order gradients with little memory requirement" [29]. The stochastic gradient descent (SGD) allows working with large data sets and a high number of parameters, since SGD only uses mini-batches to estimate gradients and only first order derivatives of the parameters are to be computed. Both of these properties are inherently found in the data set at hand and to be expected respectively.

Lastly, the learning rate is adapted. Here again, the same concept as for finding a well-performing network architecture is applied. Starting from the default learning rate, it is changed in the increasing and decreasing direction as long as it improves the model's performance. As described before, the learning curve and minimum validation loss are assessed to decide, whether this process is repeated or the minimal validation loss for the configuration at hand is found.

4 Results

In the following, the results of constructing the neural network and the subsequent analyses are presented. Starting with the final model, then the model's performance based on various metrics is assessed and lastly the comparison to a quadratic regression model contrasts its performance to the baseline approach.

4.1 Model Selection

In the subsequent sections, the derivation of the final model is shown in a step-wise manner in accordance with the methods proposed in section 3.3.

4.1.1 Layer Setup

The final model consists of five hidden layers of sizes 45-60-75-60-45, with the numbers representing the number of neurons in each layer. Therefore, the numbers of neurons are multiples of 15. The guideline to pursue multiples of 15 and powers of two was dropped at a lower complexity level, after simpler models using multiples of 15 out-performed such using powers of 2 as illustrated in figure 9. The model consisting of hidden layers of sizes 16-32-64-32-16 with input length 8, totaling $5,493$ trainable parameters, is outperformed by the model using sizes 15-30-45-30-15. The latter model was tested for the smallest minimum validation loss at an input length of 10. As clearly visible in figure 9, at $3,965$ parameters it requires significantly fewer parameters than the model using powers of 2.

In the following, model complexities were solely increased using multiples of 15. Said network constellations are summarized in table 2. The table consists of the models shown in fig 9. Additionally, the last few models that were run, after the premise of using power was dropped down, are listed as well. For each setup only the, in terms of validation loss, best-performing input length out of the ones that were tested, is displayed in the table. The last three setups that were run are of layer sizes 8-30-45-60-45-30, 10-60-75-60-5, and the best-performing 8-45-60-75-60-45-5. Even though the

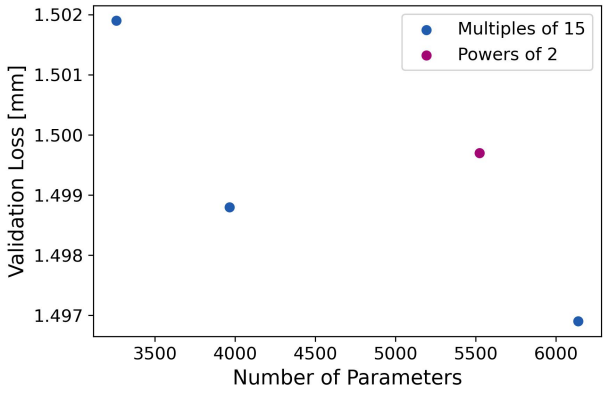

Figure 9: Comparison of models using powers of 2 versus multiples of 15. Layer setups from LTR: 10-30-45-30-5, 10-30-45-30-15-5, 8-16-32-16-8-5, 8-45-60-45-5. Detailed information on them is found in table 2.

Model Summary	Parameters	Min. Val. Loss [mm]	Epochs Trained
9-30-45-30-5	3,230	1.5019	435
10-15-30-45-30-15-5	3,965	1.4988	658
8-16-32-64-32-16-5	5,493	1.4997	393
8-45-60-45-5	6,140	1.4969	479
8-30-45-60-45-30	8,705	1.4937	440
10-60-75-60-5	10,100	1.4938	488
8-45-60-75-60-45-5	15,275	1.4935	328

Table 2: Detailed information on models run while optimizing the layer setup. Numbers under *Model Summary* list the number of neurons per layer, the first one being the input layer and the last one the output layer. All batch size are set to 65,536

.

number of parameters was almost doubled from the 8-30-45-60-45-30 model to the 8-45-60-75-60-45-5, the minimal validation exhibits only a slight decrease. The constellation of hidden layers of form 45-60-75-60-45 was the last one to be tested. It also represents the model with the highest level of complexity, containing eight layers and a total number of 15,275 parameters. To summarize the search for a well-performing layer setup, the best-performing model found uses an input length of 8 and hidden layers of sizes 45-60-75-60-45, achieving a validation loss 1.4935mm.

4.1.2 Batch Size

As described in section 3.3.3, the batch size is to be decreased as long reducing this hyperparameter decreases the validation loss. Following up on the previous section, the batch size for the model of topology 8-45-60-75-60-45-5 is reduced from 65,536 to 4,096. Each reduction step represents a division by 2. As shown in figure 10, a continuous decrease is recorded as the batch size is decreased. This behavior lasts up until batch size 8,192 where it shows a dip, before increasing again in validation loss for batch size 4,096. For this dip at batch size 8,192 the model's validation loss denotes 1.4890mm averaged over all five prediction steps.

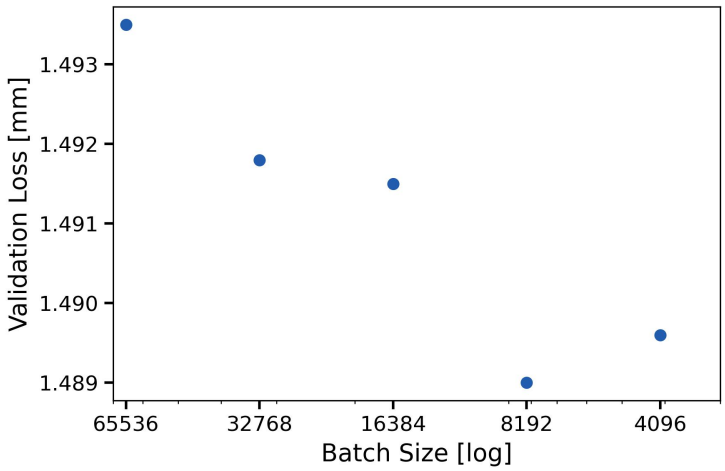

Figure 10: The minimal validation loss of the model with layer sizes 8-45-60-75-60-45-5 for different batch sizes

Learning Rate	Validation Loss [mm]	Optimizer
0.0005	1.4921	Adam
0.001	1.4890	Adam
0.0015	1.4914	Adam
0.002	1.4917	Adam

Table 3: Optimizing the learning rate for the model of topology 8-45-60-75-60-45-5

In addition to the lower validation loss, the model using a batch size of $8,192$ also shows a better learning behavior by the standards of figure 8. The learning curves for batch sizes $8,192$ and $4,096$ are shown in figure 11 and 12 respectively. The model with batch size $8,192$ displays a, by comparison, smooth downwards trend for both the validation and the training loss, eventually stabilizing at around 1.49mm for the training loss. The validation loss shows similar behavior as for its overall trend, but with rapid but small fluctuations, ultimately reaching a minimum of 1.4890mm. The model using a batch size of $4,096$ exhibits similar behavior for the training loss. For the validation loss on the other hand, the graph oscillates heavily, leading to larger discrepancies between the validation and training loss at certain epochs. The epochs affected by these larger discrepancies show no apparent pattern.

4.1.3 Learning Rate

Lastly, again continuing at the grounds of the results of the previous section, where the batch size was optimized, different learning rates are compared based on their respective validation losses. The results are depicted in table 3. As displayed there, the default learning rate of 0.001 achieves the lowest learning rate with 1.4890mm, remaining at the level of the in the previous section deemed lowest validation loss.

This step concluded the hyperparameter optimization. The resulting model is a prediction model of sort 8-in-5-out. That is to say, it takes an input of length 8 and makes five predictions, one for each step length. The final layer setup and hyperparameter setting are summarized in table 4.

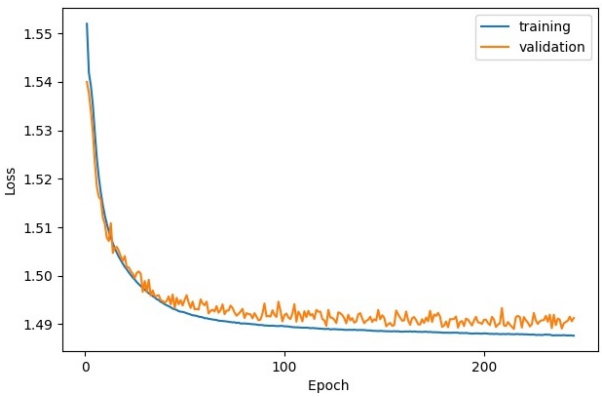

Figure 11: History of the training and validation loss for the 8-45-60-75-60-45-5 model with batch size $8,192$

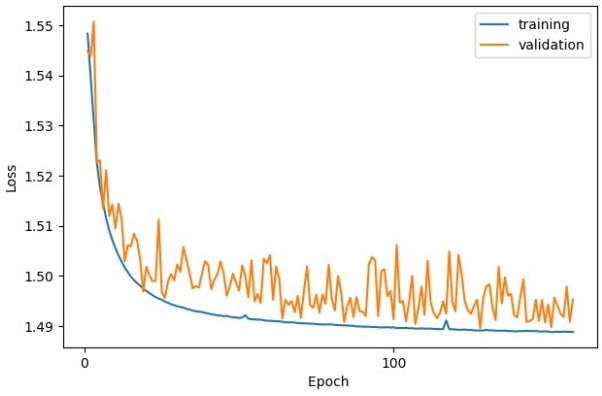

Figure 12: History of the training and validation loss for the 8-45-60-75-60-45-5 model with batch size $4,096$

Layer Setup	Input Length	Output Length	Optimizer	Learning Rate	Batch Size
8-45-60-75-60-45-5	8	5	Adam	0.001	8,192

Table 4: Layer setup and hyperparameter setting of the selected model

Step Length	Set	Mean [mm]	Median [mm]	σ [$\sqrt{mm}$]	Min. [mm]	Max [mm]
1	Entire	1.3081	1.2553	0.4945	0.0493	5.3525
5	Entire	1.6053	1.5324	0.6063	0.0898	5.7272
1	Test	1.4864	1.3231	0.7691	0.0453	12.5778
5	Test	1.7789	1.5598	0.9721	0.0454	12.4636

Table 5: Statistics on the distribution of MAEs for various step lengths. *Entire* stands for predictions on all time series in the entire data set using all their full lengths, *Test* for only the test set

4.2 Deformation Predictions

In the following sections, the selected model's (as summarized in table 4) performance is assessed by the means of various metrics. First these metrics are applied globally, to examine the overall quality of the predictions. Subsequently, individual time series are assess to highlight certain properties and characteristics.

4.2.1 Mean Absolute Error

The histograms in figures 13 and 14 depict histograms of the MAEs for step lengths 1 and 5 for predictions that were made based on the entire data set. The MAEs are computed for each time series and step length. Evaluating the histogram for step length 1, it displays a right-tailed behavior with a mean value of 1.3091mm and a standard deviation (σ) of $0.4945\sqrt{mm}$. Detailed statistics on the histograms found in figures 13 and 14 are listed in table 5. Comparing the two step length visually, a more distinct right-tailed behavior for the higher step length becomes apparent, which is reflected in its standard deviation, found in table 5, as an increase of $0.1118\sqrt{mm}$. Additionally, the mean value has increased to 1.6053mm. As a consequence the MAEs of more than 2mm have gained significantly in frequency.

Contrasting the results illustrated in the previous paragraph to the unseen data of the test set a notable drop in performance is recorded. As stated in table 5, both step length show similar increases for their respective mean values of approximately 0.17mm. The MAEs of predictions

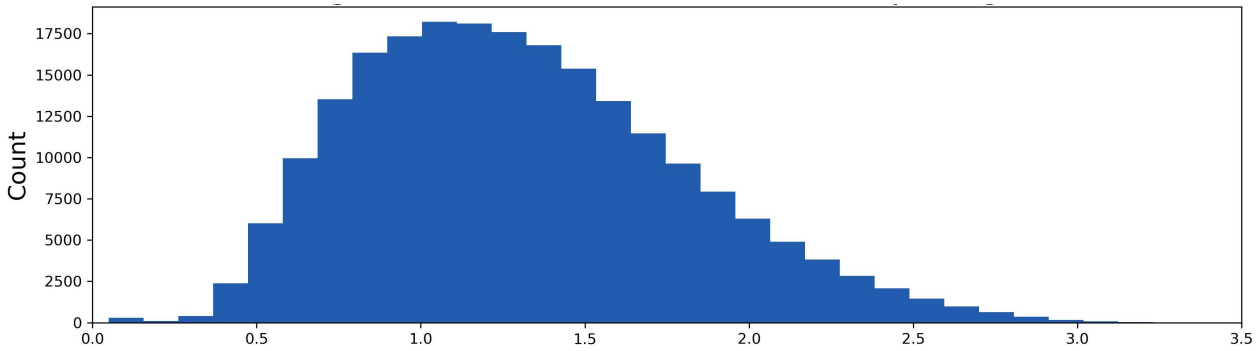

Figure 13: Distribution of MAEs for predictions of step length 1 on all time series in the data set

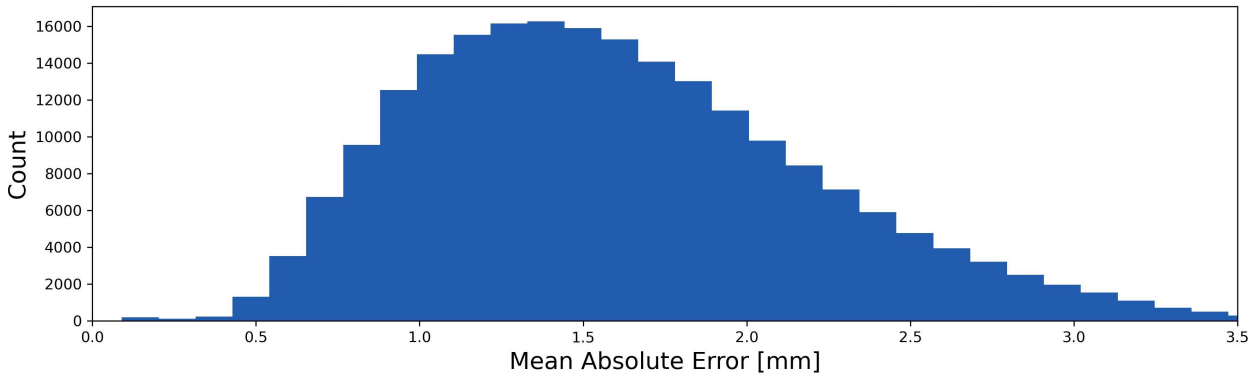

Figure 14: Distribution of MAEs for predictions of step length 5 on all time series in the data set

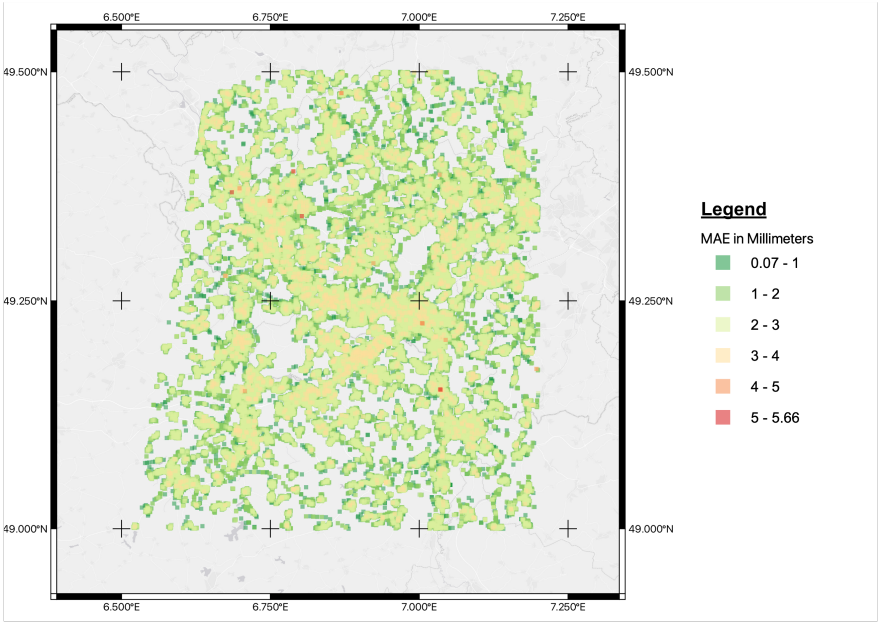

Figure 15: Spatial distribution of MAEs for the length 1 on all times series in the data set

on the test set also show increases in terms of their standard deviations. They total to $\sigma_{1,test} = 0.7691$mm for a step length of 1 on the test set and $\sigma_{5,test} = 0.9721$mm for a step length of 5. Comparing the increases, both measures experienced, compared to their counter-parts on the entire data set, increases of $\Delta\sigma_1 = 0.2747\sqrt{mm}$ for step length 1 and $\Delta\sigma_5 = 0.3658\sqrt{mm}$ for step length 5 respectively.

However, the median values for settings shown in table 5 is lower than the mean value. For the predictions made based on the test set the discrepancies are larger, being maximal for step length 5 with a difference of 0.2191mm.

The spatial distribution of the MAEs shown in figure 13 is illustrated in figure 15. The MAEs are split into 6 classes as shown in the legend to the right side of the map. The data points are visualized based on their MAE in an increasing fashion, therefore visually focusing on higher valued MAEs. Outliers, meaning classes 5 and 6 with MAEs of more than 4mm, are spatially clearly separated from each other. Values of class 5 $(3 - 4$mm$)$ are recorded primarily, but not exclusively, for build environments such as Saarbrücken $(49.23°N\ 7.01°E)$ or Saarlouis $(49.32°N\ 7.50°E)$.

4.2.2 Residuals

Another measure to asses the models performance and validity is the distribution of residuals. Figure 16 shows the distributions for step lengths 1 and 5 for all time series using their full number of time steps. The mean value for step length 1 is marginally larger with a value of 0.0068mm than the mean value for the longer step length of 5, which corresponds to 0.0060. Step length 1 shows a standard deviation of $2.1308\sqrt{mm}$ and step length 5 of $2.2887\sqrt{mm}$.

4.2.3 Symmetric Mean Absolute Percentage Error

First introduced in equation 3.4 in section 3.3.1, the sMAPE measures the symmetric mean absolute percentage error. Compared to the MAE it also incorporates changes between time steps. Its symmetry around zero expands its range from 0 to 200, compared to the 0 to 100 for the asymmetric version. For predictions of step length 1 on the the entire length of all time series in the data set, the sMAPE represents 103.5% and for the maximal step length of 5: 128.5%. The distribution of the sMAPE values are displayed in figures 17 and 18. Comparing their distributions to the distributions of the MAEs in figures 13 and 14 oppositional behavior is observed.

4.2.4 Smoothness

Predictions only add information if the predicted change exceeds the noise level, otherwise the noise level and not the actual trend is predicted. For the data set at hand, the noise level is unknown. Nevertheless, smoothness is computed against a linear form to assess whether a time series shows rapid changes between epochs possibly later impacting the predictions. Higher orders are not pursued, due to the unknown noise level being amplified for higher orders. The distribution of smoothness in terms of absolute values of second order derivatives is illustrated in figure 19. A value of 0 is the equivalent of perfectly linear time series. A lower value for this measure therefor means smoother behavior for this context. Evaluating the distribution in figure 19 suggests that a very large majority of the given time series deviates strongly from linear behavior.

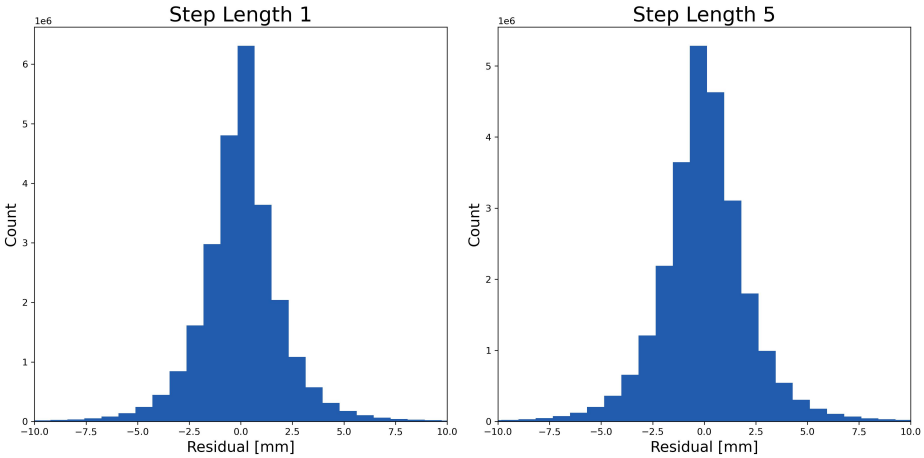

Figure 16: Distributions of residuals for 1- and 5-step predictions on the entire data set

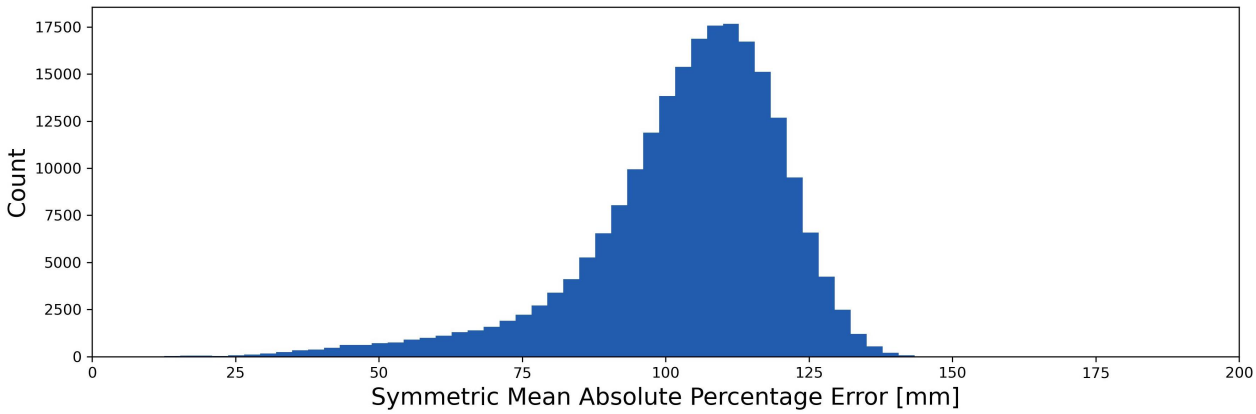

Figure 17: Distribution of the sMAPE for 1-step predictions on the entire data set

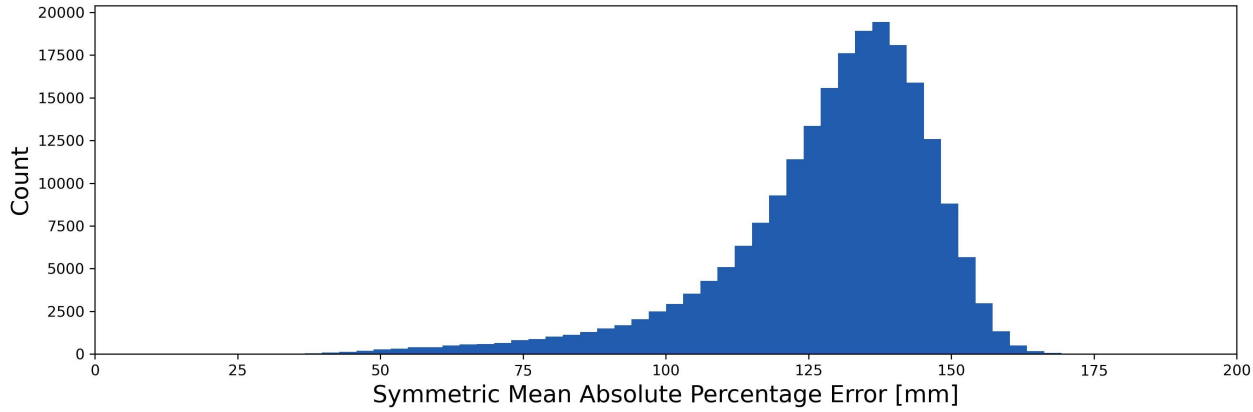

Figure 18: Distribution of the sMAPE for 5-step predictions on the entire data set

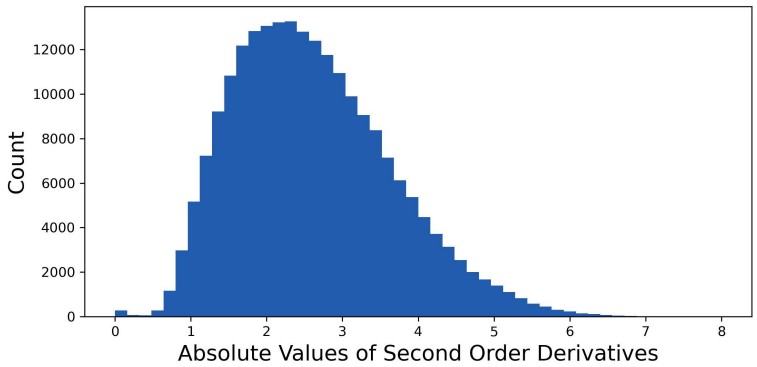

Figure 19: Distribution of second order derivatives for all time series to indicate smoothness

Index	MAE [mm]	MAE Test [mm]	sMAPE [%]	Smoothness
51, 179	5.3525	6.6655	143.31	12.4191
94, 809	0.0556	0.0598	16.80	0.0498

Table 6: Statistics on TS51179 and TS94809 for 1-step predictions

4.2.5 Individual Time Series

Lastly, after focusing on the overall performance in the previous parts of this chapter, individual time series are visualized and assessed. For illustration purposes two time series are selected. One where, based on the MAE, the quality of the prediction is high compared to the others (TS94809) and one where it is comparably low (TS51179). An overview over the selected time series' performance for 1-step predictions is given in table 6, summarizing different measures: the MAE for the entire data set, the MAE for the test set, sMAPE for the entire data set, and smoothness as discussed in section 4.2.4 is given in table.

TS51179 is visualized in figures 20 and 21, where the vertical red line represents the split between the training and test set. The measured time series oscillates rapidly, as its smoothness indicator in table 6 suggested. Both the 1-step-predictions and 5-step-predictions resemble a smoothened version of the observed time series. This effect appears more distinctive for the longer step length. Another distinct feature, appearing in both time series but being more prominent in TS51179's 1-step-prediction, is the at times delayed prediction minima and/or maxima. This is clearly visible in said visualization with the maximum measured at approximately epoch 60, but by the model predicted for approximately one epoch later. A similar behavior is shown around epoch 102 for a local minimum. When following the courses of the measured time series and the predicted values, many more of these artifacts stand out.

The second time series that is investigated more thoroughly in this section is TS94809 visualized in figures 22 and 23. As suggested by its smoothness value, it indeed follows a smoother pattern, even though its overall trend is nonlinear. The in the previous paragraph discussed delays also occur for this time series. They are most prominent around epochs 105 to 125 for the one-step-prediction. Furthermore, spikes are observed in the predicted time series, e.g. around epoch 30 for the 1-step-prediction. These spikes appear to follow decreases or increases in the input data to which the model overreacts by predicting to much change. A last very distinctive feature is observed for the 5-step-prediction. Even though the prediction quality is lower compared to the one-step-prediction, as also observed when assessing the overall performance, a large portion of the MAE appears to originate from a small number of epochs. This is reflected in the comparison of the MAE to the Median Absolute Error, which is 0.09741mm to 0.0742mm in favor of the Median Absolute Error, confirming what was previously observed visually.

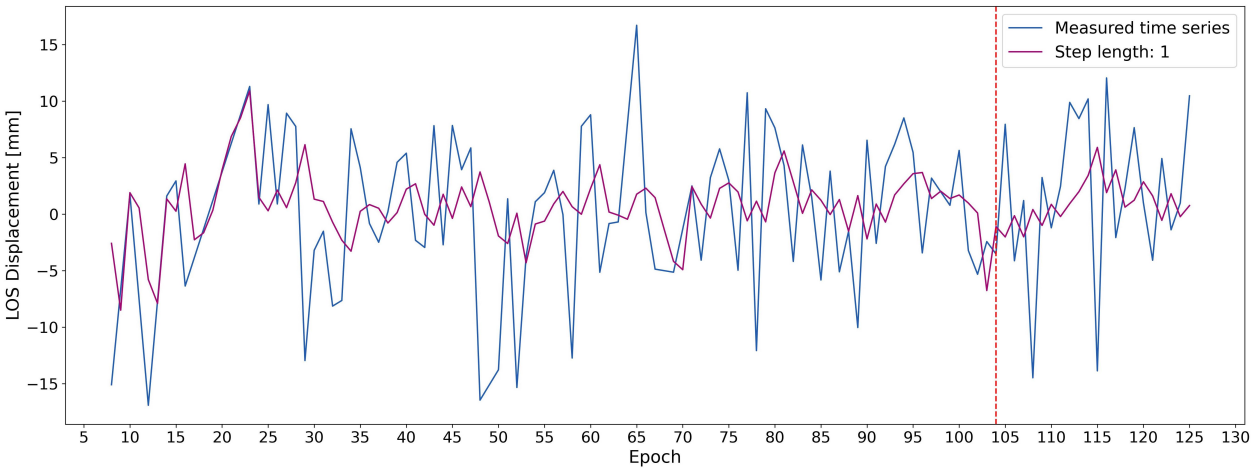

Figure 20: TS51179: Measured time series vs. 1-step prediction

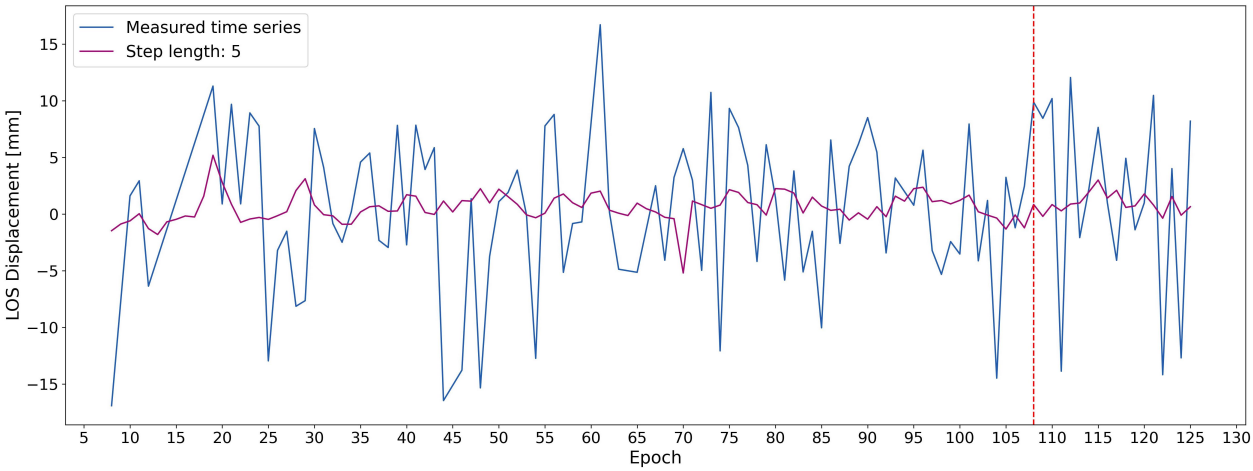

Figure 21: TS51179: Measured time series vs. 5-step prediction

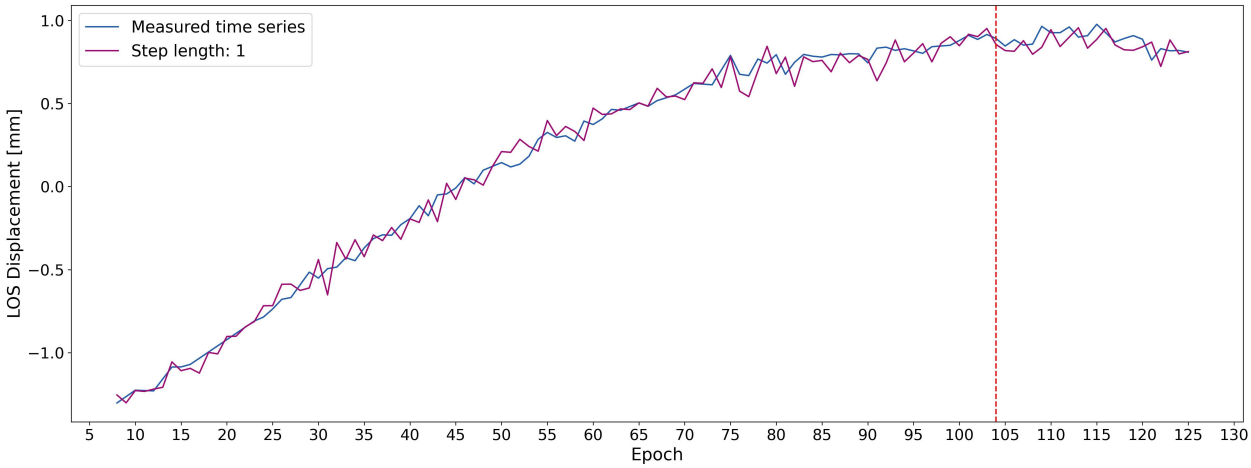

Figure 22: TS94809: Measured time series vs. 1-step prediction

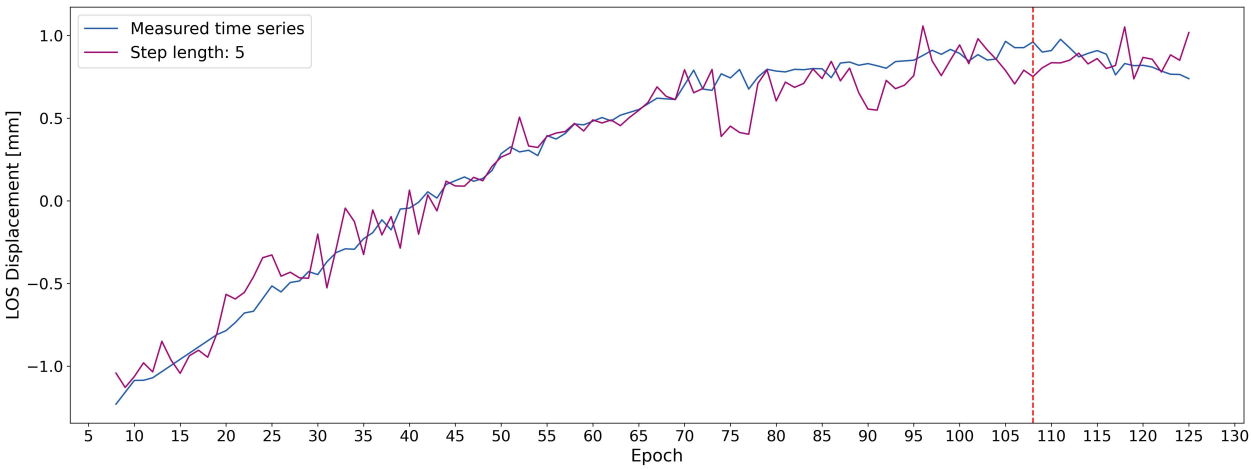

Figure 23: TS94809: Measured time series vs. 5-step prediction

Lower Bound	Upper Bound	Bin Count	MAE QR [mm]	MAE MLP [mm]	Improvement[%]
None	0.5	427	0.1384	0.1271	8.16
0.5	1.0	5,436	0.7623	0.4724	38.03
1.0	2.0	62,903	1.3695	0.8200	40.12
2.0	3.0	78,331	2.1413	1.2569	41.30
3.0	4.0	48,575	2.9286	1.6976	42.03
4.0	5.0	17,938	3.6737	2.1200	42.29
5.0	*None*	5,027	4.4340	2.5555	42.38

Table 7: Comparison of quadratic regression (QR) and the selected model using MLP-based prediction accuracies for 1-step predictions

4.3 Comparison to Quadratic Regression

In this section, to asses the selected model's prediction quality, it is put in contrast to a baseline approach to test its capabilities compared to a more established approach. In the following, the prediction accuracy, in form of MAEs, is compared to the MAEs of a quadratic regression model. Each time series is treated individually. To account for the input length of 8, used by the neural network, each time series is split into sequences of length 8. For each of these sequences, the parameters are estimated individually and a 1-step and a 5-step prediction is performed. The quadratic regression model scores a MAE of 2.2344mm for step length 1, which corresponds to a 70.81% increase compared to the in this work derived model's MAE of 1.3081mm for the same step length. For the 5-step prediction the quadratic regression models scores 8.0288mm in contrast to 1.6053mm on the MPL's part, further increasing the margin in favor of the neural network.

Table 7 states the MAE-based prediction accuracies for different levels of smoothness (as defined in section 4.2.4). The selected model outperforms the quadratic regression approach in all categories. The margin is minimal when tested against the smoothest 427 time series showing an improvement of 8%. Singular time series can be predicted more precisely using the Quadratic Regression approach, such as the in the previous section (4.2.5) TS94809. But these are based on table 7 assumed to be exceptions.

5 Discussion

In the subsequent chapter, the results presented in section 4 are discussed. Starting with the selected model and the results of the training process. The discussion is focused on what could still be improved based on the given results. In the next section the focus shifts to the selected models performance, starting with each measure separately to lastly combine all findings and put them in context. In the last section the comparison to the quadratic regression model, used as a baseline model, is discussed.

5.1 Discussion on the Model Selection

As mentioned in the previous introduction, the focus in this section lies on possible improvements to the network and the methodology based on the results at hand. The selected MLP is a 8-layered construct with layer sizes 8-45-60-75-60-45-5 as summarized in further detail in table 4. Compared to all other tested models, it scored the lowest validation loss averaged for all prediction step lengths, naturally qualifying it to be selected. The fact that within the scope of this thesis no model of higher complexity could be run, excludes the claim for optimality. This circumstance is due to time limitations. Further tests would be necessary to find a setup that scores a higher validation loss than the current one to indicate that this type of network construct has reached its optimal prediction quality for the given application, without claiming global optimality. In the following, possible improvements in terms of the methodology are discussed.

As presented previously, the guideline to use powers of 2 was dropped during the training process. Although the results displayed in figure 9 suggested such measures, more elaborate analyses on the matter would be beneficial. Continuing to run setups using powers of 2 could potentially deliver further evidence for this claim, supporting the already present argument. While the scope of this thesis did not allow for these tests to be run, the findings suggest that multiples of 15 lead to more accurate predictions, further investigations however would need to be conducted to undoubtedly conclude the hypothesis.

A similar limitation applies to the choice of input length. As part of this work, input lengths were only tested up until length 10, working with the baseline assumptions to limit the number of possible combinations and additionally keep the number of samples per time series above 90. Various models performed best using the exact input length of 10, suggesting that longer input lengths might be beneficial. Again, further investigations would be needed to verify if this is the case.

Lastly, a limitation that is most certainly implied by the applied methods is leaving tuning the batch size and learning rate for the very end. Whilst there was a good reason to keep the batch size large as described in section 3.3.3, it is very likely to have influenced the choice of layer setup, favoring the larger batch sizes. This results in a heavier correlation between the samples, which ultimately could have limited the model's prediction accuracy. The effects of adjusting the learning rate at a late stage can be presumed as less influential than excluding tuning the batch size when different layer configurations were tested. This is associated with the fact that a lot of research was put into optimizing Adam, therefore resulting in a suitable default setup for most applications.

5.2 Discussion on the Model's Performance

In this section the model's resulting performance is discussed. First focusing on each individual measure and then combining the findings of the analysis of the two exemplary time series presented in the previous chapter.

5.2.1 Discussion on Mean Absolute Errors

The statistical measures mean value and standard deviations both provide information on how two distributions of MAEs compare to each other in terms of performance. The results lead to the conclusion that the prediction quality decreases for increased step lengths. This is reflected in the minimal and maximal MAE for the entire data set as well as all mean values and standard deviations. Both of them show increases with increasing step lengths. The increases in mean values and standard deviation are also notable in the visual representation of the MAEs. The increased mean value results in a visual shift to the right for the distribution, whilst the higher standard deviation causes an elongation of the tail. The right-tailed behavior illustrates an asymmetry when comparing both extremes, in this case favoring larger MAEs over smaller ones.

The Median Absolute Error and the MAE differ for all sets of predictions in table 5. The Median Absolute Error with its robustness against outliers being lower suggests that there must be prediction outliers where the absolute error deviates significantly from what is observed for most other predictions. Possible causes for such phenomena are high levels of noise on the respective parts of the data or grave prediction errors. Though, the first option is more likely, since the concerning predictions are suggested to be outliers by analyzing the Median Absolute Error. Ultimately further analysis is required, where said predictions would be tracked and examined individually in regard to their large deviance.

Shifting the focus to assessing the model's prediction accuracy solely based on the test set, an increases is denoted for all statistical measures. This implies a lower prediction quality, which can be explained by the fact that this portion of the data was not used for training. Data behavior displayed in this set is unknown to the network, resulting in less accurate predictions. The observed maximal MAEs are significantly larger than for the entire data set. In addition to that, the longer step length has the lower maximum MAE than the shorter one. Looking at the numerical value and how high it is, it is more likely than not to be an extreme outlier. This is supported by the medians computed for each of the respective step lengths and their discrepancies to the mean values and the medians are larger for the test set than for the entire data set. It is important to mention that this is very likely not the only contributing factor to a generally higher maximum MAE. Other factors such as the overall less effective prediction behavior and the smaller sample size may very well amplify this phenomenon.

In terms of spatial distribution, outliers do not show a spatial pattern, leading to the conclusion that they are very likely to be the result of heavy noise or wrongful detections persistent scatterers. Additionally, the spatial distribution suggests on average a lower prediction quality, in terms of MAEs, in built environments. This could be due to two plausible reasons. First, in urban areas there could potentially be a higher number of disruptive contributions due to e.g. power lines, resulting in higher noise levels. The second reason could be that there is no pattern to detect after all. There are significantly more persistent scatterers contained in the data set which are located in built environments, leaving a higher chance for the model scoring higher MAEs on some of them. To definitively determine the causes, further investigations would be required. A possible starting

point would be to examine time series in built areas on the basis of their smoothness and exact geographical locations, possibly identifying disruptive factors.

Conclusively, it is important to note that whilst the MAE provides a good measure for assessing the quality of predictions, it does not respect expected changes. E.g. if the expected change for a time series is 0.5mm and the MAE lies at 1.2mm, the predictions will not be sufficiently accurate for this time series even though, compared to all the other time series, the MAE is better than the average. The same applies to the other end of the spectrum as well. For a time series showing change of 10mm or more, a MAE of 2mm could be sufficient. In addition to that, the noise levels for this data set are unknown, which again limits the range of conclusions to be drawn from the resulting MAEs, due to the fact that predictions are only adding information about deformation trends if they exceed the noise level.

5.2.2 Discussion on Residuals

Residuals can be used as an indicator of model bias. Based on the mean value of the residuals it can be determined whether the predictions have a tendency to overshoot or undershoot the measured time series. In the results at hand, a mean value of 0.0060mm indicates that predictions overshooting the true value are slightly more common. Due to its small magnitude, it is very likely that the model shows no biases and no further investigations are needed. It could be tested, however, if large spikes are more commonly found in the data than deep troughs. Based on the fact that the model is trained to predict the average behavior observed in the training process, it has a smoothening effect, resulting in larger errors for large spikes and deep troughs. Since the mean value of all residuals is positive, this would imply that larger spikes occur more frequently. This theory is yet to be vindicated with empirical evidence though.

The analysis of residuals also allows for statements about the quality to a certain extent. Similar to what could be concluded from analyzing the MAEs, the standard deviation of the residuals increases from the 1-step-prediction to the 5-step-prediction. A larger spread of residuals implies larger errors on individual predictions, therefore suggesting a lower prediction accuracy for the higher step length.

5.2.3 Discussion on Symmetric Mean Absolute Percentage Errors

Compared to the MAE, the sMAPE also accounts for the magnitudes of changes in between time steps. The average sMAPE value of 103.5% is considered high, suggesting that on average the error per prediction corresponds to approximately half of the measured change between the two points in time. Being left-tailed with a high mean value, the distributions of the sMAPEs and MAEs represent opposites. This can be an indication that rapid changes are found in the underlying data, due to the fact that such occurrences are difficult to predict based on average behavior. The sMAPE, being a percentage based measure on a set interval, then reacts more sensitively to errors in these specific scenarios than MAEs, resulting in opposite behavior.

5.2.4 Discussion on Smoothness

At many places in the previous chapters, the smoothness or the changing behavior of time series in the temporal domain seemed to be of the relevance. Due to the unknown noise level, a detailed analysis of the time series smoothness is not possible without the risk of over-analyzing what might simply be characterized as noise. Therefore, smoothness is compared to linear behavior in this

context. As previously stated in section 4.2.4, higher order derivation would result in an amplification of noise, skewing the results. The two exemplary time series found in section 4.2.5 TS51179 and TS94809, show this measures effectiveness for this application despite its rough estimation of smoothness. TS51179 containing a high level of noise scores a high value, corresponding to unsmooth behavior and TS94809 despite showing a quadratic underlying trend scores a very low value, representing smooth behavior.

5.2.5 Discussion on the Overall Performance and Individual Time Series

In the following section, the results from section 4.2.5 are discussed by combining the findings from the previous sections of chapter.

Delayed Reactions
Evaluating the predictions for TS51179, it is noticed that the predictions at times seemingly reacted late to changes in the course of the time series. This is a typical artifact as a result of forecasting. In most cases, the delay is linked to the step length of the prediction, since the model uses input that is further back in time, allowing it to assume changes only when it is fed the first signs of this change. This phenomenon is seen most distinctively for rapid change, since there are few to no signs of the change available before its occurrence. These delayed reactions to changes are relevant contributor to the MAE, since particularly for rapid and large changes these delays contribute significantly to the error.

Smoothening Effect
For both of the individual time series that were highlighted, the predicted time series seems to show smoothened behavior compared to the measured time series. This fact is linked to the model's basic assumption that all points are treated equally regardless of their geographical location. This results in an average behavior, based on all time series' trends and characteristics, learned by the model in the training stage. When a time step is predicted, the input is analyzed and the output is predicted according to the most likely behavior, which in this case is not optimized for the specific location but for the average behavior of all locations, resulting in a flattening of extreme values.

Overreactions
Predicting based on average behavior can also lead to the opposite of the in the previous paragraph described smoothening effect. For time series that show very smooth behavior, small changes can result in outputs predicting to much change. An example of a time series is TS94809 highlighted in section 4.2.5. This phenomenon is likely due to the model predicting based on the average behavior learned across all points during the training process as well. As illustrated in figure 19, a large majority of time series do not show linear behavior. This results in the average change being larger than the changes displayed in the smoothest of time series in the data set, ultimately resulting in overreactions to smaller changes on said smoother time series.

5.3 Discussion on the Comparison to Quadratic Regression

As stated in section 4.3, the MLP-based model achieves improvements reaching from 8% up to over 40% depending on the smoothness of the time series for a 12-day prediction (step length 1). The two lowest improvements are achieved for the very smoothest time series, since they most resemble linear or quadratic behavior, as seen for TS94809, where the quadratic regression model scores a slightly lower MAE than the selected model. For all other bins, the MLP clearly outperforms the quadratic regression. Though, the exact noise level is unknown, it may be assumed that time series

oscillating more rapidly, which are also considered less smooth by the measure used in this thesis, could contain higher noise levels or at least resemble time series containing higher levels of noise. The fact that the MLP's MAE is significantly lower for those bins suggests that this type of model could be more resistant to noise than quadratic regression. This potential property could be very valuable since even after sophisticated processing steps, there can still be noise present. This does not imply a sufficient prediction accuracy though. It only suggests a possibly higher robustness to noise.

Besides the fact that the selected model achieves significantly higher prediction accuracies on average, particularly for longer prediction step lengths, it also offers more flexibility. While a quadratic regression models needs to estimate its parameter for each input, resulting in immense computational costs, the MLP, once trained, is fast to be applied even on larger data sets.

6 Conclusions

This thesis aimed to construct a neural network using MLPs to predict earth deformation based on InSAR time series and assess its applicability for this purpose. Based on the finding in this thesis, the 8-layer construct, described in table 4, could potentially be improved by further investigating different layer configurations. Promising setups are more likely to consist of layer sizes being multiples of 15 rather than powers of 2. Although indications were found for multiples of 15 being more effective, the latter cannot be conclusively excluded. Furthermore, it can be concluded that the resulting model is likely to incorporate room for improvement by adapting the methodological approach, such as incorporating tuning of all hyperparameters at an earlier stage.

The analyses conducted in this thesis prove the model to show virtually no signs of biases with the capability to make predictions with a Mean Absolute Error of as low as 1.3081mm for a prediction of 12 days and 1.6053mm for a 60-days horizon. Though, no conclusive statements on the model's applicability to real life earth deformation prediction purposes can be made based on the Mean Absolute Error, due to the lacking knowledge on the exact noise level contained in the data set. Nevertheless, the analyses show that increasing the prediction step length results in a decrease in prediction accuracy.

While no definitive statement on the absolute quality of the predictions can be made, the comparison to a baseline model using quadratic regression shows clear advantages on the MLP's part. It scores lower MAEs for all smoothness-levels of time series, apart from those showing close to perfect quadratic behavior, where the regression-based model performs marginally better. For lower levels of smoothness the MLP reaches an improvement of over 40%, suggesting a possibly higher robustness to noise. The selected MLP-based model denotes an average 41% improvement for 12-day predictions and 80% for the 60-day predictions compared to the baseline approach on the data set at hand. Furthermore, the MLP-based approach offers more flexibility. The same weights can be applied to all geographical locations, while the parameters of the regressional model need to be estimated separately for each time series. In sum the MLP-based model concludes to be a promising approach for future research.

Based on the findings in this thesis, possible next steps could be to further improve the MLP-based approach, but also explore more sophisticated deep learning approaches, such as long short-term memory (LSMT) networks, to possibly further expand the pallet of found-to-be promising neural network approaches for earth deformation prediction. Furthermore, comparisons with other statistical prediction tools should be performed to further assess the MLP-based approach's performance.

7 References

[1] D. Massonnet and K. Feigl, "Radar interferometry and its application to changes in the earth's surface," *Reviews of Geophysics*, vol. 36, Nov. 1, 1998. DOI: 10.1029/97RG03139.

[2] European Space Agency. "ESA: Sentinel 1 – instrument." (2022), [Online]. Available: https://www.esa.int/Applications/Observing_the_Earth/Copernicus/Sentinel-1/Instrument (visited on 03/02/2022).

[3] N. Anantrasirichai, J. Biggs, K. Kelevitz, *et al.*, "Deep learning framework for detecting ground deformation in the built environment using satellite InSAR data," arXiv, arXiv:2005.03221, May 12, 2020, type: article. DOI: 10.48550/arXiv.2005.03221. arXiv: 2005.03221[cs,stat]. [Online]. Available: http://arxiv.org/abs/2005.03221 (visited on 06/06/2022).

[4] Z. Yang, Z. Li, J. Zhu, Y. Wang, and L. Wu, "Use of SAR/InSAR in mining deformation monitoring, parameter inversion, and forward predictions: A review," *IEEE Geoscience and Remote Sensing Magazine*, vol. 8, no. 1, pp. 71–90, Mar. 2020, ISSN: 2168-6831, 2473-2397, 2373-7468. DOI: 10.1109/MGRS.2019.2954824. [Online]. Available: https://ieeexplore.ieee.org/document/8977523/ (visited on 06/06/2022).

[5] P. González, R. Walters, E. Hatton, K. Spaans, A. Hooper, and T. Wright, "LiCSAR: Tools for automated generation of sentinel-1 frame interferograms," *undefined*, 2016. [Online]. Available: https://www.semanticscholar.org/paper/LiCSAR%3A-Tools-for-automated-generation-of-frame-Gonz%C3%A1lez-Walters (visited on 06/06/2022).

[6] K. J. Bergen, P. A. Johnson, M. V. de Hoop, and G. C. Beroza, "Machine learning for data-driven discovery in solid earth geoscience," *Science*, vol. 363, no. 6433, eaau0323, Mar. 22, 2019, ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.aau0323. [Online]. Available: https://www.science.org/doi/10.1126/science.aau0323 (visited on 06/06/2022).

[7] E. N. Grishchenkova, "Development of a neural network for earth surface deformation prediction," *Geotechnical and Geological Engineering*, vol. 36, no. 4, pp. 1953–1957, Aug. 2018, ISSN: 0960-3182, 1573-1529. DOI: 10.1007/s10706-017-0438-y. [Online]. Available: http://link.springer.com/10.1007/s10706-017-0438-y (visited on 06/06/2022).

[8] H. Chen, Z. Zeng, and H. Tang, "Landslide deformation prediction based on recurrent neural network," *Neural Processing Letters*, vol. 41, no. 2, pp. 169–178, Apr. 2015, ISSN: 1370-4621, 1573-773X. DOI: 10.1007/s11063-013-9318-5. [Online]. Available: http://link.springer.com/10.1007/s11063-013-9318-5 (visited on 06/06/2022).

[9] S. Roberts, A. Delorey, C. Johnson, R. Guyer, R. Alfaro-Diaz, and P. Johnson. "Using surface deformation and machine learning to determine state of stress changes at the coso geothermal field, california USA." (2021), [Online]. Available: https://www.semanticscholar.org/paper/Using-Surface-Deformation-and-Machine-Learning-to-Roberts-Delorey (visited on 06/06/2022).

[10] K. Barkhordari and H. Entezari Zarch, "Prediction of permanent earthquake-induced deformation in earth dams and embankments using artificial neural networks," *Civil Engineering Infrastructures Journal*, vol. 48, no. 2, pp. 271–283, Dec. 1, 2015, Publisher: University of Tehran, ISSN: 2322-2093. DOI: 10.7508/ceij.2015.02.004. [Online]. Available: https://ceij.ut.ac.ir/article_55705.html (visited on 06/06/2022).

[11] B. Osmanoğlu, F. Sunar, S. Wdowinski, and E. Cabral-Cano, "Time series analysis of InSAR data: Methods and trends," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 115, pp. 90–102, May 2016, ISSN: 09242716. DOI: 10.1016/j.isprsjprs.2015.10.003. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0924271615002269 (visited on 04/11/2022).

[12] R. F. Hanssen, *Radar Interferometry - Data Interpretation and Error Analysis*, ser. 2. Dordrecht: Kulwer Academic Publishers, 2001, 308 pp., ISBN: 0-306-47633-9. [Online]. Available: http://kluweronline.com.

[13] A. K. Gabriel, R. M. Goldstein, and H. A. Zebker, "Mapping small elevation changes over large areas: Differential radar interferometry," *Journal of Geophysical Research*, vol. 94, p. 9183, B7 1989, ISSN: 0148-0227. DOI: 10.1029/JB094iB07p09183. [Online]. Available: http://doi.wiley.com/10.1029/JB094iB07p09183 (visited on 04/11/2022).

[14] A. Duckstein, "Recent geodynamic and antrhopogenic structures in the upper rhine graben area," Feb. 7, 2022.

[15] A. Hooper, H. Zebker, P. Segall, and B. Kampes, "A new method for measuring deformation on volcanoes and other natural terrains using InSAR persistent scatterers: A NEW PERSISTENT SCATTERERS METHOD," *Geophysical Research Letters*, vol. 31, no. 23, Dec. 16, 2004, ISSN: 00948276. DOI: 10.1029/2004GL021737. [Online]. Available: http://doi.wiley.com/10.1029/2004GL021737 (visited on 05/24/2022).

[16] D. Bekaert, R. Walters, T. Wright, A. Hooper, and D. Parker, "Statistical comparison of InSAR tropospheric correction techniques," *Remote Sensing of Environment*, vol. 170, pp. 40–47, Dec. 2015, ISSN: 00344257. DOI: 10.1016/j.rse.2015.08.035. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0034425715301231 (visited on 05/24/2022).

[17] N. Mazroob Semnani, M. Breunig, M. Al-Doori, A. Heck, P. Kuper, and H. Kutterer, "TOWARDS INTELLIGENT GEO-DATABASE SUPPORT FOR EARTH SYSTEM OBSERVATION: IMPROVING THE PREPARATION AND ANALYSIS OF BIG SPATIO-TEMPORAL RASTER DATA," in *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, ISSN: 1682-1750, vol. XLIII-B4-2020, Copernicus GmbH, Aug. 25, 2020, pp. 485–492. DOI: 10.5194/isprs-archives-XLIII-B4-2020-485-2020. [Online]. Available: https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLIII-B4-2020/485/2020/ (visited on 05/24/2022).

[18] S. Corbel, J. Kaiser, and S. Vicentin, "Coal mine flooding in the lorraine-saar basin: Experience from the french mines," p. 6,

[19] M. Adya and F. Collopy, "How effective are neural networks at forecasting and prediction? a review and evaluation," *Journal of Forecasting*, vol. 17, no. 5, pp. 481–495, 1998, ISSN: 1099-131X. DOI: 10.1002/(SICI)1099-131X(1998090)17:5/6<481::AID-FOR709>3.0.CO;2-Q. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291099-131X%281998090%2917%3A5/6%3C481%3A%3AAID-FOR709%3E3.0.CO%3B2-Q (visited on 06/10/2022).

[20] K. Schindler, "Deep learning," Lecture Multivariate Statistics and Machine Learning, Zurich, Jun. 5, 2021.

[21] C. M. Bishop, *Pattern recognition and machine learning*, ser. Information science and statistics. New York: Springer, 2006, 738 pp., ISBN: 978-0-387-31073-2.

[22] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, ser. Adaptive computation and machine learning. Cambridge, Massachusetts: The MIT Press, 2016, 775 pp., ISBN: 978-0-262-03561-3.

[23] "TensorFlow," TensorFlow. (2022), [Online]. Available: https://www.tensorflow.org/?hl=de (visited on 05/19/2022).

[24] "Keras: The python deep learning API." (2022), [Online]. Available: https://keras.io/ (visited on 05/19/2022).

[25] E. Lewinson. "Choosing the correct error metric: MAPE vs. sMAPE," Medium. (Nov. 1, 2020), [Online]. Available: https://towardsdatascience.com/choosing-the-correct-error-metric-mape-vs-smape-5328dec53fac (visited on 05/23/2022).

[26] K. G. Sheela and S. N. Deepa, "Review on methods to fix number of hidden neurons in neural networks," *Mathematical Problems in Engineering*, vol. 2013, pp. 1–11, 2013, ISSN: 1024-123X, 1563-5147. DOI: 10.1155/2013/425740. [Online]. Available: http://www.hindawi.com/journals/mpe/2013/425740/ (visited on 06/04/2022).

[27] T. K. Gupta and K. Raza, "Optimizing deep feedforward neural network architecture: A tabu search based approach," *Neural Processing Letters*, vol. 51, no. 3, pp. 2855–2870, Jun. 2020, ISSN: 1370-4621, 1573-773X. DOI: 10.1007/s11063-020-10234-7. [Online]. Available: http://link.springer.com/10.1007/s11063-020-10234-7 (visited on 06/04/2022).

[28] M. Uzair and N. Jamil, "Effects of hidden layers on the efficiency of neural networks," in *2020 IEEE 23rd International Multitopic Conference (INMIC)*, Bahawalpur, Pakistan: IEEE, Nov. 5, 2020, pp. 1–6, ISBN: 978-1-72819-893-4. DOI: 10.1109/INMIC50486.2020.9318195. [Online]. Available: https://ieeexplore.ieee.org/document/9318195/ (visited on 06/04/2022).

[29] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, Jan. 29, 2017. [Online]. Available: http://arxiv.org/abs/1412.6980 (visited on 05/23/2022).