

Smart data representations: impact on the accuracy of deep neural networks

Oliver Neumann, Nicole Ludwig, Marian Turowski, Benedikt Heidrich, Veit Hagenmeyer, Ralf Mikut

Angaben zur Veröffentlichung / Publication details:

Neumann, Oliver, Nicole Ludwig, Marian Turowski, Benedikt Heidrich, Veit Hagenmeyer, and Ralf Mikut. 2021. "Smart data representations: impact on the accuracy of deep neural networks." In *Proceedings - 31. Workshop Computational Intelligence: Berlin, 25.-26. November 2021*, edited by Horst Schulte, Frank Hoffmann, and Ralf Mikut, 113–30. Karlsruhe: KIT Scientific Publishing.
<https://doi.org/10.58895/ksp/1000138532-8>.

Nutzungsbedingungen / Terms of use:

CC BY-SA 4.0



Smart Data Representations: Impact on the Accuracy of Deep Neural Networks

Oliver Neumann¹, Nicole Ludwig², Marian Turowski¹,
Benedikt Heidrich¹, Veit Hagenmeyer¹, Ralf Mikut¹

¹ Institute for Automation and Applied Informatics,
Karlsruhe Institute of Technology
Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen

² Cluster of Excellence Machine Learning,
University of Tübingen
Maria-von-Linden Str. 6, 72076 Tübingen
E-Mail: oliver.neumann@kit.edu

Abstract

Deep Neural Networks (DNNs) are able to solve many complex tasks with less engineering effort and better performance. However, these networks often use data for training and evaluation without investigating its representation, i.e. the form of the used data. In the present paper, we analyze the impact of data representations on the performance of DNNs using energy time series forecasting. Based on an overview of exemplary data representations, we select four exemplary data representations and evaluate them using two different DNN architectures and three forecasting horizons on real-world energy time series. The results show that, depending on the forecast horizon, the same data representations can have a positive or negative impact on the accuracy of DNNs.

DOI: 10.58895/ksp/1000138532-8 erschienen in:

Proceedings - 31. Workshop Computational Intelligence : Berlin, 25. - 26. November 2021

DOI: 10.58895/ksp/1000138532 | <https://www.ksp.kit.edu/site/books/m/10.58895/ksp/1000138532/>

1 Introduction

Deep Neural Networks (DNNs) can better solve complex tasks such as image classification [1, 2], object detection [3], or instance segmentation [4, 5] with less effort than traditional approaches. Nevertheless, DNNs require data for training and evaluation. However, data is often used by DNNs without further investigation of different data representations.

Since data representations influence what DNNs learn and which architectures can be used, data representations should be investigated further. The representation of the data can be changed through transformations such as reshaping, aggregation, or selection. Although recent literature, including work on feature engineering [6, 7, 8], introduces new data representations and compares them [9, 10, 11, 12, 13], it does not systematically investigate the influence of data representations on the performance of DNNs.

In this paper, we analyze the impact of data representations on the performance of DNNs at the commonly investigated example of energy time series forecasting (see e.g. [14, 15, 16, 13]). For this purpose, we investigate the time series in its original form and the derivative of the time series, and both reshaped as an image. For the analysis, we use two different architectures, namely a Fully Connected Network (FCN) and a Convolutional Neural Network (CNN).

The remainder of the paper is structured as follows. In the second chapter, we introduce different transformations for data representations. In the third chapter, we present an energy forecasting use case to demonstrate the impact of different data representations. In the fourth chapter, we discuss the findings of this paper, before we finally give a conclusion.

2 Transformations for Data Representations

In this chapter, we introduce different transformations for changing data representations based on the data types vectors or matrices. Transformations allow to convert data from one data representation to another. Unlike data

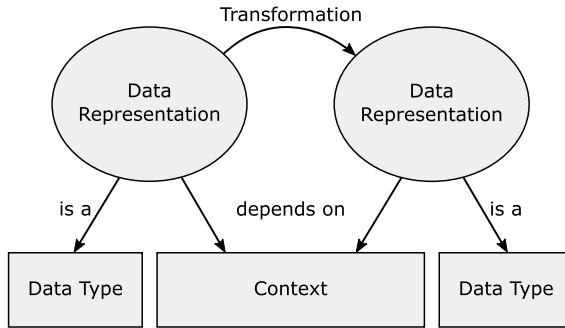


Figure 1: Relationship between data representation, data type, transformation, and context. A transformation converts a data representation into another, while a data representation is a data type and depends on a context.

types, data representations are context dependent and thus are typically difficult to characterize. For comprehensibility, we thus present transformations, which are per se context independent (see Figure 1). For the present paper, we consider reshaping, selection, aggregation, differences, convolution, rescaling, clustering, and latent space transformation as exemplary transformations for converting data from one representation into another. For each transformation, we briefly describe its key idea and underlying concept.

Reshaping Vectors and matrices can be transformed by changing their shape. This reshaping allows the use of different model architectures. Reshaping is defined by a function that maps each element of a vector or matrix to a resulting vector or matrix with a different dimensionality.

Selection Data representations can be transformed by selecting specific elements. To select certain elements, one can specify the related indices, which then define a subset of the considered vector or matrix. Choosing specific elements of a vector or matrix can be beneficial because a data representation can contain unnecessary information.

Aggregation The aggregation of data can be used as a transformation for data representations. Aggregations can help a DNN to achieve higher performances because aggregations can, for example, reduce the dimensionality and noise in the data. They can be applied on single vectors or matrices along one or more axes, leading to a matrix or vector depending on the dimensionality of the input.

Differences Calculating the differences is similar to the discrete derivation and can be applied to vectors and matrices. The data is transformed by subtracting the values of certain points or axes, vectors, or matrices depending on the input data representation. For vector inputs, the difference between certain points is calculated. For matrix inputs, the differences are calculated for certain subvectors or submatrices depending on the dimensionality of the matrix and along which axis the differences should be calculated.

Convolution A convolution is a mathematical operation that combines two functions. The convolution, e.g. a frequency filter, can be described by a kernel that is multiplied iteratively over the input data and summed afterward. Both the kernel and the input data can be vectors or matrices.

Rescaling The representation of data can be transformed by fitting a function on the data and resampling from that function. Thereby, data can be upscaled or downscaled, where upscaling increases and downscaling decreases the amount of data. Exemplary methods to approximate the underlying function are linear, cubic, or spline interpolation.

Clustering Data can also be clustered such that it is represented by cluster representatives. The cluster representatives are determined by similarity measures based on, for example, density or distances. Common clustering approaches are *k*-Means [17], fuzzy *c*-means [18], BIRCH [19], OPTICS [20], or DBSCAN [21].

Latent Space Transformation Latent space transformations learn a latent data representation of a dataset. This latent data representation has a lower dimensionality as the original dataset and could be a vector or matrix data representation. There are several approaches to use the latent space information and reduce the dimensionality like Principal Component Analysis [22], Linear Discriminant Analysis [23], or Autoencoder [24].

3 Energy Forecasting Use Case

In this chapter, we show how data representations affect the forecasting accuracy of a Deep Neural Network (DNN) when forecasting the German electricity demand and using four different data representations, namely *naive*, *naive differences*, *reshaped*, and *reshaped differences*. In the following, we first describe the data for this use case and the data representations. Afterward, we present the selected baselines and DNN architectures to forecast the electricity demand. In the last section, we present the results and compare the evaluated data representations.

3.1 Data

For the electricity demand, we use data from the European Network of Transmission System Operators for Electricity provided by Open Power System Data [25]. We select the electricity demand for Germany from the beginning of 2015 up to the end of 2019, which is the last complete year of data. The data contain typical daily, weekly, and seasonal patterns, which we account for with the help of calendar information. As calendar information, we use hour, day of week, day of year, weekend, and holiday, where the first three are encoded as sine and cosine functions and the last two are encoded as Boolean variables.

3.2 Evaluated Data Representations

This section describes the four data representations chosen to analyze the impact of data representations on the forecasting performance of DNNs, i.e. the

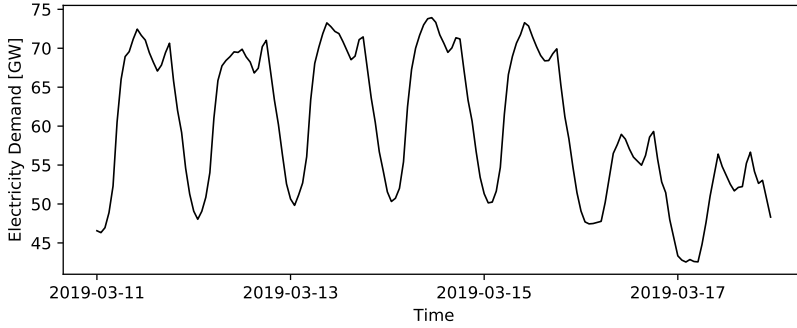


Figure 2: The *naive* data representation that comprises the electricity demand of the past 168 hours shown for the exemplary forecast origin 18.03.2019.

naive, the *naive differences*, the *reshaped*, and the *reshaped differences* data representations. Overall, this selection results in two vector-based and two matrix-based data representations. These data representations are used as input for the evaluated DNN architectures, whose output also depend on these representations.

Naive The *naive* data representation comprises the vector of the last 168 hours of the electricity demand (see Figure 2). It is defined as

$$\begin{bmatrix} x_k \\ \vdots \\ x_{k-167} \end{bmatrix}, \quad (1)$$

where $x \in X$ is the set of historical electricity demand values and k the forecast origin.

Naive Differences The *naive differences* data representation is again comprised of a vector of the last 168 hours of the electricity demand. However, instead of using the raw values as in the *naive* data representation, we now use differences. The lag of these differences depends, in our case, on the forecast

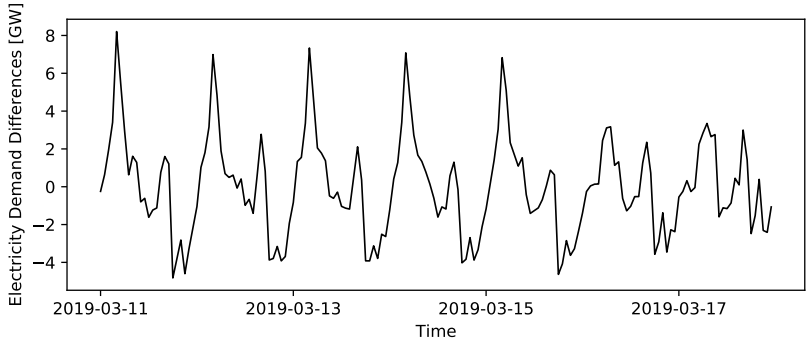


Figure 3: The *naive differences* data representation that comprises the electricity demand differences with $h = 1$ of the past 168 hours for the exemplary forecast origin 18.03.2019.

horizon, e.g., for a one-day ahead forecast, we calculate differences with a lag of one day. An exemplary week is illustrated in Figure 3 and the data representation is then defined as

$$\begin{bmatrix} x_k - x_{k-h} \\ \vdots \\ x_{k-167} - x_{k-167-h} \end{bmatrix}, \quad (2)$$

where $x \in X$ is the set of historical electricity demand values, k the forecast origin, and h the lag used for differencing that we set as the forecast horizon (e.g. 1, 24, 168).

Reshaped The *reshaped* data representation uses the *naive* data representation and reshapes it into a two-dimensional matrix such that each row represents a day. Consequently, the *reshaped* data representation consists of a 7x24 dimensional matrix (see Figure 4) and is defined as

$$\begin{bmatrix} x_k \\ \vdots \\ x_{k-167} \end{bmatrix} \Leftrightarrow \begin{bmatrix} x_k & \dots & x_{k-23} \\ \vdots & \ddots & \vdots \\ x_{k-144} & \dots & x_{k-167} \end{bmatrix}, \quad (3)$$

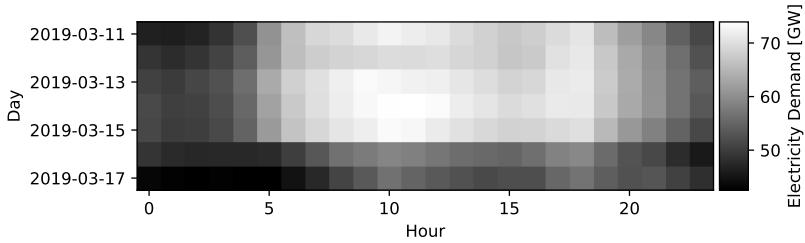


Figure 4: The *reshaped* data representation that comprises the electricity demand of the past 168 hours shown for the exemplary forecast origin 18.03.2019.

where $x \in X$ is the set of historical electricity demand values and k the forecast origin.

Reshaped Differences The *reshaped differences* data representation is equivalent to the *reshaped* data representation but uses the *naive differences* data representation as its basis. The *reshaped differences* data representation, therefore, also consists of a 7x24 dimensional matrix (see Figure 5 for an exemplary week) and is defined as

$$\begin{bmatrix} x_k - x_{k-h} \\ \vdots \\ x_{k-167} - x_{k-167-h} \end{bmatrix} \Leftrightarrow \begin{bmatrix} x_k - x_{k-h} & \dots & x_{k-23} - x_{k-23-h} \\ \vdots & \ddots & \vdots \\ x_{k-144} - x_{k-144-h} & \dots & x_{k-167} - x_{k-167-h} \end{bmatrix}, \quad (4)$$

where $x \in X$ is the set of historical electricity demand values, k the forecast origin, and h the lag used for differencing that we set as the forecast horizon (e.g. 1, 24, 168).

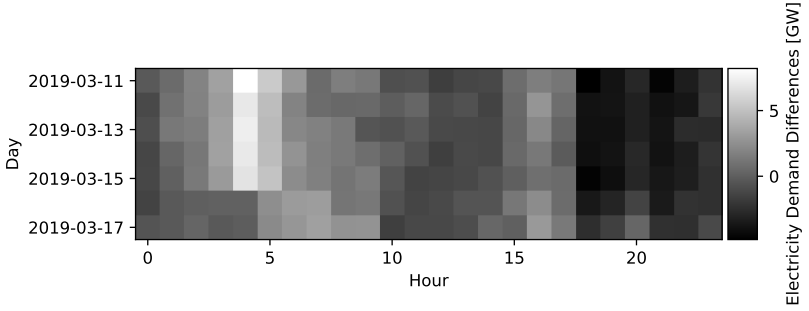


Figure 5: Exemplary week of the *reshaped differences* data representation that reshapes the electricity demand differences with $h = 1$ of the past 168 hours as a 24×7 matrix for the exemplary forecast origin 18.03.2019.

3.3 Experimental Setup

In this section, we introduce the evaluated DNN architectures and the selected baselines before we describe the experimental setup including the train validation test split, the considered forecast horizons, the number of performed runs, the used evaluation metrics, and the implementation.

To investigate the impact of data representations in energy time series forecasting, we use two DNNs (see Figure 6). The first DNN is a Fully Connected Network (FCN). It only consists of fully connected layers, whose input is a vector of historical energy data. This FCN is applied to both naive data representations. The second DNN is a Convolutional neural network (CNN). It consists of convolutional layers followed by fully connected layers that process the energy time series as a matrix. This CNN is applied to both reshaped data representations.

The FCN consists of two parts. The first part processes the energy time series vector input into a 64 dimensional latent vector representation and consists of two layers. The second part joins the 64 dimensional latent energy vector with the calendar feature vector and processes the concluding vector to the single forecast output.

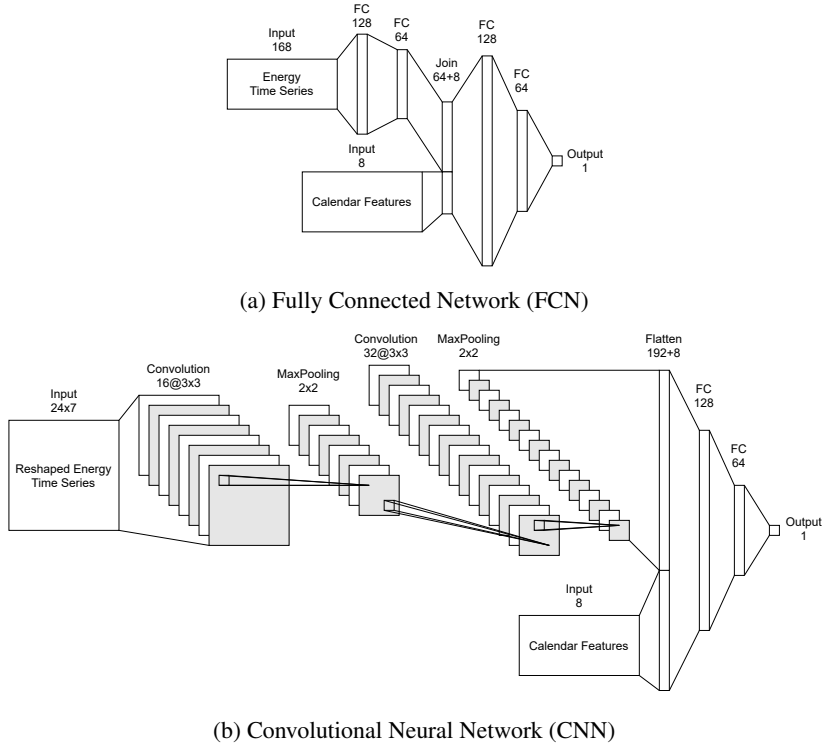


Figure 6: DNN architectures used for forecasting the energy time series, where the numbers indicate the number of neurons in the fully connected layers and the number of features with the corresponding kernel sizes in the convolutional layers (separated with @).

The CNN is also split into two parts. The first part processes the reshaped energy time series with two stacked convolutional layers. This first part results in a latent representation of matrices, which is then flattened and joined with the calendar features. The second part processes the vector-based latent representation of the energy time series and calendar features with three hidden layers.

As baselines for the general forecasting result, we choose two linear regression models that forecasts the electricity demand based on selected past electricity demand or rather electricity demand difference values and calendar features. For the electricity demand, we use the same data as for the *naive* data represen-

tation. Analog for the electricity demand differences, we use the same data as for the *naive differences* data representation. Regarding the calendar features, we use the same calendar features as for the FCN and CNN models. Thus, the linear regression for the electricity demand is defined as

$$\hat{x}_{k+h} = \alpha + \sum_{i=0}^{167} \beta_i x_{k-i} + \sum_{j=1}^8 \gamma_j c_{k+h,j}, \quad (5)$$

where $x \in X$ is the electricity demand, $c \in C$ the calendar features, and h the forecast horizon. For the electricity demand differences the linear regression is defined as

$$\hat{\Delta}(x_{k+h}, x_k) = \alpha + \sum_{i=0}^{167} \beta_i (x_{k-i} - x_{k-i-h}) + \sum_{j=1}^8 \gamma_j c_{k+h,j}, \quad (6)$$

where $x \in X$ is the electricity demand, $c \in C$ the calendar features, and h the forecast horizon.

To apply the mentioned architectures and benchmarks, we run the following setup: Regarding the train-validation-test split, we use the years 2015 to 2017 for training, 2018 for validation, and 2019 for testing. With regard to the forecast horizon, we forecast a specific hour for each model, i.e. one-hour, one-day, and one-week ahead. We evaluate the forecast horizons with the Mean Absolute Error (MAE) defined by $\frac{\sum_{i=1}^N |y_i - \hat{y}_i|}{N}$, where y_i is the ground truth and $\hat{y}_i$ the prediction. For each combination of the four evaluated data representations and the three forecast horizons, we run the respective network with ten different seeds. We report the mean and standard deviation of these ten runs and use this mean to calculate the relative advantage in percent compared to the *naive* data representation, i.e. $\frac{MAE_{compare}}{MAE_{naive}} - 1$.

The whole experimental setup is implemented in Python. For this purpose, we use PyTorch [26] for realizing the DNN architectures and pyWATTS [27] for defining a reproducible and reusable pipeline. The implementation is available on GitHub¹.

¹ <https://github.com/KIT-IAI/SmartDataRepresentations>

Table 1: Forecasting MAE results of the four evaluated data representations in GW for the three selected forecast horizons. For the *naive* and *naive differences* data representations, we use the FCN, while we apply the CNN for the *reshaped* and *reshaped differences* data representations.

Forecast Horizon	Data Representations			
	Naive	Naive Differences	Reshaped	Reshaped Differences
One-Hour	0.420 $\pm$ 0.025	0.376 (-10.3%) $\pm$ 0.027	0.531 (+26.5%) $\pm$ 0.015	0.385 (-8.3%) $\pm$ 0.003
One-Day	1.197 $\pm$ 0.128	1.289 (+7.6%) $\pm$ 0.118	1.209 (+0.9%) $\pm$ 0.017	1.240 (+3.5%) $\pm$ 0.010
One-Week	1.677 $\pm$ 0.084	1.775 (+5.8%) $\pm$ 0.102	1.742 (+3.9%) $\pm$ 0.028	1.820 (+8.5%) $\pm$ 0.012

3.4 Results

In this section, we present the results of the evaluated data representations regarding the energy time series forecast. For the 2019 test data, we report the results for the one-hour, the one-day, and the one-week ahead forecast (see Table 1). In addition, for all forecast horizons, we consider the *naive* data representation as a benchmark. More specifically, we compare the mean of all runs of each data representation to this benchmark before comparing the best data representations to the baseline.

For the one-hour ahead forecast, the *naive differences* data representation is best with an improvement of 10%. In contrast to this data representation, the *reshaped* data representation reduces the forecasting accuracy up to 27%. The *reshaped differences* data representation performs similarly as the *naive differences* data representation with an improvement of 8% compared to the *naive* data representation.

For the one-day ahead forecast, the *naive* data representation performs best, while the *reshaped* data representation performs quite similar with a higher MAE of 1%. The *naive differences*, and *reshaped differences* data representations reduce the forecasting performance between 3% to 8%.

For the one-week ahead forecast, the *naive* data representation is the best performing data representation. The *reshaped* data representation has a 4% higher MAE. However, both data representations based on differences, namely the *naive differences* and the *reshaped differences* data representations, perform worse than the *naive* data representation with higher MAEs between 5% and 9%.

Table 2: Forecasting MAE results of the baseline and the best evaluated data representations in GW for the three selected forecast horizons. For the baseline, we employ a linear regression. For the *naive* data representation, we use the FCN, while we apply the CNN for the *reshaped differences* data representation.

Forecast Horizon	Baselines		Best Data Representation	
	<i>Naive</i>	<i>Naive Differences</i>		
One-Hour	0.506	0.470	0.376 (-20.0%) $\pm$ 0.027	<i>Reshaped Differences</i>
One-Day	1.639	1.691	1.197 (-27.0%) $\pm$ 0.128	<i>Naive</i>
One-Week	1.975	2.128	1.677 (-15.1%) $\pm$ 0.084	<i>Naive</i>

For all forecast horizons, the best evaluated data representation performs better than the selected baselines. For example, the *reshaped differences* data representation improves the forecasting accuracy by at least 20% for the one-hour ahead forecast. The *naive* data representation obtains a 27% better forecasting accuracy for the one-day ahead forecast compared to the best performing baseline. For the one-week ahead forecast, the *naive* data representation achieves an 15% better forecasting accuracy. We additionally run the *naive* and *naive differences* data representation experiments on a simple Multilayer Perceptron (MLP) with one hidden layer consisting of ten neurons and achieve similar results as for the linear regression model.

4 Discussion

This section discusses the results of the energy forecasting use case. Our results show that the evaluated data representations, despite essentially containing the same information, result in different accuracies in the energy forecasting use case. Although the *reshaped* and the *reshaped differences* data representations are based on a similar concept, only the *reshaped differences* data representation outperforms the naive benchmark in the single case of the one-hour ahead forecast. Furthermore, depending on the forecast horizon, the data representations perform differently. Nevertheless, there is no data representation that offers the best forecasting accuracy for all evaluated forecast horizons. For example, the *naive differences* data representation is the best data representation for one-hour ahead forecasts. However, for the one-day and one-week ahead forecast, the *naive* data representation performs best. As a consequence,

it should be investigated in which way various data representations influence the forecasting accuracy of Deep Neural Networks (DNNs) and how they are affected by different forecast horizons and architectures given a specific use case.

In the evaluated energy forecasting use case, we consider four data representations and two DNN architectures. For these data representations and architectures, our results show that the forecasting accuracy varies. However, the ambiguous results do not allow for a general statement regarding the impact of data representations on the performance of DNNs. Moreover, the considered use case does not provide insights on the transferability of the results to other not yet evaluated data representations and to use cases from other domains. In addition, this work only investigates the impact of data representations on the accuracy of DNNs but does not examine other relevant metrics such as computational effort, robustness, or interpretability. Altogether, one should investigate further data representations, architectures, and use cases with regard to various metrics.

5 Conclusion

The present paper analyzes the impact of data representations on the performance of Deep Neural Networks (DNNs) at the example of energy time series forecasting. Based on an overview of exemplary data representations, we select four different data representations, namely the *naive*, the *naive differences*, the *reshaped*, and the *reshaped differences* data representation. We evaluate these data representations using two different DNN architectures and three forecasting horizons on real-world energy time series.

The results show that, depending on the forecast horizon, the same data representations can have a positive or negative impact on the accuracy of DNNs in the considered energy forecasting use case. Overall, there is no best performing data representation for all forecasting horizons. For example, reshaping the energy time series decreases the forecasting accuracy up to 27% compared to the *naive* data representation. However, reshaping the differences of the energy

time series is beneficial for one-hour ahead forecasts and yields an up to 8% higher forecasting accuracy compared to the *naive* data representation.

In future work, we plan to investigate the impact of various data representations on DNNs for datasets from different domains and other DNN architectures. In the case of energy forecasting, for example, data representations for additional information like weather could be investigated. For datasets from the given and other domains, the impact of data representations could also be evaluated regarding the problem complexity and dataset size as well as other metrics such as computational effort and interpretability. Furthermore, one could verify the results reported in the present paper using different DNN architectures and datasets. Lastly, it could be interesting to also investigate data representations within DNNs and probabilistic data representations.

Acknowledgements

This project is funded by the Helmholtz Association's Initiative and Networking Fund through Helmholtz AI, the Helmholtz Association under the Program "Energy System Design", and the German Research Foundation (DFG) under Germany's Excellence Strategy – EXC number 2064/1 – Project number 390727645.

References

- [1] Y. Liu, Y. Zhong, and Q. Qin, "Scene Classification Based on Multiscale Convolutional Neural Network," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, pp. 7109–7121, Dec. 2018.
- [2] P. M. Kamble and R. S. Hegadi, "Handwritten Marathi Character Recognition Using R-HOG Feature," *Procedia Computer Science*, vol. 45, pp. 266–274, Jan. 2015.
- [3] H. Zhu, X. Chen, W. Dai, K. Fu, Q. Ye, and J. Jiao, "Orientation robust object detection in aerial images using deep convolutional neural

- network,” in *IEEE International Conference on Image Processing (ICIP)*, pp. 3735–3739, IEEE, Sept. 2015.
- [4] G. Jader, J. Fontineli, M. Ruiz, K. Abdalla, M. Pithon, and L. Oliveira, “Deep Instance Segmentation of Teeth in Panoramic X-Ray Images,” in *31st Conference on Graphics, Patterns and Images (SIBGRAPI)*, pp. 400–407, IEEE, Oct. 2018.
 - [5] T. Scherr, K. Löffler, M. Böhland, and R. Mikut, “Cell segmentation and tracking using CNN-based distance predictions and a graph-based matching strategy,” *PLOS ONE*, vol. 15, id. e0243219, Dec. 2020.
 - [6] K. Faust, S. Bala, R. van Ommeren, A. Portante, R. Al Qawahmed, U. Djuric, and P. Diamandis, “Intelligent feature engineering and ontological mapping of brain tumour histomorphologies by deep learning,” *Nature Machine Intelligence*, vol. 1, pp. 316–321, July 2019.
 - [7] F. Seide, G. Li, X. Chen, and D. Yu, “Feature engineering in Context-Dependent Deep Neural Networks for conversational speech transcription,” in *IEEE Workshop on Automatic Speech Recognition Understanding*, pp. 24–29, IEEE, Dec. 2011.
 - [8] D. Zimbra, M. Ghiassi, and S. Lee, “Brand-Related Twitter Sentiment Analysis Using Feature Engineering and the Dynamic Architecture for Artificial Neural Networks,” in *49th Hawaii International Conference on System Sciences (HICSS)*, pp. 1930–1938, IEEE, Jan. 2016.
 - [9] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, “Geometric Deep Learning on Graphs and Manifolds Using Mixture Model CNNs,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5115–5124, IEEE, July 2017.
 - [10] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, “Graph neural networks: A review of methods and applications,” *AI Open*, vol. 1, pp. 57–81, Jan. 2020.
 - [11] P. Jahankhani, V. Kodogiannis, and K. Revett, “EEG Signal Classification Using Wavelet Feature Extraction and Neural Networks,” in *IEEE John*

Vincent Atanasoff 2006 International Symposium on Modern Computing (JVA'06), pp. 120–124, IEEE, Oct. 2006.

- [12] J. Jia and H.-C. Chua, “Solving two-spiral problem through input data representation,” in *Proceedings of ICNN - International Conference on Neural Networks*, vol. 1, pp. 132–135, IEEE, Nov. 1995.
- [13] B. Heidrich, M. Turowski, N. Ludwig, R. Mikut, and V. Hagenmeyer, “Forecasting energy time series with profile neural networks,” in *Proceedings of the Eleventh ACM International Conference on Future Energy Systems*, pp. 220–230, Association for Computing Machinery, June 2020.
- [14] J. G. Ordiano, S. Waczowicz, V. Hagenmeyer, and R. Mikut, “Energy forecasting tools and services,” *WIREs Data Mining and Knowledge Discovery*, vol. 8, no. 2, id. e1235, Dec. 2018.
- [15] M. Imani and H. Ghassemian, “Sequence to Image Transform Based Convolutional Neural Network for Load Forecasting,” in *27th Iranian Conference on Electrical Engineering (ICEE)*, pp. 1362–1366, IEEE, Apr. 2019.
- [16] G. Hafeez, K. S. Alimgeer, and I. Khan, “Electric load forecasting based on deep learning and optimized by heuristic algorithm in smart grid,” *Applied Energy*, vol. 269, id. 114915, July 2020.
- [17] C. Chinrungrueng and C. Sequin, “Optimal adaptive k-means algorithm with dynamic adjustment of learning rate,” *IEEE Transactions on Neural Networks*, vol. 6, pp. 157–169, Jan. 1995.
- [18] J. C. Bezdek, R. Ehrlich, and W. Full, “FCM: The fuzzy c-means clustering algorithm,” *Computers & Geosciences*, vol. 10, pp. 191–203, Jan. 1984.
- [19] T. Zhang, R. Ramakrishnan, and M. Livny, “BIRCH: an efficient data clustering method for very large databases,” *ACM SIGMOD Record*, vol. 25, pp. 103–114, June 1996.

- [20] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, “OPTICS: ordering points to identify the clustering structure,” *ACM SIGMOD Record*, vol. 28, pp. 49–60, June 1999.
- [21] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, Aug. 1996.
- [22] K. Pearson, “On lines and planes of closest fit to systems of points in space,” *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, pp. 559–572, Nov. 1901.
- [23] P. Cohen, P. Cohen, S. G. West, and L. S. Aiken, *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences*. Psychology Press, 2 ed., Apr. 2014.
- [24] M. A. Kramer, “Nonlinear principal component analysis using autoassociative neural networks,” *AIChE Journal*, vol. 37, no. 2, pp. 233–243, Feb. 1991.
- [25] F. Wiese, I. Schlecht, W.-D. Bunke, C. Gerbaulet, L. Hirth, M. Jahn, F. Kunz, C. Lorenz, J. Mühlenpfordt, J. Reimann, and W.-P. Schill, “Open Power System Data – Frictionless data for electricity system modelling,” *Applied Energy*, vol. 236, pp. 401–409, Feb. 2019.
- [26] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “PyTorch: An Imperative Style, High-Performance Deep Learning Library,” in *Advances in Neural Information Processing Systems*, vol. 32, Curran Associates, Dec. 2019.
- [27] B. Heidrich, A. Bartschat, M. Turowski, O. Neumann, K. Phipps, S. Meisenbacher, K. Schmieder, N. Ludwig, R. Mikut, and V. Hagenmeyer, “pyWATTS: Python Workflow Automation Tool for Time Series,” *arXiv*, id. 2106.10157v1, June 2021.