RESEARCH **Open Access**

Estimating time-delayed variables using transformer-based soft sensors

Jelke Wibbeke[1,2*], Darian Alves[1] and Sebastian Rohjans[1]

From The 12th DACH+ Conference on Energy Informatics 2023
Vienna, Austria. 4-6 October 2023. https://www.energy-informatics2023.org/

*Correspondence:
jelke.wibbeke@jade-hs.de

[1] Department for Civil Engineering Geoinformation and Health Technology, Jade University of Applied Science, Ofener Str. 16/19, 26121 Oldenburg, Germany
[2] Energy Department, OFFIS-Institute for Information Technology, Escherweg 2, 26121 Oldenburg, Germany

Abstract
In the course of digitization, there is an increased interest in sensor data, including data from old systems with a service life of several decades. Since the installation of sensor technology can be quite expensive, soft sensors are often used to enhance the monitoring capabilities. Soft sensors use easy-to-measure variables to predict hard-to-measure variables, employing arbitrary models. This is particularly challenging if the observed system is complex and exhibits dynamic behavior, e.g., transient responses after changes in the system. Data-driven models are, therefore, often used. As recent studies suggest using Transformer-based models for regression tasks, this paper investigates the use of Transformer-based soft sensors for modelling the dynamic behavior of systems. To this extent, the performance of Multilayer Perceptron (MLP) and Long Short-term Memory (LSTM) models are compared to Transformers, based on two data sets featuring dynamic behavior in terms of time-delayed variables. The outcomes of this paper demonstrate that while the Transformer can map time delays, it is outperformed by MLP and LSTM. This deviation from previous Transformer evaluations is noteworthy as it may be influenced by the dynamic characteristics of the input data set, and its attention-based mechanism may not be optimized for sequential data. It is important to mention that the previous studies in this area did not focus on time-delayed dynamic variables.

Keywords: Supervised learning, Transient response, Time series, Regression, Dynamic systems

Introduction

Soft sensors are models that use easy-to-measure variables to predict hard-to-measure variables. They are often used where conventional physical sensors cannot be used or are too expensive. Since there is an increased interest in obtaining additional data from systems and equipment in the course of digitization, the latter often applies to legacy systems that are already in operation. For many systems, it is not uncommon to have a service life of several decades, such as converter stations in the power grid. Thus, a lot of sensor technology is not installed in these systems, which makes the enhancement of the sensing capabilities particularly costly (Kabugo et al. 2020; Maschler et al. 2021). However, even if physical sensors are available, soft sensors are often used in parallel for condition monitoring or anomaly detection purposes (Alzawaideh et al. 2021; Filios et al. 2022).

The development of soft sensors is not always straightforward. Especially when the exact physical conditions and relationships of the system are unknown or complicated, data-driven soft sensors rather than physics-based soft sensors (like Kalman filters) are more convenient (Maschler et al. 2021; Sun and Ge 2021).

A particular challenge in modeling data-driven soft sensors is the calculation of dynamic behavior, e.g., transient states after changes in the system. These so-called transient responses occur when a process variable of a system has been changed, and the system has not yet reached the updated steady state again.

Transient responses exist in almost all technical systems. They can be caused by thermal inertia (e.g. heat capacity of a hotplate), rotational inertia (e.g. of a spinning generator), electric capacitance/inductance (e.g. of circuit boards) or the speed of chemical reactions. A simple example of a transient response is a hotplate, which is not immediately hot after switching on, but has a warm-up phase due to its heat capacity, causing the temperature to lag. Consequently, transient target variables can often not, or only moderately, be determined with the current snapshot of the system. Hence, past measured values in the form of time series have to be included (Ge et al. 2017).

The use of Long Short-term Memory (LSTM) networks, which represent a form of recurrent neural networks, has become particularly established in this context.

However, in 2017, Transformer networks were introduced (Vaswani et al. 2017). Transformers were initially developed for natural language processing, where they have become best practice, offering substantially better performance than LSTMs (Karita et al. 2019). In recent years, several studies have been presented suggesting an application of Transformers for regression problems, which seem to be on par or even better than LSTMs, at least for forecasting applications (Wen et al. 2022; Wu et al. 2020; Zhou et al. (2021).

Therefore, the research question that motivates this work is: How well can Transformers-based soft sensors be used to estimate time-delayed variables?

To answer this question, a Multilayer Perceptron (MLP), a LSTM, and a Transformer model are trained from two real-world data sets. The target variable of each data set is the temperature of the system, which reacts delayed to changes in the system and thus represents an overdamped transient response. To compare the three models, the hyperparameters are optimized to evaluate the performance in terms of accuracy, training time, and amount of training data needed.

The remainder of the paper is structured as follows: In section Related Work, the related work and state of the art are discussed. The used Transformer model is presented in section Estimation of Time-delayed Variables. Section Experimental Study explains the approach of the intended comparison, followed by a presentation and discussion of the results. In section Conclusion, a summary of the results is presented.

Related work

In recent years, neural networks have shown great promise in modelling dynamic variables. One such application is in the field of wind turbine anomaly detection and failure prediction, where Alzawaideh et al. (2021) proposes the use of LSTM networks to estimate the operating temperature of wind turbine modules (e.g. the gearbox). Another application is proposed by Chen et al. (2020), who demonstrates the efficiency of LSTMs in very short-term forecasting of photovoltaic power generation, and Azman et al. (2020), who uses LSTMs to predict the transient stability of a power system subjected to disturbances.

Multiple data-driven methods are compared by Kabugo et al. (2020) to train a soft sensor to predict the syngas heating value and hot flue gas temperature in a biomass waste-to-energy plant. This analysis shows that the neural network-based nonlinear autoregressive model performs better than the other static models tested due to the ability to use time series.

In 2017, Vaswani et al. (2017) proposed the Transformer model architecture, which utilizes the attention mechanism to improve natural language processing significantly (Karita et al. 2019). Transformers do not rely on a sequential structure and allow for parallel processing, which can dramatically accelerate computation time when using GPUs. Based on the attention mechanism multiple network architectures and use cases are proposed. Wu et al. (2020) demonstrates the superiority of Transformer-based time series forecasting over LSTMs in predicting influenza prevalence. Zhou et al. (2021) shows that the Transformer outperforms LSTMs in long sequence time series forecasting, highlighting the benefits of the non-sequential processing of Transformers in reducing computation time during inference. Lim et al. (2021) presents the Temporal Fusion Transformer to fuse data for multi-horizon forecasting. A review by Wen et al. (2022) presents a taxonomy of Transformer network modifications and application domains, analyzing the performance of various proposed architectures on time series forecasting tasks.

In addition to forecasting, Transformers have also been applied in anomaly detection, with Tuli et al. (2022) reporting that their Transformer-based approach outperforms various competitors, including LSTMs, with less data and reduced training times. Geng et al. (2021) uses a combination of gated convolutional neural networks and Transformers for dynamic soft sensor modelling to measure the solvent content of two industrial chemical processes. The results indicate that, albeit omitting hyperparameter optimization for LSTM and Transformer, the Transformer outperforms the LSTM.

To move away from specialized applications, a generalized framework called Time Series Transformer is proposed by Zerveas et al. (2021), which can be used for various multivariate modelling problems, including regression, classification, forecasting, and sample imputation. It is shown that the proposed framework outperforms comparable models on nearly all occasions, especially with limited training data. However, the focus is not on dynamic variables, whereby no comparison with an LSTM for regression is addressed.

Regardless of the grand reception, not everyone agrees on the superiority of Transformers to LSTMs. For example, Zeng et al. (2022) question the necessity of Transformers in long time series forecasting, presenting an empirical study in which a

Transformer-based network is outperformed by a simpler one-layer linear MLP-based model.

In summary, in the field of regression, Transformers are mainly used for forecasting applications focusing on long time series, leveraging the particular advantage over the sequential architecture of LSTMs. Outside of long time series forecasting, the approach of Zerveas et al. (2021) is the most popular. For example, Agrawal et al. (2022) uses the approach to predict the maximum power point for solar photovoltaic cells, showing promising results, albeit not comparing it to other approaches as a baseline. However, the usability of transformers for dynamic variables has not yet been covered.

Estimation of time-delayed variables

Time-delayed variables are variables, whose value does not react immediately to changes in the system, but require a certain amount of time to adjust. An example of such a variable is temperature measurement. When ambient temperature changes occur, the thermometer requires a certain amount of time to register and display this change. This time delay is called response time. The length of the response time depends on the system and can be in the microsecond range, as in the case of voltage spikes in electrical circuits, or in the minutes to hours range in the case of temperature changes.

Most physical systems do not respond instantaneously to changes in the system by jumping to the steady state, but have an inertia that delays the adaptation to the changes. This results in a period of dynamic state in the system between the initial static state $t_{steady,1}$ and the steady state after the change $t_{steady,2}$, during which the system transitions. Depending on the number of changes in the system and the duration of the transient response, it is possible that the target variable permanently lags behind the actual system state. Figure 1 shows an example of a time-delayed variable's response to a system change.

The temporal aspect must be integrated into the model for soft sensors of dynamic variables. Otherwise, the model is not able to know that the response time has not yet elapsed after a system change, and thus the system is in a dynamic state.

For the integration of temporal dependencies, the input data are divided into a sequence of time points. The distance between points and the number of past points of time to be included in the model is defined by the response time and must be sufficient to capture it.

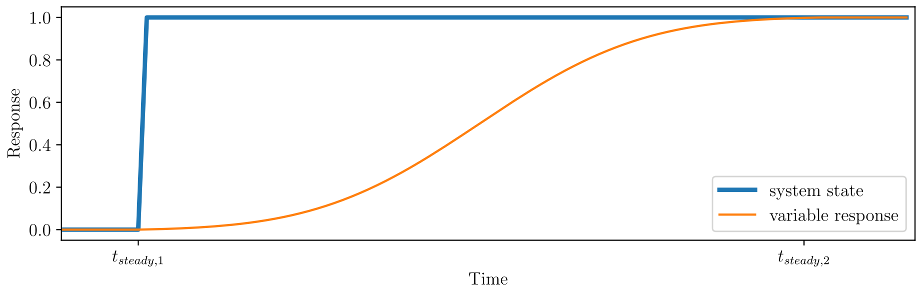

Fig. 1 Typical response of a time delayed variable to a change in system state

LSTMs are a particular type of recurrent neural network used for modeling time series data. It is capable of learning the pattern of past input variables and making a prediction for the current values of the output variables. LSTMs thus offer good possibilities for modeling time-delayed variables. Transformer architectures can also be used, but these must first be adapted to multivariate sequences.

Transformers for multivariate regression

The Transformer architecture presented by Vaswani et al. (2017) originates in natural language processing and is designed and optimized for these applications. At an abstract level, language processing applications are univariate sequence-to-sequence learning tasks in that a sequence in the form of a text is given the model, and also a text is obtained. However, the regression problem addressed here for the estimation of transient variables is a multivariate sequence-to-one problem. Each input sample X consists of w feature vectors $x_t : X = [x_1, x_2, ...x_w]$ with t being the time steps $t = 0, ..., w$ of a multivariate time series of length w. Each feature vector x consists of m different variables. Thus, each sample X is of dimensions $X \in \mathbb{R}^{m\times w}$. The predicted value is of size $\hat{y} \in \mathbb{R}^n$, where n is the number of scalars to be estimated, here $n = 1$.

For the structure of the Transformer, the architecture presented by Zerveas et al. (2021) is used in this paper, being one of the most established architecture designed for regression tasks. Unlike the original Transformer for natural language processing, the Time Series Transformer architecture does not consist of an encoder and decoder part, but uses only the encoder. However, the core of the encoder, and the encoder blocks are identical. A detailed description of the encoder blocks is referred to Vaswani et al. (2017).

In the following, we discuss the most important properties of the used Transformer model for multivariate time series. However, Zerveas et al. (2021) provides more detailed descriptions. The basic structure of the Transformer is shown in Fig. 2.

Upstream of the positional encoding, a linear layer is used to linearly project each feature vector x into a d–dimensional space, where d is the model dimension of the Transformer. The resulting model input vector corresponds to the word vectors of natural language processing Transformers.

$$u_t = W_p x_t + b_p \qquad \text{with } W_p \in \mathbb{R}^{d\times m}, b_p \in \mathbb{R}^d \tag{1}$$

$$U = [u_1, u_2, ...x_w] \tag{2}$$

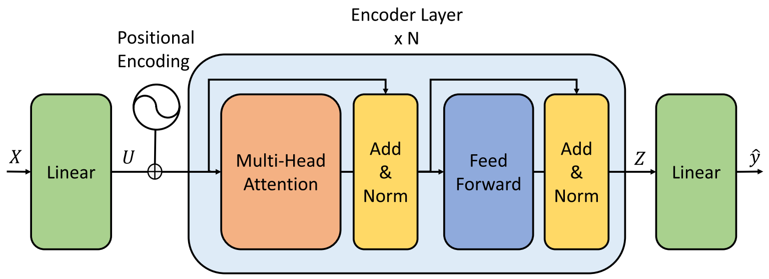

Fig. 2 Schematic of the used Transformer architecture, based on the architecture proposed by Zerveas et al. (2021)

The positional encoding is followed by a sequence of encoder layers, consisting of multi-headed attention, normalization, a fully connected feed-forward network, and another normalization (cf. Fig. 2).

Lastly, a linear layer acting as an output layer is added to the model. The encoder layer output vectors $z_t \in \mathbb{R}^d$ are concatenated into a single vector $Z \in \mathbb{R}^{d \cdot w} = [z_1; ...; z_w]$, serving as the input. The layer linearly projects the encoder output to the scalars $\hat{y} \in \mathbb{R}^n$ to be estimated, where n is the number of scalars to be estimated for the regression problem.

$$\hat{y} = W_o Z + b_o \qquad \text{with } W_o \in \mathbb{R}^{n \times (d \cdot w)}, b_o \in \mathbb{R}^n \tag{3}$$

The parameters of both linear layers are learnable parameters, optimized during model training.

Experimental study

In this section, the used data sets and models are described first. Subsequently, the training scenarios are presented, and the results are discussed.

Data sets

All tests are performed on two different data sets.

The first data set is a publicly available data set from a PV array (Boyd et al. 2017). The array consists of 312 PV modules, which provide 73.3 kW of power. From the data set, 100,000 samples are used, consisting of 13 features with a resolution of 0.1 Hz. The target variable is the back sheet temperature of one of the PV modules, which changes only slowly and with a time delay due to the heat capacity of the material. An abnormal increase in back sheet temperature might be caused by deteriorating solder bonds or partial shading, and in extreme scenarios, it has the potential to result in fire hazards (Bosman et al. 2020). Wind speed, outside temperature, solar radiation, and power are examples of used input values. A detailed list of the used features and which pre-processing steps were taken can be found in the Appendix: PV-Dataset. Since the time delay of the temperature is approximately 1 min, sequences with a length of 18 samples (3 min) are generated from the data set. In the remainder of this paper, the data set will be referred to as PV-Data.

The second data set is a proprietary data set of sensor values of a high-power inverter. The temperature of the insulated-gate bipolar transistor (IGBT) module of an inverter is used as the target variable. Physically, this is interesting in that a high IGBT temperature indicates internal losses in the IGBT, and a too-high temperature can lead to the destruction of the inverter. Due to the heat capacity of the module, the temperature is also time-delayed. Again 100,000 samples were used, which were recorded within 1 year, at a rate of 0.2 Hz. Since the time delay of the temperature is about 30 s, sequences with a length of 18 samples (1.5 min) are generated from the data set. In the following, this data set is referred to as Inv-Data.

A summary of the properties of both data sets is shown in Table 1.

Table 1 Summary of properties of the used PV and inverter data sets

	PV-Data	**Inv-Data**
Target variable	Module back sheet temperature	IGBT temperature
Input features	13	32
Samping rate	0.1 Hz	0.2 Hz
Sequence length	18 samples (3 min)	18 samples (1,5 min)
Number of samples	100.000	100.000

Table 2 Evaluated hyperparameter for the different model architectures

	PV-Data	**Inv-Data**
Multilayer Perceptron		
Batch size	16, 32, 64, **128**, 256	16, 32, 64, **128**, 256, 512
Number of layers	1, 2, **3**, 4	1, **2**, 3, 4
Layer size	128, 256, **512**, 1024	128, **256**, 512, 1024
Dropout	**0.1**, 0.2	**0.1**, 0.2
Long Shot-term Memory		
Batch size	16, **32**, 64, 128, 256	16, **32**, 64, 128, 256
Number of layers	2, **3**, 4	2, **3**, 4
Layer size	64, 128, **256**, 512	32, **64**, 128, 256, 512
Dropout	**0.1**, 0.2	**0.1**, 0.2
Time Series Transformer		
Batch size	64, **128**, 256, 512	64, 128, **256**, 512
Model dimension	32, **64**, 128	32, **64**, 128
Number of attention heads	8, 16, **32**, 64	**8**, 16, 32, 64
Number of encoder blocks	2, 4, **6**, 8	2, 4, 6, **8**
Feedforward layer size	32, **64**, 128, 256	32, 64, 128, **256**

The best hyperparameters are bold

Models

In the following, the structure of the three model architectures is described. For all models, an "Adam" optimizer with an initial $\textit{learning rate} = 0.001$ and a 80/20 train/test-split are chosen. The root-mean-square error (RMSE) is used as loss function. To compare the model performance, a hyperparameter optimization is performed for each model architecture. All optimized hyperparameters can be found in Table 2.

The MLP models are networks of ordinary fully connected dense layers. All hidden layers have dropout and a "ReLu" activation function. As MLPs are not capable of processing multivariate sequences $X \in \mathbb{R}^{m \times w}$, the input to the model is flattened first to $X \in \mathbb{R}^{m \cdot w}$.

For the LSTM, the implementation of PyTorch is used, which is based on the architecture presented by Sak et al. (2014). As each LSTM-layer outputs a sequence, the last layer is followed by a linear layer to project the output sequence into the desired scalar value. The dropout is applied to every layer except the last.

For the Transformer, the architecture presented in section Transformers for Multivariate Regression is used. Additionally, the recommended learnable positional encoding and batch normalization instead of layer normalization is applied and a "gelu" activation function is used (Zerveas et al. 2021). In addition, models are trained which are previously subjected to an autoregressive unsupervised pretraining. For this purpose, the input is subjected to a binary noise mask and the model is set to predict the masked values (Zerveas et al. 2021).

Results and discussion

After the hyperparameters were optimized, 10 models were trained for each architecture. For this purpose, an early stopping was used, which terminated the training if no improvement of the loss was achieved within 5 epochs. The course of the training is shown in Fig. 3.

Looking at these progressions, it is noticeable that the loss of the Transformer seems to change a lot from epoch to epoch. A possible reason for this could be the selection of the wrong hyperparameters. This cannot be ruled out completely, since a complete analysis of the hyperparameter is not feasible and, therefore, inherently limited. In order to minimize the negative influence of untested hyperparameters, values described by Zerveas et al. as “reasonably well performing” were used, such as the learnable positional encoding (Zerveas et al. 2021). However, this behavior was observed across all hyperparameters tested, including the pretrained models. The most likely responsible hyperparameter is the learning rate. To investigate this, a one magnitude smaller learning rate (0.0001) was tested, but without influence on the result. Another reason could be an over-fitting of the model. To minimize over-fitting, different amounts of dropout ([0.1, 0.5]) and weight decay of the Adam optimizer ([0.1, 0.5]) were tested as regularization. Both with no influence on the result.

Subsequently, the best performing model from each training run was selected and compared in terms of loss and training time. The results regarding loss are shown in Table 3. They indicate that if only the best performing model per training run is considered, the loss of each run is comparable. Even though the Transformers performance during training fluctuates from epoch to epoch. Nevertheless, the Transformers are outperformed by LSTM and MLP in both data sets. The LSTM performs best in each case. In the case of time delayed variables it also seems to be the case, that an unsupervised pretraining has no beneficial effect on the model. However, this is not beyond expectation, since also Zerveas et al. report positive effects in only about half of the data sets Zerveas et al. (2021). When comparing the training time, it can be seen that the LSTM

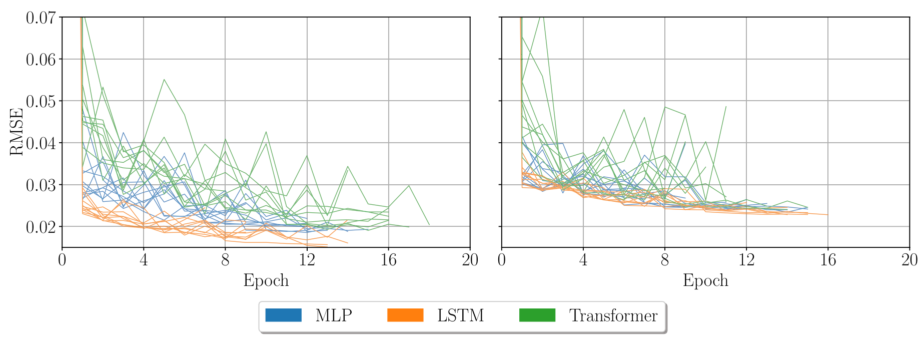

Fig. 3 Model performance progress during training on Inv-Data (left) and PV-Data (right) data set. For each architecture, 10 models were trained

Table 3 RMSE of best-performing models from each of the 10 training runs

	PV-Data		**Inv-Data**	
	Average	**Best**	**Average**	**Best**
MLP	0.0250 ± 0.0013	0.0234	0.0201 ± 0.0014	0.0186
LSTM	0.0240 ± 0.0016	**0.0228**	0.0173 ± 0.0013	**0.0151**
Transformer	0.0260 ± 0.0013	0.0241	0.0215 ± 0.0025	0.0191
(pretrained) Transformer	0.0287 ± 0.0010	0.0273	0.0246 ± 0.0029	0.0208

Bold values indicate the best result in comparison to the different model types

has the longest training time. This is expected, due to the sequential data processing (cf. Table 4).

Lastly, the performance of the models was investigated in dependence on the amount of training data. For this purpose, the number of training samples was reduced, while the test set remained unchanged. Again, 10 models were trained per architecture. The results are presented in Fig. 4. As expected, they indicate that the performance of LSTM and MLP deteriorates with decreasing sample number. However, the performance of the Transformer does so to the same extent. Therefore, a particular suitability of Transformers for smaller data sets cannot be determined. Individual outliers in the Transformer performance also suggest that some Transformer models only converge to a local minimum.

The presented results are not in line with the majority of publications about Transformers for regression (cf. Related work). However, said publications do not appeal to dynamic variables, but mostly long time series forecasting (e.g. Zhou et al. (2021)). In their proposal of the used Transformer architecture, Zerveas et al. also used

Table 4 Average total training time (until maximum accuracy is recorded) in seconds

	PV-Data	**Inv-Data**
MLP	892 ± 283	2058 ± 251
LSTM	10916 ± 2793	6032 ± 1977
Transformer	4635 ± 1263	4834 ± 1267

For each architecture 10 models were trained

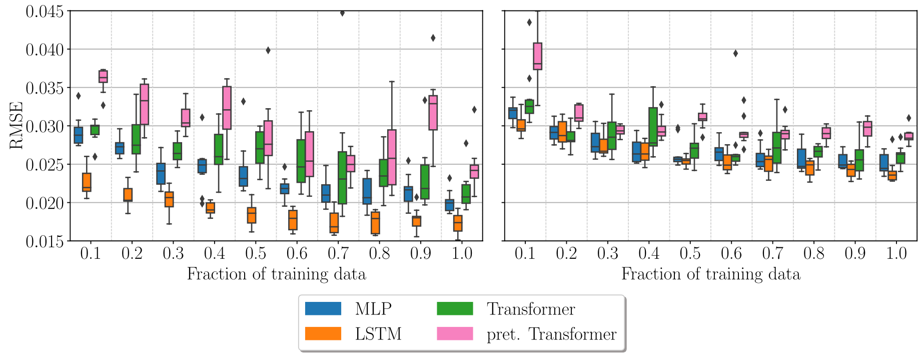

Fig. 4 Dependence of model performance on the amount of training data in the Inv-Data (left) and PV-Data (right) data sets. A fraction of 1 is equivalent to using the complete training data set of 80,000 samples. For each data fraction 10 models were trained

non-forecasting data sets, showing superior performance of the Transformer. However, results may vary as no LSTM was used for comparison, and no dynamic variables were considered (Zerveas et al. 2021).

Apart from the superior performance of sssLSTMs, Transformers are more challenging to train, being more complex and less researched and established. In retraining models, the high performance fluctuation also might open up questions regarding traceability and robustness when deployed as soft sensors. This could be relevant for continual learning approaches.

The research question “How well can Transformers-based soft sensors be used to estimate time-delayed variables?” can therefore be answered in that Transformers are indeed usable and have a reasonable performance, but more convenient solutions are available.

Conclusion

This study investigated the performance of Transformer models in comparison to LSTM and MLP architectures for two data sets focused on time-delayed variables. The results showed that despite efforts to optimize hyperparameters, the Transformer model was more difficult to train and tended to fluctuate during training. However, when considering only the best-performing model per training run, the loss of each model was comparably consistent. Nevertheless, the LSTM and MLP architectures outperformed the Transformer for both data sets. Moreover, an investigation of the models’ performance with decreasing sample size showed that the Transformer did not exhibit any particular advantage for small data sets compared to LSTM and MLP. Whether the performance fluctuation during training of the Transformer might cause problems for retraining models when deployed as continual learning soft sensors, has to be further investigated.

It should be noted that the results and conclusions of this study are specific to the investigated data sets and might not be generalizable to other types of data sets, featuring long time series or non-time-delayed variables.

In conclusion, for the investigated data sets with medium-range sequence length and time-delayed variables, LSTM and MLP architectures are better suited than Transformer models considering consistency, performance, and training time.

Appendix: PV-Dataset

The PV data set is a publicly available data set from a PV array located on the NIST campus in Maryland, USA Boyd et al. (2017). The array consists of 312 PV modules, which provide 73.3 kW. The data is recorded every second over a period from 2015 to 2018. Many of the 115 recorded sensor values and features in the data set are irrelevant for determining the backsheet temperature. Therefore, only 13 feature are used. Additionally, the time period is narrowed down to 2016 and the sampling rate of the data set is adjusted to match that of the temperature sensors (0.1 Hz). The timestamp of the measured values is only used to create the sequences. The feature “WindValid” is also only used for filtering. Both are not used as input features for the models. The formation of hotspots in the module is only relevant under solar irradiation, since no electricity can be produced without it. Therefore, all samples with no power at the inverter input are

Table 5 Used features and value bounds of the PV-data data set

Feature name	Description	Value-bounds
Timestamp	Date and time	Year 2016
Pyra1_Wm2 - Pyra8_Wm2	Millivolt output from pyranometer 1-8	[0, 1000]
WindDir_deg	Wind direction	–
WindSpeed_ms	Wind speed	–
WindValid	Wind sensor data validity	$\{-1\}$
WindRef_V	Internal reference voltage in wind sensor	–
AmbTemp_C	Outdoor ambient temperature	$[-30, 100]$
InvPDC_kW	Inverter DC input power, meas. by inverter	(0, 100]
RTD_C_1	Temperature of module backsheet	$[-20, 300]$

"–" indicates no filtering

dropped. Subsequently, outliers per feature are filtered out of the data set by defining lower and upper bounds, taking physical plausibility into account. As a last step all features are normalized between 0 and 1 and the number of samples is reduced to 100,000 using random sampling. An enumeration of the used features and value boundaries is shown in Table 5.

Acknowledgements
We sincerely thank SMA Solar Technology AG for providing us with an extensive data set.

About this supplement
This article has been published as part of Energy Informatics Volume 6 Supplement 1, 2023: Proceedings of the 12th DACH+ Conference on Energy Informatics 2023. The full contents of the supplement are available online at https://energyinformatics.springeropen.com/articles/supplements/volume-6-supplement-1.

Author contributions
JW: Conceptualization, methodology, software, writing, data curation, formal analysis, investigation. DA: Software, visualization, investigation, data curation. SR: Supervision, project administration, funding acquisition, reviewing

Funding
This research was funded by the German Bundesministerium für Bildung und Forschung (BMBF) Grant Number 13FH094PB8.

Availability of data and materials
The PV-data set was published by Boyd et al. (2017) and is publicly available and downloadable under https://pvdata.nist.gov. For training the models, the open source deep learning python library PyTorch was used. It is available under https://pytorch.org. For the hyperparameter tuning the open source python library Ray Tune was used. It is available under https://www.ray.io. The Transformer model from Zerveas et al. (2021) was used, who provided their framework free to user under https://github.com/gzerveas/mvts_transformer.

Declarations

Competing interests
The authors declare that they have no competing interests.

Published: 19 October 2023

References
Agrawal P, Bansal HO, Gautam AR, Mahela OP, Khan B (2022) Transformer-based time series prediction of the maximum power point for solar photovoltaic cells. Energy Sci Eng 10(9):3397–3410
Alzawaideh B, Baboli PT, Babazadeh D, Horodyvskyy S, Koprek I, Lehnhoff S (2021) Wind turbine failure prediction model using scada-based condition monitoring system. In: 2021 IEEE Madrid PowerTech, pp. 1–6
Azman SK, Isbeih YJ, El Moursi MS, Elbassioni K (2020) A unified online deep learning prediction model for small signal and transient stability. IEEE Trans Power Syst 35(6):4585–4598
Bosman LB, Leon-Salas WD, Hutzel W, Soto EA (2020) PV system predictive maintenance: challenges, current approaches, and opportunities. Energies 13(6):1398

Boyd M, Chen T, Dougherty B (2017) Nist campus photovoltaic (pv) arrays and weather station data sets. National Institute of Standards and Technology [Data Set]

Chen B, Lin P, Lai Y, Cheng S, Chen Z, Wu L (2020) Very-short-term power prediction for PV power plants using a simple and effective rcc-lstm model based on short term multivariate historical datasets. Electronics 9(2):289

Filios G, Kyriakopoulos A, Livanios S, Manolopoulos F, Nikoletseas S, Panagiotou SH, Spirakis P (2022) Data-driven soft sensing towards quality monitoring of industrial pasteurization processes. In: 2022 18th International Conference on Distributed Computing in Sensor Systems (DCOSS), pp. 167–174

Ge Z, Song Z, Ding SX, Huang B (2017) Data mining and analytics in the process industry: the role of machine learning. IEEE Access 5:20590–20616

Geng Z, Chen Z, Meng Q, Han Y (2021) Novel transformer based on gated convolutional neural network for dynamic soft sensor modeling of industrial processes. IEEE Trans Ind Inform 18(3):1521–1529

Kabugo JC, Jämsä-Jounela S-L, Schiemann R, Binder C (2020) Industry 4.0 based process data analytics platform: a waste-to-energy plant case study. Int J Electric Power Energy Syst 115:105508

Karita S, Chen N, Hayashi T, Hori T, Inaguma H, Jiang Z, Someki M, Soplin NEY, Yamamoto R, Wang X (2019) A comparative study on transformer vs RNN in speech applications. In: 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pp. 449–456

Lim B, Arık SÖ, Loeff N, Pfister T (2021) Temporal fusion transformers for interpretable multi-horizon time series forecasting. Int J Forecast 37(4):1748–1764

Maschler B, Ganssloser S, Hablizel A, Weyrich M (2021) Deep learning based soft sensors for industrial machinery. Procedia CIRP 99:662–667

Sak H, Senior AW, Beaufays F (2014) Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In: INTERSPEECH, pp. 338–342

Sun Q, Ge Z (2021) A survey on deep learning for data-driven soft sensors. IEEE Trans Ind Inform 17(9):5853–5866

Tuli S, Casale G, Jennings NR (2022) Tranad: deep transformer networks for anomaly detection in multivariate time series data. Proc VLDB Endow. 15(6):1201–1214

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. Adv Neural Inf Proces Syst. 30

Wen Q, Zhou T, Zhang C, Chen W, Ma Z, Yan J, Sun L (2022) Transformers in time series: a survey. arXiv preprint arXiv:2202.07125

Wu N, Green B, Ben X, O'Banion S (2020) Deep transformer models for time series forecasting: the influenza prevalence case. arXiv preprint arXiv:2001.08317

Zeng A, Chen M, Zhang L, Xu Q (2022) Are transformers effective for time series forecasting? arXiv preprint arXiv:2205.13504

Zerveas G, Jayaraman S, Patel D, Bhamidipaty A, Eickhoff C (2021) A transformer-based framework for multivariate time series representation learning. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 2114–2124

Zhou H, Zhang S, Peng J, Zhang S, Li J, Xiong H, Zhang W (2021) Informer: Beyond efficient transformer for long sequence time-series forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 11106–11115

Publisher's Note