Sustainable Energy, Grids and Networks 32 (2022) 100851

Contents lists available at ScienceDirect

Sustainable Energy, Grids and Networks

journal homepage: www.elsevier.com/locate/segan

# Applying Deep Learning-based concepts for the detection of device misconfigurations in power systems

David Fellner [a,*], Thomas I. Strasser [a,b], Wolfgang Kastner [b]

[a] AIT Austrian Institute of Technology, Giefinggasse 4, Vienna, 1210, Vienna, Austria
[b] TU Wien, Karlsplatz 13, Vienna, 1040, Vienna, Austria

ARTICLE INFO

*Article history:*
Received 9 December 2021
Received in revised form 31 May 2022
Accepted 5 July 2022
Available online 22 July 2022

*Keywords:*
Power distribution system
Artificial intelligence
Deep learning
Device malfunctions
Operational data
Malfunction detection

ABSTRACT

The electrical energy system is undergoing major changes due to the necessity for more sustainable energy generation and the following increased integration of novel grid-connected devices, such as inverters or electric vehicle supply equipment. To operate reliably in novel circumstances, as created by the decentralization of generation, power systems usually need grid supportive functions provided by these devices. These functions include control mechanisms such as reactive power dispatch used for voltage control or active power reduction depending on the voltage. As the main contribution of this work, an approach for the development of the detection of misconfigured (e.g., wrongly parameterized control curve) grid devices using solely operational data is proposed. By generating and analyzing operational data of power distribution grids, a Deep Learning-based approach is applied to the detection problem given. An end-to-end framework is used to synthesize and process the data as well as to apply machine learning techniques to it. The results offer insights into the applicability and possible ways to improve the proposed solution and how it could be employed by grid operators. The findings show that DL methods, in contrast to traditional machine learning, can be used for the problem at hand and that the framework developed offers the necessary tools to fine-tune and scale the solution for broader usage.



## 1. Introduction

Today, especially power distribution system operators (DSO) have to cope with new challenges arising due to the transformation of the energy system. A major shift in paradigm is the increasing penetration of decentralized power generation [1], which leads to technical challenges in the transmission and storage of power. Standing out is the impact of high photovoltaics (PV) proliferation, but also of other grid-connected devices such as electric vehicle supply equipment (EVSE) [2]. In case of generation outdoing demand locally, bidirectional power flows on different voltage levels as well as voltage rises are the consequences [3]. If the voltage is lifted too much this can lead to voltage band violations, which consist of voltages above or below the admissible limits. Control mechanisms are employed to allow for a reasonable decentralized generation of renewable energy without creating said violations. For this purpose, voltage regulation is the preferred strategy [4], which is made possible by generation units implementing grid supporting functions. These approaches target the frequency as well as the voltage amongst others. Apart from limiting the dispatch of active power, which is a possible solution in the EVSE case [5] of undervoltage, one of the most common ways to influence the voltage is via the power factor and followingly the reactive power exchanged with the network, usually controlled by a local droop control [6].

Such controls are configured, as the grid codes demand, in controllable decentralized grid-connected devices. However, they are configured once at installment and subsequently not monitored. As a result shifts in configuration, such as a reset of a control curve, can go unnoticed given the current layout of the grid's metering infrastructure and the DSOs' overall metering capacities.

Fig. 1 illustrates the functions of these reactive power controls; on the left the power factor ($\cos\phi$) is varied depending on the active power (P) dispatch, allowing for reactive power (Q) infeed, whereas the right side shows the impact of Q on the voltage (V) [7]. The active power control depending on the voltage applied to EVSEs is described in more detail later in the work.

To ensure that these grid supporting functions are actually delivered, DSOs need to monitor the operation of grid-connected devices, for instance, PV inverters or EVSE, as to be sure that the network works in a stable manner. As the available information about grid components' characteristics is often limited, a data-driven approach is a favorable option [9] for a monitoring solution

* Corresponding author.
*E-mail addresses:* David.Fellner@ait.ac.at (D. Fellner), Thomas.I.Strasser@ieee.org (T.I. Strasser), Wolfgang.Kastner@tuwien.ac.at (W. Kastner).

https://doi.org/10.1016/j.segan.2022.100851

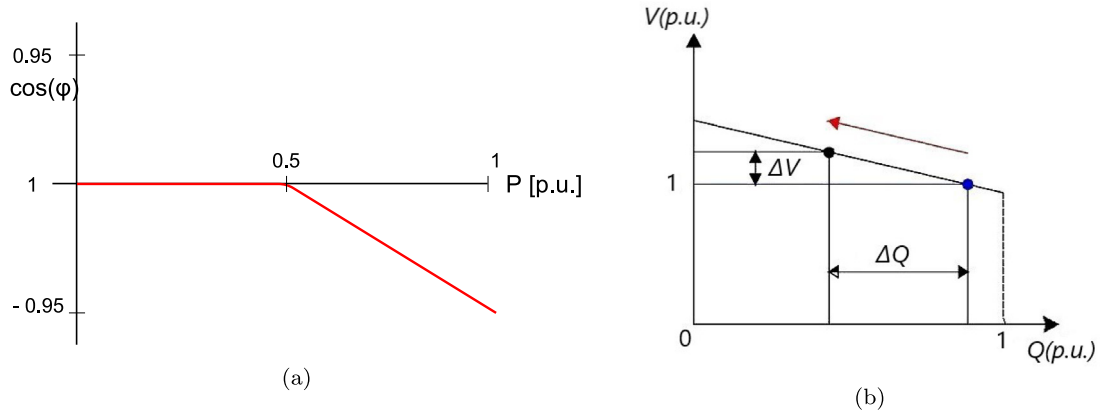


**Fig. 1.** Control schemes: (a) Q(P), (b) voltage droop [8].

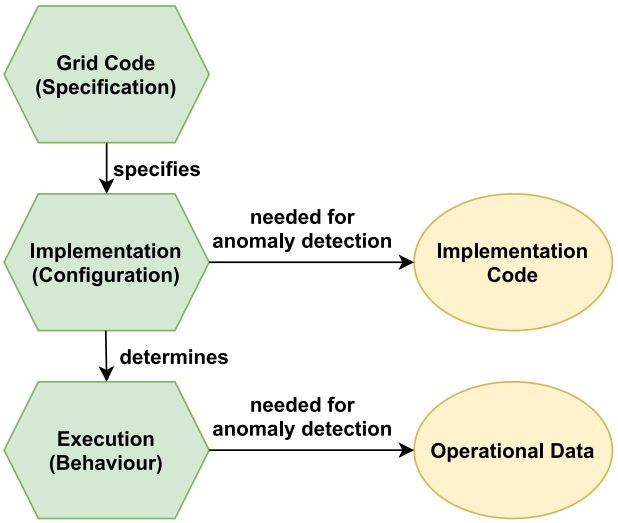


**Fig. 2.** Definitions of terms and requirements for the detection of wrong implementations (code needed) respective misconfigurations (data needed).

that is actually feasible and therefore useful to DSOs. Such a solution can be crafted in a way as to only use operational data of the grid-connected devices, in order to detect misconfigurations of the same. These deviations of configurations from the specifications – as defined by grid codes – can have two reasons; firstly, a different configuration than the normative one can be purposely implemented. Secondly, the configuration can change due to malfunctions or faults. Here misconfiguration stands for the latter meaning a deviation from previous implementation of a control curve that is assumed to be initially correct. Fig. 2 depicts how these terms are linked and what is needed to detect anomalies with respect to the type of anomaly. It becomes obvious that for the detection of involuntary misconfigurations only detection of the execution of functionalities is necessary, which does not require knowledge about an implementation code or the fundamental specification and, thus, follows a black-box approach. Therefore, only operational data is used for this purpose.

This detection of misconfigurations while only having operational data, meaning no topology information or information about the configuration other than the previous one ingrained in the data, is becoming more and more relevant as the transformation of the energy system paces on with the installment of PVs and EVSE. At the same time, more and more data at the connection points of these devices, meaning at the Smart Meters, are becoming available and could be used. However, there is no approach to this particular problem and therefore the means of developing and assessing novel ones are needed. This leads to the formulation of the condensed question to be answered:

*What approach, applying data-driven methods and algorithms solely on operational data at Smart Meter level is suited best to detect misconfigurations of functions of grid supporting devices in a low voltage distribution grid?*

To answer this question a number of objectives have to be fulfilled. These are:

- Obtain data that reflects cases of relevant misconfigurations in operational grid data.
- Assess and process this data to make it usable for the development of detection methods
- Select and apply detection methods to the data
- Pick and refine the best-suited method found.

Therefore, the main contribution of this work, which is an invited, revised, and extended version of [10], is the detailed description of an end-to-end framework that can be used to handle grid operational data and to detect misconfigurations. First, this framework is employed to either select or generate, clean and label data for further use. This is necessary since grid data in the form needed is almost impossible to obtain. DSOs have no metering in place that would yield data indicating whether a misconfiguration is present or not. The so-created datasets are then preprocessed by, for example, scaling, in order to make it fit for usage by, and training of the detection methods. Subsequently, various detection mechanisms can be applied to the data, which lastly are evaluated and compared against each other. In this work, Deep Learning (DL) approaches are under scrutiny, in addition to being benchmarked against traditional Machine Learning (ML) approaches. DL approaches are chosen for investigation because of voltage curves being highly non-linear and, therefore, features cannot be easily derived from them at a low sampling rate as the one of Smart Meter data. However, our previous work indicates a detectable impact of misconfigurations on the voltage [11]. This makes DL an interesting approach [12]. The extension over the conference version [10] consists of an extra use case concerning EVSE misconfigurations, under investigation along with a benchmarking against traditional ML methods as well as sensitivity analysis concerning the parameters of the DL models and their training.

The remaining part of this work can be summarized as follows: In Section 1 a detailed discussion of monitoring needs and issues in power distribution grids is conducted. Section 2 describes the state-of-the-art related to malfunctions in power systems as well as the usage of artificial intelligence for detecting them. In Section 3, the functionality and implementation details of the

detection framework are lined out and in Section 4 a description and results of the approaches explored using the framework are presented. Finally, Section 5 provides the discussion, conclusions, and an outlook about potential further work.

## 2. Related work

### 2.1. Classical data analysis

In the work of [13], electricity consumption data is modeled using a combination of polynomial regression and Gaussian distribution. This is done to detect anomalies in the electricity demand of several schools. This approach could be used for anomaly detection of grid-connected devices, however, the models have to be fitted individually for each device making the application less suitable for broad usage.

In [14], consumption patterns of medium voltage transformers at substations are clustered using algorithms, such as k-means and fuzzy c-means. Abnormal consumption is then identified by employing the local outlier factor (LOF) of hourly load data as a measure. Indicators such as irregular peak unusual consumption, broadest peak demand, sudden large gain, and nearly zero demand unusual consumption are used as features here. Even if not applicable to this particular problem, this shows that there are features present that allow for general detection of anomalous behavior from operational data.

[15] proposes a fault detection in microgrid using traditional machine learning approaches such as Support Vector Machine (SVM), k Nearest Neighbor (kNN), or Decision Trees (DT) in the form of Random Forests. Data of high resolution is used as well as the grid topology known. However, as only Smart Meter measurements are to be used which are only available in a low time resolution, also topology detection is not feasible [16] for the problem at hand. The low resolution and lack of topology knowledge make this approach impracticable here since features could probably not be extracted. Nevertheless, the traditional ML methods of SVM, kNN, and DT are to be tried out and used as a benchmark for other approaches.

### 2.2. Feature identification using artificial intelligence approaches

#### 2.2.1. Recurrent deep learning architectures

This can be exploited by using DL. As elaborated in [17], Recurrent Neural Networks (RNN) can be used to classify time series data; an Elman network structure is applied to classify a time series. This includes a feed-forward part and a memory part which feeds network activation's from a previous time step as inputs to the network to influence predictions at the current time step. This is achieved through back propagation through time (BPTT); here the gradient of the cost function is propagated with regard to the parameters of the network, like weight matrices, for every time point of the sequence and each layer by unfolding the recurrent connections through time [18]. The parameters are updated using the gradient in a way that minimizes the cost function. The cost function is selected according to the task, such as classification or regression [19]. For classification a cost function as the cross-entropy loss is a common choice since it yields a linear gradient structure, as does the mean squared error used for regression. This is of particular importance to avoid a vanishing gradient while back-propagating it through time [20]. Processing the input as a sequence adds a temporal dimension to the information gained and allows a more flexible window of information to be used in contrast to a feed-forward network. Here, the most frequent classification result yielded by the output neurons is used as a classification result. This might be feasible for grammar checking but might need alteration for the problem addressed in the work here. Especially because RNNs are mostly used for prediction, they have trouble with longer time-series because of a, regardless of the cost function, disappearing gradient and, additionally also due to their limited features w.r.t. parallelization [21].

The RNN approach nevertheless has some deficiencies, most prominently its lacking ability to capture long-term dependencies in sequential data, as lined out in [22]. In the Long Short-Term Memory (LSTM) RNNs recurrent hidden layers, so-called 'memory blocks' are contained; they are made of memory cells that store the network temporal state using self-connections and control the exchange of information through 'gates', which are multiplicative units. Namely, these are the input, output, and forget gates, which, respectively control the inflow or output of activation's to or from the cell or scale its internal states before using them recurrently, which can be interpreted as forgetting. This makes LSTM RNNs an interesting approach when working with longer time series. This is also due to the LSTMs ability to filter non-relevant inputs through using their gates giving it an advantage when modeling dependencies that vary over time [23, 24].

Another approach to model long-term dependencies better are Gated Recurrent Unit (GRU) RNNs; they address the same vanishing gradient issues as the LSTM approach when back-propagating the gradient of the cost function through time using a simpler structure. Only two gate types are employed by the GRU; an update gate that controls the inflow of information as well as a reset gate that decides over forgetting past information [25]. In contrast to the LSTM architecture, the GRU architecture allows for discarding of past information entirely. Still exploding gradients remain an issue, which is however tackled by gradient clipping. This makes the GRU RNN have fewer parameters in comparison to an LSTM RNN and is, therefore, more lightweight and has been observed to outperform the latter in several tasks. This is also the case for univariate time series classification, which is applicable for classification problems in power systems when, for example, only voltage data is available [26]. Because of these properties, GRU RNNs could also be interesting for a distributed application in a detection mechanism and also for frequent retraining if needed.

#### 2.2.2. Feed-forward architectures with attention mechanisms

An alternative is posed by so-called Transformer architecture [27]. Here, attention mechanisms are used that enable capturing of global dependencies between the input and output, regardless of the positions of the sample points in the time series or sequence. Here, no recurrent computation is used, allowing for better parallelization. Instead 'self-attention' is employed to reach a representation of a sequence through setting the positions of the sequence in relation . An encoder–decoder setup is used, where it performs a mapping of the input to an internal representation, which the decoder then processes to generate the output auto-regressively. The encoder and decoder both consist of feed-forward networks as well as multi-head self-attention mechanisms. This attention mechanism projects a query and key-value tuples on an output which is calculated using the weighted values. These weights are computed in turn on the query and the respective key. This yields an attention value for every query–key-value item and therefore a representation of the sequence. Multi-head attention now enables processing information from a higher dimensional query–key-value set at various positions in contrast to a single attention head, which is helpful. Additionally, positional encodings are simply added to the initial inputs to insert some hint about the positions of the points of the sequence for the feed-forward networks. This non-recurrent approach could be also a computationally interesting option.

**Table 1**
Non-functional requirements (NFR) fulfilled (X) or unfulfilled (–) by approaches in related publications cited.

| NFR | Reference | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | [14] | [13] | [15,16] | [17,21] | [22–24] | [18,25,26] | [27,29–31] | [28] |
| Scalability | – | – | – | – | – | – | X | X |
| Adaptability | – | – | – | X | – | X | X | X |
| Integrability | X | X | X | X | X | X | X | X |
| Usability | X | X | – | X – | X | X | X | X |
| Data Retention | X | – | X | X | X | X | X | X |
| Robustness | – | X | – | X | X | X | – | X |
| Quality | X | X | – | – | X | X | X | X |

#### 2.2.3. Recurrent architectures with attention mechanisms

The R-Transformer concept follows a similar idea as the aforementioned transformer approaches [28]. The main improvement proposed over the regular transformer consists in additional capturing of the sequential information in the data. This is done by positional encoding in the regular transformer, which only yields a scant impact [29], and is often limited to a certain sequence length to be able to set into a context [30]. If positional information is to be retained for a flexible sequence length in an effective manner, high efforts are required to tailor such solutions [31]. These disadvantages take their toll on the robustness of a solution built on traditional attention mechanisms. Furthermore, local structures are neglected because of the sheer number of other positions which allows only for a small signal at a local position, even if these structures might be of quite an importance. To combat these flaws, the R-Transformer uses local RNNs sliding over the sequence, applying windows of a defined length to encode the sequential information in the data and capture local structures in the time series. Thus, latent representations are generated equally for each of the windows treated by the local RNN and are not dependent on any of the other windows. Therefore, information about its local surroundings is ingrained in each data point's representation. Additionally, by sliding the RNN over the time series, the global sequentiality of the data is taken into account as well. The effect of the local RNNs can be compared to a one-dimensional convolution operation, which has the advantage of being parallelizable, but also taking into account sequential information. The gained and encoded local information of one position is then, like in the aforementioned transformer, directly connected to all other positions in the sequence through the multi-head attention mechanism. In a similar application to the one at hand (MNIST dataset with 784x1 sequences), the R-Transformer outperforms both the regular, convolutional Transformer as well as simple recurrent approaches such as LSTM and GRU, whereas an RNN performed significantly worse than all other approaches. This makes the R-Transformer an interesting approach.

### 2.3. Summary and open issues

Summarisingly, the work on anomaly detection (see Table 1) in the electrical grid domain shows that there are approaches that are not flexibly applicable to new devices or are only applicable at a transformer level or with more information or data of properties which is not available. However, the domain of DL-based approaches offers methods that are, at least in theory, well suited for developing a solution to bridge this gap. Nevertheless, no applications to this specific problem can be found in the literature, and therefore, explorations and assessments of these have to be conducted. This is done by the introduction of a novel framework allowing for generating and/or handling data that is specific to the detection problem at hand. The framework also allows for the development and assessment of detection applications, in order to set up, pick and refine data-driven methods.

## 3. Scenarios for monitoring and detection

### 3.1. Employed framework

To overcome the shortcomings of present approaches for detecting misconfigurations by the development of a new method, an environment is introduced which is able to handle different detection scenarios, grid setups, and data properties. In general, the approach to detecting devices in misconfigured states is novel in itself. This kind of framework (see Fig. 3) is used to either synthesize or clean, process, and analyze data as well as apply ML and DL methods to it. Either real-world operational grid data or data of simulations using grids that are specifically designed for simulation purposes – like the ones that form the SIMBENCH [32] project – are being used.

If operational data is to be synthesized, the grid data used is extracted from the respective files and prepared for further use in simulations, as indicated in Fig. 3 under 'Grid Data'. Those are data such as the number of connection points and the specifications of their connections and the substations as well as the consumption and dispatch of loads and generation in the grid. In this manner, the grid topology is checked and generation and load profiles, as well as control curves, are defined and handed over to the grid simulation software. This is done in the 'Grid Preparation' box. The next step is 'Grid Setup': a grid model is set up in a grid simulation software by placing elements, and adding specifications and profiles to these elements. Using simulations another plausibility and – if necessary – scaling of, for example, loads is conducted and a final grid model is yielded. This model is then used for running simulations in which parameters like the time resolution of the data synthesized, the misconfiguration of interest and its position, as well as the control curve to be monitored, can be varied, as indicated under 'Scenario Settings'. The simulation then delivers operational data of the grid including data of a malfunction, which is then labeled and saved. This is represented by the 'Simulation' box which specifies the simulation method as quasi dynamic load flow simulations, which can also be altered to be a simulation of individual load flows. These individual load flows are necessary to implement voltage-dependent controls, as the one applied to EVSEs, as described later. These voltage-dependent controls run through inner control loops in order to find an adequate setpoint for the operational state. These inner loops slow down the simulation and, therefore, the entire data generation, making it very time-consuming to collect large amounts of data in this manner. To solve this, the framework also allows for the use of so-called 'quasi dynamic simulation language models' ('QDSL models') in combination with individual load flows. These QDSL models perform the inner loops of device controls, speeding data generation up by a factor of 7. Moreover, the misconfiguration is set up and the raw load flow data of the grid simulation is exported as well as information about when and where a defined misconfiguration occurred. These results are finally used to pick relevant data such as data of connection points having a PV unit or an electric vehicle charging station, add noise to it and, therefore, create datasets. These datasets are used

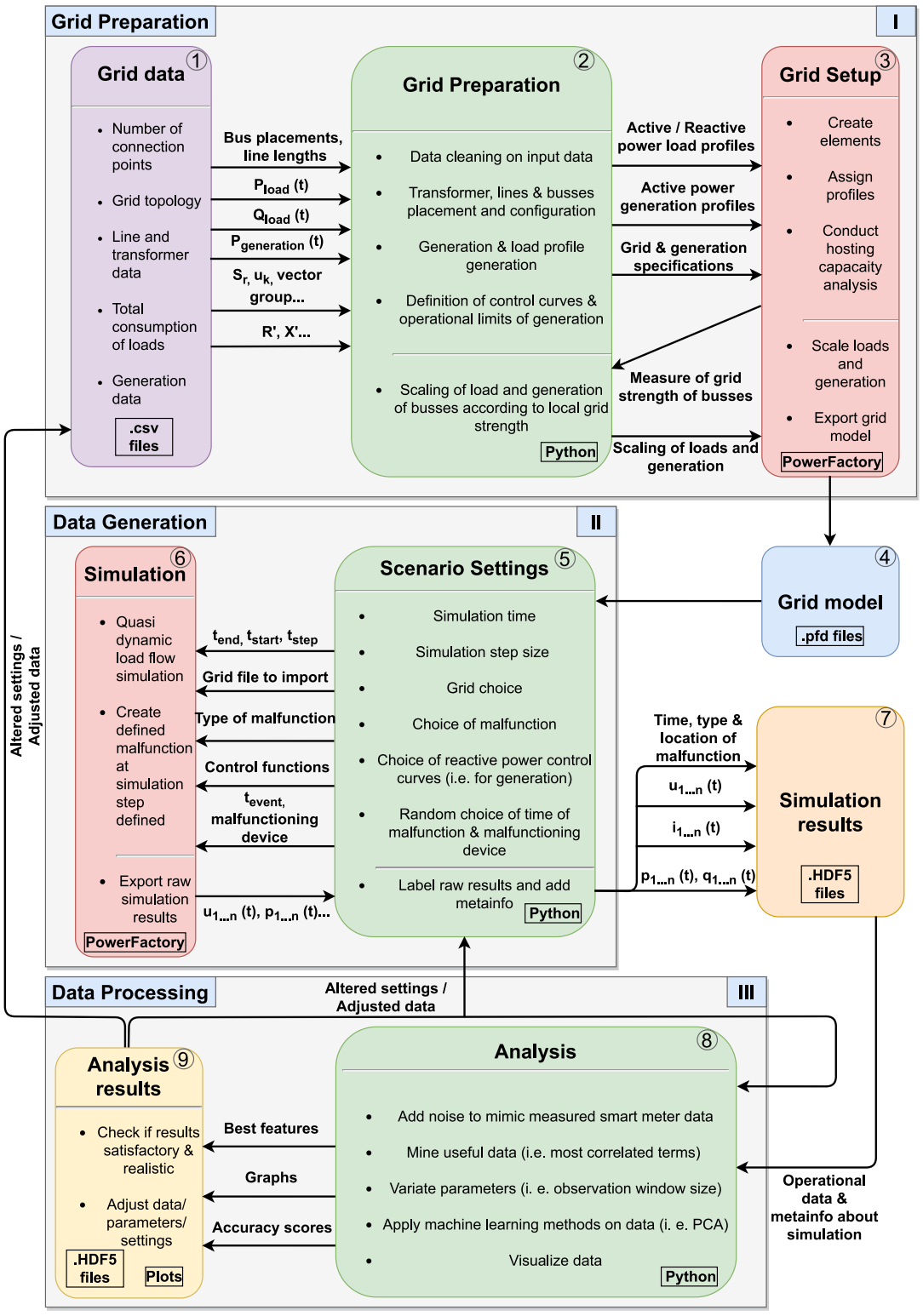


**Fig. 3.** Framework used for generation and handling of data of misconfigured devices in power grids as well as for assembling datasets using this data and applying and assessing methods and algorithms for misconfiguration detection.

to assess the applicability of machine learning detection methods, especially DL approaches in this case. This is done in the last step two steps, 'Analysis' and 'Analysis Results', of the framework; training of machine learning classifiers can be conducted as well as architecture exploration or hyperparameter optimization using grid search. The results can be used to make statements about the best-suited methods as well as to gain insights into the quality and property of the data on which the classification has been conducted.

### 3.2. Tackled scenarios

What this looks like in practice, is illustrated by the schematic of a distribution grid with household loads and PV generation in

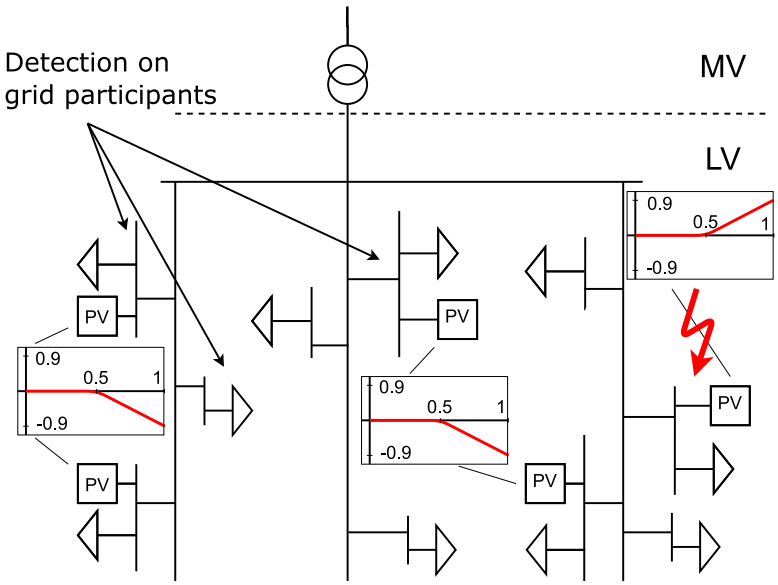


**Fig. 4.** Schematic grid used to generate data [7].

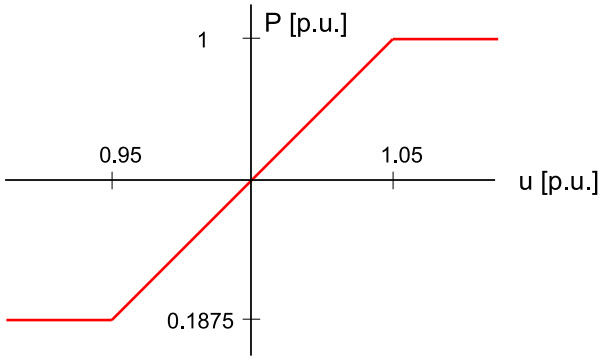


**Fig. 5.** P(U) control curve applied to EVSEs.

Fig. 4; one possible misconfiguration is shown. Here all PV inverters follow a certain control curve regarding to the power factor. As mentioned above, this is meant to help regulate the voltage in case of high active power infeed through the variation of reactive power dispatch. One of the PV units inverts its control curve, it is therefore misconfigured and the voltage is not controlled as intended anymore, which is to be detected. For PV inverters, other possible misconfigurations involve a flat control curve, which equals no control, and different maximal or minimal power factors. This allows an assessment of how grave a misconfiguration has to be found as to be detected by certain approaches.

Another misconfiguration scenario concerns the EVSEs; the control curve employed in the electric vehicle charging station simulated is a P(U) control, which is an active power control depending on the voltage. The curve used in the simulations is depicted in Fig. 5; the EVSE is charging at its rated power above a voltage of 1.05 per unit, whereas the charging power is gradually reduced if the voltage of the connection point is lower than this. At a voltage value of 0.95 per unit, this reduction is halted at the minimal charging power of 18.75% of the rated charging power. Therefore, this control should help keep voltages within limits. The misconfiguration is assumed as a flat control curve, meaning no reduction in charging power depending on the voltage level.

These misconfigurations, but also misconfigurations in other devices such as battery energy storages, are supposed to be equally detectable using this approach; being grid supporting, a change in behavior should leave a similar impact on the operational data, such as the voltage. The similarity of features should therefore make a detection possible.

The voltage at the coupling points of the loads and the grid-connected devices, such as EVSEs and PV units is recorded, for example, with a sample rate of 15 min to mimic smart meter data. This data is then turned into a dataset by creating samples of a certain sequence length, labeling the same in classes 0 (regular behavior) and 1 (misconfiguration present) as well as choosing the ratio of classes, either balanced or unbalanced to an arbitrary degree, to fit the capabilities of the methods applied later. Finally, these labeled samples are fed into a data-driven detection method to train on them and assess its performance in detecting a malfunction by recognizing the correct classes. The datasets compiled and used consist of either weekly or daily time-series sampled in 15 min intervals (i.e., common for power system applications), which leaves us with either 96 or 672 data points per sample sequence. This allows for an assessment of the impact of sequence length on the performance of the applied DL methods, which is supposed to stem from their respective handling of long-term dependencies in a sequence.

The novel data used for this work are created using up to 5 of the aforementioned SIMBENCH grids, which are either classified as rural or semi-urban since in such networks voltage issues are prevalent over current issues, making the misconfigurations described relevant in these grids. For the first scenario of a PV misconfiguration, Fig. 6 shows in two weekly time-series samples the impact left by the misconfiguration on the operational data gained, namely the voltage. The variation in voltage for class 0 ('regular behavior') is much smaller than for the malfunctioning class 1 ('malfunction/misconfiguration present'). This behavior is what is expected here since the control is implemented to keep the voltage within certain admissible limits. Therefore, this different impact of the misconfigured power factor control curve is to be detected. For this case, various datasets with up to 200,000 samples of these kinds with balanced classes, to enable proper learning of features and classification using DL [33], were split into a train and test set and used for the adaption and assessment of the DL detection approaches described in the following. Furthermore, a dataset containing 20,000 samples sourced from a single grid containing PVs and EVSEs was created to assess the applicability of the DL methods in detecting the above-described malfunction of EVSEs.

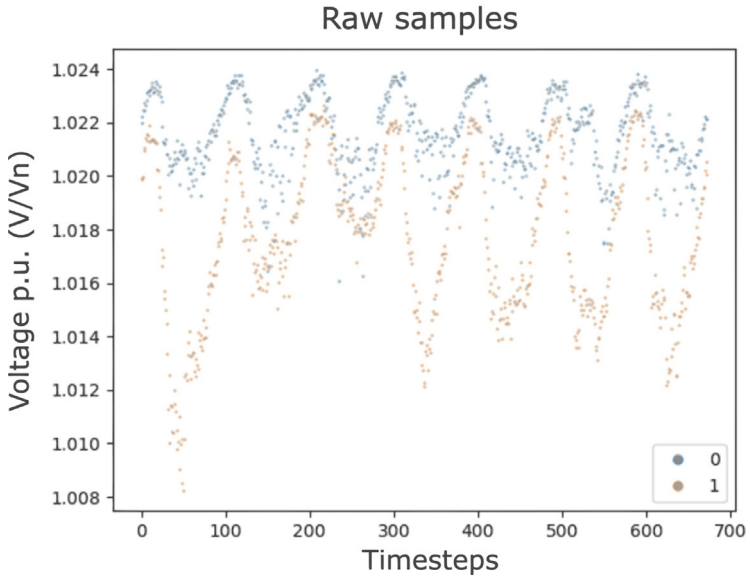


**Fig. 6.** Samples of both classes (0 (blue): regular; 1 (orange); misconfiguration present in grid connected PV device) used for Deep Learning. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

## 4. Applied learning methods and achieved results

### 4.1. Data used

The data shown above has been slightly preprocessed; before its usage in the different DL-based methods by subtracting its mean from every sample to eliminate the influence of a grid feeder-based voltage offset, as well as scaled to a range between −1 and 1. The scaler for this was fit on the training set, scaling all zero-meaned training samples between −1 and 1, and then later applied to the test set. Such samples were assembled to datasets of different sample sizes (1 day and 7 days respective 96 and 672 positions time-series length) and sample numbers (1,000, 5,000, 10,000 for preliminary analysis, and 20,000 respective 200,000 for the method comparisons). Bigger sample sizes imply more data in this case, but longer time-series might not be able to propagate back the gradients through time through the entire time-series.

### 4.2. Method implementation

For a baseline and benchmark for the DL methods, traditional machine learning algorithms were applied. Namely, these are the Support Vector Machine (SVM, NuSVM), k Nearest Neighbor (kNN) as well as Decision Trees (DT) algorithm. for all of them the implementations found in the Scitkit-learn python library were used.[1,2,3,4] All of these algorithms are supervised learning algorithms, which are applicable to the labeled data at hand. For the SVM and NuSVM, the kernels used to form the decision boundary were varied. For the DT the purity measure was varied, meaning the measure by which data is segmented into classes. For the kNN algorithm, the distance measured to the next neighboring samples was varied to either count all neighbors equally or weighted based on their distance.

As a loss criterion for the DL models, PyTorchs CrossEntropyLoss[5] is applied, which combines the LogSoftmax and negative log likelihood loss (NLLLoss). The input is expected to be the raw, untreated score of each of the two classes, as well as a class label. The CrossEntropyLoss function can be denoted as

$$loss(x, class) = -log(\frac{exp(x[class])}{\Sigma_j exp(x[j])}) = -x[class] + log(\Sigma_j exp(x[j])) \quad (1)$$

where *x[class]* denotes the output for the true target class and *j* spans across all classes, meaning that *x[j]* is the output for the *j*th class.

Fig. 7 depicts the most basic structure of the Elman network trained. There, a simple RNN with 2 layers with 6 features in the hidden states each as well as a fully connected layer with 6 neurons and 2 output neurons is presented. The output neurons obviously predict the classes 0 and 1. Each time step is fed into the network, and the output of the final time step, as it is the 'most informed' output, is used for calculating the loss and updating the weights as well as for making a classification. This approach was used during the first assessments of recurrent approaches implemented, as described in the following.

The first goal was to train at least a weak learner, meaning that the output of the classifier should be more accurate than guessing. The initial assessments described in the following were performed with regard to scenarios of misconfigured PVs; in the case of the malfunction detection task presented before, this was achieved at a sample number of 5,000 for the 1 day time-series dataset as well as for the 7 days time-series dataset. This was achieved only using data created using one grid to be able to tell if there was even enough information in the data to make a meaningful classification (i.e., for this task the F-score using the most data reached by the network was slightly over 0.5). Furthermore, a very small learning rate of $10^{-6}$ had to be chosen to reach sufficiently good results with a standard stochastic gradient (SGD) optimizer. The learning rate was controlled in a manner so as to increase the learning rate by a factor of 1.1 in case the loss between epochs diminishes, and decrease it in turn

[1] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC

[2] https://scikit-learn.org/stable/modules/generated/sklearn.svm.NuSVC.html#sklearn.svm.NuSVC

[3] https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

[4] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

[5] https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html

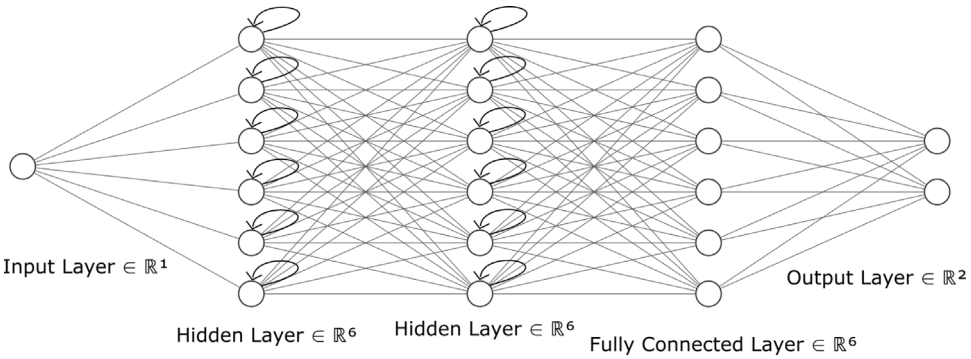


**Fig. 7.** Schematic depiction of the RNN trained and used.

by a factor of 0.9 at an increasing loss. Training was conducted for up to 100 epochs. A comparison with a linear model showed that the linear classifier did no better than guessing and therefore only reached an F-score of 0.33 on the balanced datasets. The RNN architecture put to trial here consisted of 5 RNN layers each consisting of 20 hidden units and a feed-forward layer with 20 neurons as well. Training here and in the following experiments is always conducted for 20 epochs with a learning rate of $10^{-3}$ if not stated otherwise. The RNN approach was trained using SGD and Adam optimizer on the 1-day and 7-day samples datasets, with 200,000 samples from 5 grids and 20,000 samples from 1 grid.

As a first alternative to the simple RNN structure, an LSTM RNN was tried out. The architecture used also consisted of 5 LSTM layers and a feed-forward layer with 20 hidden units, respectively neurons per layer, arranged in the same manner as for the simple RNN. An SGD optimizer was used for training.

To be able to compare the 'improved' simple recurrent approaches, for the GRU RNN, the same architecture was chosen as for the LSTM RNN. As optimizers, SGD and Adam were used when training on the same data as above. The transformer as the only non-recurrent detection approach using an attention mechanism was used with an architecture of 5 feed-forward layers with 20 neurons each. The attention mechanism constituted of a multi-head attention with one head at first. Here, an SGD optimizer was used.

Finally, the most sophisticated architecture used is the so-called R-Trans-former, following [28] which incorporates both attention mechanisms as well as recurrent and feed-forward neural networks, as lined out in Fig. 8. The multi-head attention approach allows to relate a part of a sequence to any other part of the sequence as it treats them all equally but encodes them positionally at the same time. This helps to learn global dependencies while neglecting local structures, which might also be of great interest during the course of a day. Therefore, each part of the sequence is processed beforehand by an RNN; a window of a certain number of points is slid over the sequence capturing local sequential information. In this architecture, this window had a size of 7 data points. Furthermore, the local RNNs were GRU RNNs of which 4 layers with 3 hidden units each were used. This was decided following a singular experiment conducted on the 7-day 200k dataset in which GRU reached an F-score of 0.51 after training for 47 epochs at a learning rate of $10^{-5}$, outperforming RNN and LSTM. The multi-head-attention used had one head to be able to assess the impact the recurrence has in comparison to the regular Transformer. In a first approach, only one block of stacking a local RNN, a multi-head-attention network, and a feed forward layer were used. An SGD optimizer was used.

After conducting experiments based on the initial strategy of using only the last, 'most informed' output for backpropagation as well as classification of samples, a 'majority vote' as described

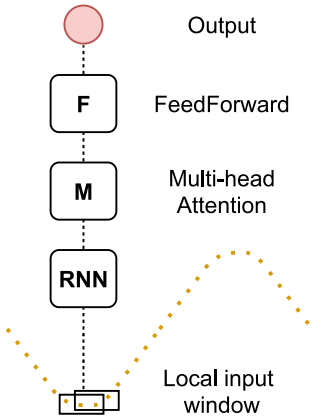


**Fig. 8.** Structure of the R-Transformer used.

in [34] was implemented. This majority vote uses the outputs of a portion of the entire sequence, or of the whole sequence, and calculates a loss depending on them. The absolute loss is then divided by the number of outputs used to have a comparable loss in all cases. This also allows for an evaluation of how many outputs should be used ideally to perform the majority vote. This can be done as a hyperparameter optimization, performed, for example, as a grid search.

### 4.3. Achieved results and discussion

The code used to produce the datasets and results can be found in the corresponding GitHub repository.[6] [For the comparison of the methods as a main] result metric the expressive F-score was used, which combines and balances Precision (i.e., how many of the found misconfigurations are actually ones), and Recall (i.e., how many of the misconfigurations present have been found). This allows a quick understanding of how helpful a result is to a grid operator since a DSO wants to balance between false alarms and finding all occurrences.

To provide a baseline, traditional machine learning algorithms were applied. Table 2 gives an overview of the methods applied as well as their parameters and the results yielded. The depicted results apply to the dataset containing data of PV misconfigurations. This assessment was conducted to provide a baseline and serve as a benchmark and additional justification of DL approaches in this case. All experiments were run applying 3-fold cross-validation.

As this assessment makes clear, various common traditional machine learning algorithms fail in delivering meaningful results, even if parameters are varied to optimize their performance.

[6] https://github.com/DavidFellner/Malfunctions-in-LV-grid-dataset

**Table 2**
Overview of the results found when detecting a PV misconfiguration using traditional machine learning methods.

| Model | Decision Trees | | kNN | | SVM & NuSVM |
|---|---|---|---|---|---|
| Parameter varied | Impurity measure | | Distance measure | | Kernel |
| 1 day-dataset (sequence length: 96) | Entropy | Gini | Uniform | Distance | Linear, Sigmoid, RBF Polynomial (degree 2–6) |
| Better than guessing (better than linear model) | No | No | No | No | No |

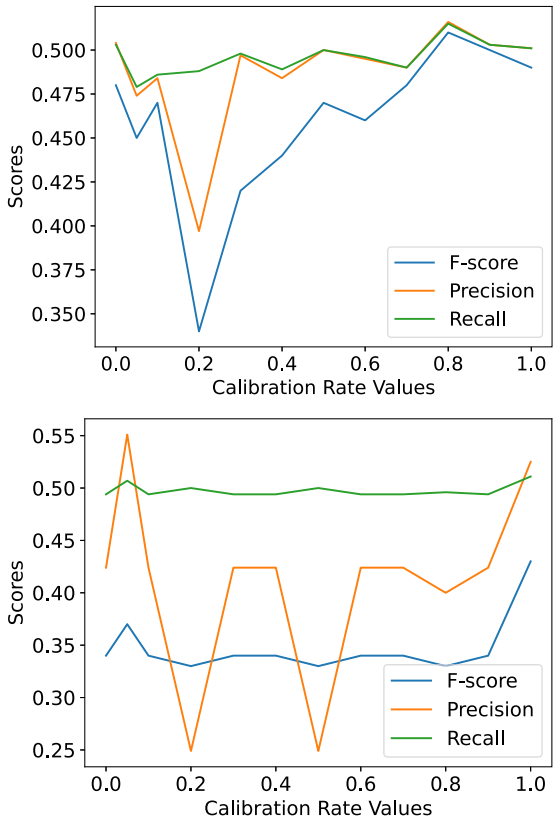


**Fig. 9.** Grid search to assess the performance of the majority vote classification; top: RTransformer, bottom: LSTM.

The aforementioned majority vote classification was assessed using a grid search hyperparameter optimization. As can be seen in Fig. 9, the so-called 'calibration rate' was varied for this purpose: this rate determines what portion of the sequence, meaning how many of the first data points of the sample processed, are used for calibration. The outputs of these first data points are not used for the majority vote classification. This means that a calibration rate of 0.8 corresponds to the last 20 percent of the sequence's outputs being used for the classification. A calibration rate of 1 corresponds to using only the last 'most informed' output for classification. On the left side of the figure, we can see the performance of the R-Transformer architecture, whereas the right side depicts the score of the LSTM architecture as described before. The dataset used consists of 20,000 one-day samples, which are samples with 96 data points, sourced from a single grid containing only loads and PVs. Therefore, the misconfiguration under scrutiny here concerns a PV unit's control curve.

The results of the assessment , only using the last, 'most informed' output for classification, for the small dataset sourced from 1 grid as well as the big dataset collected from 5 grids when detecting a PV misconfiguration are summarized in Table 3.

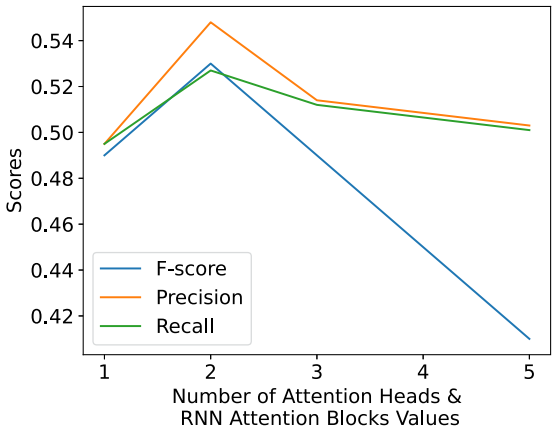


**Fig. 10.** Hyperparameter optimization done on the number of Attention Heads of the R-Transformer.

This is done for a setting with a PV proliferation of 25 percent, meaning every fourth load has a photovoltaic installation. In this context, a Weak Learner is performing better than the linear model which only guesses and therefore reaches an F-score of 0.33. The results achieved here are not good enough for actual usage, however, they provide a good orientation for further refinement of methods.

For the EVSE misconfiguration a less encompassing assessment was conducted; using data sourced from one grid with a PV and EVSE proliferation of 25 percent each, meaning every fourth load has solar generation and/or an electric vehicle charging station a dataset of 20,000 7-day samples was assembled. This dataset comprises, in contrast to the datasets used thus far, of samples of data of EV charging stations that are either misconfigured or in a regular state. Once again, only the last output is used for classification. The performances of the various methods applied to detect this EVSE misconfiguration are displayed in Table 4: once more the results are not satisfactory for a final solution but provide a guideline for further research on refined methods.

After these assessments, a first phase of hyperparameter optimization was conducted on the dataset containing PV misconfigurations of 1 grid with a sample length of one day. As the R-Transformer architecture was found to be the best fit for this application, it was also the one chosen to be tuned for better performance. Amongst others, a grid search on the number of Attention Heads was conducted. The number of Attention Heads for the Transformer as well as the number of underlying RNN Attention Blocks were varied, either separately or on par with one another. A model dimension of 30 was chosen to accommodate a higher number of Attention Heads or Blocks. As the joined adjustment of the number of blocks showed the best results, Fig. 10 shows the results of this assessment; the best number of heads was found to be 2 for both the Attention heads as well as the RNN Attention Blocks, yielding an F-Score of 0.53, which is a 4% improvement over the base configuration, which was setting the parameter to 1.

**Table 3**
Overview of the results found when detecting a PV misconfiguration using different sequence length, dataset sizes and classifiers: the F-score balances Precision and Recall.

| Model | RNN | | LSTM RNN | | GRU RNN | | Transformer | | R-Transformer | |
|---|---|---|---|---|---|---|---|---|---|---|
| Setup<br>#grids & #samples | 1 grid<br>20 k | 5 grids<br>200 k | 1 grid<br>20 k | 5 grids<br>200 k | 1 grid<br>20 k | 5 grids<br>200 k | 1 grid<br>20 k | 5 grids<br>200 k | 1 grid<br>20 k | 5 grids<br>200 k |
| F-score 1 day-dataset<br>(sequence length: 96) | 0.33 | 0.33 | 0.34 | 0.33 | 0.47 | 0.33 | 0.33 | 0.33 | 0.49 | 0.47 |
| F-score 7 day-dataset<br>(sequence length: 672) | 0.33 | 0.33 | 0.37 | 0.33 | 0.39 | 0.33 | 0.33 | 0.33 | 0.52 | 0.51 |
| Weak Learner<br>(better than linear model) | No | No | Yes | No | Yes | No | No | No | Yes | Yes |

**Table 4**
Overview of the results found when detecting a EVSE misconfiguration using different classifiers on a single dataset: the F-score balances Precision and Recall..

| Model | RNN | LSTM RNN | GRU RNN | Transformer | R-Transformer |
|---|---|---|---|---|---|
| Setup<br>#grids & #samples | 1 grid<br>20 k | 1 grid<br>20 k | 1 grid<br>20 k | 1 grid<br>20 k | 1 grid<br>20 k |
| F-score 7 day-dataset<br>(sequence length: 672) | 0.20 | 0.27 | 0.47 | 0.47 | 0.46 |
| Weak Learner<br>(better than linear model) | No | No | Yes | Yes | Yes |

**Table 5**
Overview over approaches investigated.

| Approach | Task | Comment |
|---|---|---|
| Most-informed output | Classification strategy | Best option in general for classification tasks as is yields the best scores overall |
| Majority vote | Classification strategy | Can offer an alternative classification method for specific goals i.e. avoiding false alarms |
| RNN | PV misconfiguration | Not able to extract features therefore, not better than guessing |
| | EVSE misconfiguration | Mislearning features, leading to even more misclassifications than through guessing |
| LSTM RNN | PV misconfiguration | Only partly able to extract features; slightly better than guessing in simple scenario |
| | EVSE misconfiguration | Mislearning features, leading to slightly more misclassifications than through guessing |
| GRU RNN | PV misconfiguration | Able to extract features making it better than guessing in simple scenario |
| | EVSE misconfiguration | Well able to extract features making it one of the best solutions |
| Transformer | PV misconfiguration | Not able to extract features therefore, not better than guessing |
| | EVSE misconfiguration | Well able to extract features making it one of the best solutions |
| R-Transformer | PV misconfiguration | Well able to extract features making it the best solution in all scenarios |
| | EVSE misconfiguration | Well able to extract features making it one of the best solutions |

Based on the results of this first round of tuning, another round was conducted on the R-Transformer. This time the parameter Key Size of the underlying RNN blocks of the transformer was found to improve performance at a certain setting. The Key Size determines the length of the sequence that is processed by the underlying RNN. Fig. 11 depicts the outcome of this exploration; a Key Size of 8, instead of 7, allows for an F-score of 0.60 which is another 7% increase in performance compared to the first phase of tuning and a total of 11% enhancement over the base case.

These efforts on tuning show that the performance of the solutions can be augmented by extensive architecture exploration. This is to be done for every architecture for a specific use case, however, there are no additional hurdles except for increased computational demand.

## 5. Conclusions

### 5.1. Achievements

As the necessary integration of decentralized renewable energy generation and other newly introduced grid-connected devices proceeds, grid operators need novel ways to monitor the functionalities of these generation units and devices provide. They are crucial to the safe and reliable operation of power distribution grids. Thus, the framework described in this work allows for the development of such monitoring capabilities by extracting and handling data as well as using them for the development and assessment of machine learning methods for this purpose. By its implementation and usage to generate data the first two goals

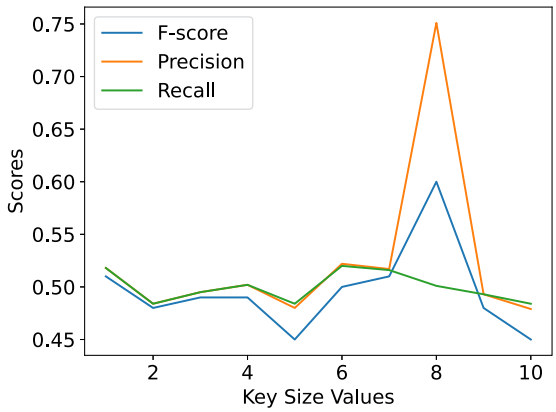


**Fig. 11.** Hyperparameter optimization done on the key size of the RNN blocks of the R-Transformer.

set initially were fulfilled. Several traditional ML, as well as DL-based approaches, have been described and compared in varying settings. In combination with the sensitivity analysis used to find a best-fitting solution, the two remaining objectives were tackled.

### 5.2. Discussion & conclusion

The initial assessment of traditional machine learning methods for anomaly detection did not yield results pointing to the applicability of the same. Even after parameters of various methods have been varied conducting a sensitivity analysis in order to provide a meaningful benchmark, not satisfactory results could be achieved. This can be attributed to the low dimensionality of the data. Even the great amount of data does not allow the traditional machine learning algorithm to succeed here. This leaves DL methods to be explored as they pose the most promising option.

The assessment of the majority vote classification versus using only the last 'most informed' output for classification offers various findings. Generally, using only the last output, or only the last portion of the sequence, stably appears to give better results than using a major part of the sequence for the classification of PV misconfigurations. However, this overall advantage is mainly rooted in a higher F-score achieved, meaning that the methods can be fine-tuned by choosing a certain calibration rate to fulfill specific requirements: depending on the priorities of the user, a certain share of the sequence can be used for classification allowing for higher precision, in cases where no false positives are wanted, or also a higher recall, in cases when all occurrences of misconfigurations are to be found.

The results of the assessment of the different methods using only the 'most informed' last output lead to the following conclusions: The RNN approach presented already demonstrates the applicability of DL for this task. This quite simple approach already yielded a weak learner for the 1-grid case, that can be extended to an ensemble method or be replaced by more sophisticated algorithms and network structures. Nevertheless, training had to be conducted with a very low learning rate and for a long time. When only trained for fewer epochs and with a higher learning rate, the RNN cannot tackle the problem and does no better than the linear model on the PV misconfiguration tasks. The RNN shows even worse performance for the EVSE case, seeming to misclassify samples. This could be due to the RNN learning wrong features, leading to indicating the improper class and showing that the RNN is not up to this task.

The LSTM and GRU RNN approaches both provide an improvement in the PV case, both yielding a weak learner for the 1-grid case. This shows that training can be done much faster with these approaches than with the simple RNN, probably because of the better back propagation of gradients through time. The GRU RNN performed significantly better than the LSTM RNN especially in the 1-day case, making it the more efficient structure. Therefore, GRU was chosen as the local RNN for the R-Transformer. Both approaches failed to provide a meaningful result on the dataset sourced from multiple grids. When put to the task of classifying the EVSE misconfiguration, the LSTM shows similar behavior as the RNN; it appears to fail to properly extract features and does not yield a weak learner within the given training frame. The GRU performs significantly better here, even yielding some of the best results in this setup. This could be attributed to the GRU's capability to discard past information, which is not of value anymore, more easily in comparison to the LSTM. Moreover, the GRU has fewer parameters than the LSTM. This might leave the GRU less confused after a shorter period of training.

The Transformer as the sole fully non-recurrent method showed that in the setting chosen feed-forward-only architectures do not yield satisfactory results as neither in the 1-grid nor in the 5-grid setup the linear model could be outperformed. At least this holds true for the PV case. In the EVSE case, the Transformer architecture yields, along with the GRU model, the most promising results. This might be due to the less sequential character of the features to be learned in the EVSE case in contrast to the PV case. Therefore, for this case, also non-recurrent approaches seem applicable.

The R-Transformer posed the most complex approach under scrutiny, which also yielded the best results for the 1 grid 20k samples dataset, remarkably showing better performance on the 7-day data for classification of a PV misconfiguration. This marks the impact the attention mechanism has as it improves the handling of longer sequences in comparison to the other recurrent approaches. Comparing tthe reults of the feed-forward Transformer the advantage of using the local GRU RNN becomes obvious as the R-Transformer manages to provide meaningful classification. Especially on the 200k samples dataset from 5 grids the combination of these two features shows its strength as the R-Transformer is the only architecture that manages to gain traction in this setup and yield a weak learner. The performance is slightly higher for the smaller dataset though, probably due to the simple network architecture used and a resulting lack in capacity. A similar phenomenon might become obvious when applying the R-Transformer on the EVSE case; the results are slightly worse than for the regular Transformer as well as for the simpler GRU architecture. This could be attributed to the complexity of the R-Transformer. Since it has many parameters, it might not be able to learn within the training time given. Even if this complexity might not be needed here the R-Transformer yields at least a viable solution to the problem. Moreover, the results of the hyperparameter tuning for one use case showed that the performance of the R-Transformer can be increased significantly in this way. As the basic way of conducting such hyperparameter tuning is the same for all the use cases, it could be extended to all the other use cases. This would have to be part of a study focused on this specific problem, as computational resources are limited in the one at hand. For the practical application of the solution, this should not be a problem, as the architectures that are found to be optimal for a certain use case only have to be trained once. Table 5 summarizes the approaches investigated as well as their assessment.

The study conducted shows how the framework can be utilized to explore methods, which lead in this case to the finding that the R-Transformer generally outperformed its competitors, which however still provided mostly functional solutions. Moreover, the applicability of solutions might differ between use cases. Additionally, the framework offers easy-to-use functionalities of tuning the architectures to obtain better performances, given the required computational power which was a limiting factor here.

### 5.3. Outlook

The presented work is a foundation for a future decision support tool for power grid operators which helps them to implement central monitoring of low voltage grids using DL detection approaches. Further work includes extensive architecture exploration in order to find the best fitting approach and an optimal model thereof for the tasks at hand. This architecture exploration was only conducted partly here since the computational resources available were limited. For a practical application, this would be no hurdle since the optimal architecture for a certain application only needs to be determined once, and only models with the best-suited parameters need to be trained then. When such models are found, a field trial in real-world grids for validation and further refinement of the method can be conducted. The sole availability of simulated data for this study can be understood as another limitation at this point. Furthermore, the range of use cases is to be expanded by training models on data of malfunctioning devices such as battery energy storage or heat pumps, which could also not be implemented yet due to the limitations of computational power already mentioned before. This would then lead to an implementation in said decision support tool and therefore integration into a grid operators toolbox for further monitoring capabilities.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

This work received funding from the Austrian Research Promotion Agency (FFG) under the "Research Partnerships – Industrial Ph.D. Program" in DeMaDs (FFG No. 879017). Furthermore, this paper is an extended version of a contribution [10] to the 2021 International Conference on Smart Energy Systems and Technologies (SEST).

## References

[1] E. Brown, J. Cloke, J. Harrison, Governance, Decentralisation and Energy: A Critical Review of the Key Issues, Loughborough University, 2015.
[2] C. Dharmakeerthi, N. Mithulananthan, T. Saha, Impact of electric vehicle fast charging on power system voltage stability, Int. J. Electr. Power Energy Syst. 57 (2014) 241–249, http://dx.doi.org/10.1016/j.ijepes.2013.12.005.
[3] J. Von Appen, M. Braun, T. Stetz, et al., Time in the sun: The challenge of high pv penetration in the german electric grid, IEEE Power Energy Mag. 11 (2) (2013) 55–64.
[4] N. Mahmud, A. Zahedi, Review of control strategies for voltage regulation of the smart distribution network with high penetration of renewable distributed generation, Renew. Sustain. Energy Rev. 64 (2016) 582–595.
[5] L. Wang, Z. Qin, T. Slangen, P. Bauer, T. Wijk, Grid impact of electric vehicle fast charging stations: Trends, standards, issues and mitigation measures - an overview, IEEE Open J. Power Electron. PP (2021) 1, http://dx.doi.org/10.1109/OJPEL.2021.3054601.
[6] P.P. Vergara, T.T. Mai, A. Burstein, P.H. Nguyen, Feasibility and performance assessment of commercial pv inverters operating with droop control for providing voltage support services, in: 2019 IEEE PES Innov. Smart Grid Techn. Europe, ISGT-Europe, 2019, pp. 1–5.
[7] D. Fellner, Data driven detection of malfunctions in power systems, in: Proceedings 9th DACH+ Conference on Energy Informatics, 2020.
[8] S. Conti, R. Nicolosi, S. Rizzo, H. Zeineldin, Optimal dispatching of distributed generators and storage systems for mv islanded microgrids, IEEE Trans. Power Deliv. 27 (2012) 1243–1251.
[9] W. Luan, J. Peng, M. Maras, J. Lo, B. Harapnuk, Smart meter data analytics for distribution network connectivity verification, IEEE Trans. Smart Grid 6 (4) (2015) 1964–1971.
[10] D. Fellner, T.I. Strasser, W. Kastner, Detection of misconfigurations in power distribution grids using deep learning, in: 2021 International Conference on Smart Energy Systems and Technologies, SEST, 2021, pp. 1–6, http://dx.doi.org/10.1109/SEST50973.2021.9543350.
[11] D. Fellner, H. Brunner, T. Strasser, W. Kastner, Towards data-driven malfunctioning detection in public and industrial power grids, in: 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, 2020, pp. 1–4.
[12] N. Mehdiyev, J. Lahann, A. Emrich, D. Enke, P. Fettke, P. Loos, Time series classification using deep learning for process planning: A case from the process industry, Procedia Comput. Sci. 114 (2017) 242–249.
[13] W. Cui, H. Wang, A new anomaly detection system for school electricity consumption data, Information 8 (2017) 151.
[14] D.D. Sharma, S. Singh, J. Lin, et al., Identification and characterization of irregular consumptions of load data, J. Mod. Power Syst. Clean Energy 5 (3) (2017) 465–477.
[15] C. Cepeda, C. Orozco-Henao, W. Percybrooks, J. Pulgarín-Rivera, O. Montoya, W. Gil-González, J. Vélez, Intelligent fault detection system for microgrids, Energies 13 (2020) https://www.mdpi.com/1996-1073/13/5/1223.
[16] G. Cavraro, R. Arghandeh, A. Meier, K. Poolla, Data-driven approach for distribution network topology detection, 2015, CoRR. abs/1504.00724 http://arxiv.org/abs/1504.00724.
[17] M. Hüsken, P. Stagge, Recurrent neural networks for time series classification, Neurocomputing 50 (2003) 223–235, http://dx.doi.org/10.1016/S0925-2312(01)00706-8.
[18] S. Kanai, Y. Fujiwara, S. Iwamura, Preventing gradient explosions in gated recurrent units, in: Advances in Neural Information Processing Systems, Vol. 30, Curran Associates, Inc., 2017.
[19] J. Chen, L. Cheng, X. Yang, J. Liang, B. Quan, S. Li, Joint learning with both classification and regression models for age prediction, J. Phys. Conf. Ser. 1168 (2019) 032016, http://dx.doi.org/10.1088/1742-6596/1168/3/032016.
[20] R. Shoham, H. Permuter, Amended cross entropy cost: Framework for explicit diversity encouragement, 2020, arXiv:2007.08140.
[21] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, P.-A. Muller, Deep learning for time series classification: a review, Data Min. Knowl. Discov. 33 (4) (2019) 917–963, http://dx.doi.org/10.1007/s10618-019-00619-1.
[22] H. Sak, A. Senior, F. Beaufays, Long short-term memory recurrent neural network architectures for large scale acoustic modeling, in: INTERSPEECH, 2014, pp. 338–342.
[23] B. Lindemann, T. Müller, H. Vietz, N. Jazdi, M. Weyrich, A survey on long short-term memory networks for time series prediction, Proc. CIRP 99 (2021) 650–655, http://dx.doi.org/10.1016/j.procir.2021.03.088, 14th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 15-17 2020.
[24] Y. Wang, S. Zhu, C. Li, Research on multistep time series prediction based on lstm, in: 2019 3rd International Conference on Electronic Information Technology and Computer Engineering, EITCE, 2019, pp. 1155–1159, http://dx.doi.org/10.1109/EITCE47263.2019.9095044.
[25] B.C. Mateus, M. Mendes, J.T. Farinha, R. Assis, A.M. Cardoso, Comparing lstm and gru models to predict the condition of a pulp paper press, Energies 14 (21) (2021) https://www.mdpi.com/1996-1073/14/21/6958.
[26] N. Elsayed, A.S. Maida, M. Bayoumi, Gated recurrent neural networks empirical utilization for time series classification, in: 2019 International Conference on Internet of Things (IThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, in: Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 2019, pp. 1207–1210, http://dx.doi.org/10.1109/iThings/GreenCom/CPSCom/SmartData.2019.00202.
[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, 2017, CoRR abs/1706.03762 arXiv:1706.03762.
[28] Z. Wang, Y. Ma, Z. Liu, J. Tang, R-transformer: Recurrent neural network enhanced transformer, 2019, arXiv preprint arXiv:1907.05572.
[29] M. Dehghani, S. Gouws, O. Vinyals, J. Uszkoreit, L. Kaiser, [Universal transformers], 2018, CoRR abs/1807.03819 arXiv:1807.03819.
[30] R. Al-Rfou, D. Choe, N. Constant, M. Guo, L. Jones, Character-level language modeling with deeper self-attention, 2018, CoRR abs/1808.04444 arXiv:1808.04444.
[31] Z. Dai, Z. Yang, Y. Yang, J.G. Carbonell, Q.V. Le, R. Salakhutdinov, Transformer-xl: Attentive language models beyond a fixed-length context, 2019, CoRR abs/1901.02860 arXiv:1901.02860.
[32] S. Meinecke, et al., Simbench—a benchmark dataset of electric power systems to compare innovative solutions based on power flow analysis, Energies 13 (12) (2020) 3290.
[33] B. Krawczyk, Learning from imbalanced data: open challenges and future directions, Prog. Artif. Intell. 5 (4) (2016).
[34] S. Satapathy, A. Jagadev, S. Dehuri, Weighted majority voting based ensemble of classifiers using different machine learning techniques for classification of eeg signal to detect epileptic seizure, Informatica 41 (2017) 99–110.