Solar Energy Advances 3 (2023) 100033

Contents lists available at ScienceDirect

Solar Energy Advances

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

# Fault detective: Automatic fault-detection for solar thermal systems based on artificial intelligence

Lukas Feierl [a,*], Viktor Unterberger [b], Claudio Rossi [c], Bernhard Gerardts [a], Manuel Gaetani [c]

[a] SOLID Solar Energy Systems GmbH, Graz, Austria
[b] BEST Bioenergy and Sustainable Technologies GmbH, Graz, Austria
[c] Links Foundation, Torino, Italy

ARTICLE INFO

*Keywords:*
Solar thermal
Fault detection
Anomaly detection
Machine learning
Artificial intelligence

ABSTRACT

Fault-Detection (FD) is essential to ensure the performance of solar thermal systems. However, manually analyzing the system can be time-consuming, error-prone, and requires extensive domain knowledge. On the other hand, existing FD algorithms are often too complicated to set up, limited to specific system layouts, or have only limited fault coverage. Hence, a new FD algorithm called *Fault-Detective* is presented in this paper, which is purely data-driven and can be applied to a wide range of system layouts with minimal configuration effort. It automatically identifies correlated sensors and models their behavior using Random-Forest-Regression. Faults are then detected by comparing predicted and measured values.

The algorithm is tested using data from three large-scale solar thermal systems to evaluate its applicability and performance. The results are compared to manual fault detection performed by a domain expert. The evaluation shows that *Fault-Detective* can successfully identify correlated sensors and model their behavior well, resulting in coefficient-of-determination scores between $R^2$=0.91 and $R^2$=1.00. In addition, all faults detected by the domain experts were correctly spotted by *Fault-Detective.* The algorithm even identified some faults that the experts missed. However, the use of *Fault-Detective* is limited by the low precision score of 30% when monitoring temperature sensors. The reason for this is a high number of false alarms raised due to anomalies (e.g., consecutive days with bad weather) instead of faults. Nevertheless, the algorithm shows promising results for monitoring the thermal power of the systems, with an average precision score of 91%.

## 1. Introduction

Although heat accounts for approximately 50% of the global energy demand, its production is still primarily dominated by fossil fuels [27]. Hence, renewable alternatives are desperately needed to reduce carbon emissions and reach the climate goals. One technology that may play a vital role in this process is solar thermal energy, which can provide renewable heat at stable prices. Therefore, more and more solar thermal systems have been installed in recent decades [27]. However, reliable monitoring is needed to ensure the performance of these systems over their long lifetime of around 25 years.

To do so, plant operators frequently analyze the monitoring data to check for any unusual behavior and react to faults quickly. However, with an increasing number of sensors and systems that need to be monitored, it is not easy to perform this task manually. For example, the datasets used within this work contain about 100 different sensors per plant that need to be checked by the monitoring personnel. This assessment must be done frequently and fast to spot problems in time while keeping the monitoring affordable. However, considerable effort is needed to interpret the complex (nonlinear, time-dependent, multivariate) solar thermal data. Hence, manually performing monitoring can be time-consuming, error-prone (as some faults might be missed out), and requires extensive knowledge about the system. As a result, there is a high potential for automatic fault detection approaches to support the monitoring personnel and speed up their work.

The topic of fault detection (FD) has been studied for several decades. Some FD algorithms for solar thermal applications have been introduced, as summarized by [8] and more recently by [12]. However, these existing methods often lack at least one of the following attributes (see Section 2 for a more detailed analysis):

**Flexibility:** Solar thermal systems are often explicitly designed to meet the needs of their customers, which leads to a wide range of unique system layouts. Thus, FD algorithms must also be very flexible to be applied to a multiplicity of different systems. However, many methods require specific measurement devices or monitoring conditions or are specifically designed for one system only.

* Corresponding author.
*E-mail address:* l.feierl@solid.at (L. Feierl).

https://doi.org/10.1016/j.seja.2023.100033
Received 31 August 2022; Received in revised form 4 December 2022; Accepted 9 January 2023
Available online 10 January 2023

| Name | Type | Year | Flexibility | Easy Configuration | Fault-Coverage | Validated |
|---|---|---|---|---|---|---|
| FUKS | Expert-System | 1995 | 0 | 1 | 1 | 1 |
| Sun et al | Expert-System | 1995 | 0 | 1 | 1 | 1 |
| IP-Solar | Expert-System | 2009 | 0 | 1 | 1 | 1 |
| InSun | Expert-System | 2015 | 0 | 1 | 1 | 1 |
| Methodiqa | Expert-System | 2016 | 1 | 1 | 1 | 1 |
| FeDet (Kassel) | Expert-System | 2017 | 1 | 1 | 2 | 2 |
| Solar-Check (Kassel) | Expert-System | 2021 | 1 | 1 | 2 | 2 |
| TRNSYS-Simulation | Parity Space | 2013 | 2 | 0 | 1 | 2 |
| ISFH IOC | Parity Space | 2006 | 1 | 1 | 1 | 2 |
| Performance Check (GRS) | Parity Space | 2020 | 2 | 1 | 0 | 2 |
| D-CAT | Parity Space | 2020 | 1 | 0 | 0 | 1 |
| ISTT | Parity Space | 1999 | 2 | 0 | 1 | 0 |
| FSC (Kassel) | Parity Space | 2021 | 0 | 1 | 0 | 0 |
| Kalogirou et al. | Identification | 2014 | 1 | 0 | 1 | 0 |
| Grossenbacher | Identification | 2003 | 1 | 0 | 1 | 1 |
| Ferreiro Garcia | Identification | 2014 | 1 | 1 | 1 | 1 |
| Correa-Jullian et al | Identification | 2019 | 2 | 1 | 2 | 0 |
| Timma et Blumberga | Identification | 2013 | 2 | 1 | 2 | 0 |
| Jiang et al | Classifier | 2019 | 2 | 0 | 1 | 1 |
| Ruiz-Moreno et al. | Classifier | 2022 | 0 | 0 | 1 | 0 |
| He et al. | Classifier | 2012 | 2 | 0 | 2 | 1 |
| Fault-Detective | Identification | 2022 | 2 | 2 | 2 | 2 |

Legend

2 fully satisfied
1 partly satisfied
0 not satisfied

**Fig. 1.** Comparison of related work for Fault-Detection and Diagnosis for solar thermal systems. Each work is categorized by the type of FDD approach based on [47] and rated in terms of *Flexibility, Easy Configuration, High Fault-Coverage,* and *Extensive Validation* based on the authors' opinions.

**Easy-Configuration:** It is not always easy to integrate FD methods into the active monitoring of a system. For example, it might be required to adapt the algorithm to a particular system design, set hyperparameters correctly, or perform simulations beforehand. This setup typically requires detailed knowledge about the system, system control, and fault detection method. As a result, some algorithms might require too much time and effort to integrate them into monitoring.

**High fault-coverage:** In addition, algorithms must be able to detect faults both reliably and fast. However, some methods are too inaccurate such that faults are discovered too late or only allow for detecting some particular faults. On the other hand, false positives also need to be prevented, as false alarms may generate considerable overhead for the monitoring personnel.

**Extensive validation:** Finally, some fault detection methods are only validated using simulation data. For example, validation data might get created by modeling components in "faulty" and "fault-free" states. However, this does not guarantee that the algorithm will work flawlessly with actual measurement data. That is because measurement uncertainties and unmodeled influences are not considered and might introduce false alarms.

A promising attempt to deal with these issues is to use artificial intelligence. Using machine learning techniques, required information can be extracted automatically from the available sensor data, and algorithms can be trained to model the system's behavior. Hence, this work proposes a new FD algorithm called *Fault-Detective.* Based on its purely data-driven approach, it can be applied to any system and requires only minimal configuration. It uses Random-Forest-Regression to identify the nonlinear relationships between sensors and models their behavior. Faults can be detected by comparing the predicted values with new measurement data - raising alarms if the difference exceeds a confidence threshold. The algorithm can be applied without knowing the structure of the plant or choosing any sensors by hand, which makes it very *flexible* and ensures an *easy configuration*. Since the algorithm adapts itself based on historical data, continuously updated by the latest measurement, it can achieve a *high fault coverage*. Finally, the algorithm is *extensively validated* with measurement data from three large-scale solar thermal plants, allowing an in-depth analysis of *Fault-Detective*'s results.

The main contributions of this work are thus (1) a novel algorithm called Fault Detective, which is based on artificial intelligence with a focus on being *flexible* and *easy to configure* to be widely applicable, and (2) its *extensive evaluation* with measurements of three solar thermal systems to quantify the prediction accuracy and the quality of the fault detection.

The paper is structured as follows: Section 2 lists related work and discusses the advantages and disadvantages of existing fault-detection methods compared to *Fault-Detective*. Section 3 presents the proposed algorithm and describes how it can be applied to a new system. Next, Section 4 deals with the evaluation method, explaining how the measurement data is used to test the algorithm. Section 5 presents the evaluation results. It shows which sensors were identified to model the system behavior, analyses the prediction accuracy, and compares the algorithm with manual fault detection. Finally, the conclusion is given in Section 6, and the proposed work is discussed in Section 7.

## 2. Related work

This section lists related work on Fault Detection and Diagnosis (FDD) for solar thermal systems. A good overview of existing FDD methods is provided by [8] and more recently by [12]. The following discusses the advantages and disadvantages of the methods to show how *Fault-Detective* might improve the current status quo. For clarity, related work is grouped based on the type of fault-detection approach, as defined by [47]. Fig. 1 contains a summary of the discussion and a rating of each method based on the authors' personal opinions in terms of the desired attributes described in the Introduction: *Flexibility, Easy Configuration, High Fault-Coverage, and Extensive Validation*.

### 2.1. Expert systems

Expert system methods try to utilize the domain knowledge of solar thermal experts and mimic how the expert would detect and diagnose faults. Often, this results in if-then-else rules that are implemented as algorithms.

One research project that uses this approach is IP-Solar [11,21,29]. They first performed an FMEA (failure mode and effects analysis) to de-

termine faults that often occur at solar thermal systems. Next, domain experts were asked to find rules to detect and diagnose these individual faults based on their experience. Finally, algorithms were implemented in a monitoring tool to detect the faults automatically as soon as new data gets available. The same idea has also been applied in the follow-up project Methodiqa [30] and by Sun et al. [43], the FUKS project [2] and the InSun project [34]. In addition, the University of Kassel and their research projects FeDet [15,25,42] and Solar-Check [16,39,41] deal with expert systems as well.

The considerable advantage of expert-system methods is that their results are easy to understand, and faults are often directly diagnosed. In addition, the calculations are typically simple enough to be implemented on PLC (programmable logic control) units and do not require extensive computational power.

However, one disadvantage is that algorithms are tailored to specific faults. Hence, many algorithms must be developed to cover all critical faults. Consequently, this also means that each algorithm must be configured for each plant, which can be time-consuming. As a result, many projects focus on a flexible configuration to speed up the process. For example, [11] provide templates to let users compose the hydraulic layout of the system and assign algorithms accordingly. Similarly, [25] use standardized sensor names, while [15,16] use a graph-based approach to represent the position of sensors in the system and assign them to algorithms automatically. Nevertheless, experts must still provide many necessary parameters and configure the correct hydraulic setup.

In addition, [41] report that many false-positive alarms are generated by the existing algorithms if tested on many systems. They argue that some operating conditions were not considered during algorithm development and that further development is needed to make algorithms more stable and flexible [41]. Instead, *Fault-Detective*'s data-driven approach might allow it to automatically model all relevant operating conditions without requiring a resource-intensive setup.

### 2.2. Parity space methods

Parity-Space methods use physically-derived models to describe the fault-free behavior of solar thermal systems. By using these mathematical models, estimations for sensor values can be calculated. If the difference between estimated values (i.e., expected behavior) and measured values (i.e., actual behavior) is too high, this thus indicates a fault.

For example, [9] use the simulation software TRNSYS to model the whole solar thermal system. Using measurement data, the model can estimate the solar yield of the corresponding timeframe. An alarm is raised if the measured solar yield is too low compared to the estimated one. The authors show that the algorithm can detect both immediate faults (e.g., stagnation) and degradation faults (e.g., small leakages). However, not much information about the source of the fault can be provided, leaving the fault diagnosis to the user. Comparing the simulation data and crosschecking with measurement data, however, may help pinpoint faults to components. The drawback of the method is that it requires a valid TRNSYS model, which in turn needs detailed knowledge about the simulation software, the solar thermal system, and its system control.

In contrast, the ISFH Input-Output Controller method [32] focuses mainly on the solar circuit and requires less simulation experience. It uses the collector Keymark equation from the EN/ISO 9806 to model the collector performance and also includes estimates for losses through storage and pipes. Again, estimated values for the solar yield are compared to measured ones. In contrast to a TRNSYS simulation, the models used in this method are more straightforward and can even be integrated into system control units. The authors report that the yield estimations have an accuracy of 10%. Hence, severe faults can be detected well, while problems with less influence on the solar yield stay undetected. Apart from that, the method can only detect faults in the solar circuit and cannot pinpoint the location of the fault.

The Performance Check [10,28,46] also uses the collector Keymark equation but focuses solely on the collectors. Hence, the influences of heat storage and pipes are not considered explicitly, which makes the calculations more straightforward. To improve the accuracy of the results, estimates are only provided if certain operating conditions are met. The method is excellent for monitoring solar collectors and checking guarantees between the collector manufacturer and the designer. However, its use for fault detection is limited as it only focuses on the collector. In contrast, faults at other system parts cannot be detected. In addition, the required operating conditions might not be met for a large portion of the year.

The D-CAT (Dynamic Collector Array Test) method implemented by [31] is another method focusing on the solar circuit. In contrast to the Performance Check, it applies a more detailed model for the collectors, which can be used with fewer restrictions on the operating conditions. Compared to the Performance check, their results are more accurate and give more insights into faults. However, setting up the algorithm is much more involved and requires implementing a new D-CAT model for each new system.

The *in situ* short-term test method (ISTT)[1] developed by [3] primarily aims at checking collector guarantees but can also be used for fault detection. The method first requires a simulation model of the solar thermal system (e.g., TRNSYS) to be set up. Next, a sensitivity analysis is performed for each component parameter used in the model. The parameters that strongly influence the simulated solar yield are then fitted with measurement data using the simulation model. As the last step, a new simulation is carried out using design conditions but with the fitted component parameters. The results are than compared with the initial guaranteed solar yield. For fault-detection, however, it might be more interesting to look for high changes in the component parameters, which might indicate faults or degradations (e.g., by reduced collector Keymark parameters). Unfortunately, the method has not been tested as FDD method to the best of the authors knowledge. In addition, the method requires a TRNSYS simulation model to be set up beforehand.

Another example of a Parity-Space method is the FSC-based approach studied by [15,38–40], which was analyzed as part of the Solar-Check project. It utilizes an empirical correlation between fractional solar consumption (FSC) and fractional auxiliary energy savings (f_save). As [26] showed, the correlation can be modeled using a polynomial function. If measured values for the f_save lie outside the confidence margin, it indicates that the solar thermal system does not work as expected. However, the authors note that one year of data is needed to compute the required values, which limits the response time of the algorithm. In addition, faults can only be detected if performance drops more than 11% [40]. Unfortunately, the method has not yet been successfully tested on operating data. Instead, experiments with actual measurement data did not show clear evidence that the method works as expected [39].

Most parity-space methods above unfortunately focus on a subpart of the system and hence do not provide a full fault coverage of the system. In addition, the methods often require setting many parameters or even creating simulation models beforehand. Instead, *Fault-Detective* tries to gather all required information based on the data and is not restricted to a particular component of the system.

### 2.3. Identification-based methods

Identification-based methods are very similar to the Parity-Space methods in that they provide predicted values (assuming fault-free operation) that are compared to measurement ones. The difference is that the models are learned based on measurement data rather than derived from mathematical formulas by domain experts.

[1] This method is listed as Identification-based algorithm by [12]. In contrast, we interpret it as Parity-Space method as it uses domain-knowledge in the form of mathematical functions (i.e., simulation models) to compare the design component parameters to measured ones.

For example, [13] model the yield of a solar parabolic-through collector field using an Artificial Neuronal Network (ANN). As input, they use the pressure difference, volume flow, and temperature difference of the collectors. The data for training and validation is acquired using a test rig that emulates the behavior of a solar thermal system. Faults are induced experimentally by altering the system control of the test rig. Using this data, fault detection is done by comparing values predicted by the ANN with measured values for the power. In addition, they also introduce a rule-based (expert system) analysis that classifies/diagnoses the faults. The downside of the method is that the input features and parameters of the ANN were derived by expert knowledge and trial and error. This means that it cannot be used directly for different systems. Similarly, adaptions are needed to the rule-based diagnosis if the algorithm is applied to a new system.

Similarly, [23,24] use ANNs to learn and predict the temperatures at two solar water heating systems in Cyprus and France. The behavior of sensors at multiple locations of the system is modeled using data from TRNSYS simulations. After training, the ANNs are supplied with measurement data from the two plants. Instead of directly comparing measured and predicted values, derived data points are calculated and compared (e.g., the temperature increase in the collector). This choice was made to ease the interpretation of the results. The diagnosis module then alarms the user if a high deviation between predicted and measured data is detected for more than five consecutive timesteps. The algorithm is validated using TRNSYS simulations and varying system parameters to mimic fault-induced system behavior. The results show excellent performance of the ANN in detecting faults. [45] use a nonlinear autoregressive (NARX) neuronal network for detecting faults. In contrast to typical ANNs, this structure allows using the previous timestamps of the predicted values, which typically increases the modeling accuracy of time-series data. The authors test different training strategies and provide a method to automatically configure the correct number of neurons in each layer. The fault detection works similarly to Kalogriou et al. by predicting the measurement values of the system and comparing the predictions with actual measured values. The algorithm is trained and validated on simulated data using TRNSYS. The method's success is shown by the example of a pump failure simulation using a fault-induced TRNSYS model.

Correa Jullian et al. [6] compare more sophisticated ANN models to detect faults. The analyzed architectures include long-short-term memory (LSTM) networks, recurrent neuronal networks (RNN), and deep layer neuronal networks (DNN), which are all reported to work very well with time-series data. For testing, data has been simulated using a fault-free and a fault-induced TRNSYS model for a solar thermal system in Chile. The results show excellent coefficient-of-determination scores of $R^2$=0.99 and higher, with the best results for the LSTM. On the other hand, fault detection proved to be more challenging, as many false positives were reported. As noted by the authors, validation with actual measurement data is missing yet.

A slightly different identification-based[2] approach is the spectral method developed by [17,36]. They measure the temperature of the collector flow and analyze its temperature change every time the pump starts its operation. The main idea is that this signal contains information about how the heat is distributed inside the collector. Thus, faults like dust on the collectors or degrading collector performance can be identified. By performing this analysis, the temperature change of "typical" plant operation is determined using measurement data of the system. The signals are converted to the frequency domain using Fourier-Transformations and average values for each frequency are calculated. Faults are then detected by applying the same transformation to new data and checking if the frequency spectrum is similar to reference values in "typical" system operation. The algorithm did well in correctly detecting covered collector panels and differences in pump speed setpoints. However, the author notes that only some faults can be detected with this method. In addition, the algorithm is more effective in detecting (long-term) degradation instead of (short-term) immediate faults and focuses only on the collectors. The required data logging rate of one measurement per second and the required 300 days of training data further complicate the configuration and flexibility of the algorithm.

As *Fault-Detective* belongs to the same type of FDD method, it shares many similarities with the approaches above. Especially the work of [24] highly influenced this work. However, the downside of most methods is that TRNSYS models are involved in training the ANNs. While this offers the opportunity to detect faults present at the solar thermal system directly after commission, it requires considerable knowledge and expertise to set up these models. Additionally, most authors did not have access to actual monitoring data to validate their approaches in real-life environments. The only exception is [17], which, however, can only detect a small section of faults and mainly focuses on long-term degradation.

### 2.4. Classification

This type of FD method also uses historical data to detect faults. However, the main difference to the Identification-based methods is that the Classification-based methods directly diagnose the fault. For example, these methods typically return the probability that one or multiple faults are present at a system at a given time.

One example is provided by [22], who apply Support-Vector-Machines (SVMs) and Demster-Shafer (DS) evidence theory. Their goal is to detect faults at a solar water heating test lab located in Beijing, China. In the first step, the measurement data is transformed using various preprocessing techniques relying on wavelet transforms to create a rich feature set. Next, SVMs are applied to isolate specific faults. The training is done by experimentally introducing faults to the system and using the corresponding measurement data to fit the support vector machines. In the last step, evidence theory is used to combine the individual predictions to make the final decision of whether a specific fault occurred. The results show that the algorithm works well, with a high fault-detection accuracy of above 90%. Unfortunately, the method requires that measurement data is available for each fault that should be detected. Hence, each fault must be observed and labeled at least once before the algorithm can be trained. This process must likely be done for each new solar thermal system, as the measurement data might not be affected similarly for different plants. This requirement considerably increases the complexity of the configuration.

Another classification-based method for solar thermal systems is provided by [20]. They apply an Adaptive-Resonance-Theory (ART) neuronal network with hierarchical layers (h-ART). In principle, it allows the users to group similar operating states of the system. If new data arrives that does not fit into any existing group, a new one is added to the h-ART network. Hence, such data is regarded as untypical and interpreted as a fault. As a further advantage, the h-ART automatically provides a severity for the fault as a direct consequence of the hierarchical layout of the algorithm. The approach is tested using the monitoring data of a solar hot water test lab. Faults are manually introduced to the system by covering the collectors, simulating pump failure with system control, and similar methods. The faults could be detected very well, with few false positives. The only disadvantage of the model is that multiple years of data are needed to train the neuronal network. Hence, the authors suggest and use data from a TRNSYS simulation of the test lab to create the training data. However, the authors note that creating such detailed simulation models requires detailed knowledge and expertise. In addition, it is not reported how well the algorithm can deal with adaptions in system control or exchanges of components. Drastic changes in the operation might create a new group in a low hierarchy

[2] Faure et al. [12] interpret this method as Classification method. However, the spectral method does not allow to label/classify detected faults, but only compares the measured spectral signal with the signal during fault-free operation. Hence, we regard it as Identification-based method.

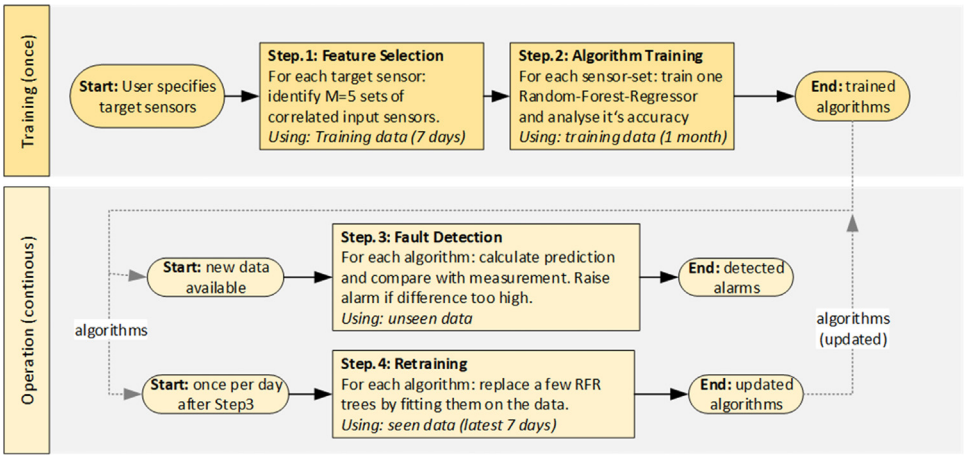


**Fig. 2.** Flowchart showing the concept of *Fault-Detective*.

layer, rendering the other groups ineffective. As a result, the entire hierarchy may need to be built again, leading to many false alarms. Hence, drastic changes may require retraining the algorithm with an updated TRNSYS model.

More recently, [37] developed an ANN-based classification method targeted at concentrating solar power plants. In contrast to the Identification-based algorithms, their network is trained to return the probability of specific faults. In addition to the ANN, they also propose two additional checks that can further isolate the location of the fault. In total, their method can distinguish between fault-free operation and three distinct kinds of faults related to the volume flow, heat losses, or the optical efficiency of the collectors. To train and test the method, the authors use simulation data of the ACUREX plant, mimicking faulty operation by inducing faults in the simulation models. The results show that each fault can be identified and isolated well, with an accuracy of over 80%.

In conclusion, one drawback of classification-based methods is that they need measurement data with faults to train the machine-learning models. Hence, it is only possible to learn about faults that have already happened (at least once) to the system. As this limits the fault coverage of the algorithms, simulation data might be used instead, which, however, increases the effort to set up the algorithm. The exception is [20], whose method can detect new faults without an additional training phase and labeled data. Unfortunately, their algorithm needs a lot of fault-free data to model typical system behavior, requiring simulation data.

## 3. Fault detective

This section describes how *Fault-Detective* works and explains why it is designed this way. The first part contains a rough overview of the method before each algorithm step is described in detail in its individual section.

### 3.1. Overview

*Fault-Detective* aims to detect faults by modeling multiple target sensors and comparing the predictions with the measurement values. If the difference between prediction (expected behavior) and measurement (actual behavior) is too high, an alarm is raised.

This functionality is provided using a four-step approach, as depicted in Figs. 2 and 3. At the Feature-Selection (Step.1), one week of measurement data is analyzed to identify correlated sensors. This step is needed to determine which sensors can be used to model a specific target sensor. However, the algorithm searches for multiple sets of sensors as this allows to detect a variety of faults. At the Algorithm Training (Step.2), each identified set is used to train a Random-Forest-Regressor (RFR). After the training, predictions for the target sensor can be made using the data of correlated sensors. Next, at the Fault-Detection (Step.3), new measurement data is supplied to *Fault-Detective*. By comparing predicted and measured values, alarms can be raised if the differences are too high compared to the training. This is done for each trained regressor targeting different sensors. Finally, the Retraining step (Step.4) continuously improves the RFSs to increase their accuracy and adapt to seasonal changes in the measurement data.

All these steps are performed automatically by *Fault-Detective,* without a need to change any hyperparameters. The only configuration that needs to be done by the user is specifying which sensors should be targeted (see Fig. 2).

### 3.2. Step1: Feature Selection

The Feature-Selection aims to identify sensors that can be used to predict the target sensor. Knowing these sensor correlations is crucial for training accurate machine-learning models in the Algorithm-Training step.

More precisely, we want to find *multiple, minimal* sets of input features (i.e., sensors) that can sufficiently model a target sensor. The motivation behind this is the following:

We want to have a *minimal* number of features as the modeling accuracy of machine learning algorithms typically increases if only relevant input features are used. That is because irrelevant features do not hold valuable information but are still processed by the algorithm. In the best case, the machine-learning algorithm discovers that the data is irrelevant to the prediction. In the worst case, however, the algorithm tries to infer incorrect relationships between the unrelated sensors, resulting in poor performance. Another advantage of using only relevant sensors is that considerably less data needs to be processed. For example, the datasets of this work contain about 100 sensors per plant. However, only a few sensors are needed to provide good models for a specific target sensor (see results in Section 5) while taking less time to parse. Plus, having fewer input features also allow users to better understand what the algorithm does. For example, imagine a machine-learning algorithm that predicts the volume flow based on the rotation speed of a pump. The relation between these two measurements is more or less straightforward, allowing solar thermal experts to interpret the algorithm results quite well. In contrast, sensemaking of the prediction is way more difficult if hundreds of sensors are involved.

Moreover, we want to have *multiple* feature sets because we want to discover faults from many directions. For example, let us assume we want to detect faults by modeling the volume flow of the primary solar circuit. One way to model the volume flow might be by using the rotation-speed signal of the pump. With this correlation, a user can check, for example, if there is a volume flow present if the pump is

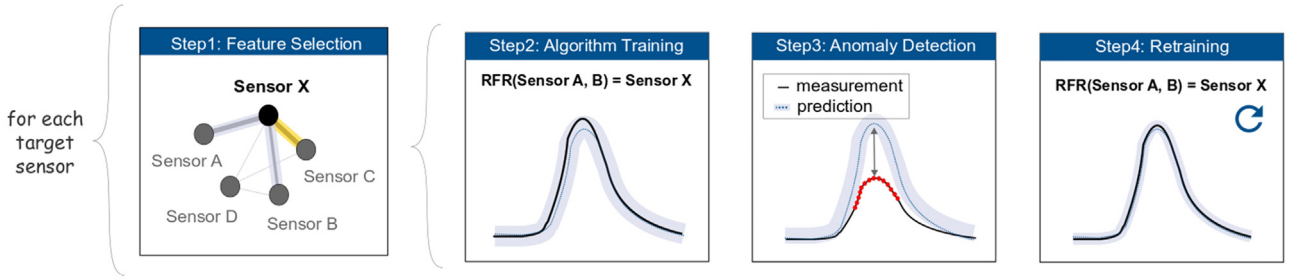


**Fig. 3.** Sketch showing each step of *Fault-Detective's* approach.

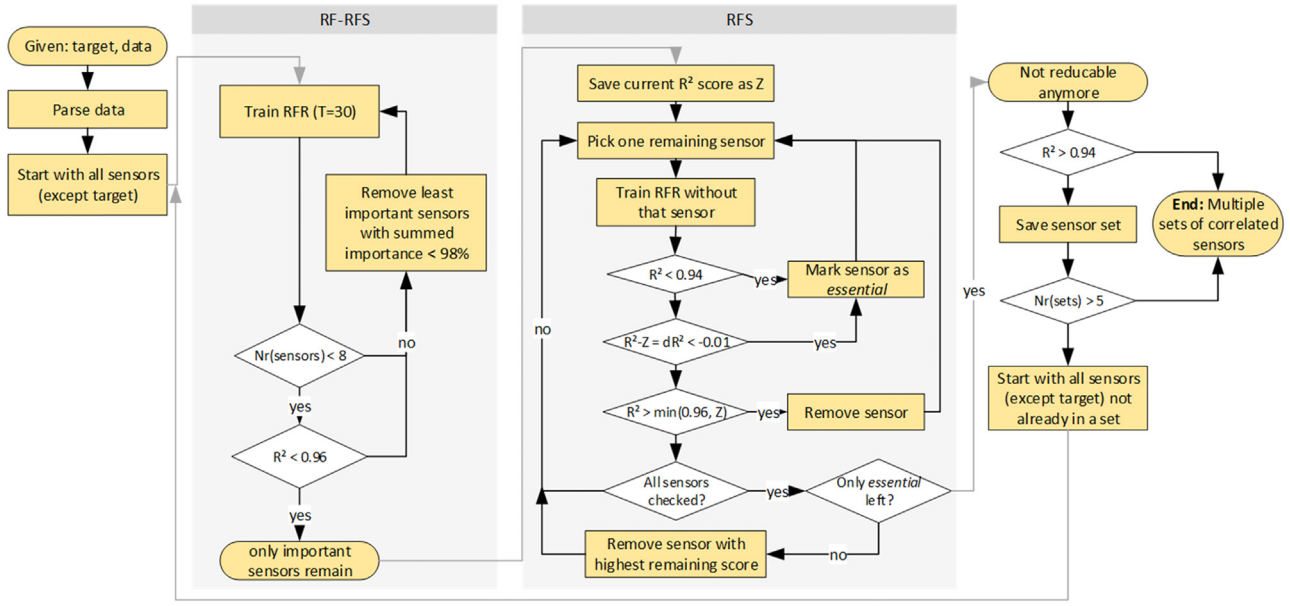


**Fig. 4.** Flowchart showing the concept of the Feature-Selection step.

switched on. Another set of features for predicting the volume flow might be the collector temperature and the irradiation. In contrast to the other example, these measurements might be correlated based on the system's control. Hence, it might be used to check that the pump starts correctly as soon as the collector temperature or the irradiation is above a certain threshold.

As these examples show, multiple small sets of correlated features allow us to detect different faults. The idea behind the Feature-Selection step is thus to identify these individual correlations between sensors automatically. It allows *Fault-Detective* to be *flexible* as it discovers the relations based on the data and still provides a *high fault coverage* as target sensors are monitored from multiple perspectives.

The algorithm to identify these multiple minimal input-feature sets for each target sensor is depicted in Fig. 4:

First, *Fault-Detective* requires a small part of historical data to analyze the correlations. This data must contain enough variability of operating conditions to represent typical system behavior. Otherwise, it will not be possible to infer correct relationships. For example, the algorithm cannot identify correlated sensors accurately if the data only contains measurements of rainy days with the plant not running. We found that using at least seven days of data typically fulfills this criterion.

The measurement data must be prepared by discarding unreadable values and omitting monotonic sensors (e.g., accumulating heat meter sensors). The latter is done as monotonically increasing/decreasing values pose problems for the Random-Forest used by *Fault-Detective* in further steps. The reason is that decision trees cannot extrapolate values if new data lies outside of the range experienced during training. This requirement would be continuously violated if using accumulating sensors for modeling.

After the preprocessing, the data is used to train a Random-Forest-Regressor (RFR) to model the target sensor. As shown by [1], this machine-learning algorithm works well to model the multi-dimensional and nonlinear behavior of solar thermal systems. In addition, RFRs provide insights into which features (i.e., sensors) are the most important for predicting the target sensors' measurements. That allows assigning a 'feature-importance' [5] to each input sensor, with higher values for more important sensors and summing up to 100%. Highly correlated features can then be discovered by iteratively dropping features with the lowest feature importance and checking if the remaining sensors still allow modeling the target sensor sufficiently well. This procedure is called Random-Forest Recursive-Feature-Selection (RF-RFS). It is also applied, for example, by [7,48]. However, while [48] only drop one feature per iteration, we keep features with an accumulated feature-importance below 98%. This threshold was chosen to eliminate many unrelated features at each iteration without accidentally removing too much important data. We stop the recursion if the number of remaining features is small enough (less than $N = 8$ features), and the out-of-bag score' [5] would drop below a certain threshold ($R^2$<0.96) if any more features were discarded. For the number of remaining features, a value of $N = 8$ was selected as models with more than eight features are difficult to interpret based on feedback from domain experts. In addition, a value of $R^2$=0.96 was chosen based on the authors' experience with solar-thermal and other datasets, as models with similar or higher scores typically yield sufficiently accurate predictions.

After the RF-RFS, all irrelevant features should have been eliminated. However, the set might still contain sensors that are very similar to each other and hence contain redundant information. To further minimize the number of features, conventional Recursive-Feature-Selection (RFS) is applied. Hence, features are eliminated systematically, always discarding the least relevant feature. It is constantly checked that dropping a feature would not decrease the model's accuracy too much, requiring a minimum score of $R^2$>0.94 after the deletion. In addition, features are not dropped if the new score would decrease more than $dR^2$>0.01 compared to the previous score. In contrast, if the score is increasing or still above $R^2$=0.96, the feature is dropped immediately. These parameters were selected based on trial and error using the validation dataset.

The process above is repeated until no more valid correlations are found or at least $M = 5$ feature-sets have been determined for each target sensor. Within the Feature-Selection step, the Random-Forest-Regressors is trained using $T = 35$ estimator trees and a max-depth of 20. These settings are used to limit the computation time, as the goal is to identify correlated features instead of providing the most accurate predictions. The other paramters are set to the Sci-Kit-Learn defaults [33].

The complexity of the Random Forest algorithm is $O(v \cdot n \log n)$, where n is the number of records and v is the number of features. The Recursive-Feature-Selection adds a factor of O(v) to such complexity in the worst case. In contrast, the complexity of the conventional Recursive-Feature-Selection is $O(v^3 \cdot n \log n)$ and hence more involved. However, note that the number of input features $v$ has already been reduced to a maximum of $v = N = 8$ in the previous step.

### 3.3. Step 2: Algorithm Training

The information from the Feature-Selection is now used in the Algorithm-Training to create machine-learning models for each identified feature set. The aim is to get the most accurate models to predict the behavior of the respective target sensors. In addition, we also want to know how accurate the predictions are. By determining a confidence interval, it is possible to distinguish whether deviations are caused by faulty system behavior or poor modeling. The complete process is depicted in Fig. 7.

In principle, any machine learning architecture that can handle the nonlinear multi-dimensional solar thermal data could be used for modeling the target sensor. For example, good modeling performance has been shown for various Artificial-Neuronal-Network architectures [6] and tree-based ensemble methods like Random-Forest-Regression [19]. In this work, Random-Forest-Regressions are used as they do not need much preprocessing for the data, are resistant to overfitting, and out of convenience as they are also used in the Feature-Selection step. Hence, in this step, each correlation is trained using a Random-Forest-Regressor using $T = 200$ estimator trees and no restriction on the depth of the trees. This setup yielded sufficiently accurate results independently of the target and input sensors.

To further increase the modeling accuracy, we use 'Input-Lagging' [14,35] to consider the temporal dependencies within solar thermal data. Hence, predictions also take previous timestamps of the input features into account. That can improve the accuracy of the algorithms, as it allows modeling phenomena like the slow heating-up and cooling-down of fluids. An example can be seen in Fig. 5, where the thermal power of Plant 1 is predicted using Random-Forest-Regression with and without lagging. In this case, the unlagged Random-Forest-Regressor cannot predict the behavior of the plant well ($R^2$=0.92), while the lagged RFR yields superior results ($R^2$=0.96). Hence, the measurements are lagged before handing the data to the Random-Forest Regressors. In this work, the last two timestamps (i.e., the last 10 min) are lagged. This rather small value allows us to consider short-term dependencies without risking overfitting due to uncorrelated data.

Finally, the second part of the Algorithm-Training is to estimate the uncertainty of the prediction. This step is crucial to interpret the differences between predictions and measurements (i.e., the residuals) and distinguish between typical and faulty behavior. By estimating a confidence interval, it is possible to tell whether high mismatches are caused by anomalies in the data or simply because of lousy modeling. However, it cannot be assumed that the confidence margin is a constant value. For example, the behavior of the collector temperature might be very different when the pump is running than during the night, as seen in Fig. 6. In this case, temperatures above 60 °C are predicted very well, while temperatures below 60 °C are modeled less accurately. Hence, the machine learning model might have different accuracies based on the operation conditions. To accommodate this, we estimate the prediction errors for different regions of the target sensor's domain. Using this strategy, the algorithm can be made more sensitive in regions where the model works well while having a higher confidence interval for regions less understood by the algorithm.

Thus, the flowing procedure is applied: After the training, the residuals are calculated using the 'out-of-bag' [5] predictions. Next, the domain of the target sensor is split into 20 bins, computing the standard derivative of the residual inside each bin. Assuming that the residuals follow a gaussian distribution centered at $\mu = 0$, that would mean that residuals of more than two times the standard derivative (i.e., $2\sigma$) only have a 5% chance of being correct. In contrast, residuals of more than $3\sigma$ should be very unlikely. While the assumption that the error follows a gaussian distribution in each bin is not generally valid, it is still used as a best guess for defining the confidence margin.

### 3.4. Step3: Fault-Detection

The trained algorithms can then be used to detect abnormal data. When provided with new measurements, each Random-Forest-Regressor generates predictions for its respective target. An alarm is triggered if the difference between prediction and measurement is unusually high compared to the confidence interval.

The following procedure is applied (see Fig. 8): When provided with new data, the regressor generates predictions for the target sensor. If the residual exceeds three times the standard derivative (i.e., $3\sigma$) it is regarded as *suspicious*. If it even exceeds four times the standard derivative of the residuals during training (i.e., $4\sigma$) it is regarded as an *anomaly*. However, warnings are only raised if the residuals are *suspicious* for at least $X = 5$ consecutive timestamps and the mean residual of the past X timesteps is above the respective thresholds. That is done to limit false alarms.

Unfortunately, the Random-Forest-Regressor cannot extrapolate beyond the range of values experienced during training. As a result, it cannot predict values above/below certain thresholds. This behavior can lead to false alarms, for example, if collector temperatures rise to unprecedented levels due to seasonal changes. Hence, warnings are not raised if the prediction is outside the known domain of the Random Forest Regressor. That values are labeled as *out of bounds* if both the measured value is outside the domain of at least 50% of the decision trees and the predicted value is outside the domain of at least one decision tree. The first criterion evaluates if the measurement data is outside the typical range, which might be caused by seasonal changes but also due to faults or other events. In contrast, the second criterion ensures that the extrapolation issue affects at least one prediction.

### 3.5. Step4: Retraining

At least seven days of data are initially used to train *Fault-Detective* in the Algorithm-Training step. Unfortunately, it cannot be expected that this small dataset can sufficiently describe the behavior of the system for its whole lifetime. Instead, seasonal changes may affect the system, for example, higher collector temperatures in summer compared to the winter. Another example is a change between summer and winter load profiles which will affect the solar thermal system's behavior even more drastically. Hence, *Fault-Detective* must be retrainable to adapt to these

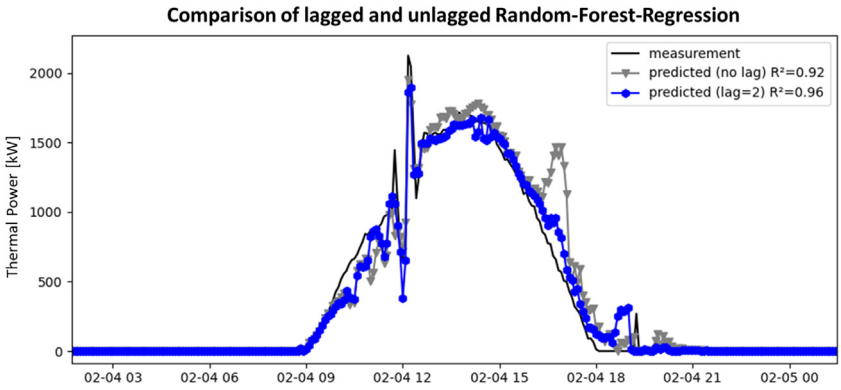


**Fig. 5.** Example of Random-Forest-Regression with and without lagging. The prediction of the lagged Random-Forest-Regressor (blue) shows a higher agreement to the measurement values (black) compared to the unlagged regressor (gray).

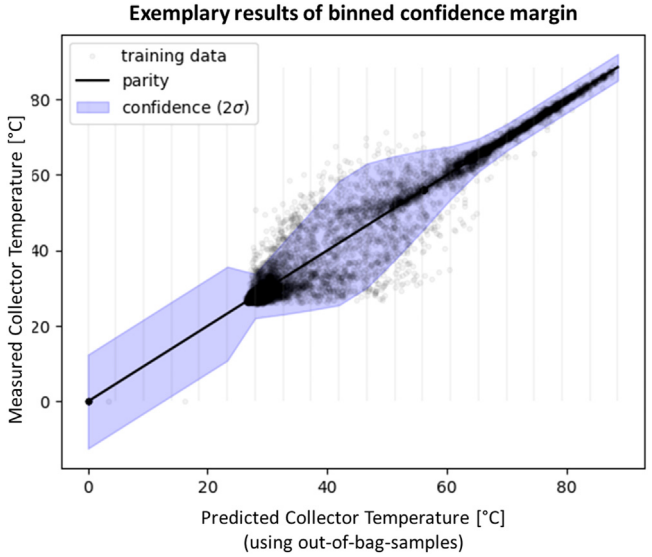


**Fig. 6.** Example of the binned confidence margin. The graph compares measured (y-axis) and predicted (x-axis) values for the solar collector temperature. Ideally, all points lie on the diagonal (black line). The confidence margin (blue area) shown is two times the standard derivative inside each bin (confined by the gray lines).

changes and learn from new measurement data. That will keep the prediction and fault-detection accuracy high throughout the year. Hence, after the Fault-Detection step is finished, an additional Retraining step is performed. It parses the data again, updating the Random-Forest-Regressor and the confidence margin. Similar to the Feature-Selection, we found that at least seven days of data are necessary for retraining. This amount of time includes enough variability of the operating conditions to infer correct models by the Random Forest Regressor.

The procedure is as follows (see Fig. 9): If provided with retraining data, *Fault-Detective* trains some new decision trees and randomly discards the same amount of old decision trees. The training is performed in the same way as at the Algorithm Training step. By default, the retraining is scheduled daily, replacing $T = 10$ decision trees (i.e., 5% of all trees) per day. In addition, the confidence margin is updated by calculating the standard derivative of the residuals in each bin using the new data. The results are then added to the previous statistics as shown in equation (Eq. (1)).

$$\sigma_{combined} = \sqrt{\frac{(\sigma_{old})^2 \cdot N_{old} + (\sigma_{new})^2 \cdot N_{new}}{N_{old} + N_{new}}} \tag{1}$$

However, care must be taken to exclude faulty data before retraining the algorithm. Otherwise, the algorithm would model faulty system behavior and no longer yield reliable predictions. Hence, data is excluded where *anomalies* (i.e., residuals of more than then $4\sigma$) occur 2 h before or after each respective timestamp. This way, only fault-free data is used for improving the model accuracy.

Unfortunately, faults are not the only events that cause the data to deviate from typical system behavior. For example, changes in system control, services, and repairs might also permanently change the systems' behavior. Such events will likely be identified as an *anomaly* by *Fault-Detective* and hence would be excluded from retraining. However, in contrast to faults, it is desired that *Fault-Detective* retrains on this data to adapt to permanent changes. Otherwise, false alarms will be raised continuously every day. To distinguish between these permanent changes and faults, we look for *anomalies* that persist for at least 30 min for multiple days (i.e., 4 out of 7 days). Here, the 30 min are used to filter out short-term faults, while requiring the anomaly to be present 4 of

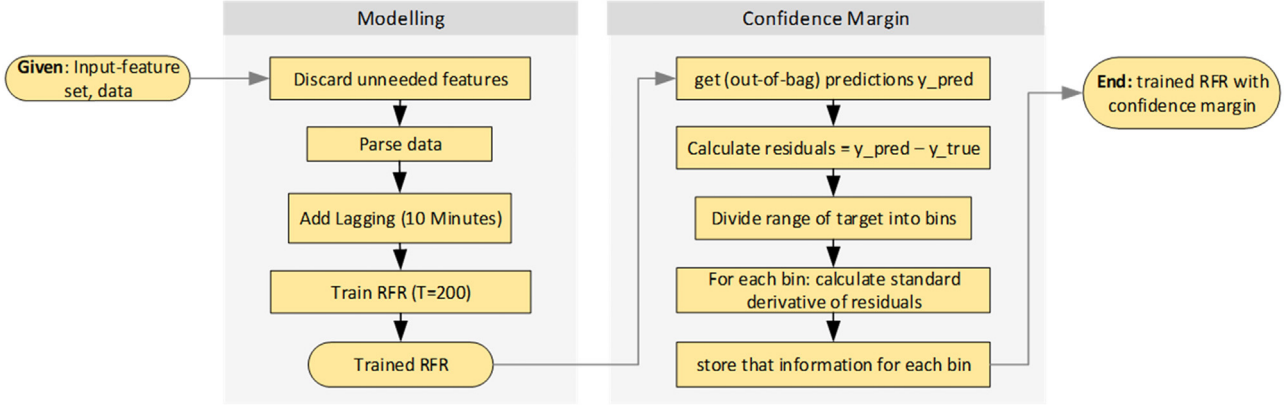


**Fig. 7.** Schematic showing the concept of the Algorithm-Training step.

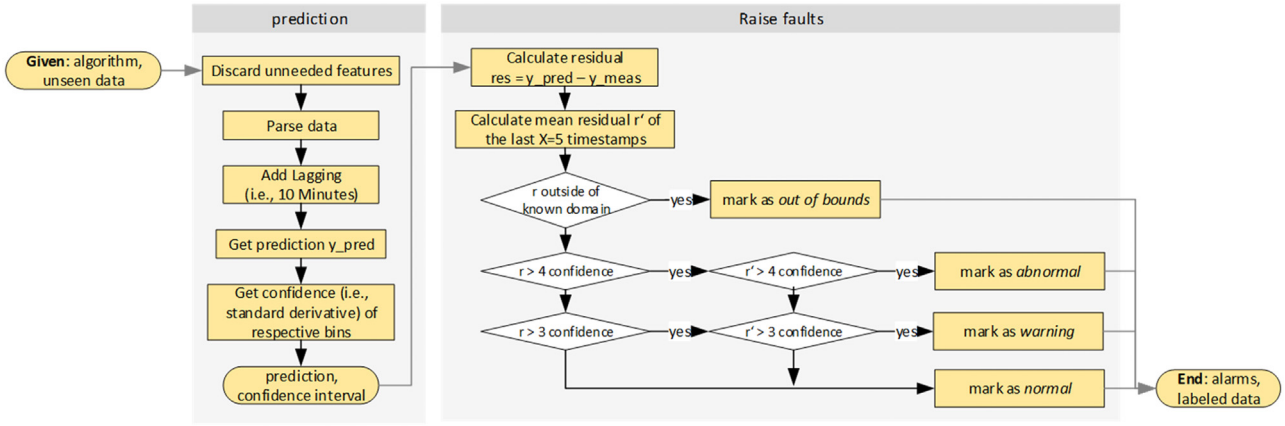


**Fig. 8.** Schematic showing the concept behind the Fault-Detection step.

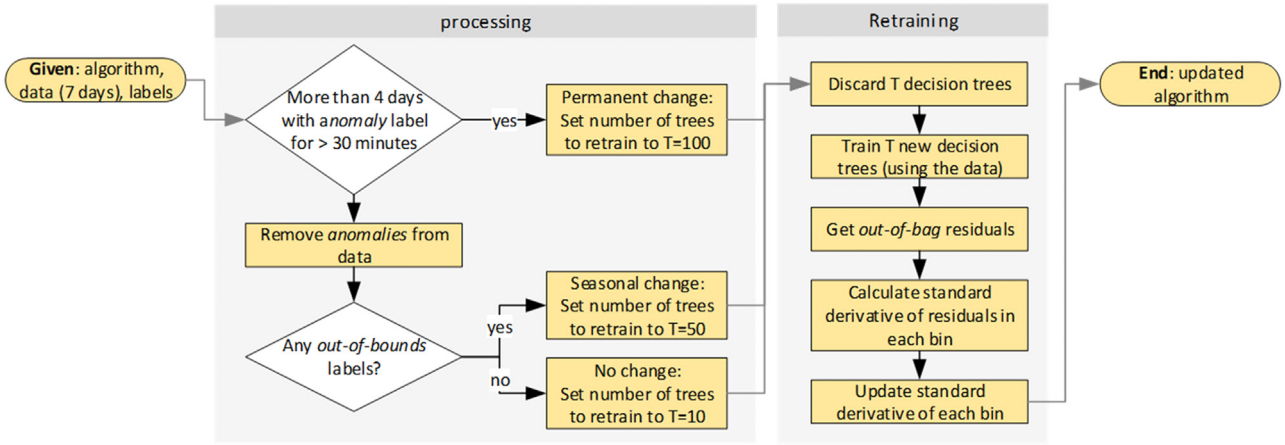


**Fig. 9.** Schematic showing the concept of the Retraining step.

7 days allows verifying if the anomaly is persistent. If such a permanent change is detected, *Fault-Detective* informs the user about it and retrains 50% of the decision trees (i.e., $T = 100$ trees) on the retraining data to adapt to the changes faster.

In addition, seasonal changes might cause extrapolation problems, leading to wrong predictions in case the values are outside of the range experienced in the past. If such events are detected (see Fault-Detection step), $T = 50$ decision trees (i.e., 25% of the tree) are replaced instead to increase the adaption rate.

## 4. Evaluation method

This section explains how *Fault-Detective* is evaluated using the measurement data of three large-scale solar-thermal systems of different sizes and structures. For this purpose, the available data is described, and then the measures for validating the Adaptability, Modelling Accuracy, and the Fault Detection Performance are introduced and discussed.

### 4.1. Available data

The performance of *Fault-Detective* is extensively evaluated using measurement data of three large-scale solar thermal systems with sizes of more than 1000 $m^2$ of collector area. The plants are used for different applications: cooling, heating, or hot water supply, and have different layout schemes and control strategies. The data is accessed using a database providing measurements with a resampling rate of 5 min. One year of data from each system is used, which allows for testing *Fault-Detective's* ability to adapt to different seasonal conditions. The datasets contain around 100 sensors per plant. About half of the sensors contained in the dataset measure collector temperatures in different collector rows. In addition, each dataset contains measurements for the flow- and return temperature and pump rotation speed of each thermal circuit. The thermal power, volume flow, valve positions, electric consumption, and storage temperatures at different layers are also recorded. For the evaluation, three sensors insightful for monitoring solar thermal plants are used as a target at each plant: one randomly selected collector temperature, the flow temperature of the primary circuit, and the solar thermal power.

### 4.2. Computational resources

The training and evaluation of *Fault-Detective* have been done using an Intel Core i5–8250 U processor and 16 GB RAM. The implementation is done in Python 3.9 using the libraries Sci-Kit-Learn [33], Pandas [44], and NumPy [18]. The Feature-Selection took 18 s per target sensor on average, identifying one input feature set every 4 s. For each identified set, the Algorithm-Training took 2.6 s on average to train a new Random-Forest-Regressor. The Fault-Detection took 0.07 s per day and algorithm, while the Retraining took 0.23 s per day and algorithm on average. Hence, 26 min were required to parse one year of data targeting all three sensors for each of the datasets.

### 4.3. Data usage for evaluation

The evaluation uses walk-forward validation, meaning that data is fed to Fault-Detective walking forward in time. That is done as con-

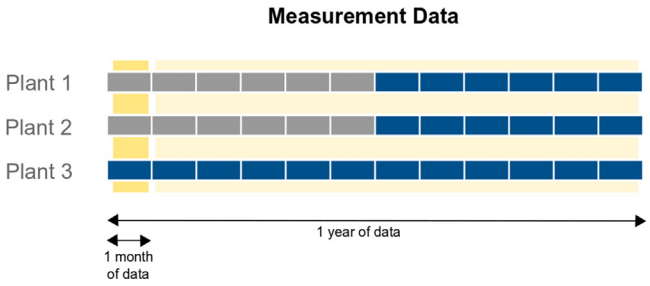


**Fig. 10.** Summary of the data usage: Only the data of two plants are used as the validation set (gray boxes). The remaining data (blue boxes) is used as the test set for the final performance evaluation.

ventional cross-validation strategies do not work well on time-series data. Since there are temporal correlations between data points, shuffling would result in poor generalization and overfitting [4]. Instead, the data of each system is provided to *Fault-Detective* as follows: The first month of data is used for the feature selection (step 1) and the initial training (step 2). These are the steps required to set up Fault-Detective. The remaining months are used for fault detection (Step.3) and retraining (Step.4) - providing *Fault-Detective* with daily chunks of data. That process emulates how new data is passed to *Fault-Detective* in an *in-situ* setting.

Datasets for machine learning are typically divided into three parts: The *training set* allows us to train the algorithm, the *validation set* can be used to check how the trained algorithm performs and adjust it accordingly, and the *test set* is used to evaluate the final algorithm in terms of performance. Hence, the test set is only used once to ensure that the statistics hold true for unseen data. This work uses the first six months of Plant1 and Plant2 as training- and validation sets to develop *Fault-Detective* and set all its hyperparameters. In contrast, the remaining six months of Plant 1 and Plant 2 and the whole dataset of Plant 3 (see Evaluating Adaptability) were used for the final testing. The whole process is illustrated in Fig. 10.

### 4.4. Evaluating adaptability

One of the benefits of *Fault-Detective* is that it can automatically adapt to multiple solar thermal systems, providing flexibility and easy installation. To prove this claim, the same algorithm with the same hyperparameters is used at all plants, ensuring no system-specific configuration is needed. For this reason, the data from Plant 3 is not used in the validation step. Instead, *Fault-Detective* is directly applied to the dataset of Plant 3 without any manual adjustment and with the same parameters used for the other plants.

### 4.5. Evaluating modelling accuracy

The modeling accuracy is evaluated using the following metrics for each of the target sensors: The mean absolute error (MAE), the root mean squared error (RMSE), and the coefficient of determination ($R^2$). All scores are calculated using the sci-kit-learn package [33].

The mean absolute error (MAE) indicates the average difference between predicted $\hat{y}$ and measured values $y$. Hence, MAE scores are close to zero in case of perfect predictions and have the same unit as the target value $y$.

$$MAE(y,\ \hat{y}) = \frac{1}{n}\sum_{i=0}^{n-1}\left|y_i - \hat{y}_i\right| \tag{2}$$

The root-mean-squared error (RMSE) is closely related to the MAE. However, it adds a higher penalty to more significant deviations between predicted and measured values. Again, values close to zero indicate an excellent prediction, while high values indicate bad prediction performance.

$$RMSE(y,\ \hat{y}) = \sqrt{\frac{1}{n}\sum_{i=0}^{n-1}\left(y_i - \hat{y}_i\right)^2} \tag{3}$$

Finally, the coefficient of determination ($R^2$) indicates how well the variance of the predicted values can explain the variance of the measured values. It can generally be interpreted as goodness-of-fit, with a score of $R^2$=1 for perfect predictions. At the same time, low values indicate bad prediction performance.

$$R^2(y,\ \hat{y}) = 1 - \frac{\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{n}\left(y_i - \bar{y}\right)^2} \tag{4}$$

Where $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ is the average measured value.

### 4.6. Evaluating fault detection performance

In addition, *Fault-Detective* is compared with manual fault detection to evaluate the fault detection performance. To do so, a solar thermal expert analyses the data of all three plants. Any events (e.g., faults, anomalies, or maintenance events) that occur at the plant are documented. Similarly, *Fault-Detective* is applied to the test and validation dataset, executing all four algorithm steps. Each anomaly the algorithm identified is then manually examined to determine whether a false alarm or an actual fault occurred. The performance is evaluated by analyzing the number of true-positive and false-positive alarms and comparing the number of faults detected by manual and automatic fault detection. As noted above, this process is only done once after the development of the algorithm has finished, using the same parameters at all three plants and requiring no user input except specifying the target sensor.

## 5. Results

This section shows the results of *Fault-Detective* using the test datasets of all three solar thermal systems. The first part contains an analysis of the Feature-Selection step, showing which correlations were identified by *Fault-Detective*. In the second part, the modeling accuracy in terms of the coefficient of determination ($R^2$), mean absolute error (MAE), and root mean squared error (RMSE) is presented. The last part shows which faults have been found by *Fault-Detective*. The results are compared to the manual fault detection performed by the monitoring personnel as well.

### 5.1. Features selection

The first step of *Fault-Detective* is to identify correlated sensors in the Feature-Selection step, using the first week of the training data. As written in Section 4, three sensors are targeted at each system: one randomly selected collector temperature, the flow temperature of the primary circuit, and thermal power. The results can be seen in Fig. 11.

Targeting the collector temperature, *Fault-Detective* primarily identifies correlations to other collector temperature sensors. This is expected as collector rows next to each other also experience the same return flow and irradiation and are adjusted to yield similar flow temperatures. Hence, the modeling accuracy on the training set is also very high, with an average score for the coefficient of determination of $R^2$=0.98. Interestingly the scores for Plant 2 are lower than the others, while more input sensors are identified at Plant 3.

| Set | System | Target | $R^2$ score (Training) | Input Sensors |
|---|---|---|---|---|
| 0 | Plant 1 | Coll. Temp. 02 | 1.00 | Collector Temp 01 |
| 1 | Plant 1 | Coll. Temp. 02 | 1.00 | Collector Temp 13 |
| 2 | Plant 1 | Coll. Temp. 02 | 1.00 | Collector Temp 14 |
| 3 | Plant 1 | Coll. Temp. 02 | 1.00 | Collector Temp 04 |
| 4 | Plant 1 | Coll. Temp. 02 | 0.99 | Collector Temp 15 |
| 5 | Plant 2 | Coll. Temp. 63 | 0.97 | Collector Temp 33 |
| 6 | Plant 2 | Coll. Temp. 63 | 0.98 | Collector Temp 57 |
| 7 | Plant 2 | Coll. Temp. 63 | 0.97 | Collector Temp 42 |
| 8 | Plant 2 | Coll. Temp. 63 | 0.97 | Collector Temp 56 |
| 9 | Plant 2 | Coll. Temp. 63 | 0.98 | Collector Temp 69 |
| 10 | Plant 3 | Coll. Temp. 05 | 0.97 | Collector Temp 09 |
| 11 | Plant 3 | Coll. Temp. 05 | 0.99 | Collector Temp 07, Collector Temp. 11 |
| 12 | Plant 3 | Coll. Temp. 05 | 0.99 | Collector Temp 03, Collector Temp. 13 |
| 13 | Plant 3 | Coll. Temp. 05 | 0.98 | Collector Temp 21, Collector Temp. 15 |
| 14 | Plant 3 | Coll. Temp. 05 | 0.98 | Collector Temp 01, Collector Temp. 15 |
| 15 | Plant 1 | Solar Power | 0.95 | Collector Temp 05, Solar Flow Temp., Power from Storage to Consumer, Irradiation |
| 16 | Plant 1 | Solar Power | 0.96 | Storage Temp Bottom, Collector Temp 3, Collector Temp 29 |
| 17 | Plant 1 | Solar Power | 0.95 | Collector Temp 4, Solar Return Temp |
| 18 | Plant 1 | Solar Power | 0.95 | Collector Temp 52, Collector Temp 37, Collector Temp 02 |
| 19 | Plant 1 | Solar Power | 0.95 | Solar Return Temp. (redundant), Collector Temp 30, Collector Temp 01 |
| 20 | Plant 2 | Solar Power | 1.00 | Solar Temp Difference, Solar Volume Flow |
| 21 | Plant 2 | Solar Power | 0.98 | Solar Return Temp, Solar Flow Temp, Solar Circuit Pressure |
| 22 | Plant 2 | Solar Power | 0.98 | Storage Return Temp, Storage Flow Temp, Electric Consumption |
| 23 | Plant 3 | Solar Power | 0.99 | Solar Temp Difference, Solar Volume Flow |
| 24 | Plant 3 | Solar Power | 0.95 | Solar Flow Temp, Solar Return Temp |
| 25 | Plant 1 | Flow Temp. | 1.00 | Solar Flow Temp (redundant) |
| 26 | Plant 1 | Flow Temp. | 0.99 | Collector Temp 04, Storage Temp Top |
| 27 | Plant 1 | Flow Temp. | 0.99 | Collector Temp 14, Storage Temp Bottom |
| 28 | Plant 1 | Flow Temp. | 0.97 | Collector Temp 13, Solar Return Temp |
| 29 | Plant 1 | Flow Temp. | 0.98 | Solar Return Temp (redundant), Collector Temp 15 |
| 30 | Plant 2 | Flow Temp. | 1.00 | Solar Flow Temp (redundant Sensor) |
| 31 | Plant 2 | Flow Temp. | 0.96 | Solar Temp Difference, Storage1 Return Temp, Storage1 Temp Top |
| 32 | Plant 2 | Flow Temp. | 0.96 | Solar Return Temp. to Storage1, Solar Return Temp. Storage2, Storage1 Flow Temp to Consumer |
| 33 | Plant 2 | Flow Temp. | 0.96 | Collector Temp 64, Solar Flow Temp. Storage2, Consumer Flow Temp |
| 34 | Plant 2 | Flow Temp. | 0.96 | Solar Return Temp, Storage 1 Return Temp from Consumer, Solar Flow Temp. Storage2, Electric Consumption |
| 35 | Plant 3 | Flow Temp. | 0.99 | Solar Flow Temp (redundant) |
| 36 | Plant 3 | Flow Temp. | 0.97 | Collector Temp. 19, Storage Temp Middle |
| 37 | Plant 3 | Flow Temp. | 0.97 | Temp Diff Storage to Consumer, Collector Temp 21 |
| 38 | Plant 3 | Flow Temp. | 0.96 | Solar Volume Flow, Collector Temp 17 |
| 39 | Plant 3 | Flow Temp. | 0.97 | Flow Temp Storage to Consumer, Collector Temp 15 |

**Fig. 11.** Results of Feature-Selection, showing which sets of sensors were identified to be correlated to the target sensor, sorted by their feature importance. In addition, the out-of-bag score ($R^2$) for each sensor set is shown.

Concerning the power measurements, *Fault-Detective* also manages to identify some apparent relations. For example, the thermal power can be calculated using the volume flow and the temperature difference of the solar circuit. This correlation is successfully identified at both Plant 1 and Plant 3. However, no volume-flow-related sensor was selected targeting Plant 2. The reason might be that the pump speed is running at discretized levels. Hence, volume flow can be well estimated based on the temperature sensors.

In addition to the obvious input-feature sets, more insightful correlations are also identified. For example, the electric consumption and the pressure at the solar circuit correlate with the volume flow. Hence, they are used to predict the power in the case of Plant 2. As another example, the irradiation and some storage temperatures are selected at Plant1 as they are connected to system control. Surprisingly, Fault-Detective does not select the irradiation in the case of Plant 2 and Plant 3.

| Set | System | Target | $R^2$ score (Training) | $R^2$ score (Validation) | $R^2$ score (Test) | MEA (Test) | RMSE (Test) | Nr. Alarms | True Positives |
|---|---|---|---|---|---|---|---|---|---|
| 0 | Plant 1 | Coll. Temp. 02 | 1.00 | 1.00 | 0.99 | 0.52 | 1.70 | 17 | 100% |
| 1 | Plant 1 | Coll. Temp. 02 | 1.00 | 1.00 | 0.99 | 0.77 | 1.96 | 39 | 77% |
| 2 | Plant 1 | Coll. Temp. 02 | 1.00 | 1.00 | 0.99 | 0.81 | 2.02 | 32 | 56% |
| 3 | Plant 1 | Coll. Temp. 02 | 1.00 | 1.00 | 0.99 | 0.61 | 1.79 | 23 | 70% |
| 4 | Plant 1 | Coll. Temp. 02 | 0.99 | 1.00 | 0.99 | 0.83 | 2.02 | 32 | 53% |
| 5 | Plant 2 | Coll. Temp. 63 | 0.97 | 0.97 | 0.96 | 1.99 | 3.79 | 66 | 21% |
| 6 | Plant 2 | Coll. Temp. 63 | 0.98 | 0.98 | 0.96 | 1.91 | 3.96 | 104 | 6% |
| 7 | Plant 2 | Coll. Temp. 63 | 0.97 | 0.94 | 0.91 | 2.96 | 5.51 | 103 | 18% |
| 8 | Plant 2 | Coll. Temp. 63 | 0.97 | 0.97 | 0.96 | 1.98 | 3.95 | 113 | 8% |
| 9 | Plant 2 | Coll. Temp. 63 | 0.98 | 0.98 | 0.97 | 1.67 | 3.01 | 120 | 25% |
| 10 | Plant 3 | Coll. Temp. 05 | 0.97 | 0.98 | 0.98 | 1.54 | 2.92 | 121 | 0% |
| 11 | Plant 3 | Coll. Temp. 05 | 0.99 | 0.98 | 0.98 | 1.42 | 2.99 | 126 | 0% |
| 12 | Plant 3 | Coll. Temp. 05 | 0.99 | 0.98 | 0.98 | 1.43 | 2.97 | 108 | 0% |
| 13 | Plant 3 | Coll. Temp. 05 | 0.98 | 0.96 | 0.95 | 2.76 | 4.80 | 167 | 0% |
| 14 | Plant 3 | Coll. Temp. 05 | 0.98 | 0.97 | 0.97 | 2.12 | 3.74 | 100 | 0% |
| 15 | Plant 1 | Solar Power | 0.95 | 0.96 | 0.95 | 60.05 | 136.78 | 66 | 86% |
| 16 | Plant 1 | Solar Power | 0.96 | 0.95 | 0.94 | 69.42 | 146.53 | 59 | 98% |
| 17 | Plant 1 | Solar Power | 0.95 | 0.96 | 0.95 | 54.34 | 132.17 | 60 | 100% |
| 18 | Plant 1 | Solar Power | 0.95 | 0.95 | 0.95 | 52.53 | 132.56 | 68 | 97% |
| 19 | Plant 1 | Solar Power | 0.95 | 0.96 | 0.95 | 51.25 | 132.50 | 72 | 100% |
| 20 | Plant 2 | Solar Power | 1.00 | 1.00 | 1.00 | 5.98 | 20.73 | 4 | 75% |
| 21 | Plant 2 | Solar Power | 0.98 | 0.96 | 0.94 | 32.92 | 86.16 | 17 | 53% |
| 22 | Plant 2 | Solar Power | 0.98 | 0.96 | 0.94 | 33.55 | 84.05 | 18 | 61% |
| 23 | Plant 3 | Solar Power | 0.99 | 0.99 | 0.99 | 7.25 | 29.84 | 0 | 100% |
| 24 | Plant 3 | Solar Power | 0.95 | 0.96 | 0.94 | 35.71 | 75.31 | 5 | 0% |
| 25 | Plant 1 | Flow Temp. | 1.00 | 1.00 | 1.00 | 0.31 | 1.51 | 15 | 93% |
| 26 | Plant 1 | Flow Temp. | 0.99 | 0.99 | 0.97 | 1.61 | 3.60 | 93 | 31% |
| 27 | Plant 1 | Flow Temp. | 0.99 | 0.99 | 0.98 | 1.56 | 3.32 | 139 | 65% |
| 28 | Plant 1 | Flow Temp. | 0.97 | 0.98 | 0.97 | 1.72 | 3.97 | 127 | 60% |
| 29 | Plant 1 | Flow Temp. | 0.98 | 0.98 | 0.96 | 1.70 | 4.15 | 138 | 57% |
| 30 | Plant 2 | Flow Temp. | 1.00 | 1.00 | 1.00 | 0.19 | 0.47 | 0 | 100% |
| 31 | Plant 2 | Flow Temp. | 0.96 | 0.94 | 0.91 | 1.27 | 2.09 | 168 | 24% |
| 32 | Plant 2 | Flow Temp. | 0.96 | 0.92 | 0.86 | 1.47 | 2.61 | 118 | 28% |
| 33 | Plant 2 | Flow Temp. | 0.96 | 0.93 | 0.88 | 1.47 | 2.37 | 133 | 26% |
| 34 | Plant 2 | Flow Temp. | 0.96 | 0.96 | 0.93 | 1.18 | 1.88 | 109 | 34% |
| 35 | Plant 3 | Flow Temp. | 0.99 | 0.99 | 0.99 | 0.58 | 1.39 | 21 | 5% |
| 36 | Plant 3 | Flow Temp. | 0.97 | 0.95 | 0.95 | 1.92 | 3.17 | 120 | 0% |
| 37 | Plant 3 | Flow Temp. | 0.97 | 0.95 | 0.95 | 1.93 | 3.12 | 137 | 1% |
| 38 | Plant 3 | Flow Temp. | 0.96 | 0.94 | 0.95 | 1.85 | 2.90 | 94 | 0% |
| 39 | Plant 3 | Flow Temp. | 0.97 | 0.93 | 0.92 | 2.26 | 3.75 | 128 | 0% |
| Average | | | 0.98 | 0.97 | 0.96 | | | 79.5 | 30% |

**Fig. 12.** Results for the modeling accuracy of each correlated sensor set, showing the coefficient of determination ($R^2$) score for the initial first week of data (Training), the validation set (Validation), and the test set (Test). In addition, the mean absolute error (MAE) and the root mean squared error (RMSE) for the test set are shown. Finally, the number of alarms raised by *Fault-Detective* and the percentage of alarms corresponding to verified faults are also shown.

To predict the flow temperature, *Fault-Detective* identifies various sensor sets. For example, *Fault-Detective* correctly recognizes the redundant sensors installed for this measurement at all three plants. As the sensors are positioned next to each other, the resulting model yields very high scores of $R^2$=0.999.

In addition, a lot of collector and storage temperatures are selected to model the solar primary flow temperature. That is expected, as the flow temperature is a mixture of the individual collector rows. Similarly, the storage temperatures often serve as system control setpoints for controlling the pumps.

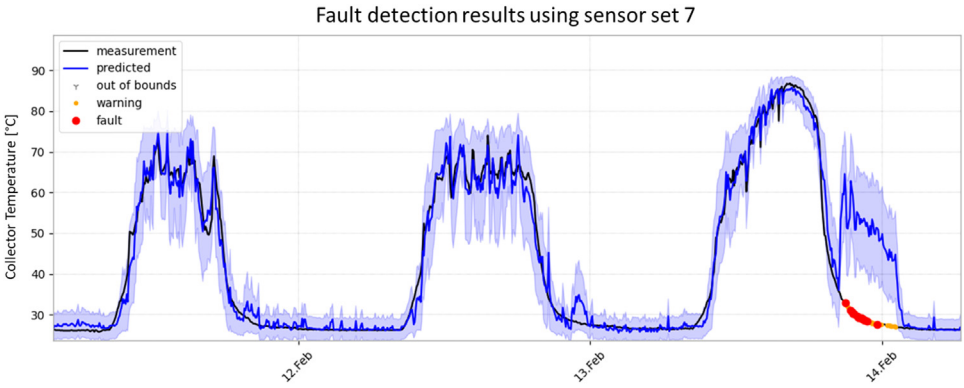


**Fig. 13.** Prediction results using sensor set 07 targeting the collector temperature of Plant 2. A false-alarm is raised on the third day (red dots) due to natural circulation affecting the collector temperature 42, which is used as the input sensor.

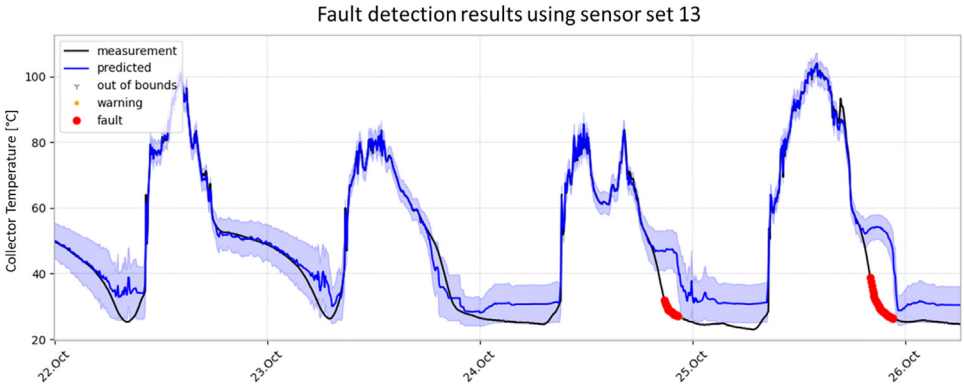


**Fig. 14.** Prediction results using sensor set 13 for targeting the collector temperature at Plant 3. On the third and the fourth day, the cooldown of some collectors is delayed, leading to false alarms and incorrect predictions.

In summary, all the feature sets identified by *Fault-Detective* are reasonable and yield good modeling results on the training dataset with values above $R^2$>0.95.

### 5.2. Modeling accuracy

As discussed in the Evaluation Method (Section 4), walk-forward validation is used for testing *Fault-Detective's* performance. Hence, the first month of data is used for the initial training of the Random-Forest-Regressors, using the identified features of the Feature-Selection. Next, the Fault-Detection step is carried out using daily chunks of data. After each day, the past seven days are used for retraining the algorithms.

Fig. 12 shows the accuracy scores of all individual regressors in terms of the coefficient of determination ($R^2$), the mean-absolute-error (MAE), and the root-mean-squared-error (RMSE). In order to compute these scores, only data from fault-free system behavior was used. Hence, verified events influencing the system behavior (i.e., faults and services verified by the monitoring personnel) were excluded from the data to compute the scores.

For most sensor sets, the $R^2$ scores for the validation and test dataset are similar to those during the Feature-Selection, which indicates that the algorithm can extrapolate to unseen data well. In addition, the MAE and RMSE scores are low, indicating that the models can predict the respective target sensor sufficiently well. However, the following sensor sets perform considerably worse on the validation and test sets compared to the training:

In the case of set 07, the temperature of collector 63 is predicted based on collector 42. During some days, the fluid in the pipes shifts due to natural circulation. When the fluid passes the temperature sensors inside the pipes, the sensor registers a significant temperature change. In contrast, other collector sensors are not affected at all. This effect is somewhat unpredictable and hence cannot be modeled by Fault-Detective well (see Fig. 13).

An example of such a case can be seen in Fig. 13. The differences between the scores for the test set ($R^2$=0.91) and the validation set ($R^2$=0.94) can be explained as more of these events happened in the year's second half, coincidently.

A very similar phenomenon can also be recognized in the case of sensor-set 13, which targets the collector temperature 05 of Plant 3. Shortly after the pumps are switched off each day, the collector and pipes start to cool down. However, hot fluid sometimes shifts towards the temperature sensor, leading to a slight temperature increase. As a result, the cooldown at some sensors is delayed by a short time, leading to high prediction errors (see Fig. 14). The same issue also affects sensor sets 5–14 and 30–39 and, unfortunately, often leads to false alarms.

In addition, there is a drop in prediction accuracy in almost all sensor sets targeting the flow temperature of Plant 2, namely sets 31–34. The reasons for the low scores are periods with concurrent days of bad weather, leading to a slight decrease in temperature in the storage and pipes. Unfortunately, this rare event (happening only twice in one year) cannot be interpreted correctly by *Fault-Detective* (see Fig. 15).

Finally, sensor set 39 targeting the flow temperature at Plant 3 also yields low accuracy scores. One reason is that the primary solar pump is sometimes switched on in the afternoon for a short time. As a result, the flow temperature decreases rapidly as the collector's cooled-down fluid reaches the sensor. Again, *Fault-Detective* fails to predict this phenomenon correctly (see Fig. 16).

Apart from these issues, the Random-Forest-Regressors also often fail to correctly predict the system's behavior when the pump starts for the first time of the day. Due to the already hot collector temperatures and the low temperatures in the pipes, high values for the solar power can occur. Similarly, collectors are suddenly cooled down by the cold return flow, while the hot temperature from the collectors passes the flow temperature sensors. While *Fault-Detective* can model these effects to some extent, the exact temperatures and the specific timing of these changes cannot always be predicted by the algorithm well (see Fig. 15).

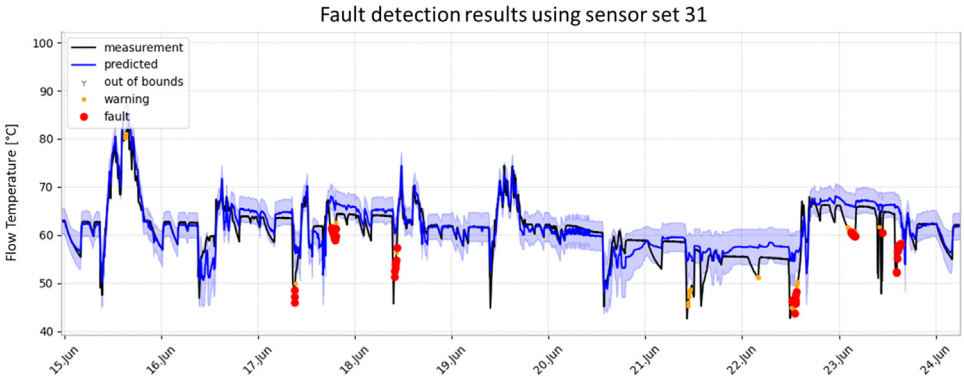


**Fig. 15.** Prediction results for sensor set 31 for Plant 2. Starting on the 20th of June, a series of bad-weather days occurs, leading to false alarms. In addition, the pump start-up in the morning leads to false alarms between the 17th and 19th of June.

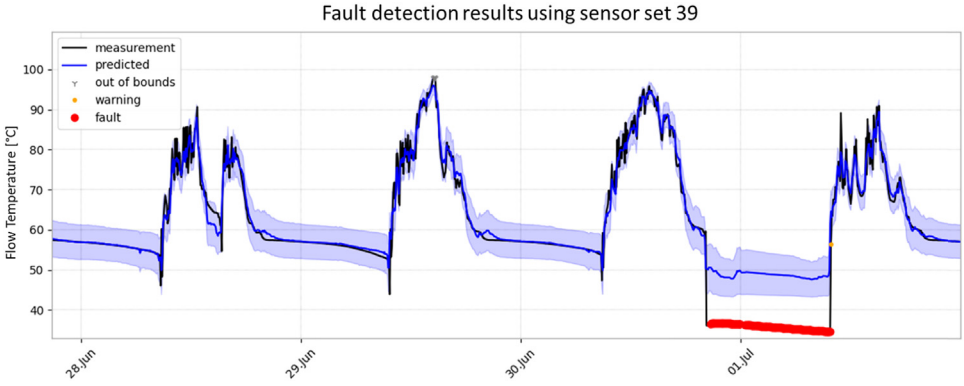


**Fig. 16.** Prediction results for senor set 39 targeting the flow temperature of Plant 3. On some days, the primary solar pump is switched on in the afternoon for a short time, decreasing the flow temperature and leading to false alarms.

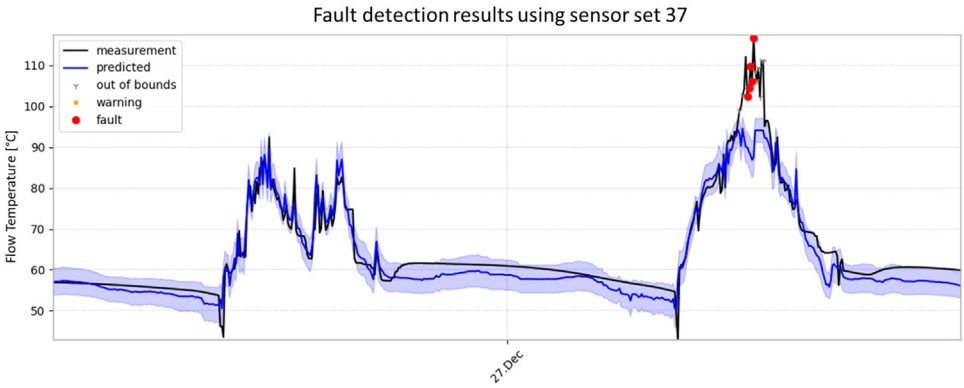


**Fig. 17.** Correct detection of Limited Extraction event via sensor set 37, which targets the flow temperature of Plant 3. Due to the limited heat extracted from the storage, the collector temperatures increase for a short while. The anomaly was spotted even though the temperatures did not reach critical levels.

Nevertheless, the overall accuracy of the Random-Forest-Regressors is high, with an average score of $R^2$=0.96 on the test set and good visual overlap between predictions and measurements, especially during nominal system operation.

### 5.3. Fault-detection accuracy

In addition, the ability to detect faults is evaluated as well. Table 5 lists the fault and service events that were detected either by manual fault detection or with the help of *Fault-Detective*. The results show that all faults identified by the monitoring personnel were also detected by at least one Random-Forest-Regressor. Surprisingly, there are even some faults that the domain experts missed. For example, *Fault-Detective* identified "Limited Extraction" on the 27th of December at Plant 3, where the consumer side of the system extracts abnormally little energy from the primary circuit. As a result, the collector, flow-, and return temperatures increase. While the monitoring personnel missed these slight changes, *Fault-Detective* managed to identify the abnormal system behavior (see Fig. 17). Similarly, the faults "High Collector Temperatures" and "High Collector Temperatures (Subfield)" were detected successfully (see Figs. 18 and 19).

However, despite these correct detections, the statistics in Fig. 11 clearly show that the precision of most regressors is poor, with many false positives being raised. Especially the sensors-sets targeting the collector temperatures and flow temperatures of Plant 2 and Plant 3 raise many false alarms. The reason for this is the low prediction accuracy for the behavior during the "start-up" of the pumps (see Fig. 14) and the different cooldown behavior of the collector temperatures (Fig. 12). In contrast, targeting the thermal power sensor seems to be very successful, with an overall precision of 91%.

## 6. Discussion

The results of Fault-Detective are mixed. On the positive side, each algorithm step performed well. The Feature-Selection successfully identified correlated sensors. The resulting sets are both reasonable from a physical point of view and show good modeling performance. Similarly, the Algorithm-Training and Retraining step modeled typical system be-

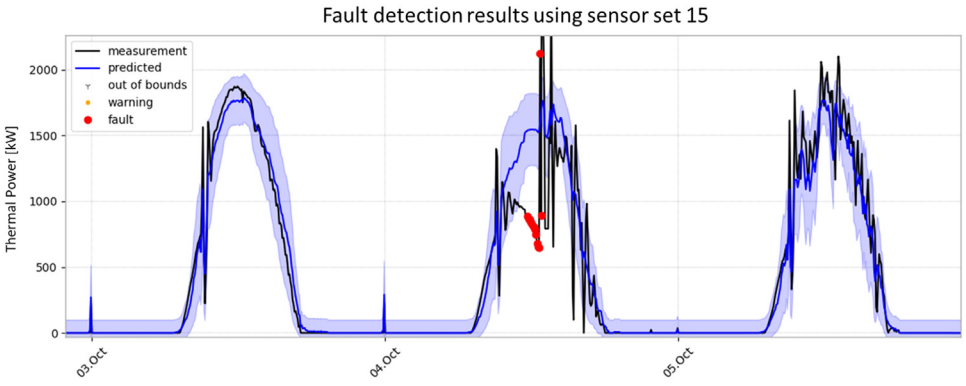


**Fig. 18.** High collector temperatures (all collectors) event detected with sensor set 15 targeting solar power of Plant 1. On the second day, heat cannot be extracted for a short period of time, increasing the collector temperatures and decreasing the thermal power.

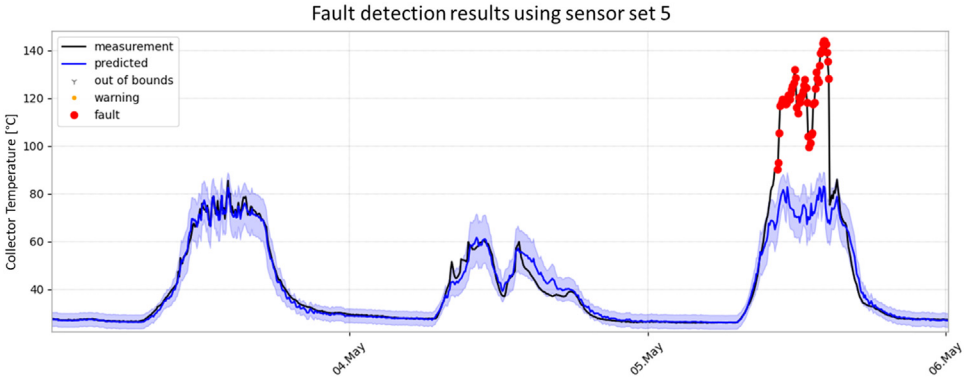


**Fig. 19.** High collector temperatures (subfield) event detected via sensor set 5 targeting the collector temperature. On the third day, the temperatures of collector 63 are rising, while collector 33 remains at low-temperature levels. *Fault-Detective* correctly raises the alarm even before temperatures above 90 °C occur.

havior well, resulting in high $R^2$ scores. In addition, the Fault-Detection step allowed detecting all faults spotted by the monitoring personnel. The algorithm even identified some faults that were missed by the expert. The evaluation also backs up the claim that Fault-Detective can be flexibly applied to different plants since the results of all data sets are similar. In summary, promising results are achieved targeting the thermal power, with a precision of 91% and almost no false alarms raised.

However, many false alarms are raised when targeting the temperature sensors (see statistics in Fig. 12). Hence, Fault-Detective cannot be used as-is for fault detection, as the false alarms would create much overhead for the personnel. While results from [41] indicate that similar false-positive rates are also expected using other FD methods, the evaluation still suggests that only thermal power can be sufficiently monitored by *Fault-Detective* in an *in-situ* setting.

Interestingly, most of the false alarms can be explained by abnormal system operation (e.g., unpredictable natural circulation, pumps starting late at night), momentarily outlier (e.g., system start-up), or other rare events (e.g., consecutive days with bad weather). That suggests that *Fault-Detective* works well in learning typical system behavior but fails to distinguish anomalies from faults. Hence, similar false alarms may also be raised by other identification-based methods that attempt to detect faults by modeling typical system behavior. In addition, the results reinforce the importance of testing fault detection algorithms with measurement data, as most false alarms were caused by system behaviors rarely modeled by simulations.

In order to limit false alarms, future work might use clustering approaches to group similar faults together. Hence, a user must only diagnose similar anomalies once, while new faults belonging to the same category are automatically labeled. This would considerably speed up the monitoring personnel's work and severely limit the impact of false alarms. In addition, future studies might also investigate interactive approaches, for example, by letting the user interact with *Fault-Detective* in a visual interface to monitor solar-thermal systems.

Furthermore, both the Feature-Selection and the Algorithm-Training might be improved:

Currently, Random-Forest-Regressors are used to model the behavior of solar-thermal systems. However, the evaluation suggests that seasonal changes in the data and rare events like multiple consecutive bad weather days lead to mispredictions. This could be improved using different sampling approaches, more extensive training periods, or different machine learning algorithms. For example, long-short-term-memory LSTM networks do not suffer from extrapolation issues and might yield better results. They also allow for retraining the algorithm without losing information from historical data and automatically can handle temporal dependencies in the data.

In addition, some expected correlations were not identified by the Feature-Selection step. For example, the irradiation sensor is not selected for predicting the thermal power at some plants, although they are correlated. The reason might be that some time is needed until the irradiation affects the thermal power due to the high inertia of the fluid and heat capacity of collectors and pipes. However, temporal dependen-

**Table 1**
Additional details about the datasets used to validate and test *Fault-Detective*.

| Name | Coll. Area | Nr. of Sensors | Sampling Rate | Nr. of Timesteps | Used for |
|---|---|---|---|---|---|
| Plant 1 | > 1000 $m^2$ | 113 | 5 min | 105,120 | Validation & Testing |
| Plant 2 | > 1000 $m^2$ | 171 | 5 min | 105,120 | Validation & Testing |
| Plant 3 | > 1000 $m^2$ | 86 | 5 min | 105,120 | Testing only |

**Table 2**
Results for Fault-Detection showing the types of faults in the datasets. In addition, the number of occurrences and the percentage of correctly detected faults via manual and automatic fault detection are shown for each category.

| Event Name | Description | Amount | Detected by Fault-Detective | Detected Manually |
|---|---|---|---|---|
| High collector temperatures | The temperature of most collectors is higher than 120 °C. | 3 | 100% | 100% |
| High collector temperatures (subfield) | The temperature of some collectors is higher than 120 °C. | 2 | 100% | 100% |
| Limited Extraction | Energy extraction is limited due to a fault on the consumer side. | 5 | 100% | 0% |
| Pump speed to low | The volume flow is too low. | 4 | 100% | 0% |
| Constant Data | The data logger records no or constant data. | 8 | 100% | 88% |
| Service Conducted | A service/repair is conducted that affects the system. | 7 | 100% | 100% |
| System Settings Changed | System control of the solar system changed permanently | 5 | 80% | 80% |

cies are currently not accounted for in the Feature-Selection step. Hence, future work could investigate using Input-lagging or similar techniques for the Feature-Selection to improve the results.

## 7. Conclusion

This work introduced a new fault detection algorithm called *Fault-Detective,* a purely data-driven approach requiring no domain knowledge of the solar thermal system. The algorithm was *extensively validated* using the data from three large-scale solar thermal systems, targeting a collector temperature sensor, a flow temperature sensor, and thermal power. The results are compared to manual fault detection performed by solar thermal experts (Tables 1 and 2).

The evaluation shows that *Fault-Detective* can adapt to multiple system layouts without any adaption, proving the *algorithm's flexibility and ease of configuration*. In addition, all faults found by the monitoring personnel could also be identified by *Fault-Detective*, proving the *high fault coverage* of the algorithm. The algorithm could detect even some additional faults that were missed by the experts. However, many false alarms are raised when targeting temperature sensors, as *Fault-Detective* cannot differentiate between faults and other (less severe) anomalies in the system data. Nevertheless, promising results are achieved targeting the thermal power, with a precision of 91% and few false alarms raised. While further studies are necessary, the work thus concludes that *Fault-Detective* is promising to support the monitoring personnel during monitoring thermal power.

## Declaration of Competing Interest

Lukas Feierl reports a relationship with SOLID Solar Energy Systems GmbH that includes: employment. Bernhard Gerardts reports a relationship with SOLID Solar Energy Systems GmbH that includes: employment. Viktor Unterberger reports a relationship with BEST - Bioenergy and Sustainable Technologies GmbH that includes: employment. Claudio Rossi reports a relationship with LINKS Foundation that includes: employment. Manuel Gaetani reports a relationship with LINKS Foundation that includes: employment. This work is based on the results of the Ship2Fair project and has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 792276.

## Acknowledgments

This paper is based on the results of the Ship2Fair project and has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 792276.

## References

[1] M.W. Ahmad, J. Reynolds, Y. Rezgui, Predictive modelling for solar thermal energy systems: a comparison of support vector regression, random forest, extra trees and regression trees, J. Clean. Prod. 203 (2018) 810–821, doi:10.1016/j.jclepro.2018.08.207.
[2] H. Altgeld, 1999. Funktionskontrollen bei kleinen thermischen Solaranlagen ohne Wärmemengenmessung (Forschungsbericht No. 1999).
[3] T. Beikircher, N. Benz, M. Gut, H. Drück, A short term test method for large installed solar thermal systems, in: Proceedings of the ISES Solar World Congress, 6, 1999.
[4] L. Börjesson, M. Singull, Forecasting financial time series through causal and dilated convolutional neural networks, Entropy 22 (2020) 1094, doi:10.3390/e22101094.
[5] L. Breiman, Random forests, Mach. Learn. 45 (2001) 5–32, doi:10.1023/A:1010933404324.
[6] C. Correa Jullian, J. Cardemil, E. López Droguett, M. Behzad, Assessment of deep learning algorithms for fault diagnosis of solar thermal systems, in: Proceedings of the ISES Solar World Congress, International Solar Energy Society, Santiago, Chile, 2019, pp. 1–12, doi:10.18086/swc.2019.08.03. Presented at the ISES Solar World Congress 2019/IEA SHC International Conference on Solar Heating and Cooling for Buildings and Industry 2019.
[7] B.F. Darst, K.C. Malecki, C.D. Engelman, Using recursive feature elimination in random forest to account for correlated variables in high dimensional data, BMC Genet. 19 (2018) 65, doi:10.1186/s12863-018-0633-8.
[8] A.C. de Keizer, K. Vajen, U. Jordan, Review of long-term fault detection approaches in solar thermal systems, Sol. Energy 85 (2011) 1430–1439, doi:10.1016/j.solener.2011.03.025.
[9] C. de Keizer, S. Kuethe, U. Jordan, K. Vajen, Simulation-based long-term fault detection for solar thermal systems, Sol. Energy 93 (2013) 109–120, doi:10.1016/j.solener.2013.03.023.
[10] ISO 24194: Solar energy - Collector fields - Check of performance, 2022. ISO.
[11] A. Dröscher, P. Ohnewein, M.Y. Haller, R. Heimrath, Modular specification of large-scale solar thermal systems for the implementation of an intelligent monitoring system, in: Proceedings of the ISES Solar World Congress, 2009, pp. 683–688.
[12] G. Faure, M. Vallée, C. Paulus, T.Q. Tran, Fault detection and diagnosis for large solar thermal systems: a review of fault types and applicable methods, Sol. Energy 197 (2020) 472–484, doi:10.1016/j.solener.2020.01.027.
[13] R. Ferreiro Garcia, J.L.C. Rolle, J.P. Castelo, M.R. Gomez, On the monitoring task of solar thermal fluid transfer systems using NN based models and rule based techniques, Eng. Appl. Artif. Intell. 27 (2014) 129–136, doi:10.1016/j.engappai.2013.06.011.
[14] A.R. Firdawanti, I.M. Sumertajaya, B. Sartono, Random forest lag distributed regression for forecasting on palm oil production, in: Proceedings of the 1st International Conference on Statistics and Analytics, Bogor, Indonesia, EAI, Bogor, Indonesia, 2020 ICSA 2019, 2-3 August 2019, doi:10.4108/eai.2-8-2019.2290493.
[15] M. Georgii, C. Schmelzer, H. Braas, J. Orozaliev, K. Vajen, A flexible software framework for self-adapting algorithm-based fault detection and diagnosis in solar heating systems, in: Proceedings of the ISES Solar World Conference and the IEA SHC Solar Heating and Cooling Conference for Buildings and Industry, International Solar Energy Society, Abu Dhabi, 2017, pp. 1–8, doi:10.18086/swc.2017.19.05.
[16] M. Georgii, C. Schmelzer, C. Sauer, J. Orozaliev, K. Vajen, Digital representation of heating systems for fault detection purposes, in: Proceedings of the EuroSun 2022, 2022, pp. 1–6.
[17] U. Grossenbacher, Qualitätssicherungssystem für Solaranlagen - Methode zur permanenten Funktionskontrolle thermischer Solaranlagen (No. 77269), EnergieBüro Grossenbacher (2003).
[18] C.R. Harris, K.J. Millman, S.J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N.J. Smith, R. Kern, M. Picus, S. Hoyer, M.H. van Kerkwijk, M. Brett, A. Haldane, J.F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, T.E. Oliphant, Array programming with NumPy, Nature 585 (2020) 357–362, doi:10.1038/s41586-020-2649-2.
[19] M.A. Hassan, A. Khalil, S. Kaseb, M.A. Kassem, Exploring the potential of tree-based ensemble methods in solar radiation modeling, Appl. Energy 203 (2017) 897–916, doi:10.1016/j.apenergy.2017.06.104.
[20] H. He, T.P. Caudell, D.F. Menicucci, A.A. Mammoli, Application of adaptive resonance theory neural networks to monitor solar hot water systems and detect existing or developing faults, Sol. Energy 86 (2012) 2318–2333, doi:10.1016/j.solener.2012.05.015.
[21] C. Holter, B. Gerardts, P. Ohnewein, A. Dröscher, F. Feichtner, K. Schgaguler, E. Meißner, P. Luidolt, A. Köstinger, R. Heimrath, W. Streicher, Development of the Prototype IP-solar: a web-based monitoring and diagnostics tool for solar thermal systems, Energy Procedia 30 (2012) 134–143, doi:10.1016/j.egypro.2012.11.017.
[22] S. Jiang, M. Lian, C. Lu, S. Ruan, Z. Wang, B. Chen, SVM-DS fusion based soft fault detection and diagnosis in solar water heaters, Energy Explor. Exploit. 37 (2019) 1125–1146, doi:10.1177/0144598718816604.
[23] S. Kalogirou, S. Lalot, G. Florides, B. Desmet, Development of a neural network-based fault diagnostic system for solar thermal applications, Sol. Energy 82 (2008) 164–172, doi:10.1016/j.solener.2007.06.010.

[24] S.A. Kalogirou, E. Mathioulakis, V. Belessiotis, Artificial neural networks for the performance prediction of large solar systems, Renew. Energy 63 (2014) 90–97, doi:10.1016/j.renene.2013.08.049.
[25] S. Kuethe, A.C. de Keizer, R. Shahbazfar, K. Vajen, Implementation of data processing and automated algorithm based fault detection for solar thermal systems, in: Proceedings of the ISES Solar World Congress 2011, International Solar Energy Society, Kassel, Germany, 2011, pp. 1–8, doi:10.18086/swc.2011.28.16. Presented at the ISES Solar World Congress 2011.
[26] Letz, T., 2002. Validation and background information on the FSC procedure, IEA SHC Task 26 - Solar Combisystems.
[27] Murdock, H.E., Gibb, D., Andre, T., Sawin, J.L., Brown, A., Ranalder, L., Andre, T., Brown, A., Collier, U., Dent, C., Epp, B., Hareesh Kumar, C., Joubert, F., Kamara, R., Ledanois, N., Levin, R., Skeen, J., Sverrisson, F., Wright, G., Passaro, F., Guerra, F., Dwi Sastriani, N.M., Yaqoob, H., Gicquel, S., Hamirwasia, V., Kifukwe, G., Yuan-Perrin, Y., Mayer, T., Williamson, L.E., Budiman, A., Chen, O., Findlay, K., Harris, A., Jones-Langley, J., Urbani, F., Mastny, L., Brumer, L., 2021. Renewables 2021 - Global status report (No. 978-3-948393- 03–8). France.
[28] J.E. Nielsen, IEA-SHC-T55-B-D.2-FACT-SHEET-collector-fields-check-of-performance, IEA SHC Task 55, 2020.
[29] P. Ohnewein, A. Dröscher, K. Schgaguler, F. Feichtner, E. Meißner, P. Luidolt, A. Köstinger, R. Heimrath, M. Jaindl, W. Streicher, IP-solar: development of a web-based monitoring and diagnostics tool for solar thermal systems, in: Proceedings of the EuroSun ISES European Solar Congress, 8, 2010.
[30] P. Ohnewein, H. Schrammel, D. Tschopp, S. Krammer, H. Poier, B. Gerardts, A. Köstinger, A. Weinhappl, METHODIQA - development of a quality assurance methodology for renewable heat systems based on intelligent operational monitoring, Energy Procedia 91 (2016) 376–383, doi:10.1016/j.egypro.2016.06.285.
[31] P. Ohnewein, D. Tschopp, R. Hausner, W. Doll, Dynamic Collector Array Test (D–CAT). Final Report FFG Project 848766 - MeQuSo. Development of Methods For Quality Assessment of Large-Scale Solar Thermal Plants Under Real Operating Conditions, AEE INTEC, Gleisdorf, 2020.
[32] Paerisch, P., Vanoli, K., 2006. Quality assurance of solar thermal systems with the ISFH- Input/Output-Procedure 7.
[33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, Scikit-learn: machine learning in python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[34] D. Pietruschka, Final deliverable report for self-detection on monitoring procedure, IEA-SHC Task 45 (2015).
[35] E. Polyzos, C. Siriopoulos, Autoregressive random forests: machine learning and lag selection for financial research, SSRN Electron. J. (2022), doi:10.2139/ssrn.4118546.
[36] R. Räber, Spektralmethode zur Fehlerfrüherkennung in Wärmetechnischen Anlagen, ETH Zurich, 1997, doi:10.3929/ETHZ-A-001845425.
[37] S. Ruiz-Moreno, A.J. Sanchez, A.J. Gallego, E.F. Camacho, A deep learning-based strategy for fault detection and isolation in parabolic-trough collectors, Renew. Energy 186 (2022) 691–703, doi:10.1016/j.renene.2022.01.029.
[38] C. Schmelzer, M. Georgii, O. Kusyy, J. Orozaliev, K. Vajen, Towards automated continuous performance benchmarking of DHW and combi systems, in: Proceedings of EuroSun 2018, International Solar Energy Society, Rapperswil, CH, 2018, pp. 1–8, doi:10.18086/eurosun2018.01.10. Presented at the ISES EuroSun 2018 Conference –12th International Conference on Solar Energy for Buildings and Industry.
[39] Schmelzer, C., Georgii, M., Kusyy, O., Sauer, C., Orozaliev, J., Vajen, K., 2021. SolarCheck: entwicklung eines einheitlichen Verfahrens und eines anschaulichen Indikators zur Feststellung der Funktionsfähigkeit thermischer Solaranlagen für Trinkwarmwasserbereitung und kombinierte Heizungsunterstützung.
[40] C. Schmelzer, M. Georgii, J. Orozaliev, K. Vajen, Fault detection for solar thermal systems - overall system evaluation or component-oriented approach, in: Proceedings of the ISES EuroSun Conference, International Solar Energy Society, 2020, pp. 1–8, doi:10.18086/eurosun.2020.04.02. 13th International Conference on Solar Energy for Buildings and Industry. Presented at the EuroSun 2020 Online.
[41] C. Schmelzer, M. Georgii, C. Sauer, J. Orozaliev, K. Vajen, Fault detection for solar thermal systems: evaluation and improvement of existing algorithms, in: Proceedings of the EuroSun, 2022, p. 8. 2022.
[42] Schmelzer, C., Georgii, M., Vajen, K., 2015. FeDet – Automatische Fehlerdetektion und Diagnose thermischer Solaranlagen. Kassel, Germany.
[43] C.K. Sun, C.W. Chan, P. Tontiwachwuthikul, Intelligent diagnostic system for a solar heating system, Expert Syst. Appl. 16 (1999) 157–171, doi:10.1016/S0957-4174(98)00068-2.
[44] The Pandas Development Team, 2022. pandas-dev/pandas: pandas. 10.5281/ZENODO.3509134.
[45] Timma, L., Blumberga, D., 2013. Application of artificial neural networks for detection of developing faults in solar combisystems 13.
[46] D. Tschopp, P. Delmas, M. Rhedon, S. Sineux, A. Gonnelle, P. Ohnewein, J.E. Nielsen, Application of Performance Check (PC) Method to Large Collector Arrays. IEA SHC FACT SHEET 55 B-D1.1, IEA SHC, 2021.
[47] V. Venkatasubramanian, R. Rengaswamy, K. Yin, S.N. Ka, A review of process fault detection and diagnosis Part I: quantitative model-based methods, Comput. Chem. Eng. (2003) 19.
[48] C. Zhang, Y. Li, Z. Yu, F. Tian, Feature selection of power system transient stability assessment based on random forest and recursive feature elimination, in: Proceedings of the IEEE PES Asia-Pacific Power and Energy Engineering Conference (APPEEC), IEEE, Xi'an, China, 2016, pp. 1264–1268, doi:10.1109/APPEEC.2016.7779696. Presented at the 2016 IEEE PES Asia-Pacific Power and Energy Engineering Conference (APPEEC).