Control Engineering Practice 164 (2025) 106427

Contents lists available at ScienceDirect

Control Engineering Practice

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

# A fault diagnosis benchmark of technical systems with incomplete data — six solutions☆

Daniel Jung [a,1,*], Erik Frisk [a], Mattias Krysander [a], Anna Sztyber-Betley [b,2], Francesco Corrini [c], Andrea Arici [c], Nicolas Anselmi [c], Mirko Mazzoleni [c], Jiamin Xu [d], Siwen Mo [d], Zixuan Xu [d], Chongpan Yang [d], Zhile Du [d], Hossein Safaeipour [e], Mehdi Forouzanfar [e,f], Vahid Mirahi [e], Anna Pinnarelli [f], Vicenç Puig [g], Qiao Deng [d], Yufei Liu [d], Jiakun Liu [d], Haobin Ke [d], Wanting Zhu [d], Silke Merkelbach [h], Maryam Ahang [i], Homayoun Najjaran [i]

[a] Vehicular Systems, Department of Electrical Engineering, Linköping University, Linköping, SE-581 83, Sweden
[b] Faculty of Mechatronics, Warsaw University of Technology, Warsaw, 02-525, Poland
[c] Department of Management, Information and Production Engineering, University of Bergamo, Via G. Marconi 5, Dalmine (BG), 24044, Italy
[d] School of Automation, Central South University, Changsha, 410083, China
[e] Department of Electrical Engineering, Ahv.C., Islamic Azad University, Ahvaz, Iran
[f] Department of Mechanical Energy and Management Engineering (DIMEG), University of Calabria, Rende, Italy
[g] Institute of Robotics and Industrial Informatics (CSIC-UPC), University of Catalonia, Barcelona, Spain
[h] Advanced Systems Engineering, Fraunhofer IEM, Paderborn, 33102, Germany
[i] Department of Electrical and Computer Engineering, University of Victoria, Victoria, V8W 2Y2, BC, Canada

ARTICLE INFO

*Keywords:*
Fault detection and isolation
Model-based diagnosis
Data-driven fault diagnosis

ABSTRACT

This paper presents a benchmark problem for fault diagnosis of an internal combustion engine that has been formulated and solved. The objective is to design a diagnosis system using and incomplete model information training data that only contains a limited set of fault realizations. Six different solutions to the benchmark, that were presented at the IFAC Safeprocess symposium 2024, are described and evaluated. The contribution of this paper is the benchmark and the presentation of six different solutions in one paper. The paper is intended to provide a starting point for engineers and researchers who work with fault diagnosis and monitoring of technical systems.

## 1. Introduction

Diagnosis systems are designed to monitor the health of a technical system by detecting abnormal behavior and identifying its cause. Based on observations from the system, e.g., sensor and actuator signals, the diagnosis system computes diagnoses, i.e., fault hypotheses. The diagnoses computed by the diagnosis system give important information about system degradation and faulty components, which is then used by, e.g., the control system to select suitable countermeasures (Blanke, Kinnaert, Lunze, & Staroswiecki, 2016). It is therefore important to identify the faulty component as fast and accurately as possible. Misclassifications and falsely rejecting a true diagnosis can

☆ This work has benefited from participation in Dagstuhl Seminar 24031 "Fusing Causality, Reasoning, and Learning for Fault Management and Diagnosis".

* Corresponding author.
*E-mail addresses:* daniel.jung@liu.se (D. Jung), erik.frisk@liu.se (E. Frisk), mattias.krysander@liu.se (M. Krysander), anna.sztyber@pw.edu.pl (A. Sztyber-Betley), francesco.corrini@unibg.it (F. Corrini), a.arici4@studenti.unibg.it (A. Arici), n.anselmi@studenti.unibg.it (N. Anselmi), mirko.mazzoleni@unibg.it (M. Mazzoleni), xujm123@csu.edu.cn (J. Xu), mosiwen@csu.edu.cn (S. Mo), 234612129@csu.edu.cn (Z. Xu), 224612196@csu.edu.cn (C. Yang), duzhile666@126.com (Z. Du), hossein.safaeipour@iau.ac.ir (H. Safaeipour), mehdi.forouzanfar@iau.ac.ir (M. Forouzanfar), vahid.mirahi@iau.ac.ir (V. Mirahi), anna.pinnarelli@unical.it (A. Pinnarelli), vicenc.puig@upc.edu (V. Puig), qiao.deng@csu.edu.cn (Q. Deng), 244611030@csu.edu.cn (Y. Liu), 224611057@csu.edu.cn (J. Liu), haobin.ke@outlook.com (H. Ke), 234611036@csu.edu.cn (W. Zhu), silke.merkelbach@iem.fraunhofer.de (S. Merkelbach), maryamahang@uvic.ca (M. Ahang), najjaran@uvic.ca (H. Najjaran).

[1] This project was partially sponsored by the Swedish research excellence center ELLIIT.
[2] This project was partially funded by a research grant from the Scientific Council of the Discipline of Automation, Electronics, Electrical Engineering and Space Technologies of Warsaw University of Technology, Poland granted in 2023.

https://doi.org/10.1016/j.conengprac.2025.106427
Received 20 February 2025; Received in revised form 19 May 2025; Accepted 23 May 2025
Available online 16 June 2025

result in unnecessary costs and increased hazards since the correct action is not taken and may also reduce the operator's trust in the diagnosis system.

Developing a diagnosis system is non-trivial and the design must satisfy several performance requirements, e.g.:

- missed detection and false alarm rate,
- time before detection, and
- fault isolation accuracy.

Selecting a suitable diagnosis system solution for a given application depends on the properties of the system and the faults to be monitored but also available information, e.g., models and training data. Fault diagnosis is complicated by, e.g., model inaccuracies, measurement noise, and limited training data from relevant fault scenarios (Dai & Gao, 2013; Theissler, Pérez-Velázquez, Kettelgerdes, & Elger, 2021). Engineers that are designing diagnosis systems must handle these complicating factors to fulfill the desired performance requirements (Zio, 2022).

To address these challenges, a benchmark problem has been developed and provided to the scientific community. The purpose of the benchmark is to have an industrially relevant system with realistic conditions for designing a diagnosis system, focusing on aspects not included in previously published benchmarks. In particular:

- Faults are rare events. It is costly, and sometimes not even possible, to conduct experiments to collect data from relevant fault realizations. This means that training data consists of a limited number of realizations of each fault type. Thus, the available training data is not representative of the actual data distribution of the different fault classes.
- There is access to some engineering knowledge about the system to be monitored, e.g., schematics or basic model structures derived from physical insights. The available models can be incomplete, e.g. parameter values are not available, or have been derived for different purposes than fault diagnosis, e.g., control design.

Because of the different types of uncertainties, the developer must use all available information about the system to design a diagnosis system that fulfills the performance requirements.

### 1.1. Related research

Different fault diagnosis approaches have been proposed in the scientific community, such as model-based diagnosis and data-driven diagnosis, e.g., Gao, Cecati, and Ding (2015), Tidriri, Chatti, Verron, and Tiplica (2016). Several survey papers present the current state of the art with a focus on either a specific design approach — such as model-based diagnosis and signal-based fault diagnosis (Gao et al., 2015), data-driven fault diagnosis (Abid, Khan, & Iqbal, 2021; Xu & Saleh, 2021) — or a specific application area, such as vibration-based condition monitoring (Stetco et al., 2019), batteries (Xiong, Sun, Yu, & Sun, 2020), traction systems in high-speed trains (Chen, Jiang, Ding, & Huang, 2020), HVAC systems (Mirnaghi & Haghighat, 2020), automotive systems (Theissler et al., 2021), and wind turbines (Liu & Zhang, 2020; Stetco et al., 2019). While these surveys give an overview of existing fault diagnosis methods it is necessary to understand when it is suitable to use a specific fault diagnosis solution.

One approach to address this need is the use of academic and industrial benchmarks to evaluate and compare different fault diagnosis methods. Melo, Câmara, Clavijo, and Pinto (2022) provide a summary of existing open benchmarks for process monitoring and fault diagnosis. Their survey discusses realistic simulators such as the Tennessee Eastman Process (TEP), which models an industrial chemical process (Downs & Vogel, 1993). A simulation-based windmill benchmark for fault-tolerant control was proposed by Odgaard, Stoustrup, and Kinnaert (2013). A review and comparison of six different diagnosis system designs submitted to an FDI competition based on this benchmark is presented in Odgaard and Stoustrup (2012). While such benchmarks are useful for comparing diagnostic solutions, they may not fully capture the challenges associated with limited training data and incomplete model information, since the provided simulation model can be leveraged to generate new training data or extract the underlying model of the simulated system.

Other benchmarks provide measurements from real systems — e.g. the DAMADICS benchmark (Bartyś, Patton, Syfert, de las Heras, & Quevedo, 2006) and the Case Western Reserve University (CWRU) data (see Wu, 2024). The PHM society has also organized various fault diagnosis and prognostics related data challenges (see Jia, Huang, Feng, Cai, & Lee, 2018; Su & Lee, 2024). The DAMADICS benchmark offers both a simulation model and real data from an industrial actuator system used in a sugar factory, along with a set of defined performance criteria. The CWRU dataset is widely used for bearing diagnostics and provides experimental vibration data from both nominal and faulty bearings — making it a popular benchmark for condition monitoring research (see, e.g., Smith & Randall, 2015). Feiyi and Jinsong (2015) reviewed several fault diagnosis system designs developed for the NASA Advanced Diagnostics and Prognostics Test-bed (ADAPT), which simulates an electrical power system in an aerospace vehicle (see Poll et al., 2007).

To address challenges such as lack of representative data and limited model information, a new industrial benchmark has been proposed. The case study focuses on the air path of a four-cylinder spark-ignited internal combustion engine (Jung, Frisk, & Krysander, 2024). The system is characterized by nonlinear dynamic behavior, including both fast and slow dynamics as well as feedback loops. Data from the engine test bench has been used previously, see, e.g., Jung (2022). Building on this previous work, the benchmark includes a detailed definition and motivation, along with a comparison of different diagnostic solutions.

To encourage the development and comparison of fault diagnosis solutions, the benchmark provides both operational data from nominal and faulty operation and a mathematical model of the system without known parameters. The available actuator and sensor signals correspond to the standard signals found in commercial vehicles. The benchmark includes faults such as leakages and sensor faults. The primary challenge is to develop a diagnosis system capable of identifying abnormal system behavior using limited model information and training data that may not represent all relevant fault scenarios.

### 1.2. Contributions

The contributions of this paper are as follows: The first contribution is the formulation of a benchmark problem that illustrates the complicating factors of many industrial applications. The benchmark is described and how its formulation relates to the challenges when designing a diagnosis system. The benchmark provides a mathematical model of the system with unknown parameters and a number of datasets representing different fault scenarios. All material is available online.[3] The model is implemented using the Fault Diagnosis Toolbox, available in Matlab and Python (Frisk, Krysander, & Jung, 2017).

The second contribution is a presentation, comparison, and discussion of the different solutions to the benchmark problem and the conclusions that can be drawn from that. This paper provides a compilation of the six contributions submitted to the benchmark challenge at the 12th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes (Safeprocess), 2024. These solutions are collected and compared in this paper with respect to the criteria that are provided in the benchmark description. Although there was a single problem formulation, the provided solutions show a big variation in solution strategies and how to address the complicating factors and compute reliable diagnoses. This can provide an initial reference for practicing engineers and researchers in fault diagnosis.

[3] https://vehsys.gitlab-pages.liu.se/diagnostic_competition/

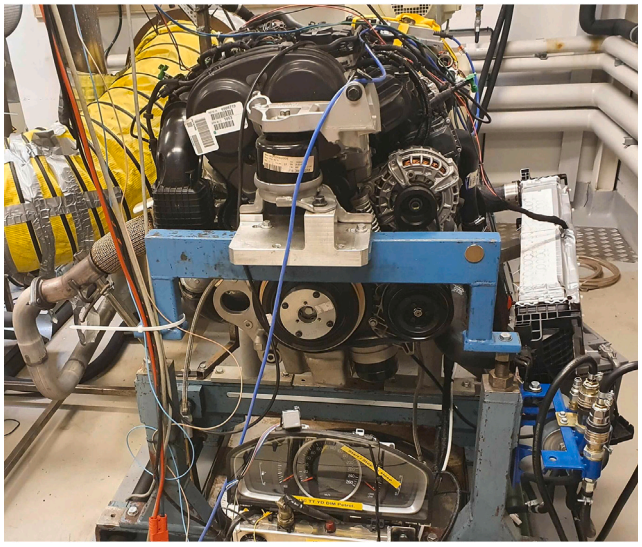

**Fig. 1.** Engine test bench used for data collection.

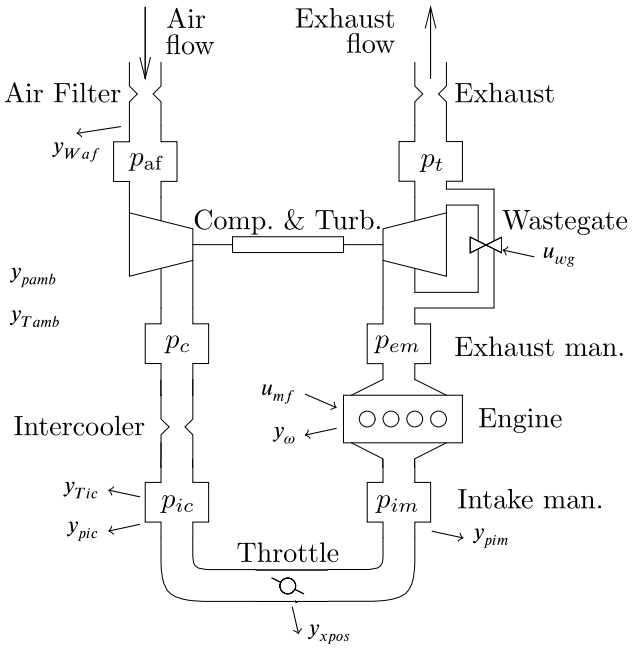


**Fig. 2.** A schematic of the model of the air flow through the engine.

### 1.3. Content of paper

The outline of the paper is as follows. Section 2 describes the benchmark including the provided data. In Sections 3–8 the solutions are described in detail. In Section 9 the different solutions are compared and discussed. Finally, conclusions and discussions about future activities are presented in Sections 10 and 11.

## 2. The LiU-ICE benchmark

The benchmark is based on a commercial four-cylinder internal combustion engine from Volvo Cars, operated on a test bench using MATLAB/Simulink (Jung et al., 2024), see Fig. 1. A Simulink-based vehicle and driver model provides reference speed trajectories from standardized driving cycles to create both static and dynamic operating conditions. Data collection and real-time fault injection are handled via an ETAS INCA system.

### 2.1. System description

Fig. 2 shows a schematic of the engine air path. The engine uses a throttle to regulate intake air, which mixes with fuel in the cylinders to generate torque. Exhaust gases drive the turbocharger via a turbine, with a wastegate controlling the flow. The engine control unit (ECU) manages torque output and maintains optimal air–fuel ratios to reduce emissions, making this subsystem key for emission control.

The engine's complex, nonlinear dynamics — featuring both fast and slow modes, feedback loops, and coupled intake/exhaust flows — make it well-suited for fault diagnosis studies, especially fault isolation. Fault effects are typically non-local and can influence multiple subsystems depending on operating conditions.

The dataset includes the following eight sensor signals:

- $y_{\text{pic}}$ — intercooler pressure
- $y_{\text{pim}}$ — intake manifold pressure
- $y_{\text{pamb}}$ — ambient pressure
- $y_{\text{Tic}}$ — intercooler temperature
- $y_{\text{Tamb}}$ — ambient temperature
- $y_{\text{Waf}}$ — air mass flow after air filter
- $y_\omega$ — engine speed
- $y_{\text{xpos}}$ — throttle position

and the two actuator signals:

- $u_{wg}$ — wastegate position
- $u_{mf}$ — injected fuel mass into the cylinders

### 2.2. Data description

A number of datasets have been collected from the engine test bench representing different fault scenarios. Sensor and actuator signals have been collected from realistic operating conditions. An example is shown in Fig. 3. Here, the realistic driving scenario is represented by the Worldwide Harmonized Light Vehicles Test Procedure (WLTP) (European Community, Japan, & United States of America, 2009) that is approximately 30 min long and include both urban and highway driving patterns, see Fig. 4. Data is sampled at 20 Hz and stored in a CSV-format.

Training data include one fault-free and seven faulty scenarios where each file contains data from one full WLTP driving cycle. The training set covers four hours of data. Faults are introduced at $\sim$ 120 seconds into each dataset and remain for the duration. The considered faults are:

- $f_{\text{ypic}}$ — fault in intercooler pressure ($y_{\text{pic}}$) sensor
- $f_{\text{ypim}}$ — fault the intake manifold pressure ($y_{\text{pim}}$) sensor
- $f_{\text{yWaf}}$ — fault in the air mass flow ($y_{\text{Waf}}$) sensor
- $f_{\text{iml}}$ — leakage in the intake manifold

Sensor faults are introduced in the ECU as multiplicative factors: $y = (1 + f_y)x$ where $y$ is the sensor signal, $x$ is the measured quantity, and the factor $f_y$ can be positive or negative. These faults may influence the system beyond the sensor output due to feedback loops. The leakage is introduced via a manually operated valve with various orifices connected to the intake manifold.

The fault magnitudes that are represented in training data are highlighted in Table 1. Note that in the training data for the fault $f_{\text{ypim}}$, only negative faults are provided and for $f_{\text{yWaf}}$ only positive faults are provided. The remaining fault magnitudes of the different faults in the table have been collected in the same way and represent the test data,

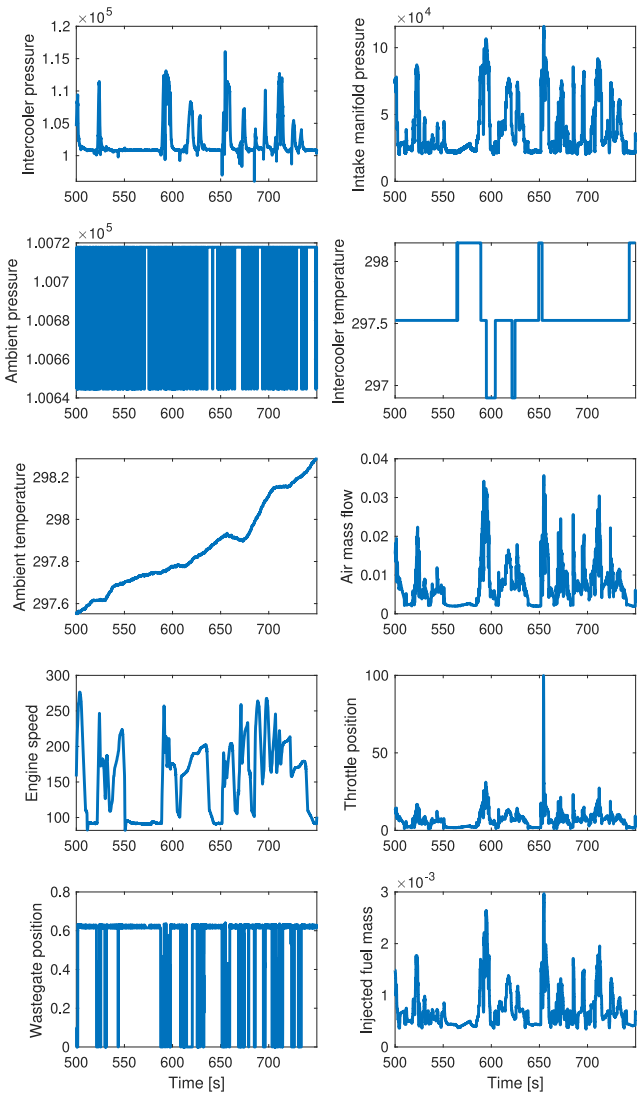


**Fig. 3.** Example of nominal measurement data.

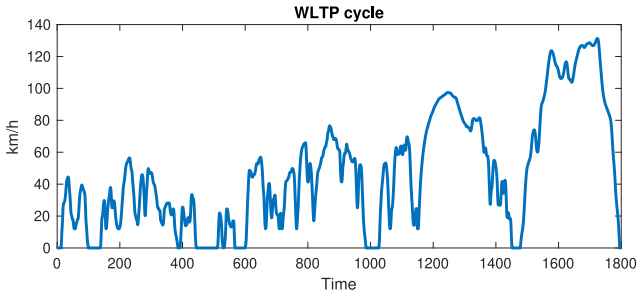


**Fig. 4.** WLTP driving cycle used for data generation.

in total 17 data files (including an additional fault-free scenario) or ∼ 8.5 hours of operational data. Test datasets were released only after the Safeprocess 2024 challenge.

### 2.3. Description of the provided model

The provided model of the system is based on (Eriksson, Frei, Onder, & Guzzella, 2002). The model structure consists of a number of control volumes separated by flow restrictions as illustrated in Fig. 2. A more detailed description of the component models is given in Eriksson et al. (2002). The model is formulated as a set of Differential Algebraic Equations (DAE) which consists of 94 equations of which 14 are dynamic equations. The known faults represented in the datasets are included in the model as signals. The model is implemented using the Fault Diagnosis Toolbox (Frisk et al., 2017) and code is provided for both Matlab and Python. Fig. 5 shows an example of the implemented model in Matlab.

**Table 1**
Summary of datasets with fault scenarios. The highlighted fault magnitudes are represented in training data, the others only in test data.

| Fault | Magnitudes |
|---|---|
| $f_{\text{ypic}}$ | −20%, −15%, −10%, −5%, 5%, 10%, 15% |
| $f_{\text{ypim}}$ | −20%, −15%, −10%, −5%, 5%, 10%, 15% |
| $f_{\text{yWaf}}$ | −20%, −15%, −10%, −5%, 5%, 10%, 15% |
| $f_{\text{iml}}$ | 4 mm, 6 mm |

```
... %Control volume intake manifold:
,p_im == m_im*R_air*T_im/V_im...
,dmdt_im == (W_th - W_ac) + fiml...
,dTdt_im == (W_th*cv_air*(T_imcr-T_im) +R_air*...
    (T_imcr*W_th-T_im*W_ac))/(m_im*cv_air)...
,DiffConstraint(dmdt_im,m_im)...
,DiffConstraint(dTdt_im,T_im)...
...
... %Control volume exhaust manifold:
,p_em == m_em*R_exh*T_em/V_em...
,dmdt_em == W_e - W_twg...
,dTdt_em == (W_e*cv_exh*(T_ti-T_em)+R_exh*(T_ti*W_e-T_em*W_twg))/(
,DiffConstraint(dmdt_em,m_em)...
,DiffConstraint(dTdt_em,T_em)...
```

**Fig. 5.** A screenshot of the implemented model in the Fault Diagnosis Toolbox in Matlab.

A structural representation of the engine model is shown in Fig. 6 where a mark in position $(i, j)$ denotes that the variable $x_j$ is included in equation $e_i$. The model variables are organized in unknown variables including state variables $\mathcal{X}$ (blue marks), known variables $\mathcal{Z}$ (black marks), and fault signals $\mathcal{F}$ (red marks). In the structural model, state variables and their derivatives are treated as separate signals (Frisk et al., 2012). Thus, to explicitly state the relation between a state variable $x_i$ and its derivative $\dot{x}_i$, additional relations have been included in the structural model, see Fig. 5,

$$\dot{x}_i = \frac{d}{dt}(x_i), \tag{1}$$

where the state variables are marked as I and their derivatives as D in Fig. 6.

## 3. Solution, first participant

*Authors:* Anna Sztyber-Betley
*Affiliation:* Warsaw University of Technology
*Contact:* anna.sztyber@pw.edu.pl

### 3.1. Introduction

This section presents the CAREngine (Classification And REsidual analysis for ENGINe diagnosis) algorithm. The algorithm is based on analyzing residuals and diagnostic signals (binary or trivalent signals indicating whether residuals exceed adaptive thresholds). The algorithm uses classifiers to detect and isolate faults. The algorithm is divided into two parts: offline and online. The offline part is responsible for building models, developing adaptive thresholds and training classifiers. The online part is responsible for computing residuals and diagnostic signals for the current sample and running classifiers for fault detection and isolation.

The offline part consists of the following steps:

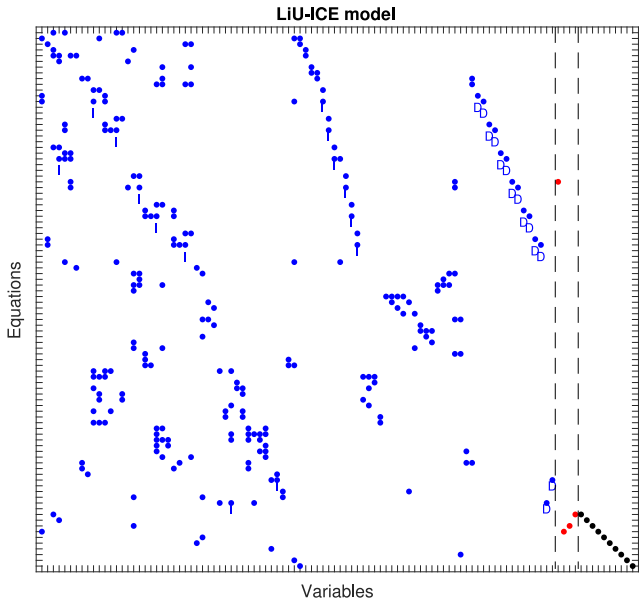


**Fig. 6.** A structural representation of the LiU-ICE model provided in the benchmark. The blue marks represent unknown variables, the black marks are known variables, and the red marks are modeled fault signals.

**Table 2**
Output, inputs and type of the residuals.

| r | Output | Inputs | Model |
|---|---|---|---|
| $r_0$ | $y_{\text{pic}}$ | $y_{\text{pim}}$, $y_{\text{Tic}}$, $u_{mf}$, $u_{wg}$ | neural1 |
| $r_1$ | $y_{\text{pic}}$ | $y_{\text{pim}}$, $y_{\text{Tic}}$, $y_{\text{waf}}$, $u_{wg}$ | neural2 |
| $r_2$ | $y_{\text{pic}}$ | $u_{mf}$ | neural1 |
| $r_3$ | $y_{\text{pim}}$ | $u_{mf}$, $y_{\text{pic}}$ | neural1 |
| $r_4$ | $y_{\text{pim}}$ | $y_{\text{xpos}}$, $y_{\text{pic}}$ | neural1 |
| $r_5$ | $y_{\text{pim}}$ | $y_{\text{waf}}$ | neural1 |
| $r_6$ | $y_{\text{pim}}$ | $y_{\text{xpos}}$ | neural1 |
| $r_7$ | $y_{\text{waf}}$ | $y_{\omega}$, $y_{\text{xpos}}$, $y_{\text{pim}}$ | neural1 |
| $r_8$ | $y_{\text{xpos}}$ | $y_{\omega}$, $y_{\text{pim}}$, $y_{\text{pic}}$ | linear |
| $r_9$ | $u_{wg}$ | $y_{\text{pim}}$, $y_{\omega}$ | neural1 |
| $r_{10}$ | $u_{mf}$ | $y_{\text{pim}}$ | neural2 |
| $r_{11}$ | $u_{mf}$ | $y_{\text{waf}}$, $y_{\omega}$, $u_{wg}$ | neural1 |
| $r_{12}$ | $y_{\text{waf}}$ | $u_{mf}$, $y_{\omega}$ | neural1 |

1. building models of process variables using normal operation data,
2. computing residuals for all models and all training data,
3. developing adaptive thresholds for residuals,
4. computing diagnostic signals for all data and all residuals,
5. training the Quadratic Discriminant Analysis (QDA) classifier to detect faults,
6. training the Linear Support Vector Classifier (LSVC) to isolate faults.

The online part consists of the following steps:

1. computing residuals and diagnosis signals for the current sample,
2. computing smoothed values,
3. running the fault detection classifier,
4. in the case of fault detection, running the fault isolation classifier.

### 3.2. Residuals

Each residual is calculated using the formula: $r_i = y_i - \hat{y}_i$, where $y_i$ represents the measured variable and $\hat{y}_i$ denotes the model output.

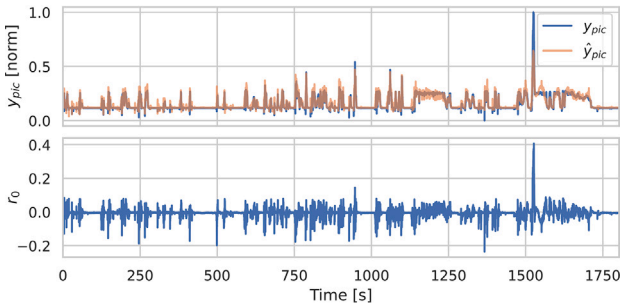


**Fig. 7.** Residual $r_0$: modeled $\hat{y}_{pic}$ and measured value $y_{pic}$ (top) and the value of the residual (bottom).

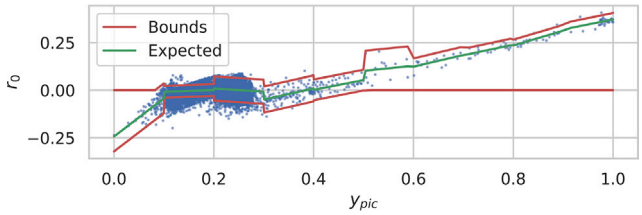


**Fig. 8.** Bounds for residual $r_0$.

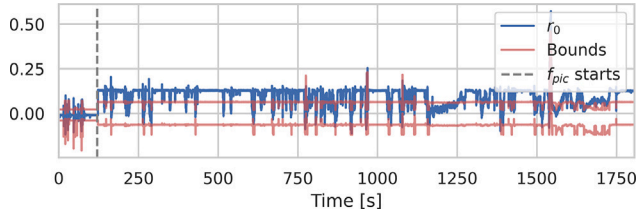


**Fig. 9.** Residual $r_0$ value with bounds for scenario $f_{pic}$ 110.

The models of the predicted values were developed using solely data that did not contain any faults. The residuals and corresponding model inputs, outputs, and types are presented in Table 2. The residuals are calculated using the following models:

- Linear — Ridge regression model using L2 regularization with regularization coefficient $\alpha = 0.01$,
- Neural1 — multilayer perceptron neural network with a hidden layer of 8 neurons, ReLu activation and kernel L2 regularization with regularization coefficient $\alpha = 0.00001$. The model was trained with the Adam optimiser.
- Neural2 — Multilayer perceptron neural network with two hidden layers and parameters analogous to Neural1.

An example of a residual $r_0$ is shown in Fig. 7. It can be observed that the model fit is less precise under dynamic conditions.

Adaptive thresholding was used to determine the bounds of the residuals. Segment linear function was fitted to residual as a function of the measured variable. The bounds were calculated as the expected value plus or minus two standard deviations within a segment. The lower bound was adjusted to be always less than or equal to zero, while the upper bound was adjusted to always be greater than or equal to zero. The bounds for the residual $r_0$ are presented in Fig. 8, while the residual $r_0$ value with bounds for fault scenario $f_{pic}$ 110 is shown in Fig. 9. It can be observed that the residual is outside the specified bounds during the fault.

**Table 3**
Confusion matrix for fault detection.

| True \ Predicted | Fault | No fault |
|---|---|---|
| Fault | 105 940 | 21 573 |
| No fault | 213 | 16 587 |

### 3.3. Detection and isolation classifiers

Quadratic Discriminant Analysis (QDA) was used for the detection classifier. The features used for training were the diagnostic signals, which were binary or trivalent signals indicating whether the residuals exceeded the adaptive thresholds. The diagnostic signals were averaged over an exponentially weighted window with a $\alpha = 0.002$ weighting coefficient. The isolation classifier used a Linear Support Vector Classifier (LSVC) with a Nystroem nonlinear feature transformation containing 50 components. The features used for training were the residuals, which were also averaged in an exponentially weighted window with $\alpha = 0.002$.

The decision function of the isolation classifier was used to determine the classification results. A decision function value of less than zero indicates a negative classification for a given class. A zero value was appended to the decision function values to determine the probability of an unknown fault. Subsequently, the softmax function was applied to extract the fault probabilities. The unknown fault would be assigned the highest rank only if the decision function values for all known faults were below zero.

### 3.4. Remarks on structural methods

The competition data included a structural engine model. Established structural methods, such as Structural Analysis (Blanke, Kinnaert, Lunze, & Staroswiecki, 2015; Frisk et al., 2012; Krysander, Aslund, & Nyberg, 2008) and the Graph of a Process (GP) (Kościelny, Bartyś, Syfert, & Sztyber, 2022; Sztyber, Ostasz, & Kościelny, 2015), were applied to extract residual structure and fault sensitivity. An initial attempt to solve the benchmark by fitting residual generators to normal data and using theoretical fault sensitivity was unsuccessful.

Using the Fault Diagnosis Toolbox (Frisk et al., 2017), all Minimally Structurally Overdetermined Sets (MSO) were computed along with their fault sensitivities and associated measurements. However, identical measurement sets appeared in multiple MSOs with varying fault sensitivities, leading to ambiguous results for data-driven models. Additionally, GP was used to construct a causal graph (assuming integral causality) and to identify partial models (Model Structures, MS). All model structures were found to be sensitive to every fault, preventing fault isolation. These issues were linked to feedback loops and system interconnections.

### 3.5. Results

Using the validation data (last 20% of each file), classifier types and hyperparameters were selected. Tables 3 and 4 show the validation confusion matrices for fault detection and isolation. Detection accuracy was 84.9% (training) and 89.6% (validation). Validation showed many false positives, likely from dynamic end-of-cycle conditions, which were reduced by training on the entire dataset. After retraining, isolation accuracy was 99.7% (training) and 82.5% (validation), with unknown fault classifications at 0.10% (training) and 12.3% (validation).

In addition, an evaluation was conducted on the test data after the competition. As neural networks are sensitive to parameter initialization, the offline part of the algorithm was repeated 10 times to analyze the distribution of results. The results for fault detection and isolation accuracy in the case of $f_{\text{pim}}$ are shown in Fig. 10. It can be observed that the results are positive for the fault size in the same direction as in the training data, while for fault sizes in different directions, the fault isolation results are poor. Furthermore, the results show a significant discrepancy between different runs.

**Table 4**
Confusion matrix for fault isolation.

| True \ Predicted | $f_{\text{iml}}$ | $f_{\text{pic}}$ | $f_{\text{pim}}$ | $f_{\text{waf}}$ |
|---|---|---|---|---|
| $f_{\text{iml}}$ | 16 639 | 226 | 0 | 1141 |
| $f_{\text{pic}}$ | 2206 | 28 606 | 1025 | 4256 |
| $f_{\text{pim}}$ | 11 | 4289 | 26 871 | 4947 |
| $f_{\text{waf}}$ | 857 | 2997 | 126 | 32 079 |

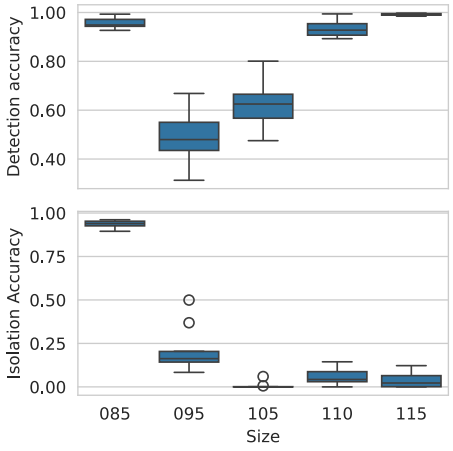


**Fig. 10.** Detection and isolation accuracy for $f_{\text{pim}}$.

### 3.6. Conclusions

The incorporation of signs of the residuals and diagnostic signal makes the algorithm sensitive to the coverage of different directions of fault magnitudes in the training data. Further research will be conducted to analyze augmentation techniques to alleviate this issue. Secondly, the results of different runs of neural network training exhibited significant discrepancies. There is a need for a reliable method of selecting the best model. Furthermore, the structural methodologies employed for residual design did not yield the desired results. Further research is required to ascertain the conditions under which the isolability results of structural methodologies can be relied upon.

## 4. Solution, second participant

*Authors:* Francesco Corrini, Andrea Arici, Nicolas Anselmi, Mirko Mazzoleni
*Affiliation:* University of Bergamo
*Contact:* francesco.corrini@unibg.it

### 4.1. Introduction

The challenge proposed in this competition requires to correctly detect and isolate four types of fault. The solution presented in this section is based on four different Recursive Least Squares (RLS) models (Ljung, 1986). Each model will produce a set of residuals, composed of two time varying coefficients of the respective RLS model. Each residual set is sensitive to a specific subset of faults. The idea is that the behavior of the time varying coefficients is able to capture changes in the working condition of the system. Fault detection and isolation are achieved looking to the behavior of the four residual sets. To do this, four anomaly detection models are developed using open-set classification techniques (Lundgren & Jung, 2022).

### 4.2. Methodologies

#### 4.2.1. Residuals generation

The proposed residual generators are composed by four RLS models, each one with a constant forgetting factor $\mu_i \in \mathbb{R}_{>0}$, $i \in \{1, \dots, 4\}$.

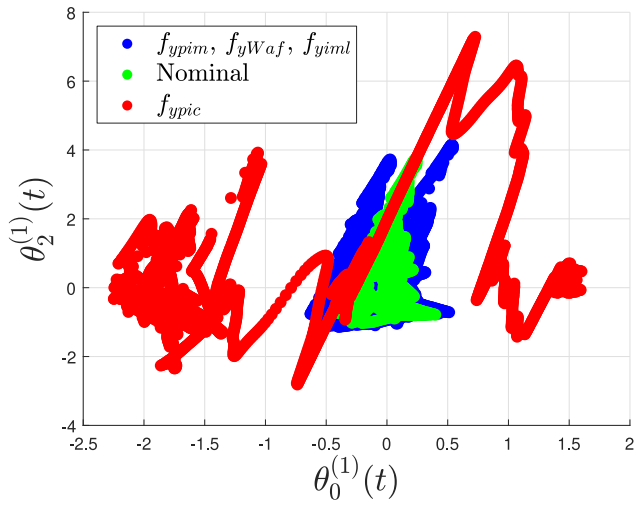


**Fig. 11.** Residuals $r_1$ computed on the eight data sets.

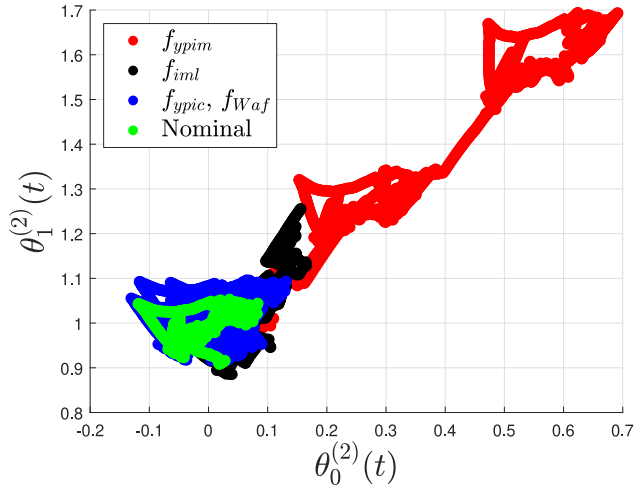


**Fig. 12.** Residuals $r_2$ computed on the eight data sets.

Hereafter, $\vartheta_j^{(i)}$ will denote the $j$th time varying coefficient of the $i$th RLS model, where $j \in \{0, \dots, 3\}$. The set of residuals signals $r_i$ consists in two of the time varying coefficients of each RLS model. Denoting with $t \in \mathbb{N}$ the discrete-time index, the first model reads as

$$y^{(1)}(t) = \vartheta_0^{(1)}(t) + x_1^{(1)}(t)\vartheta_1^{(1)}(t) + x_2^{(1)}(t)\vartheta_2^{(1)}(t), \tag{2}$$

where $y^{(1)} = y_{pic}$, $x_1^{(1)} = y_{xpos}$, $x_2^{(1)} = (y_{xpos})^2$. The features $x_1^{(1)}$ and $x_2^{(1)}$ and the output $y^{(1)}$ are selected considering the actuator valves equations of the Turbocharged Spark Ignite Engine (Eriksson, 2007). The forgetting factor used for model (2) is $\mu_1 = 0.999$, with $r_1 = \left[\vartheta_0^{(1)}, \vartheta_2^{(1)}\right]$. Fig. 11 shows that the set of residuals $r_1$ is sensitive only to the fault $f_{\text{ypic}}$. Because of this, $r_1$ can be used to detect and isolate the fault $f_{\text{ypic}}$. The second model is

$$y^{(2)}(t) = \vartheta_0^{(2)}(t) + x_1^{(2)}(t)\vartheta_1^{(2)}(t) + x_2^{(2)}(t)\vartheta_2^{(2)}(t), \tag{3}$$

with $y^{(2)} = u_{mf}$, $x_1^{(2)} = (y_{pim})^2$ and $x_2^{(2)} = y_\omega$. In this case, the features $x_1^{(2)}$ and $x_2^{(2)}$ and the output $y^{(2)}$ are selected according to the second and third state equation of the nonlinear mathematical model of the engine (Gagliardi, Tedesco, & Casavola, 2018). The forgetting factor used for model (3) is $\mu_2 = 0.9997$, with $r_2 = \left[\vartheta_0^{(2)}, \vartheta_1^{(2)}\right]$. As shown in Fig. 12 the residual set $r_2$ is sensitive to the fault $f_{\text{ypim}}$ and sometimes also to the fault $f_{\text{iml}}$. Due to this, it is possible to use the residual set

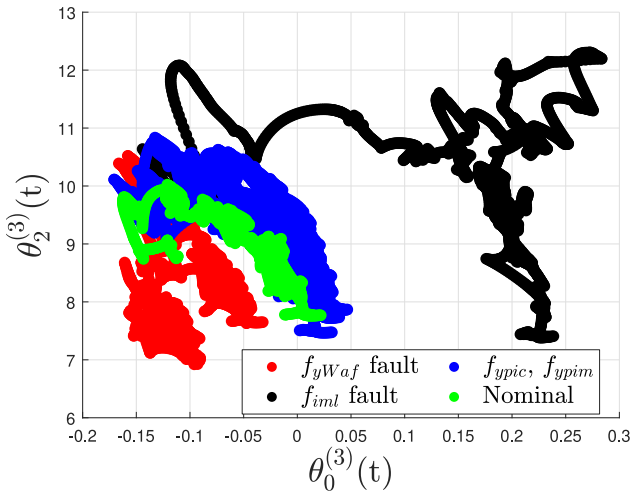


**Fig. 13.** Residuals $r_3$ computed on the eight data sets.

**Table 5**
Isolation matrix, where a 0 means that the residual must not react to isolate the fault, 1 that must react, and $*$ that the residual is irrelevant.

| Residual/Fault | No Fault | $f_{\text{ypic}}$ | $f_{\text{ypim}}$ | $f_{\text{yWaf}}$ | $f_{\text{iml}}$ |
|---|---|---|---|---|---|
| $r_1$ | 0 | 1 | 0 | 0 | 0 |
| $r_2$ | 0 | 0 | 1 | 0 | * |
| $r_3$ | 0 | 0 | 0 | 1 | 1 |
| $r_4$ | 0 | * | * | 0 | 1 |

$r_2$ to detect $f_{\text{ypim}}$ but not to isolate it. The third model is:

$$y^{(3)}(t) = \vartheta_0^{(3)}(t) + x_1^{(3)}(t)\vartheta_1^{(3)}(t) + x_2^{(3)}(t)\vartheta_2^{(3)}(t) + x_3^{(3)}(t)\vartheta_3^{(3)}(t), \tag{4}$$

with $y^{(3)} = u_{mf}$, $x_1^{(3)} = \log(y_\omega) \cdot y_{Waf}$, $x_2^{(3)} = y_{Waf}$ and $x_3^{(3)} = \log(y_{xpos})$. The variables $y^{(3)}$, $x_1^{(3)}$, $x_2^{(3)}$ and $x_3^{(3)}$ in (4) are chosen by considering the second, third and fourth state space equation of the model in Gagliardi et al. (2018). The interaction between $y_{Waf}$ and the logarithm of $y_\omega$ in the variable $x_1^{(3)}$ is obtained through data analysis. The forgetting factor for model (4) is $\mu_3 = 0.9997$, with $r_3 = \left[\vartheta_0^{(3)}, \vartheta_2^{(3)}\right]$. Fig. 13 shows that the residual set $r_3$ allows to correctly detect the faults $f_{\text{yWaf}}$ and $f_{\text{iml}}$, however such faults cannot be isolated from each other. The fourth model is

$$y^{(4)}(t) = \vartheta_0^{(4)}(t) + x_1^{(4)}(t)\vartheta_1^{(4)}(t) + x_2^{(4)}(t)\vartheta_2^{(4)}(t), \tag{5}$$

with $y^{(4)} = y_\omega$, $x_1^{(4)} = y_{pim}$ and $x_2^{(4)} = \log(y_{pim})$. The features $x_2^{(4)}$ and the output $y^{(4)}$ are selected considering the first and third state equations of the model in Gagliardi et al. (2018). Similarly to what as been done for (4), the logarithmic dependence of the feature $x_2^{(4)}$ is derived from analysis performed on data. The forgetting factor used for model (5) is $\mu_4 = 0.9997$, with $r_4 = \left[\vartheta_0^{(4)}, \vartheta_2^{(4)}\right]$. In Fig. 14 it can be observed that the residual set $r_3$ reacts only to the fault $f_{iml}$. In this way, it is possible to isolate $f_{\text{iml}}$ from $f_{\text{yWaf}}$ by observing if $r_4$ reacted. Also, it is possible to isolate $f_{\text{ypim}}$ observing when $r_2$ reacts and $r_3$ does not react, because when $r_3$ does not react $f_{\text{iml}}$ is not present. Table 5 summarizes the isolation process.

#### 4.2.2. Anomaly detector

Figs. 11–14 show that, when some faults are not present, the residual signals belong to a specific region of the residuals space. Because of this, it is possible to characterize an *insensitive region* for each residuals set, where points that are inside the region are associated with the nominal working condition of the system or to a fault condition which does not trigger the specific residual set. Instead, points outside the insensitive region are associated with a fault condition. According to

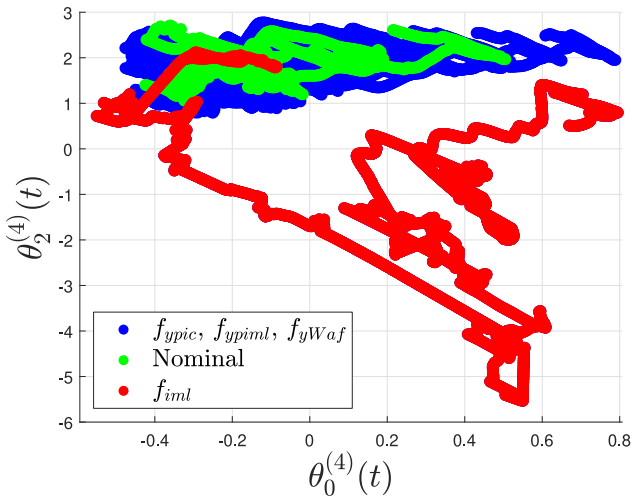


**Fig. 14.** Residuals $r_4$ computed on the eight data sets.

Table 5, it is possible to detect and isolate faults by training four classifiers able to distinguish if a point is inside or outside the insensitive region. In this work, this is done using open-set classification techniques proposed in Lundgren and Jung (2022). Each classifier is trained using all available data points that fall inside the respective insensitive region. Consider a data set $\boldsymbol{R}_i$ of points that belong to the insensitive region of the residuals set $\boldsymbol{r}_i$. The idea is to divide $\boldsymbol{R}_i$ in batches of fixed size $B$, and to estimate a probability density function (pdf) on each batch. In this work, the batch size is $B = 300$ data points and the pdfs are assumed Gaussian, so that their estimates can be obtained in closed form using the sample mean and variance–covariance matrix estimators. Consider $\Omega_i$ the set of the estimated pdfs (one for each batch of data) and two pdfs $p, q \in \Omega_i$. Since $p$ and $q$ are assumed normal, the Kullback–Leibler (KL) divergence (Zhang, Liu, Chen, Li, & Wang, 2021) $K(p \parallel q)$ can be defined as

$$K(p \parallel q) = \frac{1}{2}\left[\mathrm{Tr}\left(\Sigma_q^{-1}\Sigma_p\right) - n + \log\left(\frac{\det \Sigma_q}{\det \Sigma_p}\right) + \left(\mu_q - \mu_p\right)^T \Sigma_q^{-1}\left(\mu_q - \mu_p\right)\right], \tag{6}$$

where $n$ is the dimension of $p$ and $q$, $\mu_p, \mu_q \in \mathbb{R}^n$ are the mean vectors and $\Sigma_p, \Sigma_q \in \mathbb{R}^{n\times n}$ are the covariance matrices. Lower the value of (6), higher is the similarity between $p$ and $q$. Let $p_i'$ be a new pdf estimated using a batch of residuals obtained online. To understand if residuals in the new batch belong to the insensitive region of $\boldsymbol{r}_i$, the similarity $D_i(p_i')$ between $p_i'$ and the pdfs in $\Omega_i$ is considered, that is:

$$D_i(p_i') = \min_{q\in\Omega_i} K(p_i' \parallel q), \quad i \in \{1, \dots, 4\}, \tag{7}$$

where $p_i'$ belongs to the insensitive region of $\boldsymbol{r}_i$ if $D_i(p_i')$ is below a certain threshold $J_i$. The thresholds $J_i$ are tuned according to the method proposed in Lundgren and Jung (2022). Training the four anomaly detectors for the residuals sets $\boldsymbol{r}_i$, $i \in \{1, \dots, 4\}$, consists in estimating the four sets of pdfs $\Omega_i$, and the four thresholds $J_i$ using the eight data set provided for the competition. In particular, each set of pdfs $\Omega_i$ is trained using only the data sets that belongs to the insensitive region of the corresponding residuals set $\boldsymbol{r}_i$. When the anomaly detectors are employed online, four new pdfs $p_i'$, are computed every time 25 new samples are collected. Then, the similarity between $p_i'$ and $\Omega_i$ is checked to understand if it is below the threshold $J_i$. In this way, the classification logic in Table 5 is achieved.

### 4.3. Results

Table 6 presents the validation results of the proposed solution on the data provided for the competition. The approach used is the

**Table 6**
Performance of the fault diagnosis solution based on open-set classification in terms of Correct Detection and Isolation Rate (CIDR), Miss Detection Rate (MDR), False Positive Rate (FPR), Miss Classification Rate (MCR) and computational time evaluated on a pc running Matlab with CPU i7-12700KF.

| Data set | CDIR | MDR | FPR | MCR | Time |
|---|---|---|---|---|---|
| NF | 91.02% | | 8.98% | | 8.19 s |
| pic_90 | 93.46% | 0 | | 6.54% | 9.31 s |
| pic_110 | 94.34% | 0 | | 5.56% | 9.65 s |
| pim_90 | 85.81% | 6.87% | | 7.32% | 8.99 s |
| pim_80 | 94.57% | 0.52% | | 4.91% | 8.52 s |
| waf_105 | 85.95% | 8.91% | | 5.14% | 9.00 s |
| waf_110 | 91.75% | 0 | | 8.25% | 8.34 s |
| iml_6mm | 82.28% | 0.53% | | 17.19% | 9.00 s |

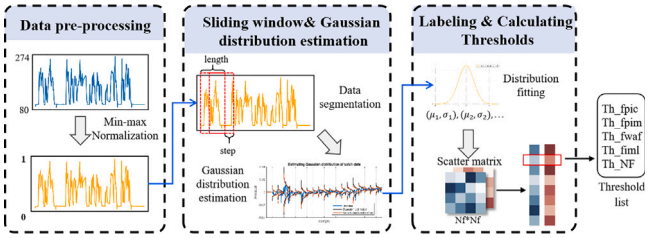


**Fig. 15.** The overall structure of the proposed method.

*Leave One Data set Out Cross Validation* (LODOCV), i.e. the solution is calibrated using seven out of the eight available data sets, and the performance are computed on the data set that has been left out.

### 4.4. Conclusions

Results in Table 6 show that the proposed solution is able to detect and isolate the known faults with high accuracy. This represents a significant result considering the low amount of data available and the low computational time required to online update the four RLS models and compute the four pdfs $p_i'$.

## 5. Solution, third participant

*Authors:* Jiamin Xu, Siwen Mo, Zixuan Xu, Chongpan Yang, and Zhile Du
*Affiliation:* Central South University
*Contact:* Xujm123@csu.edu.cn

### 5.1. Introduction

This solution adopts a distribution-based fault diagnosis method. Using a sliding window, the training dataset is segmented, and each segment undergoes distribution fitting to generate historical "experience". Fault types are then matched based on distribution similarity. The method's advantages align with the dataset's characteristics and the principles of distribution-based diagnosis:

**Measurement Noise** By focusing on distribution statistics rather than trend data, the method is less affected by measurement noise.

**Limited Training Data** The sliding window and overlap strategy increase data utilization, mitigating the impact of limited fault samples.

**Unknown Faults** Distribution similarity is compared to thresholds; low similarity flags potential unknown faults.

### 5.2. Methodologies

The overall structure of the proposed method is illustrated in Fig. 15, and each component will be detailed in the following sections.

#### 5.2.1. The training phase

**Data Preprocessing:** Each dataset is normalized via min–max scaling across sensor channels to prepare for training and testing:

$$f(x) = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{8}$$

**Data Sliding Window:** A sliding window with size 50 and step size 10 is applied. For datasets $X'$ of length 36,000, this yields 3,600 overlapping feature segments $X = X_0, X_1, \ldots$. A smaller $step\_size$ increases the number of segments and extracted features, improving diagnostic precision but also raising computational cost:

$$X_i = [x's \times i : x's \times i + w] = [x_{i,1}, x_{i,2}, \ldots] \tag{9}$$

**Gaussian Distribution Fitting:** Each segment (50 points) is modeled as a Gaussian distribution. Using NumPy's `mean` and `cov`, 3,600 distributions are generated per dataset, reflecting fault characteristics:

$$\mu_i = \frac{1}{N}\sum_{l=1}^{N} X_i(l) \;\; \sigma_i^2 = \frac{1}{N}\sum_{l=1}^{N}(x_{i,l} - \mu_i)^2 \tag{10}$$

**Labeling:** Distributions are labeled by fault type. Data from $wltp\_NF$ are all labeled "NF" (normal). For other datasets, segments after 120 s are labeled by their fault type; segments before 120 s are "NF".

**Calculating KL Divergence:** KL divergence quantifies differences between distributions from different fault types:

$$D_{KL}(P(X_i) \parallel P(X_j)) = \log\left(\frac{\sigma_i}{\sigma_j}\right) + \frac{\sigma_i^2 + (\mu_i - \mu_j)^2}{2\sigma_j^2} - 0.5 \tag{11}$$

**Calculating Fault-Specific Thresholds:** For each fault type, all $N_f$ segments' distributions are used to compute a divergence matrix $M_f$ $(N_f \times N_f)$. Flattening and sorting $M_f$, we select the value at $\text{int}(0.01 \times N_f \times N_f)$ as the threshold $th_f$. This yields the threshold list:

$$th_{list} = [Th_{NF}, Th_{F1}, Th_{F2}, Th_{F3}, Th_{F4}] \tag{12}$$

With this, the training phase concludes.

#### 5.2.2. The testing phase

In the testing phase, if fewer than 50 sampling values are available, diagnosis is skipped and no fault is assumed—this "dead zone" exists because at least 50 points are required to form a valid feature segment. Once 50 or more values are input, the latest 50 are used to create a feature segment, followed by Gaussian fitting (mean and variance).

The KL divergence between this distribution and each fault type (including the no-fault case) from the training set is then computed, yielding five KL values:

$$KL_{list} = [KL_{NF}, KL_{F1}, KL_{F2}, KL_{F3}, KL_{F4}] \tag{13}$$

Each value corresponds to a specific fault type. Diagnosis is completed by comparing the KL list with the threshold list, using the following rules:

- If each element of the KL list is greater than the corresponding element in the threshold list, and $\max(KL_{list}) \gg$ threshold list [argmax(threshold list)], it is considered to belong to an unknown fault.
- Otherwise, it is directly diagnosed as the fault type corresponding to the minimum element in the KL list. Specifically, $\text{argmin}(KL_{list}) = 0$ corresponds to no fault, $\text{argmin}(KL_{list}) = 1$ corresponds to $wltp_{f_{\text{iml}}}$, $\text{argmin}(KL_{list}) = 2$ corresponds to $f_{\text{pic}}$, and so on.

With this, the diagnostic phase of the algorithm is complete.

**Table 7**
Hyperparameters.

| Parameters | value |
|---|---|
| $window\_size$ | 50 |
| $step\_size$ in training sets | 50 |
| $step\_size$ in testing sets | 50 |
| selection criteria for thresholds | top 1% |

| | CPU | GPU | Memory | DISK |
|---|---|---|---|---|
| Specifications | Inter xeon Silver ver4116 | RTX 2080TI 11G*8 | 32G*4 | 480G SSD+2TB HDD |

**Fig. 16.** The hardware settings.

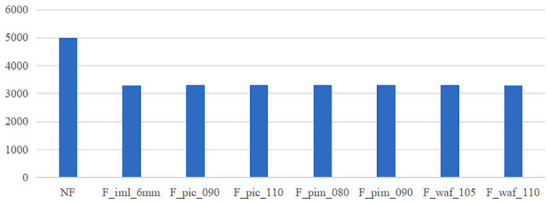


**Fig. 17.** Bar chart of the distribution quantity of different fault types.

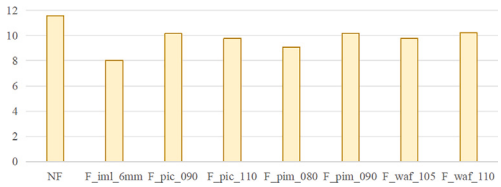


**Fig. 18.** Bar chart of the thresholds for various types of faults.

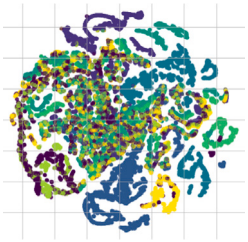

**Fig. 19.** Visualization of mean distribution.

### 5.3. Results

The proposed fault diagnosis method is here evaluated using training data. To apply the proposed method, key hyperparameters must be set. The window and step sizes determine how many feature distributions are learned per fault type. A window that is too small may fail to capture meaningful temporal patterns. The hyperparameters and hardware settings are summarized in Table 7 and Fig. 16.

#### 5.3.1. Display of distribution results from the training phase

Using the method described in Section 2, numerous distributions reflecting different fault types were obtained, as shown in Figs. 17, 18, and 19. Fig. 17 displays the number of distributions per fault type, with normal conditions (NF) having the most due to larger data volume. Fig. 18 shows the KL divergence thresholds (top 1%) for each fault type; these represent divergence between test and empirical distributions and are not directly comparable across types. Fig. 19 presents a T-SNE visualization of distribution means, illustrating their separability for diagnosis.

**Table 8**
Experimental metrics.

| Experimental metrics | Description |
|---|---|
| Time from fault occurrence until detection | \ |
| Fault alarm rate | FAR = $\frac{FP}{FP+TN}$ |
| Missed detection rate | MDR = $\frac{FN}{FN+TP}$ |
| Fault isolation accuracy | FIA = $\frac{TP}{FN+TP}$ |

| Testing_set | Fault alarm rate | Missed detection rate | Fault isolation accuracy | Time from fault occurrence until detection | Computation time |
|---|---|---|---|---|---|
| wltp_f_waf_105 | 0.1 | 0.671 | 0.259 | 0.25 | 0.156 |
| wltp_f_waf_110 | 0.073 | 0.046 | 0.954 | 0.25 | 0.247 |
| wltp_f_pim_090 | 0.067 | 0.273 | 0.836 | 0.3 | 0.216 |
| wltp_f_pim_080 | 0.126 | 0.083 | 0.877 | 0.4 | 0.188 |
| wltp_f_pic_110 | 0.084 | 0.116 | 0.661 | 0.35 | 0.253 |
| wltp_f_pic_90 | 0.002 | 0.126 | 0.615 | 0.3 | 0.192 |
| wltp_f_iml_6mm | 0.121 | 0.401 | 0.739 | 1.6 | 0.204 |

**Fig. 20.** Overall experimental results.

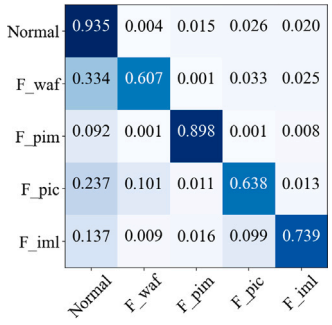


**Fig. 21.** Confusion matrix of diagnostic performance.

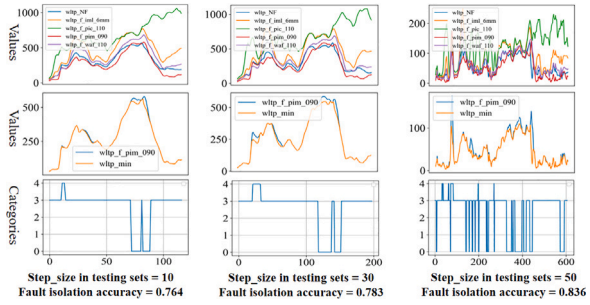


**Fig. 22.** Experimental results.

#### 5.3.2. Experimental metrics

To verify the effectiveness of the proposed method, multiple experimental metrics need to be set, as detailed in Table 8.

#### 5.3.3. Experimental results

The experimental results are shown in Fig. 20. We first present the overall diagnostic evaluation of the proposed method, followed by the confusion matrix in Fig. 21, which details performance across five fault types and normal conditions.

In terms of time efficiency, the method performs well for both fault detection latency and computation time. For diagnostic accuracy, the method shows weaker performance when diagnosing faults related to air mass flow sensors but achieves strong results for other fault types.

Additionally, we analyzed the impact of the key hyperparameter $step\_size$ in the testing phase using the dataset $wltp\_f\_pim\_090$ as a case study. As shown in Fig. 22, decreasing the step size leads to a significant drop in fault isolation accuracy. This suggests that very small step sizes reduce the variation between consecutive data segments, causing instability in diagnostic results.

### 5.4. Conclusions

In this subsection, we summarize the main contributions of the proposed method. Based on the primary characteristics of the obtained datasets, a data-driven distribution matching-based fault diagnosis method is proposed. A corresponding forgetting coefficient for the KL divergence is utilized for each type of fault. Experiments are conducted to verify the effectiveness of the proposed method across a range of evaluation metrics.

## 6. Solution, fourth participant

*Authors:* Hossein Safaeipour[1], Mehdi Forouzanfar[1,2], Vahid Mirahi[1], Anna Pinnarelli[2], and Vicenç Puig[3]
*Affiliation:* [1]Islamic Azad University, [2]University of Calabria, [3]Polytechnic University of Catalonia-BarcelonaTech
*Contact:* mehdi.forouzanfar@iau.ac.ir

### 6.1. Introduction

This solution presents a statistical method for extracting features from sensor signals, emphasizing deviations indicative of faults (Gao et al., 2015). A fixed-thresholding approach based on residual analysis is adopted to detect abnormalities without requiring full model knowledge (Blanke et al., 2016). Fault detection and isolation are performed using Adaptive Neuro-Fuzzy Inference Systems (ANFIS) (Brahimi, Hadroug, Iratni, Hafaifa, & Colak, 2024; Salahshoor, Kordestani, & Khoshro, 2010), addressing challenges due to incomplete modeling and limited fault realizations (Li & Yu, 2022; Rojas, Rojas, & Cortés-Romero, 2022).

Early detection of abnormal behavior remains critical to prevent undesirable consequences and economic losses (Jafari, Ramezani, & Forouzanfar, 2022). The proposed solution combines AI-based modeling and signal processing to enhance robustness, with evaluation based on the benchmark criteria.

### 6.2. Solution strategy

Data-driven models based on ANFIS are developed to characterize the engine dynamics. Feature selection from available signals ensures model simplicity and robustness. The model training strategy is developed under the following assumptions:

**Assumption 1.** Faults remain active long enough to be detected.

**Assumption 2.** Multiple faults can occur sequentially after an initial fault.

To improve training efficiency and diagnosis accuracy, feature selection identifies the most relevant input–output variables from sensor and actuator data. Since not all collected measurements aid fault diagnosis, selecting appropriate features enhances computational efficiency and system robustness.

#### 6.2.1. Healthy system model

During monitoring, anomalies are initially treated as unknown faults. A normal operating model (*Model_fx*), trained on healthy data, characterizes fault-free dynamics and distinguishes normal from abnormal conditions.

**Table 9**
Applied residual signals.

| Fault Class | $R_i$ | $\overline{R}_i$ |
|---|---|---|
| Anomaly Detection | √ | √ |
| '$f_{\text{ypic}}$' Isolation | √ | × |
| '$f_{\text{ypim}}$' Isolation | √ | × |
| '$f_{\text{yWaf}}$' Isolation | × | √ |
| '$f_{\text{iml}}$' Isolation | √ | × |

#### 6.2.2. Isolation models

Four distinct models (*Model_pic*, *Model_pim*, *Model_waf*, *Model_iml*) are trained to isolate specific known faults. Each model uses all datasets except the one corresponding to its target fault. After anomaly detection, the isolation module analyzes historical data and monitors new samples within the allowed window.

**Remark 1.** If no known fault is isolated within the permitted steps, the anomaly is labeled as an unknown fault ('*fx*' flag).

#### 6.2.3. Residual generation and filtering

Residual signals are generated by comparing measured outputs and model estimates:

$$\hat{R}_i(k) = y_i(k) - \hat{y}_i(k), \quad (i = 1, 2, \dots, q) \tag{14}$$

where $y_i(k)$ and $\hat{y}_i(k)$ denote the measured and estimated values of the $i^{\text{th}}$ feature at time $k$.

Under nominal conditions, residuals oscillate around zero within an uncertainty band $[\gamma^-, \gamma^+]$ caused by system uncertainties. Faults induce deviations beyond this range. To enhance detection sensitivity, a moving window averaging is applied:

$$W_{i,k} = \frac{1}{l} \sum_{j=k-l}^{k} \hat{R}_{i,j} \cdot \hat{R}^2_{i,k}, \tag{15}$$

$$R_{i,k} = \frac{1}{m} \sum_{j=k-m}^{k} W_{i,j} \cdot W^2_{i,k} \tag{16}$$

where $l$ and $m$ are the moving window lengths.

If necessary, an alternative residual enhancement method is applied based on moving standard deviation:

$$\sigma_i = \text{std}(\hat{R}_{i,k-b}, \dots, \hat{R}_{i,k}), \tag{17}$$

$$\overline{W}_{i,k} = \frac{1}{\tau} \sum_{j=k-\tau}^{k} \hat{R}_{i,j} \cdot \sigma_i, \tag{18}$$

$$\overline{\sigma}_i = \text{std}(\overline{W}_{i,k-b}, \dots, \overline{W}_{i,k}), \tag{19}$$

$$\overline{R}_{i,k} = \frac{1}{m} \sum_{j=k-m}^{k} \overline{W}_{i,j} \cdot \overline{\sigma}_i \tag{20}$$

where $b$, $\tau$, and $m$ define the respective window sizes. The residual signals used for anomaly detection and isolation are summarized in Table 9.

#### 6.2.4. Residual evaluation

To reliably detect faults, a fixed thresholding approach is employed, based on the statistical properties of residuals during fault-free operation (Safaeipour, Forouzanfar, Puig, & Birgani, 2023). The fixed thresholds are determined by scaling the maximum observed residual magnitudes as follows:

$$\gamma_i = A_i \max_k |R_{i,k}|, \overline{\gamma}_i = \overline{A}_i \max_k |\overline{R}_{i,k}|, \tag{21}$$

where $A_i, \overline{A}_i < 1$ are tuning parameters set to balance detection sensitivity and false alarm rate.

Under nominal conditions, residuals $R_i(k)$ and $\overline{R}_i(k)$ fluctuate within bounded intervals primarily due to modeling uncertainties and measurement noise. Fixed thresholds $\gamma_i$ and $\overline{\gamma}_i$ are determined as scaled maxima of residual signals, providing static decision boundaries for anomaly detection.

The detection logic activates when residuals exceed their corresponding thresholds. If the cumulative number of alarms $N_i$ within the window from the alarm start time $A_{ST}$ to the current sampling time $C_{ST}$ satisfies:

$$\frac{N_i}{C_{ST} - A_{ST}} > \beta \quad \text{and} \quad N_i > \alpha, \tag{22}$$

where $\beta \in (0, 1]$ defines the minimum acceptable alarm density over the window, and $\alpha \in \mathbb{N}$ represents the minimum number of alarms required to confirm a fault. If both conditions are met, the detection flag is raised; otherwise, the event is classified as a false alarm.

The alarm counter is reset if the following condition holds without raising a detection flag:

$$\frac{C_{ST} - A_{ST}}{N_i} > \alpha, \tag{23}$$

indicating that the frequency of alarms is insufficient for fault confirmation over the monitoring interval.

### 6.3. Fault isolation

Upon detection, the dedicated ANFIS isolation models described in Section 6.2.2 are activated. These models apply the detection logic of Section 6.2.4 to outputs estimated for fault identification. Faults are classified based on:

$$\text{Fault}_{\text{isolated}} = f(r(t), \theta), \tag{24}$$

where $r(t)$ is the filtered residual vector, and $\theta$ summarizes the membership functions, rule bases, and tuned parameters of each model.

A segment of historical residuals is initially analyzed. If isolation fails, the window is updated with new samples until a fault is isolated or the isolation window expires. Once isolated, the corresponding model is deactivated, while others remain active according to Assumption 2 in Section 6.2. If no known fault is identified within the permitted window, the anomaly is classified as unknown ('*fx*' flag) following *Remark* 1 in Section 6.2.2. After expiration, all models reset except those that flagged faults.

### 6.4. Robustness to noise

Following the residual evaluation in Section 6.2.4, a cumulative alarm mechanism enhances robustness by requiring sustained alarm activity over a monitoring window. This approach increases resilience against transient disturbances, measurement noise, and isolated threshold crossings, improving fault detection under normal and unknown operating conditions.

Using sustained deviation logic, the method remains robust to transient noise. While noise can amplify residual energy and facilitate early fault detection, it may occasionally cause missed alarms for small-magnitude faults (Blanke et al., 2016; Gao et al., 2015). The cumulative evaluation mitigates such risks, ensuring a balanced trade-off between sensitivity and specificity.

### 6.5. Results

The detection strategy in Section 6.2 is applied for both anomaly detection and isolation, adhering to the competition structure. Simulation parameters are: $l = 200$, $m = 500$, $\tau = 50$, $b = 25$, $\alpha = 6$, $\beta = 0.7$, with moving average windows selected by trial and error to optimize performance.

**Remark 2.** After the first fault isolation, $\alpha$ is doubled to apply a more conservative logic for subsequent faults.

**Remark 3.** If no known fault is isolated within 2000 samples after anomaly detection, the fault is classified as unknown ('*fx*'), based on experimental evaluation of worst-case isolation delays.

**Table 10**
Simulation results for 8 Datasets.

| Faultdescription | Type | Simulation time (s) | Detection time (Sample) | Isolation time (Sample) |
|---|---|---|---|---|
| NF | None | 13.9 | – | – |
| waf_105 | Sensor | 28.8 | 2835 (435) | 3239 (839) |
| waf_110 | Sensor | 25.1 | 2942 (542) | 3262 (862) |
| iml_6mm | Leakage | 26.4 | 2687 (287) | 2774 (374) |
| pic_090 | Sensor | 27.0 | 2500 (100) | 2506 (106) |
| pic_110 | Sensor | 26.8 | 2476 (076) | 2523 (123) |
| pim_080 | Sensor | 26.9 | 2799 (399) | 2805 (405) |
| pim_090 | Sensor | 26.8 | 3180 (780) | 4271 (871) |

* Actual fault occurrence at sample 2400.

Simulation results are summarized in Table 10.

### 6.6. Conclusions

The proposed approach integrates adaptive neuro-fuzzy modeling, residual analysis, and fixed thresholding for fault detection and isolation in complex nonlinear systems. By combining healthy-system modeling with fault-specific isolation modules, early anomaly detection and accurate fault identification are achieved, while unknown scenarios are flagged. Moving-window residual enhancement and cumulative alarm logic improve robustness against noise and transient deviations. Simulation results on the LiU-ICE benchmark confirm the method's diagnostic accuracy, efficiency, and adaptability to multi-fault industrial conditions.

## 7. Solution, fifth participant

*Authors:* Qiao Deng, Yufei Liu, Jiakun Liu, Haobin Ke and Wanting Zhu
*Academic Advisor:* Zhiwen Chen
*Affiliation:* Central South University
*Contact:* zhiwen.chen@csu.edu.cn

### 7.1. Introduction

The strong coupling and nonlinear interactions between the ten signals in the LiU-ICE benchmark complicates conventional feature-based methods. We therefore adopt Deep CCA to learn joint representations that maximize inter-view correlation, enhancing feature richness and fault discrimination (Chen, Liang, et al., 2021). However, Deep CCA's reliance on multi-objective optimization, staged training, and fixed architectures hinders generalization. To address this, we introduce the Canonical Correlation-Guided Deep Neural Network (CCDNN), an end-to-end trainable model with integrated CCA objectives and adaptive structure, combining robust cross-view correlation learning with streamlined training (Chen et al., 2025).

### 7.2. Related work

Deep Canonical Correlation Analysis (DCCA) learns joint representations for two views via deep nonlinear encoders. Let view 1 input $x_1 \in \mathbb{R}^{n_1}$ be mapped by a $d$-layer network with parameters $\theta_1 = \{W_l^1, b_l^1\}_{l=1}^d$ to output $f_1(x_1;\theta_1) \in \mathbb{R}^o$; define $\theta_2$ and $f_2(x_2;\theta_2)$ similarly for view 2. The training objective is:

$$(\theta_1^*, \theta_2^*) = \underset{(\theta_1,\theta_2)}{\arg\max} \quad corr(f_1(X_1;\theta_1), f_2(X_2;\theta_2)) \tag{25}$$

Given $m$ samples, stack top-layer outputs into $H_1, H_2 \in \mathbb{R}^{o\times m}$ and center them: $\bar{H}_i = H_i - \frac{1}{m}H_i\mathbf{1}$, where $\mathbf{1} \in \mathbb{R}^{m\times m}$ is the all-ones matrix. Define $\hat{\Sigma}_{12} = \frac{1}{m-1}\bar{H}_1\bar{H}_2^T$, $\hat{\Sigma}_{ii} = \frac{1}{m-1}\bar{H}_i\bar{H}_i^T + r_i I$, $r_i > 0$. Let $R = \hat{\Sigma}_{11}^{-1/2}\hat{\Sigma}_{12}\hat{\Sigma}_{22}^{-1/2}$. If $k$ is set to the output dimensionality $o$, this corresponds to the trace of the matrix $R$, or

$$corr(H_1, H_2) = \|R\|_{tr} = tr(R^T R)^{1/2} \tag{26}$$

Subsequently, the parameters of DCCA can be optimized to maximize this correlation measure using gradient-based optimization techniques. For further details on the methodology,please refer to Andrew, Arora, Bilmes, and Livescu (2013).

### 7.3. Methodologies

Learning multi-view representations that are maximally linearly correlated is a longstanding goal. We propose the CCDNN framework, which embeds canonical correlation into a deep neural network as a constraint rather than the primary objective (Chen et al., 2025). Unlike CCA (Chen, Ding, Peng, Yang, & Gui, 2017; Chen, Ding, Zhang, Li, & Hu, 2016; Hotelling, 1992), kernel CCA (Fukumizu, Bach, & Gretton, 2007) and DCCA (Chen, Liang, et al., 2021), CCDNN uses correlation as a constraint, allowing the main loss to be tailored (e.g. classification loss) while still enforcing view alignment. To remove redundant features induced by correlation, we introduce a redundancy filter. The overall CCDNN architecture is illustrated in Fig. 23.

Traditional DCCA and DCCAE use linear CCA, which can only compute the correlation between two features, without fully utilizing the potential of CCA. The CCDNN method proposed in this paper constructs new CCA and RF layers. This method introduces a novel mathematical approach, projecting onto each other's view subspaces to remove redundant features. Although our CCA is not directly used as the learning objective for the task, it influences the gradient backpropagation during training in the form of a constraint. For clarity, we have drawn the computation graphs of DCCAE and CCDNN for comparison, as shown in Fig. 24.

#### 7.3.1. Optimization objective

In contrast to deep CCA, taking the classification task as an illustrative example, the optimization objective in the proposed method can be formulated as follows:

$$\begin{aligned} \underset{(\theta_1,\theta_2,\theta_3)}{\arg\min} \quad & -\sum_{k=1}^{N_s}\sum_{a=1}^{N_c} g_k^a \log \hat{g}(\theta_1,\theta_2,\theta_3,J,L,\Sigma,x_k,y_k)_k^a \\ \text{s.t.} \quad & [J,L,\Sigma] = CCA(f_1(X;\theta_1), f_2(Y;\theta_2)). \end{aligned} \tag{27}$$

where $N_s$ represents the number of training samples, $N_c$ denotes the number of categories, $g_k^a$ is the label associated with category $a$, and $CCA(\cdot,\cdot)$ refers to the linear canonical correlation analysis operator.

Incorporating canonical correlation as a constraint in the loss function does not introduce any additional parameters or computational complexity. The entire network can still be trained end-to-end using stochastic gradient descent (SGD) with backpropagation, as proposed by Robbins and Monro (1951), and can be implemented easily using standard libraries without modifying the solvers. This approach is not only practical but also critical in the comparison between deep CCA and our proposed networks.

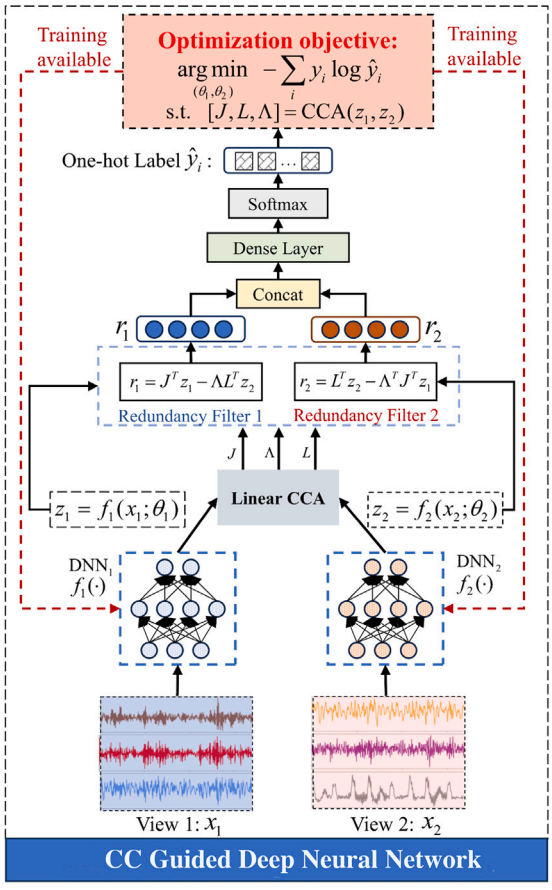


**Fig. 23.** Architectures of CCDNN (Chen et al., 2025).

The flexibility of the optimization objectives and deep network architectures is a key advantage. In this solution, experiments utilize optimization objectives such as cross-entropy, alongside deep network architectures like GRU. Other objectives and network structures are also feasible. Achieving this level of flexibility is generally more challenging for DCCA.

#### 7.3.2. Redundancy filter

The rationale behind the redundancy filter is quite straightforward. If the correlation coefficients obtained from the CCA constraint are not all equal to zero, this implies that redundancy is present in the outputs (i.e., the learned features) of the two deep networks due to the inherent correlation. Therefore, we propose a redundancy filter to mitigate such correlation-induced redundancy. The filter consists of two components and can be expressed as follows:

$$\mathbf{r}_1 = \mathbf{J}^\mathbf{T}\mathbf{u} - \mathbf{\Sigma}\mathbf{L}^\mathbf{T}\mathbf{y} \tag{28}$$

$$\mathbf{r}_2 = \mathbf{L}^\mathbf{T}\mathbf{y} - \mathbf{\Sigma}^T\mathbf{J}^\mathbf{T}\mathbf{u} \tag{29}$$

Taking the first part in Eq. (28) as an example, it can be rewritten as

$$\mathbf{r}_1 = \mathbf{J}^\mathbf{T}\mathbf{u} - \mathbf{\Sigma}\mathbf{L}^\mathbf{T}\mathbf{y} = \mathbf{\Gamma}^T\mathbf{\Sigma}_u^{-1/2}(\mathbf{u} - \mathbf{\Sigma}_{uy}\mathbf{\Sigma}_y^{-1}\mathbf{y}) \tag{30}$$

where $\Sigma_{uy}\Sigma_y^{-1}\mathbf{y}$ represents the least-mean-squares estimation of $\mathbf{u}$ using the data vector $\mathbf{y}$.

To further clarify the correlation-induced redundancy, based on the properties of CCA, the covariance of the residual signal $\mathbf{r}_1$ can be expressed as follows:

$$\mathbf{\Sigma}_{r_1} = \mathbf{J}^T\mathrm{E}(\mathbf{u}\mathbf{u}^T)\mathbf{J} + \mathbf{\Sigma}\mathbf{L}^T\mathrm{E}(\mathbf{y}\mathbf{y}^T)\mathbf{L}\mathbf{\Sigma}^T - \mathbf{J}^T\mathrm{E}(\mathbf{u}\mathbf{y}^T)\mathbf{L}\mathbf{\Sigma}^T$$
$$- \mathbf{\Sigma}\mathbf{L}^T\mathrm{E}(\mathbf{y}\mathbf{u}^T)\mathbf{J} = \mathbf{I}_l - \mathbf{\Sigma}\mathbf{\Sigma}^T$$
$$= \begin{bmatrix} \mathrm{diag}((1-\rho_1^2), \ldots, (1-\rho_\kappa^2)) & 0 \\ 0 & \mathbf{I}_{l-\kappa} \end{bmatrix} \in \mathcal{R}^{l\times l} \tag{31}$$

It is evident that incorporating the correlation with $\mathbf{y}$ (where $\rho_i \neq 0$) causes the covariance matrix $\Sigma_{\mathbf{r}_1}$ to decrease.

Similar to the analysis of $\mathbf{r}_1$, the understanding of $\mathbf{r}_2$ is straightforward and will not be repeated here. From Eqs. (21) and (22), it can be observed that the correlation-induced redundancy is reduced based on the covariance of the residual signal. The filtered signals are then concatenated and fed into the dense layer. It is important to note that if $\mathrm{diag}(\Sigma) = 0$, this implies that the learned representations of both deep networks have no correlation. In this case, $\mathbf{r}_1 = J^T\mathbf{u}$ and $\mathbf{r}_2 = L^T\mathbf{y}$, meaning that the redundancy filter passes the inputs without modification, indicating that no redundancy needs to be removed.

### 7.4. Results

The selection of variables significantly affects the experimental outcomes. Before training the model, we conducted a careful variable selection process to ensure robustness. Since some variables are more susceptible to environmental influences, we selected only those less likely to be affected. From Fig. 25, we selected our experiments' six most relevant variables.

Further variable selection experiments were conducted to determine the optimal selection. The correlation matrix in Fig. 25 reveals that the four variables — *ypic*, *ypim*, *ywaf*, and *yxpos* — exhibit strong correlations. To maximize the correlation between the two views, we assigned these four variables across two separate opinions as follows:

- **View 1 (X1)**: ypic, ypim, yw
- **View 2 (X2)**: ywaf, yxpos, uwg

We used a sampling-based approach to split the dataset into training, validation, and test sets. The results of our experiments are presented in Table 11. The metrics evaluated include the Fault Alarm Rate (FAR), Miss Detection Rate (MDR), Fault Isolation Accuracy (FIA), Detection Time ($T_d$, in seconds), and Computation Time ($T_c$, in milliseconds) for each type of fault.

**Table 11**
Experimental results for fault detection and diagnosis.

| Fault | FAR | MDR | FIA | $T_d$ (s) | $T_c$ (ms) |
|---|---|---|---|---|---|
| $f_{\mathrm{iml}}$ | 0.1000 | 0.0087 | 0.9841 | 0 | 0.1459 |
| $f_{\mathrm{ypic}}$ | 0.0771 | 0.0030 | 0.9963 | 0.425 | 0.1581 |
| $f_{\mathrm{ypim}}$ | 0.0944 | 0.0109 | 0.9610 | 0.5 | 0.1449 |
| $f_{\mathrm{yWaf}}$ | 0.1596 | 0.1669 | 0.8045 | 0.025 | 0.1536 |

The experimental results show that the lowest Miss Detection Rate (MDR) of 0.0030 was observed for the Intercooler pressure sensor fault ($f_{\mathrm{ypic}}$), indicating the highest fault detection accuracy. The Fault Alarm Rate (FAR) was minimized for $f_{\mathrm{ypic}}$, with a value of 0.0771, suggesting a reliable detection process. The detection time ($T_d$) for $f_{\mathrm{iml}}$ was 0, indicating immediate detection, while other faults had varying detection times, with $f_{\mathrm{ypim}}$ taking 0.5 s. The computation time ($T_c$) is also inconsistent, with $f_{\mathrm{ypim}}$ having the shortest computation time of 0.1449 ms.

### 7.5. Conclusions

In this section, we proposed the Canonical Correlation Guided Deep Neural Network, which effectively leverages the strengths of Deep Canonical Correlation Analysis while addressing its limitations by incorporating a classification-oriented optimization objective and a redundancy filter. The findings indicate that CCDNN can reliably diagnose multiple fault types while maintaining computation and detection time efficiency. Future work will focus on further optimizing the model and exploring its applicability to other complex systems.

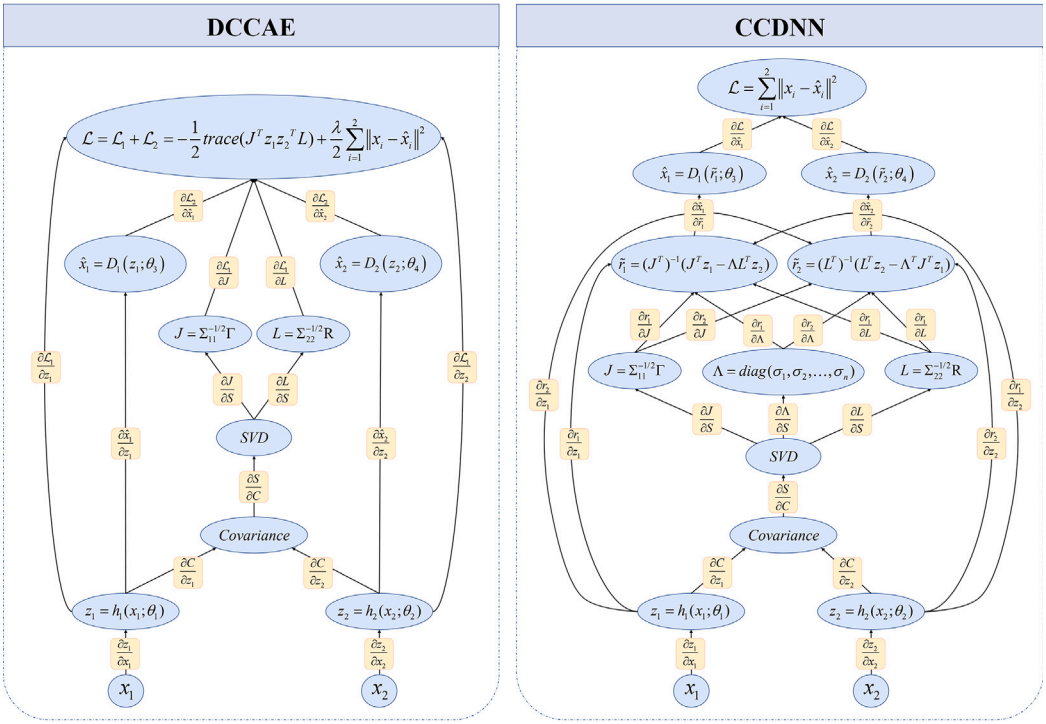


**Fig. 24.** Comparison of computation graphs (Chen et al., 2025).

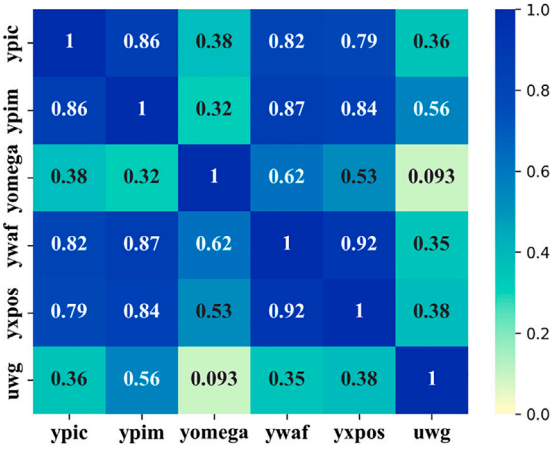


**Fig. 25.** Architectures of CCDNN.

## 8. Solution, sixth participant

*Authors:* Silke Merkelbach[1], Maryam Ahang[2] and Homayoun Najjaran[2]

*Affiliation:* [1]Fraunhofer IEM, [2]University of Victoria

*Contact:* silke.merkelbach@iem.fraunhofer.de

### 8.1. Introduction

This section introduces a hierarchical data-driven fault diagnosis method based on a variational autoencoder (VAE). As an accurate physical model of the system is hard to achieve and may change during the system's lifetime, we focus on a purely data-driven approach. Data-driven and machine learning approaches are powerful tools for handling uncertainties in condition monitoring (Ahang, Charter, et al., 2024). The proposed solution consists of three steps: (1) a VAE with a binary classifier for fault detection, (2) another VAE with a multi-class classifier for fault isolation, and (3) a one-class support vector machine (SVM) for unknown fault detection. The outputs of all these steps are aggregated to make the final diagnosis decision. We chose to use VAEs because they are robust and have been used successfully to analyze machine data, such as for condition monitoring (Ahang, Abbasi, Charter, & Najjaran, 2024). The SVM was chosen as it was applied effectively in the past for machine data analysis and novelty detection (Lundgren & Jung, 2022).

Autoencoder models consist of an encoder and a decoder. The encoder maps the input data to a latent space, usually of lower dimensions, and the decoder reconstructs the input from the latent space. The encoder of a VAE maps the input data into probabilistic distributions instead of single points. The objective function of a VAE consists of two main components: reconstruction loss and Kullback–Leibler (KL) divergence. The reconstruction loss ensures that the generated data resembles the input data, while the KL divergence regularizes the latent space to follow a normal distribution. The loss function of a $\beta$-Variational Autoencoder ($\beta$-VAE) modifies the traditional VAE loss by introducing a hyperparameter $\beta$ to scale the KL divergence term. This allows for controlling the balance between the reconstruction loss and the regularization of the latent space. A higher value of $\beta$ increases the spread of the data in the latent space, leading to better separation between data clusters. Therefore, in a noisy environment, the noise will not have a major effect due to the distance between different data groups, so the classification accuracy will increase (Higgins et al., 2017). The $\beta$-VAE objective function is expressed as:

$$\mathcal{L}_{\beta\text{-VAE}}(\mathbf{x},\mathbf{z}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - \beta D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z})\right) \tag{32}$$

Where $\mathbf{x}$ is the input data, $\mathbf{z}$ is the latent variable, $q_\phi(\mathbf{z}|\mathbf{x})$ is the encoder distribution (posterior distribution of the latent variable), $p_\theta(\mathbf{x}|\mathbf{z})$ is the decoder distribution (likelihood of the data given the latent variable), $p(\mathbf{z})$ is the prior distribution, typically a standard normal $\mathcal{N}(0, I)$, $D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z})\right)$ is the Kullback–Leibler divergence between the posterior and the prior, and $\beta$ is a hyperparameter that controls the weight of the KL divergence term.

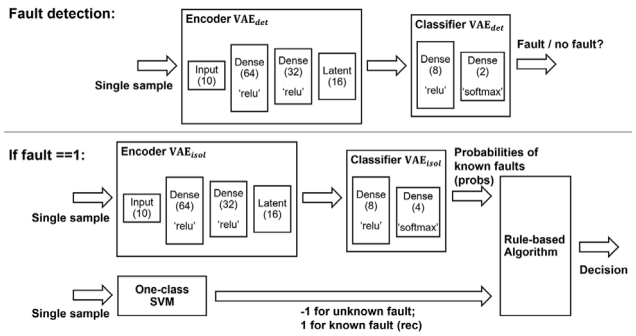


**Fig. 26.** Overview of the solution.

A One-Class Support Vector Machine (SVM) is a variation of SVM designed to detect anomalies and outliers within a dataset. This model defines a boundary around the majority class in the feature space.

### 8.2. Implementation

#### 8.2.1. Data preprocessing

The dataset consists of fault-free data and four different fault scenarios. Faults are introduced after about 120 s in each dataset, so in each fault dataset, the data after 130 s is considered faulty, and the data before 110 s is labeled fault-free. The data is normalized with a MinMaxScaler over all available training data. To make the model more robust against noise and unseen data, a random Gaussian noise with a mean of 0 and a variance of 0.05 is added to all data before the start of the training. Train and test data are randomly selected with a ratio of 80% and 20%.

#### 8.2.2. Model architecture

The architecture of our approach is shown in Fig. 26. For fault detection and diagnosis, a deep three-layer VAE with 10 inputs (as we have 10 sensors) and 64-32-16 neurons is used. This VAE is responsible for the sensor fusion, extracts and combines the input information, and maps them to distributions in its latent space. To perform fault detection, a two-layer MLP classifier with softmax activation is added to the network after the encoder part on the latent space data. Two VAEs with the same architecture are used: one for fault detection VAE$_{det}$ and one for fault isolation VAE$_{isol}$. In the first step of fault detection, a binary classifier is added to the encoder part of VAE$_{det}$. If a fault is detected, VAE$_{isol}$ and the SVM are responsible for fault isolation. The number of neurons in the last layer of the classifier after VAE$_{isol}$ is four, as we have four faults. The training of both VAEs is done over 100 epochs using the Adam optimizer with a learning rate of 5e-3 and a batch size of 64. VAE$_{det}$ is trained with all fault-free data in one class and with all faults in the other class. VAE$_{isol}$ is trained with known fault data, and the SVM is trained to recognize all known data.[4] The output of VAE$_{det}$ ($Det$), VAE$_{isol}$ ($Isol$), and the SVM ($UF$) are combined as shown in Algorithm 1. The output of the algorithm is the prediction ($Pred$) of the probability for each class.

### 8.3. Results

The data distribution in the latent space of the VAE considering all normal and faulty data is shown in Fig. 27, where the faults are separable. Here, faults with different severities are shown, and each fault has a unique distribution; only fault $f_{ywaf}$ has different distributions for different severities, making it hard to tell how the model might react to other severities of this fault.

[4] The recognition of unknown fault types was not validated for the competition.

**Algorithm 1** Ensemble Fault Isolation Algorithm

**Input:** $Det$, $Isol$, $UF$
**Output:** $Pred$
$Pred \leftarrow []$
**for** $i = 1$ to $len(data)$ **do**
  $det \leftarrow \max(Det[i])$
  **if** $det = 1$ **then**
    **if** $UF[i] = -1$ **and** $\max(Isol[i]) \geq 0.99$ **then**
      $uF_{Prob} \leftarrow 0.5$
    **else if** $UF[i] = -1$ **and** $\max(Isol[i]) < 0.99$ **then**
      $uF_{Prob} \leftarrow 1.0$
    **else**
      $uF_{Prob} \leftarrow 0.0$
    **end if**
    $Isol[i] \leftarrow \texttt{append}(Isol[i], uF_{Prob})$
    $Isol[i] \leftarrow Isol[i]/\texttt{sum}(Isol[i])$
  **else**
    $Isol[i] \leftarrow \mathbf{0}_{1\times5}$
  **end if**
  $Pred \leftarrow \texttt{append}(Pred, \arg\max(Isol[i]))$
**end for**

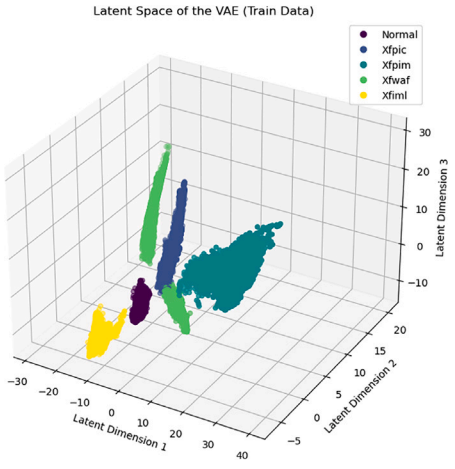


**Fig. 27.** Three-dimensional representation of the latent space of the VAE for training data.

The prediction results for all data, including the new test set with different severities that were not seen in the first set of data, alongside the primary dataset, are shown in Table 12. Since there were no unknown faults in the data, we only validated the true detection rate, the false alarm rate, and the fault isolation accuracy. The results show that our approach is capable of detecting faults with a reliability of 85% but creates false alarms quite often. The isolation accuracy has the potential for improvement. Although the true detection rate and fault isolation accuracy were really good for the training data, the performance was weak for unseen data.

### 8.4. Discussion

The results show that our approach works for data similar to the data seen during training but is not very reliable for very different data. Detection works nicely, but it detects too many false alarms and identifies the correct fault only in 57.4% of cases. Other data-driven models tested during the design phase of the approach did not improve performance as well. The fault-free data was often confused with $f_{ypim}$ which seems to be most similar to the fault-free data. The reason for the bad classification of normal data is the fault detection by VAE$_{det}$, forcing VAE$_{isol}$ to predict a fault even though there is no fault. To overcome this, detection and isolation could be merged into one VAE.

**Table 12**
Performance metrics of various datasets. Blue rows are the training data.

| Data | True detection rate | False alarm rate | Accuracy | Fault isolation accuracy |
|---|---|---|---|---|
| df_output_wltp_f_pic_080 | 0.620 | 0 | 0.579 | 0.002 |
| df_output_wltp_f_pic_085 | 0.951 | 0.000 | 0.888 | 0.950 |
| df_output_wltp_f_pic_090 | 1 | 0 | 0.933 | 0.996 |
| df_output_wltp_f_pic_095 | 0.118 | 1 | 0.177 | 0.003 |
| df_output_wltp_f_pic_105 | 0.353 | 1.000 | 0.396 | 0.001 |
| df_output_wltp_f_pic_110 | 0.998 | 1 | 0.998 | 0.998 |
| df_output_wltp_f_pic_115 | 1 | 0.003 | 0.934 | 0 |
| df_output_wltp_f_pim_080 | 0.999 | 0.087 | 0.938 | 0.999 |
| df_output_wltp_f_pim_085 | 1 | 0 | 0.935 | 1.000 |
| df_output_wltp_f_pim_090 | 1.000 | 0.002 | 0.933 | 1.000 |
| df_output_wltp_f_pim_095 | 0.225 | 1 | 0.276 | 0 |
| df_output_wltp_f_pim_105 | 0.866 | 0.550 | 0.845 | 0 |
| df_output_wltp_f_pim_110 | 0.970 | 0.002 | 0.906 | 0.968 |
| df_output_wltp_f_pim_115 | 1 | 0 | 0.934 | 0 |
| df_output_wltp_f_waf_080 | 1 | 0 | 0.933 | 0 |
| df_output_wltp_f_waf_085 | 0.966 | 0.068 | 0.906 | 0.962 |
| df_output_wltp_f_waf_090 | 1 | 0 | 0.934 | 0 |
| df_output_wltp_f_waf_095 | 1 | 0 | 0.933 | 1.000 |
| df_output_wltp_f_waf_105 | 1 | 0 | 0.933 | 1 |
| df_output_wltp_f_waf_110 | 1.000 | 0 | 0.933 | 0.995 |
| df_output_wltp_f_waf_115 | 0.732 | 0.741 | 0.733 | 0.545 |
| df_output_wltp_f_iml_4mm | 0.777 | 0.793 | 0.778 | 0.777 |
| df_output_wltp_f_iml_6mm | 1.000 | 0.683 | 0.979 | 1.000 |
| df_output_wltp_NF | – | 1 | – | – |
| df_output_wltp_NF_2 | – | 0 | – | – |
| Mean | 0.851 | 0.413 | 0.815 | 0.574 |

$f_{ypim}$ and $f_{ywaf}$ seem to be quite similar since they were confused in both directions. $f_{ypic}$ was confused with the fault-free data for the lower amplitudes, showing that the sensitivity of our approach is too low. The fault that could be detected best was $f_{iml}$, but we cannot completely rely on it since there were only two amplitudes. A slight improvement in the overall performance of our approach might be reached by considering multiple samples in a row rather than looking at individual time steps, but that does not help in handling unknown fault severities. We recommend incorporating domain knowledge into the model to achieve a more reliable prediction, using, e.g., the informed machine learning taxonomy by (Rueden et al., 2021) and/or the decision guidance proposed by (Afroze, Merkelbach, von Enzberg, & Dumitrescu, 2023). In cases like this, where an understanding of the domain is crucial, another way could be to include domain experts more closely in the model development process (Merkelbach, Von Enzberg, Kühn, & Dumitrescu, 2022).

### 8.5. Conclusions

We presented an approach consisting of two VAEs and an SVM, while one VAE was used for fault detection, and the other VAE and the SVM were used for fault isolation. We found that the approach works fine for data that is similar to the data seen during training. However, the system is too complex to be reliably predicted with our approach, especially if there is only a small amount of data available for training. We recommend incorporating domain knowledge in the approach to handle the system's complexity better.

## 9. Benchmark of the solutions

The six proposed solutions employ various strategies to solve the same problem. The diagnosis system solutions presented in the previous sections are benchmarked against different fault scenarios in the test data, see Table 1, and compared to each other. During the evaluation the following metrics were assessed for each fault scenario:

- FAR — False Alarm Rate (False positives)
- TDR — True Detection Rate (True positives)
- TIR — True Isolation Rate, i.e. when a fault is correctly detected, how often the true fault is ranked highest among the diagnoses by the diagnosis system

**Table 13**
Summary of evaluation performance for each solution. Values represent averages over the evaluation datasets.

| Solution | 100% - FAR | TDR | TIR |
|---|---|---|---|
| 1 | 92.2% | 87.3% | 50.1% |
| 2 | 100.0% | 83.1% | 85.5% |
| 3 | 58.4% | 84.5% | 48.8% |
| 4 | 99.9% | 97.2% | 68.0% |
| 5 | 30.5% | 87.4% | 37.2% |
| 6 | 36.2% | 78.6% | 42.3% |

- UR — Unknown fault Rate, i.e. when a fault is correctly detected, how often the unknown fault is ranked highest among the diagnoses by the diagnosis system

The UR is included as a metric because it is preferable for the diagnosis system to indicate an unknown fault rather than falsely identifying an incorrect fault mode as the most likely one. In the evaluation, each metric ranges between 0%–100%, where ideally, TDR and TIR should be 100% while FAR should be 0%. To simplify the comparison between different solutions, figures display 100% minus the FAR so that all curves ideally approach 100%. Note that the exact metrics were not disclosed to participants in advance to prevent cycle beating — i.e., tailoring diagnosis solutions specifically to maximize selected performance metrics.

### 9.1. Evaluation of the solutions

Figs. 28–33 show the evaluation results of each solution. A summary is presented in Table 13. Each subplot shows the diagnosis performance for a single fault mode as a function of fault magnitude. Gray intervals represent fault magnitudes used in the training data, which and excluded in the evaluation. The 0% fault magnitude represents fault-free test data used as a reference.

Note that each fault scenario starts with a fault-free system before fault injection. Thus, both false alarms and missed detections can occur in each fault scenario (except for the fault-free test data). The results show that achieving both high detection and isolation performance across all faults scenarios was a challenging task.

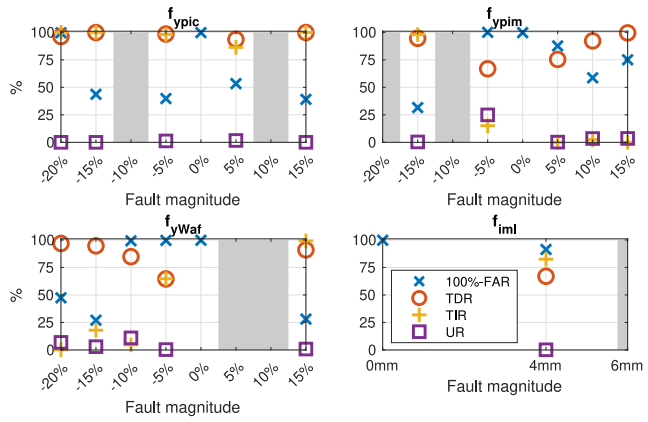


**Fig. 28.** Evaluation of solution 1.

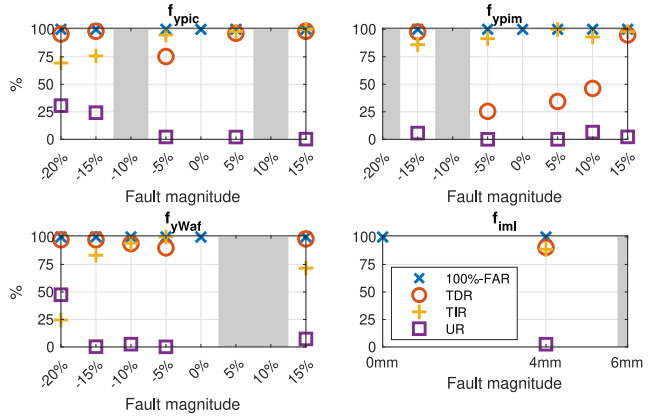


**Fig. 29.** Evaluation of solution 2.

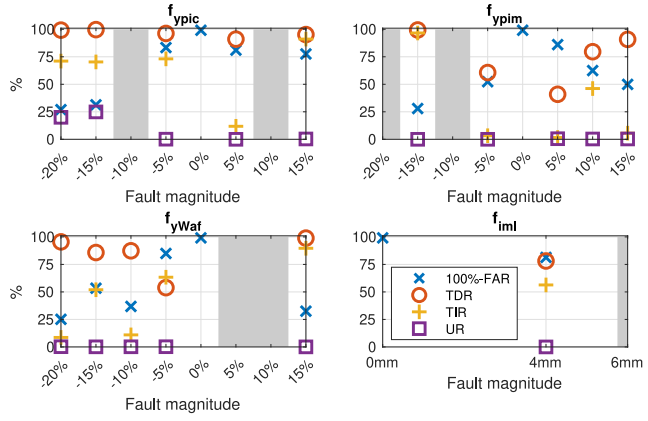


**Fig. 30.** Evaluation of solution 3.

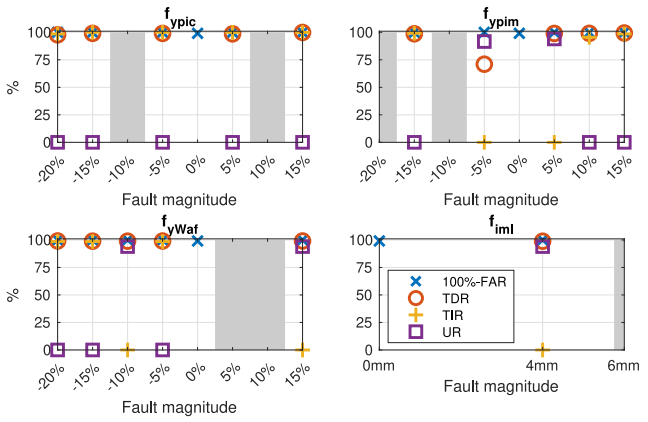


**Fig. 31.** Evaluation of solution 4.

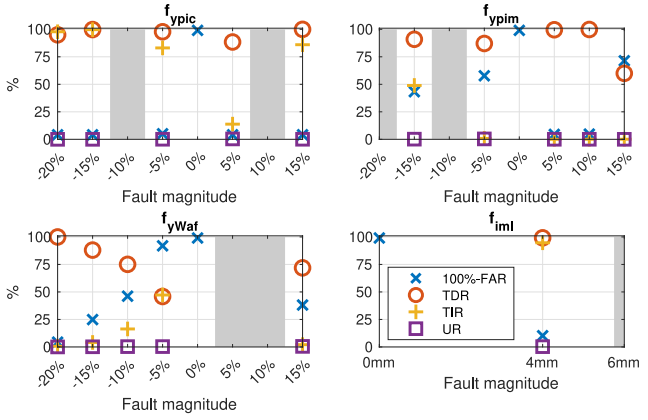


**Fig. 32.** Evaluation of solution 5.

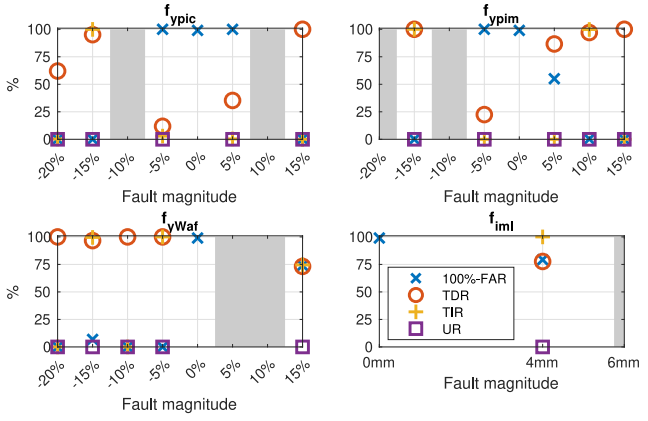


**Fig. 33.** Evaluation of solution 6.

All solutions performance well on the fault-free test data. Solutions 1, 3, 5, and 6, achieved high TDRs in the fault scenarios but also exhibited significant FARs. Solutions 2 and 4 had low FARs. A trend of increasing TDR with larger fault magnitudes is evident in Solutions 1, 2, 3, 5, and 6, whereas Solution 4 consistently achieved a high TDR across all fault scenarios.

TIR performance generally declines when deviating from training data. The drop is more pronounced for faults $f_{ypim}$ and $f_{ywaf}$ than for $f_{ypic}$, likely because the training data only includes either positive or negative magnitudes — not both, see Solutions 1, 3, and 5. In Solutions 4 and 6, isolation performance varies even with small changes in fault magnitude, which may indicate the influence of ambient conditions on measurements. Solution 2 demonstrates strong overall isolation accuracy, while Solution 4 either isolates the fault or classifies it as unknown.

### 9.2. Discussions

Based on the prerequisites of the competition, all solutions address the benchmark's key challenges, such as limited training data and the need to manage unknown faults. Nevertheless, several common

design principles are evident — many solutions implement data-driven, residual-based feature generation.

The primary challenge was generalizing from the limited training data, which led to false alarms. These alarms are likely triggered by out-of-distribution data resulting from varying ambient conditions. Engine dynamics depend on slowly changing variables such as ambient pressure and temperature. Fig. 34 shows the range of ambient pressures and temperatures measured in the datasets. One promising direction is the development of feature-generation techniques that are robust to external operating conditions but sensitive to faults. For example, Solution 2 uses estimated parameters of local residual models as features for fault classification.

When training data is limited, one approach is to explore few-shot learning. In few-shot learning, a classifier is trained using only data from a few examples of each fault (e.g., Li et al., 2023). Another strategy to address the issue of limited training data in machine learning is the use of data augmentation techniques which generate new samples from existing data to enrich the training set and potentially improve the robustness of the trained models (see, e.g., Wen et al., 2021). Transfer learning is also a viable option, allowing the use of pretrained models from similar systems to reduce the need of extensive training data, see e.g. Zhang et al. (2024).

Another suggestion of exploration is the design of hybrid diagnosis systems that combine model-based and data-driven techniques, see e.g. Tidriri et al. (2016). The solutions presented here made limited use of the mathematical model provided in the benchmark. Designing data-driven residuals from structural model information has been investigated in, e.g. Mohammadi, Krysander, and Jung (2025), Pulido, Zamarreño, Merino, and Bregon (2019). Another approach is to design graph neural networks to model the interactions between signals (Chen, Liu, Hu, & Ding, 2021). Selecting model structures of data-driven models based on physical insights could enhance generalizability and interpretability, especially when training data is scarce.

All solutions in this work employ open-set fault isolation strategies, meaning they can identify unknown fault scenarios. This capability is valuable when training data is limited and could, e.g., be used to identify new cases requiring operator intervention or that could be used to refine the fault diagnosis models. Instead of relying on general-purpose classification models, one potential improvement is to explore data-driven models that represent the behavior of individual fault modes to enhance generalizability when training data from faults is scarce, see e.g. Jung and Säfdal (2022).

## 10. Conclusions

This paper presents a fault diagnosis benchmark to illustrate the challenges that must be addressed in many industrial applications and six different solutions to the benchmark. The varying solutions show that there are different approaches to address the same problem formulation.

The participants in the competition were offered both model information and data in the benchmark. All solutions try to address the challenges of limited training data and the ability to handle unknown faults. There were attempts of utilizing the model in the diagnosis system design. However, the main approach is the use of a data-driven fault diagnosis. The evaluations showed that performance a combination of different fault diagnosis methods is needed.

The goal of the LiU-ICE benchmark is to inspire engineers and researchers developing new fault diagnosis solutions that aims to address the complicating factors faced in many industrial applications. By providing both model and data it is possible to experiment with both model-based and data-driven diagnosis system designs that can be evaluated and compared.

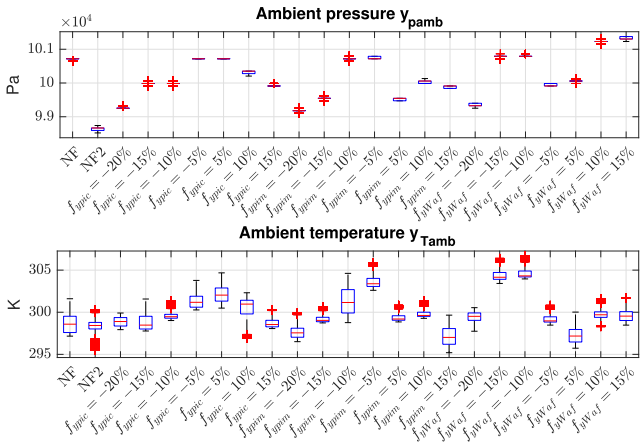


**Fig. 34.** The range of ambient pressures and temperatures measured in the different datasets.

## 11. Future benchmark activities

There are plans to arrange new competitions for the LiU-ICE benchmark. For the new competitions, there will be more datasets provided to the benchmark including different drive cycles and new fault scenarios. To help researchers who want to work with the benchmark, the different conditions of each competition will be provided on the homepage such that it is possible to develop new solutions.

## CRediT authorship contribution statement

**Daniel Jung:** Writing – original draft, Supervision, Project administration, Methodology, Investigation. **Erik Frisk:** Writing – original draft, Validation, Methodology, Investigation. **Mattias Krysander:** Writing – original draft, Validation, Methodology, Investigation. **Anna Sztyber-Betley:** Writing – original draft, Validation, Methodology, Investigation. **Francesco Corrini:** Writing – original draft, Validation, Methodology, Investigation. **Andrea Arici:** Writing – original draft, Validation, Methodology, Investigation, Formal analysis. **Nicolas Anselmi:** Writing – original draft, Validation, Methodology, Investigation. **Mirko Mazzoleni:** Writing – original draft, Validation, Methodology, Investigation. **Jiamin Xu:** Writing – original draft, Validation, Methodology, Investigation. **Siwen Mo:** Writing – original draft, Validation, Methodology, Investigation, Formal analysis. **Zixuan Xu:** Writing – original draft, Validation, Methodology, Investigation. **Chongpan Yang:** Writing – original draft, Validation, Methodology, Investigation. **Zhile Du:** Writing – original draft, Validation, Methodology, Investigation. **Hossein Safaeipour:** Writing – original draft, Validation, Methodology, Investigation. **Mehdi Forouzanfar:** Writing – original draft, Validation, Methodology, Investigation. **Vahid Mirahi:** Writing – original draft, Validation, Methodology, Investigation. **Anna Pinnarelli:** Writing – original draft, Validation, Methodology, Investigation. **Vicenç Puig:** Writing – original draft, Validation, Methodology, Investigation. **Qiao Deng:** Writing – original draft, Validation, Methodology, Investigation. **Jiakun Liu:** Writing – original draft, Validation, Methodology, Investigation. **Haobin Ke:** Writing – original draft, Validation, Methodology, Investigation. **Wanting Zhu:** Writing – original draft, Validation, Methodology, Investigation. **Silke Merkelbach:** Writing – original draft, Validation, Methodology, Investigation. **Maryam Ahang:** Writing – original draft, Validation, Methodology, Investigation. **Homayoun Najjaran:** Writing – original draft, Validation, Methodology, Investigation.

## Declaration of competing interest

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

## Acknowledgments

The authors would like to thank Volvo Cars and Aurobay for providing the engine and use of data and Tobias Lindell for the help with data collection.

## References

Abid, A., Khan, M., & Iqbal, J. (2021). A review on fault detection and diagnosis techniques: basics and beyond. *Artificial Intelligence Review*, *54*, 3639–3664.

Afroze, L., Merkelbach, S., von Enzberg, S., & Dumitrescu, R. (2023). Domain knowledge injection guidance for predictive maintenance. In *International conference on machine learning for cyber-physical systems* (pp. 75–87). Springer.

Ahang, M., Abbasi, M., Charter, T., & Najjaran, H. (2024). Condition monitoring with incomplete data: An integrated variational autoencoder and distance metric framework. arXiv preprint arXiv:2404.05891.

Ahang, M., Charter, T., Ogunfowora, O., Khadivi, M., Abbasi, M., & Najjaran, H. (2024). Intelligent condition monitoring of industrial plants: An overview of methodologies and uncertainty management strategies. arXiv preprint arXiv:2401.10266.

Andrew, G., Arora, R., Bilmes, J., & Livescu, K. (2013). Deep canonical correlation analysis. In *International conference on machine learning* (pp. 1247–1255). PMLR.

Bartyś, M., Patton, R., Syfert, M., de las Heras, S., & Quevedo, J. (2006). Introduction to the DAMADICS actuator FDI benchmark study. *Control Engineering Practice*, *14*(6), 577–596.

Blanke, M., Kinnaert, M., Lunze, J., & Staroswiecki, M. (2015). *Diagnosis and fault-tolerant control*. New York, NY, USA: Springer.

Blanke, M., Kinnaert, M., Lunze, J., & Staroswiecki, M. (2016). *Introduction to diagnosis and fault-tolerant control*. Springer.

Brahimi, L., Hadroug, N., Iratni, A., Hafaifa, A., & Colak, I. (2024). Advancing predictive maintenance for gas turbines: An intelligent monitoring approach with ANFIS, LSTM, and reliability analysis. *Computers & Industrial Engineering*, *191*, Article 110094.

Chen, Z., Ding, S. X., Peng, T., Yang, C., & Gui, W. (2017). Fault detection for non-Gaussian processes using generalized canonical correlation analysis and randomized algorithms. *IEEE Transactions on Industrial Electronics*, *65*(2), 1559–1567.

Chen, Z., Ding, S. X., Zhang, K., Li, Z., & Hu, Z. (2016). Canonical correlation analysis-based fault detection methods with application to alumina evaporation process. *Control Engineering Practice*, *46*, 51–58.

Chen, H., Jiang, B., Ding, S. X., & Huang, B. (2020). Data-driven fault diagnosis for traction systems in high-speed trains: A survey, challenges, and perspectives. *IEEE Transactions on Intelligent Transportation Systems*, *23*(3), 1700–1716.

Chen, Z., Liang, K., Ding, S. X., Yang, C., Peng, T., & Yuan, X. (2021). A comparative study of deep neural network-aided canonical correlation analysis-based process monitoring and fault detection methods. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(11), 6158–6172.

Chen, D., Liu, R., Hu, Q., & Ding, S. (2021). Interaction-aware graph neural networks for fault diagnosis of complex industrial processes. *IEEE Transactions on Neural Networks and Learning Systems*, *34*(9), 6015–6028.

Chen, Z., Mo, S., Ke, H., Ding, S. X., Jiang, Z., Yang, C., et al. (2025). CCDNN: A novel deep learning architecture for multi-source data fusion. *IEEE/CAA Journal of Automatica Sinica*.

Dai, X., & Gao, Z. (2013). From model, signal to knowledge: A data-driven perspective of fault detection and diagnosis. *IEEE Transactions on Industrial Informatics*, *9*(4), 2226–2238.

Downs, J., & Vogel, E. (1993). A plant-wide industrial process control problem. *Computers & Chemical Engineering*, *17*(3), 245–255.

Eriksson, L. (2007). Modeling and control of turbocharged SI and DI engines. *Oil & Gas Science and Technology - Revue de L'IFP*, *62*, http://dx.doi.org/10.2516/ogst:2007042.

Eriksson, L., Frei, S., Onder, C., & Guzzella, L. (2002). Control and optimization of turbo charged spark ignited engines. In *IFAC world congress*.

European Community, Japan, & United States of America (2009). Proposal to develop a new global technical regulation on worldwide harmonized light vehicle test procedures. United Nations Digital Library.

Feiyi, R., & Jinsong, Y. (2015). Fault diagnosis methods for advanced diagnostics and prognostics testbed (ADAPT): A review. In *2015 12th IEEE international conference on electronic measurement & instruments: Vol. 1*, (pp. 175–180). IEEE.

Frisk, E., Bregon, A., Aslund, J., Krysander, M., Pulido, B., & Biswas, G. (2012). Diagnosability analysis considering causal interpretations for differential constraints. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, *42*(5), 1216–1229.

Frisk, E., Krysander, M., & Jung, D. (2017). A toolbox for analysis and design of model based diagnosis systems for large scale models. *IFAC-PapersOnLine*, *50*(1), 3287–3293.

Fukumizu, K., Bach, F. R., & Gretton, A. (2007). Statistical consistency of kernel canonical correlation analysis. *Journal of Machine Learning Research*, *8*(2).

Gagliardi, G., Tedesco, F., & Casavola, A. (2018). A LPV modeling of turbocharged spark-ignition automotive engine oriented to fault detection and isolation purposes. *Journal of the Franklin Institute*, *355*(14), 6710–6745. http://dx.doi.org/10.1016/j.jfranklin.2018.06.038.

Gao, Z., Cecati, C., & Ding, S. (2015). A survey of fault diagnosis and fault-tolerant techniques—Part I: Fault diagnosis with model-based and signal-based approaches. *IEEE Transactions on Industrial Electronics*, *62*(6), 3757–3767.

Higgins, I., Matthey, L., Pal, A., Burgess, C. P., Glorot, X., Botvinick, M. M., et al. (2017). Beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR (Poster)*: *Vol. 3*.

Hotelling, H. (1992). Relations between two sets of variates. In *Breakthroughs in statistics: methodology and distribution* (pp. 162–190). Springer.

Jafari, H., Ramezani, A., & Forouzanfar, M. (2022). Parameter-dependent Lyapunov function based fault estimation and fault-tolerant control for LPV systems. *International Journal of Systems Science*, *53*(11), 2374–2389.

Jia, X., Huang, B., Feng, J., Cai, H., & Lee, J. (2018). A review of PHM data competitions from 2008 to 2017: Methodologies and analytics. In *Proceedings of the annual conference of the Prognostics and Health Management Society* (pp. 1–10).

Jung, D. (2022). Automated design of grey-box recurrent neural networks for fault diagnosis using structural models and causal information. In *Learning for dynamics and control conference* (pp. 8–20). PMLR.

Jung, D., Frisk, E., & Krysander, M. (2024). The LiU-ICE Benchmark–an industrial fault diagnosis case study. arXiv preprint arXiv:2408.13269.

Jung, D., & Säfdal, J. (2022). A flexi-pipe model for residual-based engine fault diagnosis to handle incomplete data and class overlapping. *IFAC-PapersOnLine*, *55*(24), 84–89.

Kościelny, J. M., Bartyś, M., Syfert, M., & Sztyber, A. (2022). A graph theory–based approach to the description of the process and the diagnostic system. *International Journal of Applied Mathematics and Computer Science*, *32*(2), 213–227.

Krysander, M., Aslund, J., & Nyberg, M. (2008). An efficient algorithm for finding minimal overconstrained subsystems for model-based diagnosis. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, *38*(1), 197–206.

Li, G., Li, Y., Fang, C., Su, J., Wang, H., Sun, S., et al. (2023). Research on fault diagnosis of supercharged boiler with limited data based on few-shot learning. *Energy*, *281*, Article 128286.

Li, S., & Yu, J. (2022). A multisource domain adaptation network for process fault diagnosis under different working conditions. *IEEE Transactions on Industrial Electronics*, *70*(6), 6272–6283.

Liu, Z., & Zhang, L. (2020). A review of failure modes, condition monitoring and fault diagnosis methods for large-scale wind turbine bearings. *Measurement*, *149*, Article 107002.

Ljung, L. (1986). *System identification: theory for the user*. Prentice-Hall, Inc.

Lundgren, A., & Jung, D. (2022). Data-driven fault diagnosis analysis and open-set classification of time-series data. *Control Engineering Practice*, *121*, Article 105006.

Melo, A., Câmara, M., Clavijo, N., & Pinto, J. (2022). Open benchmarks for assessment of process monitoring and fault diagnosis techniques: a review and critical analysis. *Computers & Chemical Engineering*, Article 107964.

Merkelbach, S., Von Enzberg, S., Kühn, A., & Dumitrescu, R. (2022). Towards a process model to enable domain experts to become citizen data scientists for industrial applications. In *2022 IEEE 5th international conference on industrial cyber-physical systems* (pp. 1–6). IEEE.

Mirnaghi, M., & Haghighat, F. (2020). Fault detection and diagnosis of large-scale HVAC systems in buildings using data-driven methods: A comprehensive review. *Energy and Buildings*, *229*, Article 110492.

Mohammadi, A., Krysander, M., & Jung, D. (2025). Consistency-based diagnosis using data-driven residuals and limited training data. *Control Engineering Practice*, *159*, Article 106283.

Odgaard, P., & Stoustrup, J. (2012). Results of a wind turbine FDI competition. *IFAC Proceedings Volumes*, *45*(20), 102–107.

Odgaard, P. F., Stoustrup, J., & Kinnaert, M. (2013). Fault-tolerant control of wind turbines: A benchmark model. *IEEE Transactions on Control Systems Technology*, *21*(4), 1168–1182.

Poll, S., Patterson-Hine, A., Camisa, J., Garcia, D., Hall, D., Lee, C., et al. (2007). Advanced diagnostics and prognostics testbed. In *Proceedings of the 18th international workshop on principles of diagnosis (DX-07)* (pp. 178–185).

Pulido, B., Zamarreño, J., Merino, A., & Bregon, A. (2019). State space neural networks and model-decomposition methods for fault diagnosis of complex industrial systems. *Engineering Applications of Artificial Intelligence*, *79*, 67–86.

Robbins, H., & Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, 400–407.

Rojas, H. D., Rojas, H. E., & Cortés-Romero, J. (2022). Fault diagnosis based on algebraic identification assisted by extended state observers. *Nonlinear Dynamics*, *107*(1), 871–888.

Safaeipour, H., Forouzanfar, M., Puig, V., & Birgani, P. T. (2023). Incipient fault diagnosis and trend prediction in nonlinear closed-loop systems with Gaussian and non-Gaussian noise. *Computers & Chemical Engineering*, *177*, Article 108348.

Salahshoor, K., Kordestani, M., & Khoshro, M. S. (2010). Fault detection and diagnosis of an industrial steam turbine using fusion of SVM (support vector machine) and ANFIS (adaptive neuro-fuzzy inference system) classifiers. *Energy*, *35*(12), 5472–5482.

Smith, W., & Randall, R. (2015). Rolling element bearing diagnostics using the Case Western Reserve University data: A benchmark study. *Mechanical Systems and Signal Processing*, *64*, 100–131.

Stetco, A., Dinmohammadi, F., Zhao, X., Robu, V., Flynn, D., Barnes, M., et al. (2019). Machine learning methods for wind turbine condition monitoring: A review. *Renewable Energy*, *133*, 620–635.

Su, H., & Lee, J. (2024). Machine learning approaches for diagnostics and prognostics of industrial systems using open source data from PHM data challenges: A review. *International Journal of Prognostics and Health Management*, *15*(2).

Sztyber, A., Ostasz, A., & Kościelny, J. (2015). Graph of a process - a new tool for finding model's structures in model based diagnosis. *IEEE Transactions on System, Man, and Cybernetics: Systems*, *45*(7), 1004–1017. http://dx.doi.org/10.1109/TSMC.2015.2444101.

Theissler, A., Pérez-Velázquez, J., Kettelgerdes, M., & Elger, G. (2021). Predictive maintenance enabled by machine learning: Use cases and challenges in the automotive industry. *Reliability Engineering & System Safety*, *215*, Article 107864.

Tidriri, K., Chatti, N., Verron, S., & Tiplica, T. (2016). Bridging data-driven and model-based approaches for process fault diagnosis and health monitoring: A review of researches and future challenges. *Annual Reviews in Control*, *42*, 63–81.

Von Rueden, L., Mayer, S., Beckh, K., Georgiev, B., Giesselbach, S., Heese, R., et al. (2021). Informed machine learning–a taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Transactions on Knowledge and Data Engineering*, *35*(1), 614–633.

Wen, Q., Sun, L., Yang, F., Song, X., Gao, J., Wang, X., et al. (2021). Time series data augmentation for deep learning: A survey. In *Proceedings of the thirtieth international joint conference on artificial intelligence* (pp. 4653–4660). International Joint Conferences on Artificial Intelligence Organization.

Wu, M. (2024). Rolling bearing fault diagnosis data set. http://dx.doi.org/10.21227/w41e-9y33.

Xiong, R., Sun, W., Yu, Q., & Sun, F. (2020). Research progress, challenges and prospects of fault diagnosis on battery system of electric vehicles. *Applied Energy*, *279*, Article 115855.

Xu, Z., & Saleh, J. (2021). Machine learning for reliability engineering and safety applications: Review of current status and future opportunities. *Reliability Engineering & System Safety*, *211*, Article 107530.

Zhang, Y., Liu, W., Chen, Z., Li, K., & Wang, J. (2021). On the properties of Kullback-Leibler divergence between Gaussians. CoRR. abs/2102.05485.

Zhang, J., Pei, G., Zhu, X., Gou, X., Deng, L., Gao, L., et al. (2024). Diesel engine fault diagnosis for multiple industrial scenarios based on transfer learning. *Measurement*, *228*, Article 114338.

Zio, E. (2022). Prognostics and health management (PHM): Where are we and where do we (need to) go in theory and practice. *Reliability Engineering & System Safety*, *218*, Article 108119.