Diagnosing PARC's Refrigerator Benchmark with Data-Driven Methods

Alexander Feldman and **Rui Abreu** and **Bhaskar Saha** and **Anurag Ganguli** and **Johan de Kleer**
PARC Inc.
e-mail: {afeldman,rui,bhaskar.saha,anurag.ganguli,dekleer}@parc.com

Abstract

Diagnosing real-world systems is a challenging task due to, e.g., complexity, measurement error and lack of control in experimentation. Designing controlled experiments, diagnostic algorithms, and building benchmarks to validate the algorithms pave the way to address the challenge.

More than $1\,800$ hours of power and temperature data from three identical household refrigerators, modified for the purpose of benchmark creation, were recorded. Controlled failure injection of non-destructive faults has been performed over multiple 24-hour scenarios. The thermal and electric data in the benchmark has been analyzed manually and with four data-driven machine-learning-based algorithms. The performance of each data-driven, machine-learning diagnostic algorithm has been characterized in terms of diagnostic metrics such as false positives, false negatives, and isolation time.

1 Introduction

Diagnostics of physical systems is important for the progress of science and engineering. Designing diagnostic algorithms that take into consideration the specific physical domain (such as electrical, thermal, or mechanical) promises improvement in the quality of diagnosis as different physical domains show different speeds, failure propagation rates and mechanisms of failure. While there are analogies between electrical short-circuits and thermal transfers, electrical circuits can change their state very quickly (in the order of microseconds) while thermal systems may take hours to warm-up or cool-down.

We are interested in the thermal aspects of systems such as refrigerators, air-conditioning units, cryostat units, ore smelters, infrared telescopes, and nuclear reactors. Benchmarks are needed to study the performance of diagnostic algorithms. While there is a widely-available electrical diagnostic benchmark provided by the NASA Ames Research Center [Feldman *et al.*, 2010], there is no thermal diagnostic benchmark. We have created one and we have validated the benchmark with several data-driven diagnostic algorithms.

To create thermo-electric benchmark for diagnostic algorithms we have acquired three identical household refrigerators. These refrigerators were significantly modified in order to turn them into test-beds. The modification includes the installation of temperature, voltage, and current sensors, on/off relays and door actuators, and the modification of the control mechanism to allow controlled injection of failures.

Figure 1: One of PARC's refrigerator test-beds

The thermo-electric benchmark that we present in this paper consists of more than 1800 hours of instrumentation: 24 DS18B20 semiconductor temperature sensors and a specially designed $500\,\mathrm{Hz}$ (synchronized) voltage and current logger with 24-bit resolution. Each failure or nominal scenario is 24 hours long. Failure injections and retraction were performed in a controlled manner.

A subset of the benchmark (scenarios are added to the benchmark at the time of writing of this article) has been diagnosed with four data-driven diagnostic algorithms based on machine learning: decision trees, random forests, Support Vector Machines (SVM), and Single-layer Neural Networks. Diagnostic metrics such as classification errors, false positive rates, false negative rates, isolation time, and CPU performance were computed from classification results and the failure injections.

Although the machine-learning methods resulted in more than $90\ \%$ accuracy score, translation to diagnostic metrics often led to $3 - 4$ hour long intervals (in 24 hour

scenarios) when the machine-learning-based diagnoser was wrong. This led us to conclude that future work is needed on improving data-driven diagnostic methods.

2 Comparison Framework

This article provides practical foundations of a diagnostic framework. The framework describes knowledge representation and algorithms. Its purpose is to allow comparison of various diagnostic approaches as well as validation of the algorithm design.

All concepts in this article are illustrated with the help of a fictional three-tank system, as shown in figure 2. The tanks are denoted as T_1, T_2, and T_3. They all have the same area $A_1 = A_2 = A_3 = 3$ [m^2]. We use $g = 9.8$ and assume that the liquid is "pure" water with density $\rho = 1$.

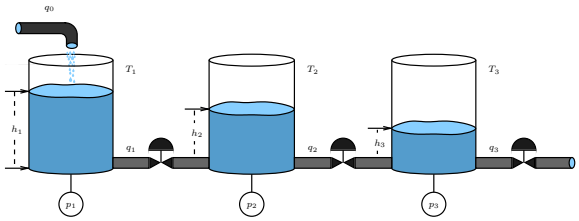

Figure 2: Three tanks running example

Tank T_1 is filled from a pipe q_0 with a constant flow of $0.75\,\mathrm{m}^3\,\mathrm{s}^{-1}$. It drains into T_2 via a pipe q_1. The liquid level is denoted as h_1. There is a pressure sensor p_1 connected to T_1 that measures the pressure in Pascals [Pa]. Starting from Newton's (and Bernouli's) equations and manipulating them (the actual derivation is irrelevant in this paper) we derive the following Ordinary Differential Equation (ODE) that gives the level of the liquid in T_1:

$$\frac{dh_1}{dt} = \frac{q_0 - k_1\sqrt{h_1 - h_2}}{A_1} \tag{1}$$

In Eq. 1, the coefficient k_1 is the product of the cross-sectional area of the tank A_1 and the area of the drainage hole and $\sqrt{2g}$ and the friction/contraction factor of the hole. We emphasize the use of k_1 because, later, we will be "diagnosing" our system in term of changes in k_1. Consider a physical valve R_1 between T_1 and T_2 that constraints the flow between the two tanks. We can say that the valve changes proportionally to the cross-sectional drainage area of q_1 and hence k_1. The diagnostic task is to compute the true value of k_1, given p_1, and from k_1 we can compute the actual position of the valve R_1.

The water levels of T_2 and T_3, denoted as h_2 and h_3 respectively, are given by:

$$\frac{dh_i}{dt} = \frac{k_{i-1}\sqrt{h_{i-1} - h_i} - k_i\sqrt{h_i}}{A_i}, \tag{2}$$

where i is the tank index ($i \in \{2, 3\}$). We assume that $k_1 = k_2 = k_3 = 0.75$.

The water level can be turned into pressure with the following equation:

$$p_i = \frac{g\,h_i\,A}{A} = g\,h_i \tag{3}$$

where i is the tank index ($i \in \{1, 2, 3\}$).

The initial water level in the three tanks is zero.

Automatic diagnostic methods are applied to devices that are built of interacting components. It is often possible, especially in non-critical situations, to wait until a component fails and to replace it as opposed to predicting remaining useful life or simply following a maintenance schedule.

Definition 1 (Diagnostic System)**.** A diagnostic system SD is defined as the pair $\langle \mathrm{COMPS}, \mathrm{OBS} \rangle$ where COMPS are component variables, and OBS are observable (sensor) variables.

In the three tanks running example the set of components is the three tanks $\mathrm{COMPS} = \{T_1, T_2, T_3\}$ and the set of observable variables $\mathrm{OBS} = \{p_1, p_2, p_3\}$.

All experiments described in this article are on dynamic systems where the **signal** is one of the most fundamental primitives. We use functions to represent both signals and systems. Signals are functions that map time (always denoted as t) into a range, often a physical quantity such as temperature or voltage. Functions use **variables**.

Definition 2 (Observation)**.** An observation $\alpha = \{\mathbf{a_1}, \ldots, \mathbf{a_n}\}$ is a set of signals associated with the observable variables $a_1, \ldots, a_n$.

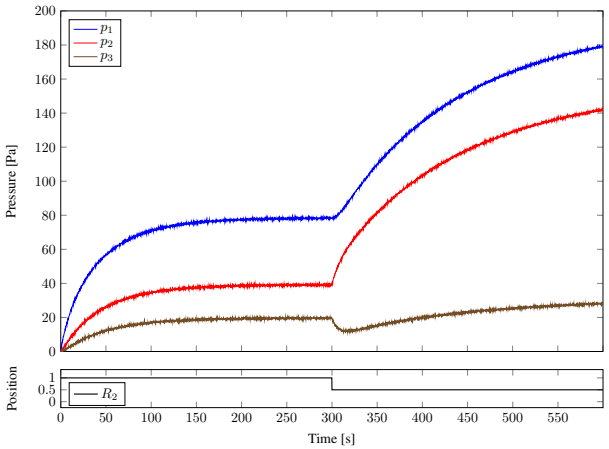

Figure 3: Simulated observation and the corresponding failure injection for the 3-tank system

It is assumed that $\mathbf{a_i} \in \mathbb{R}$ for $i \in \{1, \ldots, n\}$. An example observation for the three-tanks running example is shown in figure 3.

Definition 3 (Fault Injection)**.** A fault injection $\phi = \{\mathbf{f_1}, \ldots, \mathbf{f_m}\}$ is a set of discrete-value signals associated with the fault variables $f_1, \ldots, f_m$.

As an example for a three-tanks fault-injection we can consider a singleton set ϕ whose only function is $\mathbf{R_1} = 1$ (fully-opened) at $0 \leq t < 300$ and $\mathbf{R_1} = 0.5$ (half-opened) at $300 \leq t < 500$.

Definition 4 (Diagnosis)**.** Given a system SD with fault variables $\mathrm{COMPS} = \{c_1, c_2, \ldots, c_n\}$, a diagnosis $\omega = \{\Pr(c_i = d_{i,j})\}$ is a set of probability distribution functions specifying the probability of a component c_i to be in a state $d_{i,j}$.

Diagnostic metrics are functions that take a fault injection ϕ and a diagnosis ω and characterize the optimality of a diagnostic algorithm.

Definition 5 (Classification Errors)**.** Given a set of components COMPS, a fault injection ϕ and a diagnosis ω, the classification errors M_{err} is defined as:

$$M_{\text{err}} = \sum_{c \in \text{COMPS}} \int_0^{t_e} |\text{Pr}(\phi(c)) - \text{Pr}(\omega(c))| dt$$

where t_e is the end of the time horizon.

Classification errors have units and the unit is "error seconds". False positives and false negatives can be defined in a similar manner the sum of the false positives and false negatives is equal to the total number of classification errors. An important property of a diagnostic algorithm is how quickly it finds the root-cause-of-failure as defined in the next metric.

Definition 6 (Isolation Time)**.** Given a set of components COMPS, a fault injection ϕ and a diagnosis ω, the isolation time M_{iso} is defined as:

$$M_{\text{iso}} = \begin{cases} t_f(\omega) - t_f(\phi) & \text{if } t_f(\omega) \geq t_f(\phi) \\ \infty & \text{otherwise} \end{cases}$$

where $t_f(\omega)$ is the earliest time for which $\text{Pr}(\omega(c)) > 0$ and $t_f(\phi)$ is the earliest time for which $\text{Pr}(\phi(c)) > 0$.

A diagnostic algorithm evaluation framework can compute other metrics such as isolation accuracy and computational time. The use of the metrics depends on the application context (for example decision making or repair).

3 Refrigerator Test-Bed Design

The most important characteristics of a test-bed is the support of reproducible diagnostic experiments. To create a diagnostic benchmark we need multiple repetitions of the same diagnostic experiment (fault-injection) so we can statistically validate the correctness of our diagnostic algorithms (it does not matter if they are data-driven, model-driven, rule-driven, or probabilistic).

Notice that in the case of the refrigerators, we do not have full control over the environment. Although, the room in which the refrigerators are placed is climate controlled, there are small variations due to the outside weather, room use and maintenance or malfunctions of the main building's HVAC system. To compensate for these variations in the environment we will apply the rule: measure what you cannot control. Figure 4 shows the architecture of the test-bed. The rectangles show the various components in the test-bed and the arrows signify the type of information that is being transferred or the physical quantity that is being measured or changed.

Multiple temperature sensors measure the temperature inside the refrigerator, inside the freezer, and in the room. We disconnect the thermostat from the compressor circuit and we connect it to a digital input of the Arduino Mega board. This way we can think of the thermostat as a one-bit temperature sensor. The power is measured by a voltage and current sensor.

On the actuation side, we have a relay board that switches the fridge on and off. We also install a linear actuator that can open or close the door to simulate human activity.

The test-bed refrigerator is controlled by an Arduino MEGA 2560 board. The Arduino board is also responsible for interfacing the sensors and sending the sensed values to the Linux base. The Arduino board waits for commands from the Linux base station and actuates the linear actuator (for opening or closing the door) or the power relay (for turning the refrigerator on or off).

The line plot in figure 5 shows the readings from three DS18B20 temperature sensors and the thermostat position for the duration of one nominal experiment. The temperature in the freezer and in the refrigerator is oscillating around the set temperature as expected. The room temperature depends on the building HVAC and on the outside temperature.

The line plot in figure 6 shows the readings from all temperature sensors during one hour of nominal operation. The lower part of the figure shows a full thermostat cycle. The temperature readings are almost identical per group of sensors. The small differences are due to their location. This indicates good accuracy of the DS18B20 semiconductor temperature sensors (the data sheet specifies accuracy of $0.5\,^{\circ}\text{C}$ but we expect that it is actually higher).

4 Machine-Learning-Based Diagnostics

In this section we introduce and analyze several supervised learning methods. A straightforward way to think of diagnosis is to classify the sensor data into a number of nominal or faulty behaviors. To do this we first train a classifier which finds a function that passes or is near a set of points (in a higher-dimensional space). The classifier is then used for predicting the state of the device under test. The input for both training and run-time classification are features which are computed from the sensor-data. We also compute labels from the fault-injection. These labels are used both for training and evaluating the performance of the classifier.

Algorithm 1: TRAINCLASSIFIER(α, ϕ, K)

Input: α, observation
Input: ϕ, fault injection
Input: K, maxium rolling-window size
Result: $\mathcal{C}$, classifier function
for $\mathbf{a} \in \alpha, f \in \mathcal{F}, 1 \leq k \leq K$ **do**
  $F \leftarrow F \cup f(\mathbf{a}, k)$
end
for $\mathbf{a}, \mathbf{b} \in \alpha, g \in \mathcal{G}, 1 \leq k \leq K$ **do**
  $F \leftarrow F \cup g(\mathbf{a}, \mathbf{b}, k)$
end
$\mathbf{l} \leftarrow$ COMPUTELABELS(ϕ)
$\mathcal{C} \leftarrow$ FINDCLASSIFIER($\mathcal{F}, \mathbf{l}$)
return $\mathcal{C}$

Algorithm 1 shows the process of training. The COMPUTELABELS function projects the fault injection into a single set of labels. For single faults this is straightforward. For multiple faults it results into an exponential blow-up in the number of labels which may be a problem depending on the context. The actual training is done by the FINDCLASSIFIER subroutine.

Algorithm 1 also uses two sets of functions $\mathcal{F}$, $\mathcal{G}$ that compute features of single sensors and pairs of sensors respectively. These feature functions can be as simple as moving average or they can compute derivatives, have knowledge of physics, etc.

The actual diagnostic algorithm just takes the classifier $\mathcal{C}$ and applies to an observation α. The classification result has to be projected to a health state ω. This last step is context

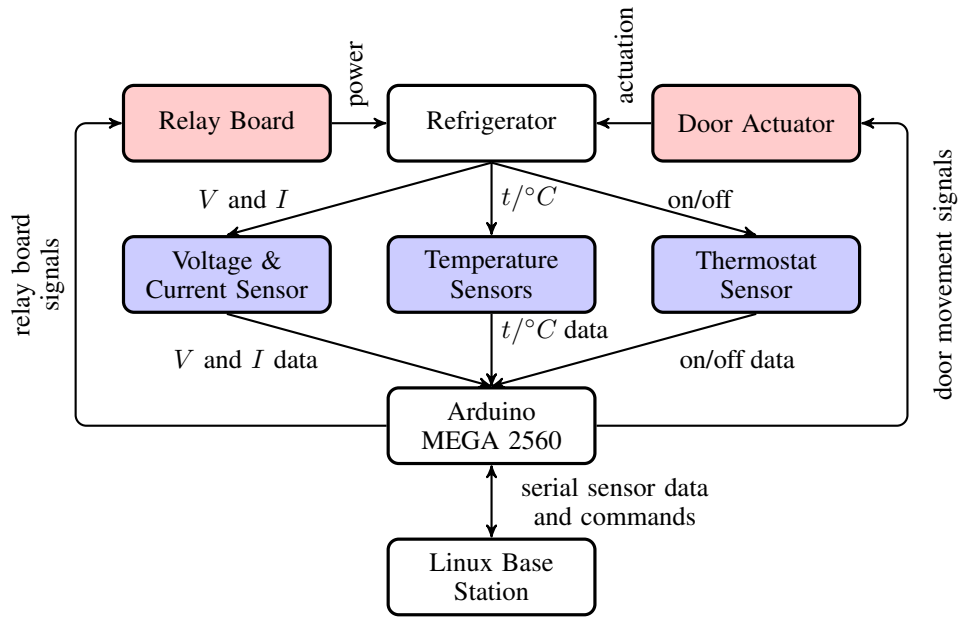

Figure 4: Test-bed overview

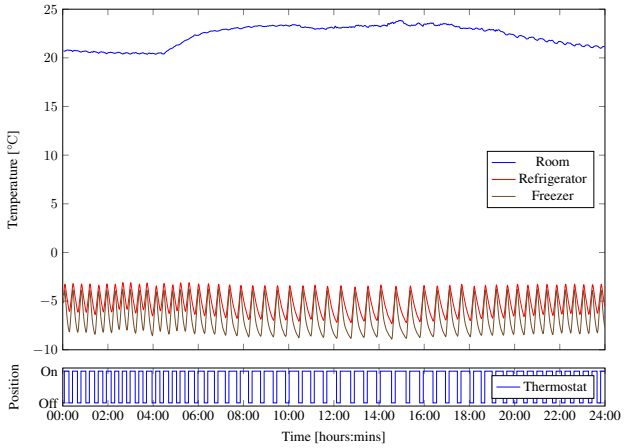

Figure 5: Refrigerator test-bed temperatures and thermostat position during a 24-hour nominal experiment

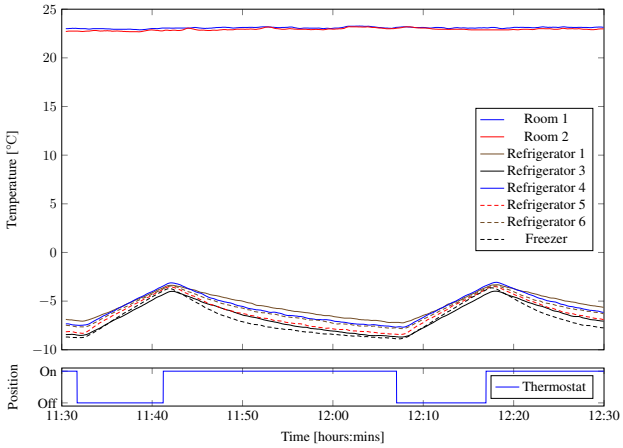

Figure 6: Refrigerator test-bed temperatures and thermostat position during a 1-hour nominal experiment

specific and is again straightforward in the case of single faults.

4.1 Feature Engineering

Features in machine learning are signals containing information about a measurable property of the system being observed. It might be a challenge to use raw sensor data in machine-learning. There may be too much data: the refrigerator, for example, uses an AC compressor and is connected to a 60 Hz power supply. The Nyquist-Shannon sampling theorem says that to recover the 60 Hz, one has to sample at at-least 120 Hz. In reality, most engineers over-sample in an attempt to capture higher frequencies that would preserve transients in the signal. In the refrigeration case-study, the sampling rate is 500 Hz which means that at every 2 ms there is a new data point. Even at this modest sampling frequency it may be computationally hard to apply the classification or regression formula. We are only interested in algorithms that can compute diagnosis at real-time. Summarizing the signals over time-windows solves the computational difficulty problem.

Another reason for using features is that classification and regression machine-learning formulas are typically not good enough to capture the complex physics of the devices we are trying to diagnose. Manually constructing features is bridging physics and model-based reasoning and machine learning. We can actually "hide" a fully-fledged model-based diagnosis engine as a feature. This feature will directly compute if the device is failing and then there is no work for the classifier to determine the state of the system.

Table 1 shows the types of features that are used for diagnosing the refrigerator. They are all computed for a sliding window of k samples. The first four features are simply common sliding-window descriptive statistics. The fifth one is the first derivative of temperature (of course, first a low-pass filter with a cut-off frequency of 1 mHz is used to

Description	Formula
Minimum value	$x_{\min} = \min\{x_i, x_{i-1}, \ldots, x_{i-k}\}$
Maximum value	$x_{\max} = \max\{x_i, x_{i-1}, \ldots, x_{i-k}\}$
Mean value	$\mu = \frac{1}{k}\sum_{j=i-k}^{i} x_j$
Standard deviation	$\sigma = \sqrt{\sum_{j=i-k}^{i}(x_j - \mu)^2}$
Derivative	$x' = x_i - x_{i-1}, \ldots, x_{i-k+1} - x_{i-k}$
Product	$p = x \times y$

Table 1: Features formulas for diagnosing the refrigerator

smooth the original signal). The last feature is the product of two signals. Notice that after computing derivatives or products, it is necessary to again compute descriptive statistics (minimum, maximum, mean, and standard deviation) over a sliding window.

In the case of the thermostat only the present value of the thermostat at the time of diagnosis is used. This approach does not use all thermostat information. To use all information it is possible to compute a feature that shows when was the last thermostat transition. Somewhat surprisingly, this feature not only does not help with any of the classifiers we have tried, its inclusion significantly decreases the isolation accuracy.

How is the length k of the sliding window determined? A more advanced approach would be to use intelligent segmentation instead of a sliding window, however segmentation implies making a step toward computing a diagnosis which imposes a bootstrapping problem. In our study we take a range of sliding window lengths and we compute a large number of features. We also compute the product of all possible temperature pairs. This results in a large number of features: 144 for the 3-tanks running example and 10 681 for the refrigerator. Later it is possible to use feature selection algorithm to select relevant features and some classification algorithms (such as random forests) will do this themselves.

The simultaneous use of multiple sliding window sizes results in families of features. One such family is shown in figure 7. The sliding window there varies from 1 s to 60 s and is stepped every 5 s.

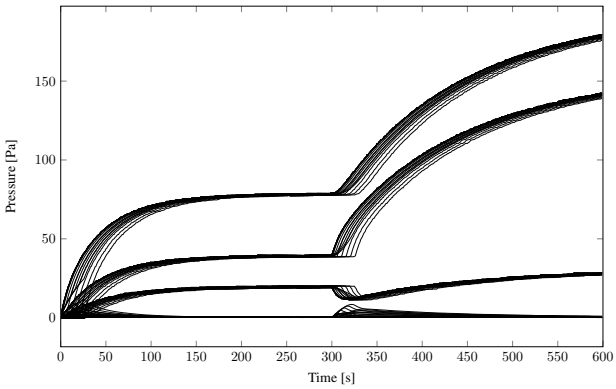

Figure 7: Descriptive statistics features for the three tanks running example

Figure 8 shows the features computed from one temperature sensor for a fixed sliding window size of 30 min.

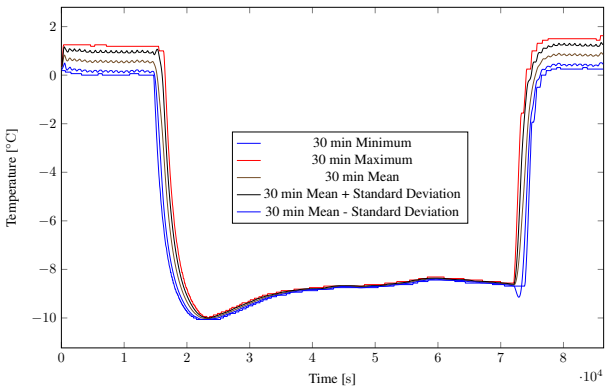

Figure 8: Subset of the rolling window features of one of the refrigerator's temperature sensors

Machine learning often normalizes the feature ranges by multiplying them with suitable constants. Experiments showed that this does not help the accuracy of classification in the case of the refrigerator and we use the features without scaling.

4.2 Diagnostic Algorithms

Machine learning has been popular in the recent decades and has given rise to a multitude of classification and regression methods. From decision trees to convolutional neural networks, the idea is to find parameters of a formula that will fit but not overfit the training data and predict the future.

Decision Trees

Decision trees are one of the easiest machine learning algorithms. Unfortunately they are prone to errors and tend to grow to be too large. Figure 9 shows a decision tree for diagnosing the three tanks running example.

There are already too many nodes in figure 9 and that is after the maximal depth of the decision tree was limited to three so the figure can fit on the page. Another problem with decision trees is that they are learned differently and the variance for prediction is very large. The three tanks example was classified 100 times with a decision tree classifier which resulted in mean isolation accuracy of 0.9962 with a standard deviation of 1.27×10^{-3}.

Random Forests

The random forest classifier is an extension of decision trees. The idea is to construct multiple decision trees and to use averaging for better isolation accuracy.

Support Vector Machines

The Support Vector Machine (SVM) classifier constructs a hyperplane in a multidimensional space [Cortes and Vapnik, 1995]. An advantage of this method is that it is fully deterministic, unlike random forests and decision trees. A disadvantage of SVM is its large computational and memory complexity during the testing phase.

With the three tanks running example, SVM achieves almost perfect isolation: only two labels per fault-mode (a total of six) are misclassified as nominal. This is close to the theoretical optimum as in the beginning of the scenario the three tanks start empty and the all pressure sensors have the same values.

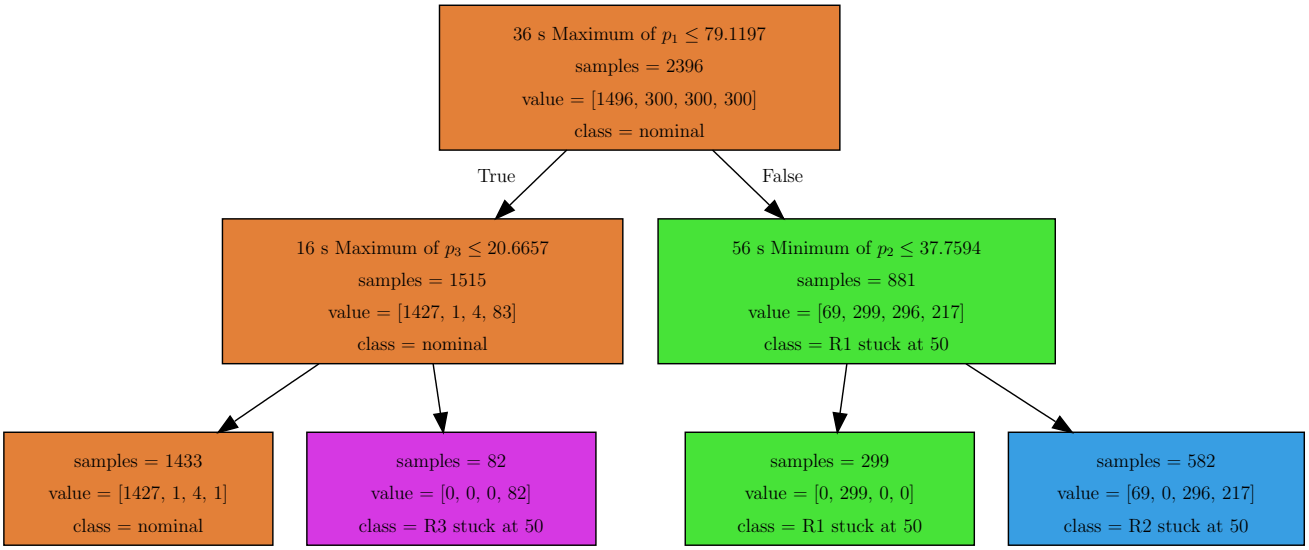

Figure 9: An example decision tree learned from the three tanks simulation data.

Neural Networks

In our quest for simple, yet efficient, diagnostic algorithms, we have considered perceptrons. These are very basic linear classifiers, one of the first methods of machine learning. Similar to SVM, the training is deterministic.

In the three tanks running example, the perceptron classifies badly 20 labels, which is 0.83 % of the total. This is slightly worse than the other methods and, as the running example is very simple, is an indication that there will be difficulties with the real-world and drastically more complex refrigeration system.

5 Experimental Results

Number of data-driven diagnostic experiments were performed on the refrigeration scenarios. Each scenario is exactly twenty-fourth hours. All experiments were performed on a 32-CPU (8 cores per CPU) Intel Xeon 3.3 GHz Linux computer with 1.5 TiB of RAM.

Half of the diagnostic scenarios (nine) were used to train the classifiers and the other nine to test the performance. Each training and testing has been repeated ten times as some of the methods use randomized learning algorithms and to measure more accurate CPU time.

Constructing a refrigeration diagnostic benchmark is challenging due to the need of non-destructive faults (at least in the beginning of our project). The isobutane refrigerant used in the system under consideration is highly flammable so experiments have to be designed with safety in mind. On the other hand electrical experiments (short-circuits and open-circuits) and experiments with a leak of thermal energy are easier to conduct. Table 2 provides a summary of the refrigeration benchmark.

The experimental results from diagnosing the refrigerator are shown in terms of false positives, false negatives, and isolation time. The units of these three metrics is time. A false positive of 1 hour for example, means that during the 24-hour scenario the diagnostic engine "thought" for one hour that there is a fault. Similar for the false negatives. The isolation time is how long did it take for the diagnostic engine to find the injection.

Fault	Type	Parameters
No fault (nominal)	N/A	None
Wrong thermostat setting	Logic	Thermostat position
Open-circuit thermostat	Electrical	None
Short-circuit thermostat	—"—	None
Open door	Thermal	Door angle

Table 2: Common-refrigerator diagnostic benchmark

Table 3 shows the refrigeration benchmark diagnostic metrics calculated by a decision tree. The results are surprisingly good for such a basic classifier. Three scenarios (nominal, bad thermostat 1, and bad thermostat 2) have almost perfect accuracy (small number of false positives and false negatives). The isolation time is also minimal which indicates that small sliding window of one minute for computing the features works satisfactory.

Thermostat open and short-circuits lead to small number of both false positives and false negatives. These false-positives can be filtered-out after the diagnosis process with an algorithm that is designed specifically for this purpose.

The problematic scenarios for decision trees are opening the door. Out of 16 hours one-fourth to one-half of the time the decision tree cannot detect the fault.

Decision tree is a very fast method with CPU time always less than two seconds. The decision tree footprint is also very small which makes this method ideal for embedding and real-time disconnected operation.

Random forests perform similar to decision trees which is of no surprise as they are collection of decision trees. Surprisingly, random forests are slightly worse compared to decision tree (see the scenario where the thermostat is set at position 3 by mistake).

While SVM is in general a good classification method it does not perform well with a large number of features which is visible in table 5. Another disadvantage of SVM is the huge computational cost. It processes twenty-four hours of test data in more than two minutes. This means that real-time performance is still achievable with the chosen Δ_ω rate

Fault	Classification errors [h:mm:ss]	False positives [h:mm:ss]	False negatives [h:mm:ss]	Isolation time [h:mm:ss]	CPU time [s]
Bad thermostat at 1	0:01:00	0:00:00	0:01:00	0:01:00	0.87
Bad thermostat at 2	0:01:00	0:00:00	0:01:00	0:01:00	1.13
Bad thermostat at 3	4:36:36	0:00:00	4:36:36	0:01:00	1.05
Nominal	0:10:18	0:10:18	0:00:00	N/A	1.06
Thermostat open circuit	0:24:29	0:10:19	0:14:10	0:08:23	1.09
Thermostat short circuit	0:48:48	0:12:20	0:36:28	0:06:58	1.03
Door opened 25%	4:37:12	0:18:48	4:18:24	0:00:54	0.99
Door opened 50%	6:23:01	0:05:48	6:17:13	0:00:49	1.04
Door opened 75%	8:55:51	0:01:18	8:54:33	0:00:45	1.05

Table 3: Decision trees performance metrics of the common-refrigerator diagnostic benchmark

Fault	Classification errors [h:mm:ss]	False positives [h:mm:ss]	False negatives [h:mm:ss]	Isolation time [h:mm:ss]	CPU time [s]
Bad thermostat at 1	0:01:00	0:00:00	0:01:00	0:01:00	0.87
Bad thermostat at 2	0:01:00	0:00:00	0:01:00	0:01:00	1.05
Bad thermostat at 3	11:13:48	0:00:00	11:13:48	0:09:18	1.14
Nominal	0:00:00	0:00:00	0:00:00	N/A	1.07
Thermostat open circuit	0:14:08	0:01:20	0:12:47	0:09:28	1.08
Thermostat short circuit	0:34:33	0:02:23	0:32:10	0:10:34	1.08
Door opened 25%	4:21:12	0:00:24	4:20:48	0:01:12	1.12
Door opened 50%	8:47:49	0:01:06	8:46:43	0:01:49	1.12
Door opened 75%	5:09:03	0:01:42	5:07:21	0:00:57	1.05

Table 4: Random forests performance metrics of the common-refrigerator diagnostic benchmark

of $1\,\mathrm{min}$.

While we are yet to find additional empirical evidence, SVMs perform better with less features while decision trees and neural networks are good in selecting optimal features for achieving low classification error rates.

Table 6 shows the diagnostic performance of the simplest neural network classifier–the perceptron. This simple linear classifier is performing surprisingly well. It recognizes almost immediately the three scenarios where the thermostat is set to a wrong position. The nominal mode, however, is not classified properly with nearly seven hours of false positives. The door opening is correctly classified in one of the three cases.

Similar to the decision trees and the random forests, the perceptron is computationally very efficient: it provides real-time performance and is capable of computing diagnosis every second second.

6 Related Work

Monitoring of refrigerators containing medical vaccines is obviously important. In 2011, McCollster and Vallbona have published their findings from monitoring for the first time, fifty-four refrigerators with vaccines [McColloster and Vallbona, 2011]. From those, $24\ \%$ have shown protracted periods of temperature below $0\,^{\circ}\mathrm{C}$. The authors find striking correlation ($r = 0.76$) between the prevalence of pertussis (whooping cough) in the regions of the study and below-zero temperatures in the refrigerators holding the pertussis vaccinations.

Many machine-learning methods can be used not only for classification but also for regression. Neural networks is one of those methods. In the case of regression one can estimate some continuous variable as opposed to a label from a set. Regression-based neural networks have been used for predicting the power consumption of a refrigerator [Ertunc and Hoşöz, 2006]. The authors show high correlation ($0.933 < r < 1$) and low relative error ($1.9\ \% - 4.18\ \%$). These results show that "inferior" models based on neural networks can have relatively high simulation accuracy of certain variables such as consumed power.

Refrigeration systems are related to Heat Ventilation and Air-Conditioning (HVAC) installations. In HVAC, in addition to the monitoring and diagnostic aspect, there is also energy-saving and psychometrics (the field of engineering studying thermodynamic and physical properties of gas-vapor mixtures). Due to the prevalence and importance of reliable and efficient HVAC there are multiple benchmarks and simulations [Wang *et al.*, 2013]. The main goal of this paper is to facilitate HVAC commissioning and to provide basis for energy saving.

7 Conclusions

We have showed the diagnostic performance of four data-driven diagnostic algorithms on a benchmark of refrigeration data. Although the raw classifiers perform relatively well, this translates to bad diagnostic accuracy and isolation time.

This article marks the beginning of several future activities. We are planning to extend the number of scenarios and to provide more failure cases: some based on the test-beds that we have built and some based on software fault-augmentation.

Fault	Classification errors [h:mm:ss]	False positives [h:mm:ss]	False negatives [h:mm:ss]	Isolation time [h:mm:ss]	CPU time [s]
Bad thermostat at 1	1:06:00	0:00:00	1:06:00	0:20:00	135.29
Bad thermostat at 2	11:23:00	0:00:00	11:23:00	0:15:00	133.08
Bad thermostat at 3	24:00:00	0:00:00	24:00:00	Never	133.7
Nominal	0:00:00	0:00:00	0:00:00	N/A	132.87
Thermostat open circuit	13:26:03	0:00:00	13:26:03	6:52:58	133.22
Thermostat short circuit	16:00:01	0:00:00	16:00:01	Never	134.52
Door opened 25%	19:59:54	0:00:00	19:59:54	Never	134.68
Door opened 50%	19:59:49	0:00:00	19:59:49	Never	135.57
Door opened 75%	19:59:45	0:00:00	19:59:45	Never	141.44

Table 5: SVM performance metrics of the common-refrigerator diagnostic benchmark

Fault	Classification errors [h:mm:ss]	False positives [h:mm:ss]	False negatives [h:mm:ss]	Isolation time [h:mm:ss]	CPU time [s]
Bad thermostat at 1	0:01:00	0:00:00	0:01:00	0:01:00	0.93
Bad thermostat at 2	0:08:00	0:00:00	0:08:00	0:01:00	1.16
Bad thermostat at 3	0:01:00	0:00:00	0:01:00	0:01:00	1.28
Nominal	6:49:00	6:49:00	0:00:00	N/A	1.26
Thermostat open circuit	0:32:53	0:06:55	0:25:58	Never	1.15
Thermostat short circuit	0:16:24	0:08:26	0:07:58	Never	1.21
Door opened 25%	3:58:54	0:00:00	3:58:54	0:01:54	1.2
Door opened 50%	0:00:00	0:00:00	0:00:00	0:02:49	1.24
Door opened 75%	3:58:45	0:00:00	3:58:45	0:01:45	1.08

Table 6: Perceptron performance metrics of the common-refrigerator diagnostic benchmark

In terms of diagnostic methods and algorithms we are interested in answering the question: "Is physics-based MBD more powerful than data-driven and, in particular, machine-learning-based diagnostics?" To do that we need electrical and thermodynamic models of the refrigerators. We will use the data from the benchmark to construct these models and to infer their parameters as consumer-grade refrigerators do not provide detailed modeling information.

We plan to experiment with "deep-learning" methods [Funahashi and Nakamura, 1993] such as recurrent neural networks. This approach develops statistical models that are capable of memorizing change and this is very helpful in the case of more subtle failures such as degradation.

References

[Cortes and Vapnik, 1995] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[Ertunc and Hoşöz, 2006] Huseyin M. Ertunc and Murat Hoşöz. Artificial neural network analysis of a refrigeration system with an evaporative condenser. *Applied Thermal Engineering*, 26(5):627–635, 2006.

[Feldman *et al.*, 2010] Alexander Feldman, Tolga Kurtoglu, Sriram Narasimhan, Scott Poll, David Garcia, Johan de Kleer, Lukas Kuhn, and Arjan van Gemund. Empirical evaluation of diagnostic algorithm performance using a generic framework. *International Journal of Prognostics and Health Management*, pages 1–28, 2010.

[Funahashi and Nakamura, 1993] Ken-ichi Funahashi and Yuichi Nakamura. Approximation of dynamical systems by continuous time recurrent neural networks. *Neural Networks*, 6(6):801–806, 1993.

[McColloster and Vallbona, 2011] Patrick McColloster and Carlos Vallbona. Graphic-output temperature data loggers for monitoring vaccine refrigeration: Implications for pertussis. *American Journal of Public Health*, 101(1):46–47, 2011.

[Wang *et al.*, 2013] Liping Wang, Steve Greenberg, John Fiegel, Alma Rubalcava, Shankar Earni, Xiufeng Pang, Rongxin Yin, Spencer Woodworth, and Jorge Hernandez-Maldonado. Monitoring-based HVAC commissioning of an existing office building for energy efficiency. *Applied Energy*, 102:1382–1390, 2013.