



HAL
open science

Aircraft Fleet Health Monitoring with Anomaly Detection Techniques

Luis Basora, Paloma Bry, Xavier Olive, Floris Freeman

► **To cite this version:**

Luis Basora, Paloma Bry, Xavier Olive, Floris Freeman. Aircraft Fleet Health Monitoring with Anomaly Detection Techniques. Aerospace, 2021, 8 (4), pp.103. 10.3390/aerospace8040103. hal-03196104

HAL Id: hal-03196104

<https://hal.science/hal-03196104>

Submitted on 12 Apr 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Article

Aircraft Fleet Health Monitoring with Anomaly Detection Techniques

Luis Basora ^{1,†,‡} , Paloma Bry ^{1,†,‡}, Xavier Olive ^{1,†,‡}  and Floris Freeman²

¹ ONERA DTIS, Université de Toulouse, Toulouse, France; luis.basora@onera.fr (L.B.); paloma.bry@protonmail.com (P.B.); xavier.olive@onera.fr (X.O.)

² KLM Royal Dutch Airlines, Postbus 7700, 1117 ZL Schiphol, The Netherlands, Floris.Freeman@klm.com (F.F.)

* Correspondence: luis.basora@onera.fr

† Current address: 2 avenue Édouard Belin, 31055 Toulouse CEDEX 4, France.

‡ These authors contributed equally to this work.

Version March 29, 2021 submitted to Aerospace

Abstract: Predictive maintenance has received considerable attention in the aviation industry where costs, system availability and reliability are major concerns. In spite of recent advances, effective health monitoring and prognostics for the scheduling of condition-based maintenance operations is still very challenging. The increasing availability of maintenance and operational data along with recent progress made in machine learning has boosted the development of data-driven prognostics and health management (PHM) models. In this paper, we describe the data workflow in place at an airline for the maintenance of an aircraft system and highlight the difficulties related to a proper labelling of the health status of such systems, resulting in a poor suitability of supervised learning techniques. We focus on investigating the feasibility and the potential of semi-supervised anomaly detection methods for the health monitoring of a real aircraft system. Proposed methods are evaluated on large volumes of real sensor data from a cooling unit system on a modern wide body aircraft from a major European airline. For the sake of confidentiality, data has been anonymized and only few technical and operational details about the system had been made available. We trained several deep neural network autoencoder architectures on nominal data and used the anomaly scores to calculate a health indicator. Results suggest that high anomaly scores are correlated with identified failures in the maintenance logs. Also, some situations see an increase in the anomaly score for several flights prior to the system's failure, which paves a natural way for early fault identification.

Keywords: aviation; predictive maintenance; prognostics and health management; condition monitoring; anomaly detection; deep learning; neural networks; autoencoders; time series

1. Introduction

Prognostics and health management (PHM) has drawn growing interest from industrial and academic research in the last few years, especially in sectors like aviation where profit margins are small and operational costs are critical. Aircraft availability and system reliability can be improved with effective health monitoring and prognostics. The purpose is the anticipation of system failures before their actual occurrences, as these cause major repair cost and operational disruptions. Condition monitoring is also key for optimising the scheduling of maintenance operations based on the estimated condition or remaining useful life (RUL) of the systems.

The increasing availability of large volumes of sensor data generated daily by aircraft in flight calls for technologies to make the best of recent progresses made in the field of machine learning (ML) and anomaly detection [1]. This data can be exploited by algorithms in order to extract patterns

31 or anomalies to be linked with the degradation of the system. In PHM, data-driven approaches are
32 particularly relevant for complex systems for which appropriate physics-based models may not exist.

33 In this paper, we investigate the use of anomaly detection to generate system health indicators
34 (HI) for a fleet of a modern wide-body aircraft. We focus our analysis on a specific aircraft system,
35 but claim that the presented methodology is generic enough to be adapted to other use cases. More
36 precisely, our contribution includes:

- 37 1. a presentation of the different sources of data available at a major airline for the maintenance
38 of an aircraft cooling system unit. We explain the difficulties to exploit such data for health
39 monitoring primarily due to the inherent uncertainties in the maintenance data;
- 40 2. the introduction of a semi-supervised anomaly detection approach for condition monitoring
41 based on autoencoders to extract meaningful information from a complex dataset with high
42 uncertainty in the labelling. For the sake of confidentiality, data has been anonymized and prior
43 knowledge about the system under study is very limited;
- 44 3. the experimentation of the proposed approach with three types of autoencoders to compute the
45 anomaly scores: a fully-connected autoencoder, a convolutional autoencoder, and a variant of a
46 long short-term memory (LSTM) autoencoder available in the literature.
- 47 4. a method to determine a health indicator from the anomaly scores by computing a threshold
48 based on the F_{β} -score metric.
- 49 5. the evaluation of the autoencoders with a set of binary classification metrics by using instance
50 weights to account for the uncertainty on the provided failure data.

51 We explore the feasibility and potential of an approach based on standard anomaly detection
52 techniques and autoencoders for condition monitoring when applied to a real and complex dataset.

53 The results of the study show that a relatively high anomaly score usually corresponds with
54 the periods where a failure has been expected by maintenance engineers. In some cases, we also
55 observe an upward trend on the anomaly scores a few flights before the failure was actually detected
56 by the maintenance operators. However, the task remains challenging because of the lack of prior
57 system knowledge (confidentiality constraints), sensor data complexity, the lack of guarantee that
58 available signal data are appropriate to detect a given failure and general uncertainties in the sensor
59 and provided failure data.

60 The paper is organised as follows. Section 2 reviews the classical and more recent anomaly
61 detection methods used in aviation, as well as the data-driven approaches specific to prognostics and
62 their application to health monitoring. Section 3 describes the system under study and the different
63 data sources available as well as the adopted approach to constitute a properly labelled dataset
64 with failure periods. Section 4 overviews the anomaly detection approach to address data labelling
65 uncertainty and compute the health indicator. Section 5 presents the design of the autoencoders used
66 for anomaly detection and the alternative options tested. Section 6 describes the details of the training
67 and testing of the autoencoders. Section 7 provides an analysis of the results and a discussion on
68 the suitability and issues related to the use of such models for health monitoring. Lastly, Section 8
69 highlights the main points of the study, the implications for health monitoring and suggests some
70 potential ideas for future work.

71 2. Literature review and background

72 2.1. Anomaly detection methods in aviation

73 Autoencoder architectures can be considered as part of the recent advances in the field of neural
74 networks and deep learning, which explains why their application to anomaly detection in aviation
75 and PHM are still relatively limited compared to other classical approaches.

76 In this sub-section, we briefly review progresses made in anomaly detection applied to aviation
77 in general, before focusing in the next subsections on data-driven methods adapted to the PHM field.

78 For more details, the reader may refer to popular surveys in the literature on anomaly detection
79 methods [2–4] including the recent advances [5] and their application to aviation [1].

80 Firstly, it is important to note that aviation data is challenging for several reasons: its large volume,
81 high-dimensionality, heterogeneity (mixed categorical and continuous attributes), multi-modality
82 (multiple modes of nominal and non-nominal operations with different types of aircraft, airports and
83 airspaces) and temporality (long time-series). In addition, the challenge is expected to be even bigger
84 in the future with the continuously growing number of sensor-equipped aviation systems.

85 Among the classical approaches for anomaly detection in aviation, Multiple Kernel Anomaly
86 Detection (MKAD) [6] developed by NASA is still one of the state-of-the-art methods for the detection
87 of anomalies in flight data. Based on kernel functions and One-Class Support Vector Machine
88 (OC-SVM) [7], its computational complexity is quadratic with respect to the number of training
89 examples. More recently, Puranik et al. [8] propose a framework based on OC-SVM to identify
90 anomalies with the aim of improving the safety of general aviation operations.

91 Anomaly detection based on clustering methods (ClusterAD) are also widely applied. For instance,
92 Li et al [9,10] uses ClusterAD methods based on DBSCAN [11] to detect anomalies in the take-off
93 and approach operations with datasets containing from a few hundreds to a few thousands flights.
94 OC-SVM and ClusterAD methods have both performance issues with large datasets. ClusterAD
95 methods have the advantage though that they can identify multiple types of flight operation patterns
96 (different nominal operations) corresponding to the identified clusters.

97 As far as we know, only a few classical techniques for anomaly detection are scalable to large
98 datasets. For instance, Oehling et al. [12] approach based on Local Outlier Probability (LoOP) [13] was
99 applied to an airline dataset of 1.2 million flights in order to detect anomalies related to safety events.

100 Classical approaches such as clustering or distance-based methods like k-Nearest Neighbours
101 (kNN) suffer from the curse of dimensionality issue when applied to high-dimensional data. The
102 solution is often the application of a dimensionality reduction technique like principal component
103 analysis (PCA) [14] as a preprocessing step. On the other hand, distance-based methods are
104 computationally expensive even during the prediction phase, which makes them inappropriate for
105 certain applications.

106 With large-scale high-complex data as the one generally available in aviation, deep-learning
107 approaches are supposed to perform better than traditional machine learning methods [5]. This is
108 because deep-learning algorithms are specifically designed to learn complex patterns in big data. A
109 diversity of neural network models are used in deep-learning, such as the traditional Fully Connected
110 Network (FCN), Convolutional Neural Networks (CNN) or Recurrent Neural Networks (RNN).
111 Deep-learning architectures are based on a sequence of layers combining one or several types of such
112 neural network models. Their goal is to progressively find a representation of the original data features
113 which is more appropriate for tasks like classification or regression.

114 In the particular case of autoencoders, they are trained to find a lower-dimensional hidden
115 representation (latent space) from which original data can be reconstructed, so they are often used as a
116 technique for non-linear dimensionality reduction. In addition, they are suitable for anomaly detection
117 based on the assumption that anomalies are incompressible and cannot be properly reconstructed
118 from the lower dimensional representation of the latent variables. In the case of time-series data,
119 autoencoders with RNN layers should be particularly adapted to exploit the temporal dependencies
120 related to anomalies. In practice, however, they present significant limitations when applied to long
121 sequences.

122 2.2. *Data-driven approaches for prognostics*

123 PHM has leveraged the increasing availability of sensor data to monitor the health of complex
124 systems. The advantage of data-driven approaches is that they generally do not require any knowledge
125 about the failure mechanisms inherent to these systems.

126 Depending on the availability of time-to-failure or end-of-life data, data-driven methods for
127 prognostics can have two types of expected outputs [15]: direct failure prediction in the form of
128 remaining useful life (RUL) estimation, or indirect failure prediction through the calculation of health
129 degradation indicators. The first type of output requires availability of enough historical failure cases
130 to train a RUL estimator. In the aerospace domain, failures are very scarce which make RUL prediction
131 challenging. This second type of results need fewer failure cases, but do require the definition of a
132 threshold to decide on the presence of a failure. Anomaly detection for prognostics falls within this
133 second category.

134 Data-driven approaches for PHM can be divided in two major categories [16–18]: statistical
135 approaches (e.g. hidden Markov models, Bayesian networks, Gaussian mixture models) and neural
136 networks approaches (e.g. autoencoders), the latter being more frequently used over the past few
137 years.

138 As for the statistical approaches, in [19] a mixture of Gaussian Hidden Markov Models is presented
139 for RUL prediction of bearings from a NASA benchmark database. Also, we can cite the work by
140 Zhao et al. [20] to detect early anomalies in an electric generator’s multivariate sensor data, based on
141 Pearson correlation between the sensors and vector quantization clustering.

142 Concerning the application of neural networks to PHM, as sensor data come naturally in the
143 form of time series, RNN have been widely used. For instance, in [21] and [22] a RNN is trained for
144 temporal feature extraction and RUL estimation. In both methods, models are trained in a supervised
145 way on run-to-failure time series data, and test on time series data for systems that have not reached
146 failure yet. More recently, Husebø et al. [23] propose a CNN autoencoder for feature extraction to
147 diagnose electrical faults in induction motors.

148 In PHM, neural networks have also been specifically used for anomaly detection. We can mention
149 hybrid neural network approaches such in [24], where a CNN is combined with a gated recurrent unit
150 (GRU) RNN in order to extract spatio-temporal features from sensor data and detect anomalies in
151 rotating machinery. Hundman et al. [25] propose an approach based on LSTM for anomaly detection
152 in spacecraft systems in real time: the comparison between the values predicted by the LSTM and
153 the actual data gives a reconstruction error, used as a health indicator. Non-parametric dynamic
154 thresholding is then used to determine whether there is a failure or not and alert operations engineers.

155 *2.3. Applications to aircraft systems health monitoring*

156 The increasing availability of condition monitoring data for aircraft fleets has made data-driven
157 PHM valuable for proactive maintenance of aviation systems. Thus, in [26] a non-parametric modelling
158 technique is described to calculate the health indicator for an aircraft air conditioning system. In [27], a
159 study is made on several feature selection techniques in combination with neural networks for RUL
160 prediction applied to a gas turbine engine dataset.

161 More recently, Schwartz et al. [28] propose a method using self-organizing maps (SOMs) and
162 kernel density estimation for fault detection and identification in aircraft jet engines. Baptista et al. [29]
163 study the use of hybrid neural networks combining RNN layers with multi-layer perceptron (MLP)
164 layers to estimate the RUL. The hybrid neural network is fed with statistical features computed on
165 time series of two real-world aircraft engine datasets.

166 Concerning the more specific use of autoencoder-based anomaly detection with flight data,
167 we can mention a few recent efforts. For instance, an approach based on fully-connected deep
168 autoencoders [30] and another one based on convolutional denoising autoencoders [31] have been
169 applied for fault detection, the first on the NASA DASHlink open database [32], and the second on a
170 dataset of customer notification reports sent over aircraft communications addressing and reporting
171 system (ACARS) to airlines for engine fault detection.

172 However, as far as we know, there is no comparative study in the literature on the use of several
173 kinds of autoencoders for the computation of a health indicator of an aircraft system, which is the
174 purpose of our research.

175 3. System and data description

176 3.1. System under study

177 The aircraft system under study is a cooling unit system belonging to a modern and widely-used
178 airliner. Each aircraft has four units installed. A pump package pumps cooling fluid sequentially
179 through each of the 4 cooling-units. Each cooling unit consists of a compressor, condenser, flash-tank
180 and evaporator. Depending on the required cooling capacity, one or more cooling units may be
181 operational. By changing the priorities of the cooling unit during each flight, the aircraft attempts to
182 spread usage equally over each of the 4 units. For most failure cases, failure starts with clogging of the
183 cooling-units filters, which can ultimately lead to failure of the compressor. When failure on any of the
184 cooling units internals is expected, the entire unit is replaced and sent to the maintenance-shop for
185 repair. Early detection of a fault in the filter could avoid a costly repair of the cooling-unit's compressor.
186 These cooling units are not safety critical, the aircraft is allowed to depart even with all cooling-units
187 inoperative. However, this would impose an operational limitation to the airline.

188 3.2. Flight and sensor data

189 The 32 GB dataset provided by the airline contains anonymous sensor and contextual data for
190 18 294 flights. Contextual flight data is a mix of categorical and continuous variables such as timestamp,
191 flight identification number, tail number, flight phase, altitude, computed airspeed (CAS), airport
192 departure and arrival. Sensitive information such as the names of the airports or the flights and tail
193 numbers have been encrypted, and timestamps do not correspond with the real departure times as
194 they have been shifted.

195 Sensor data is provided in the form of variable-length (per flight) time series of nine continuous
196 variables sampled at 1 Hz for each of the four cooling units installed. The physical nature of the
197 signals is unknown as the names of the parameters are encrypted. After a discussion with the airline,
198 it was concluded that the four units could be assumed to work independently, so the number of actual
199 samples are four times the number of flights.

200 Figure 1 shows the nine types of anonymised sensor signals for a healthy cooling unit system
201 during a flight. Each of the nine subplots displays four time series representing the sensor values of
202 the four cooling units installed in the aircraft.

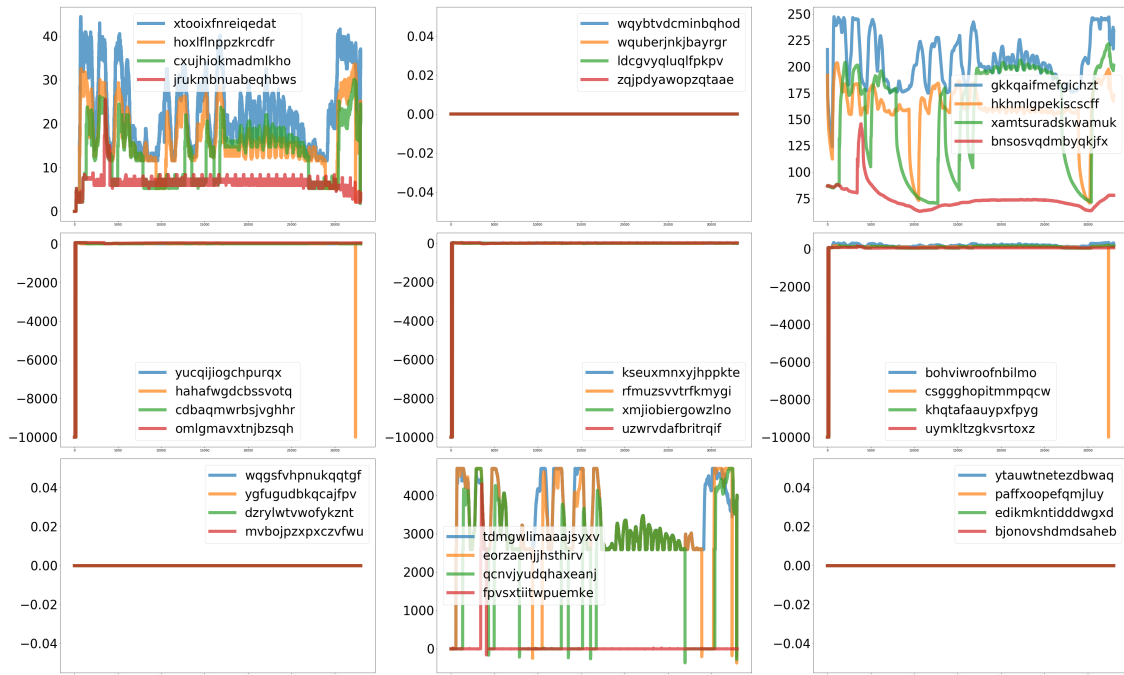


Figure 1. Anonymous sensor data during a flight for the four healthy cooling units installed in an aircraft.

3.3. Maintenance data

For the cooling unit, most replacements are triggered by aircraft maintenance messages (MMS). These messages are raised when a suspected deviation from nominal operation is detected by the aircraft. In the ideal case, a failure isolation procedure follows, which identifies the corrective action immediately, resulting in replacement of the cooling unit and disappearance of the MMS. In most cases however, trouble-shooting is less trivial, as these maintenance message are 'noisy' on its own. For example, faults in various components may result in the same maintenance message, making fault isolation challenging. Furthermore, a simple re-set of a system can make the messages disappear, only to re-occur months later. In practise, the diagnostics of whether a cooling unit is faulty and need replacement comes down to the engineering judgement of the responsible specialist. When the replaced cooling-unit is inspected at the repair-shop the existence of the fault can be confirmed, but no start-date of the fault can be inferred from this inspection. This makes labelling of the data a challenge on its own. To address this problem, we make use of multiple data sources for labelling of the data:

- Removal data: list of removal dates for all replaced cooling units. For non-safety critical components such as the cooling unit, immediate corrective action after expected failure is not mandatory. Hence, removals dates do not correspond to actual failure dates.
- Aircraft technical logbook: Contains potential issues reported by the crew. This textual data is ordered by date, but the fault description is not well structured which makes it difficult to link an issue directly to cooling-unit failure.
- Central maintenance computing function (CMCF) data: system generated messages including flight deck effects (FDE) and MMS.
- Shop reports: List the condition of the internals of the cooling unit upon inspection at the component repair shop.

For labelling the data, the following approach is taken. Firstly, the dates for all unscheduled removals of the cooling unit are collected. Only removals that have resulted in disappearance of the MMS are considered. Subsequently, only those removals that were confirmed in the shop-report were considered. For these failures, the start-date of each of these failures was selected to be the first day

230 prior to the removal, after which maintenance message appeared consistently over multiple flights.
 231 Technical specialists and predictive maintenance engineers from the airline were asked to confirm
 232 these failure dates (PM). As uncertainty about the actual failure data persisted, each failure in the PM
 233 data was tagged by the airline as either TRUE, LIKELY or DUBIOUS.

234 The final labelling data consists of expected failures of the cooling-unit, including: the aircraft
 235 tail number, the position of the failed cooling system unit, dates on which the failure was detected,
 236 dates on which the failure was fixed by the removal, the uncertainty tag (TRUE, LIKELY or DUBIOUS) and
 237 sometimes associated comments. The tag and comments can provide valuable information concerning
 238 the uncertainty on the actual occurrence of fault.

239 The airline provided the MMS as well, which might help to understand some anomalies identified
 240 by the models outside the PM failure data intervals (potential false positives). MMS and PM uncertainty
 241 tags are especially exploited when computing the performance metrics on the models (see 7.2).

242 The distribution of faults per aircraft is shown in Table 1.

Table 1. PM and MMS fault distribution per aircraft tail number.

	MMS	PM	Total
apbsayjk	9	0	9
cntxlxyh	20	1	21
cwuumlxe	55	1	56
dlkzncgy	97	4	101
ekzlmbdx	146	3	149
enwslczm	236	3	239
ibauqnxj	223	3	226
iefywfmy	103	5	108
iilvttok	105	3	108
lbhkyjhi	196	3	199
rgwwyqtt	155	3	158
tjyjdtaf	93	4	97
trmblwny	276	3	279
vazmznfq	42	0	42
whjlcda	44	0	44
wnjxbqsk	79	3	82
zcbiftrr	42	0	42
Total	1921	39	1960

243 4. Anomaly detection approach

244 4.1. Semi-supervised learning

245 Models are trained on what we assume to be normal or healthy data, i.e. flights for which a failure
 246 has been identified are removed from the training set. Thus, our setting is the most extended variant
 247 of semi-supervised anomaly detection known as learning from positive (i.e. normal) and unlabeled
 248 examples (LPUE) [33,34].

249 More precisely, we removed from the training set the flights falling within the PM failure periods.
 250 However, we do not exclude the flights for which only a MMS but no confirmation of a fault exists. In
 251 addition, we removed from the training set the flights within the 20 days preceding the detection of a
 252 PM failure, where the uncertainty on the real health status of the cooling system is higher. In spite of
 253 that, the uncertainty on the labelling makes it impossible to ensure that all flights in the training set are
 254 actually healthy.

255 .

256 4.2. Anomaly detection models

257 Anomaly detection is based on a variety of neural networks known as autoencoders. The
258 advantage of the autoencoders is their ability to reduce dimensionality, by automatically extracting the
259 most significant features in a lower dimensional latent space.

260 Three variants of autoencoders are tested: fully-connected, convolutional and RNN autoencoders.
261 RNN-based autoencoders are well adapted to find time-dependent patterns in the signals, which is
262 not the case of the fully-connected and convolutional autoencoders. In addition, RNN autoencoders
263 can accept variable-length sequences as input, whereas the other two types of autoencoders require
264 the use sliding windows to breakdown the sequences into fixed-sized vectors.

265 Autoencoders output an anomaly score which is the reconstruction loss, i.e. the difference
266 between the predicted and the actual value according to some norm (e.g. MSE, L1). Anomaly scores
267 are assumed to increase with anomalous data "not seen" by the model, since correlations in sensor data
268 should change because of some incipient degradation or fault. In addition, we expect the anomaly
269 scores to present an upward trend a few flights before the system runs into failure in order to prevent
270 its breakdown.

271 The architecture and characteristics of the three variants of autoencoders are further detailed in
272 Section 5.

273 4.3. Anomaly threshold

274 For each flight and for each cooling unit, a health indicator is produced, which is an anomaly
275 score along with a binary health status ("healthy" or "faulty"). A flight anomaly score is the mean of
276 the anomaly scores calculated at sensor level. For the models requiring a sliding window, we take the
277 average of the scores computed over the sliding windows of a flight.

278 In order to determine the health status, we need to compute a threshold. Then, we can predict
279 the label ("faulty" or "healthy") of a flight depending on whether the anomaly score of the system is
280 higher ("faulty") or lower ("healthy") than the threshold. The predicted label can then be compared
281 with the real label in order to assess the performance of our model. However, the ground truth as to
282 whether a flight is faulty or not depends on the fault information, which is uncertain for the reasons
283 we mentioned before.

284 Obviously, there is a trade-off when computing the threshold. If it is set too high, some faults will
285 be missed. If it is too low, the rate of false alarms (false positives) will become unacceptable. Selecting
286 the optimal threshold is case-specific, as it depends on the underlying failure rate of the cooling unit,
287 the performance of the anomaly detector, and the cost of (corrective and preventive) maintenance.
288 After a discussion with the airline, it seems preferable to set a relative high threshold because of several
289 reasons. Firstly, faults are rare in aviation and so they are for this cooling unit. Hence, the distribution
290 between healthy and faulty legs is biased to healthy data. For instance, if we say only 0.1% of the
291 flights are faulty, a model with a 100% of true positives and 1% of false positives will end up triggering
292 110 alerts every 10000 flights even though only 10 alerts correspond with a real fault. Secondly, too
293 many false alarms would outweigh the benefit of cheaper preventive repairs, and may require more
294 units to be held in stock. Finally, a low rate of false positives is necessary if we want our prognostic
295 alerts to be trusted and accepted by the maintenance operators. This requires that at least between 65%
296 or 70% of the total alarms are true (precision).

297 The methodology to determine the threshold is described in Section 7.2.

298 5. Autoencoder models

299 We have implemented three different autoencoders solutions which are described in the following
300 three subsections. Although we did perform some testing with other variants in order to select the
301 presented architectures, the objective was not to find the most optimized one in terms of layers,
302 units per layer, activation functions and so on. The goal was to come up with three reasonable

303 models for comparison, each one representing a different approach to autoencoders: fully-connected,
 304 convolutional-based and LSTM-based autoencoders.

305 5.1. Fully-connected autoencoder (FCAE)

306 The FCAE used in this paper is a multi-layered neural network with an input layer, multiple
 307 hidden layers and an output layer (see Figure 2). Each layer can have a different number of neural
 308 units and each unit in a layer is connected with every other one in the next layer. For this reason we
 309 call it here fully-connected, although in the literature it is usually refer to as autoencoder (AE), or to
 310 emphasise the multi-layer architecture aspect, as deep autoencoder (DAE) or stacked autoencoder
 311 (SAE). In all cases, an autoencoder performs two functions in a row: encoding and decoding.

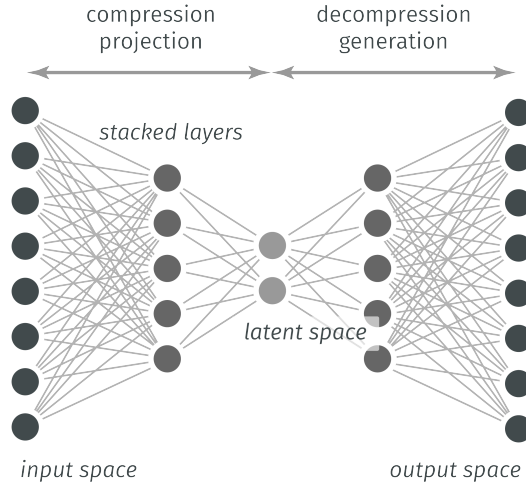


Figure 2. FCAE with multiple stacked layers

312 The encoding function maps the input data $s \in \mathbb{R}^d$ to a hidden representation $y \in \mathbb{R}^h = e(s) =$
 313 $g(w \cdot s + b)$ where $w \in \mathbb{R}^{d \times h}$ and $b \in \mathbb{R}^d$ are respectively the weight matrix and the bias vector and
 314 $g(\cdot)$ is a non linear activation function such as the sigmoid or ReLU functions. The decoding function
 315 maps the hidden representation back to the original input space according to $\hat{s} = d(y) = g(w' \cdot y + b')$,
 316 $g(\cdot)$ being most of the time the same activation function.

The objective of the autoencoder model is to minimise the error of the reconstructed result:

$$(w, b, w', b') = \operatorname{argmin} \ell(s, d(e(s))) \quad (1)$$

where $\ell(u, v)$ is a loss function determined according to the input range, typically the mean squared error (MSE) loss:

$$\ell(u, v) = \frac{1}{n} \sum ||u_i - v_i||^2 \quad (2)$$

317 where u is the vector of observation, v the reconstructed vector and n the number of available samples
 318 indexed by variable i .

319 The proposed FCAE architecture is described in Table 2. We use three blocks of layers both in the
 320 encoder and decoder, which have a symmetric structure in terms of number of neural units per layer.
 321 The output shape refers to the shape of the tensors after the layers have been applied, where bs , wl and
 322 nf refers respectively to the the batch size, (sliding) window length and number of features. The layer
 323 dimensions are defined as $dim1 = wl * nf$, $dim2 = dim1/3$ and $dim3 = dim2/2$.

Table 2. Fully-Connected Autoencoder (FCAE) architecture

Block	Layer	Output shape
Input	-	$(bs, dim1)$
Enc1	Linear($dim1, dim2$) ReLU	$(bs, dim2)$
Enc2	Linear($dim2, dim3$) ReLU	$(bs, dim3)$
Enc3	Linear($dim3, nf$)	(bs, nf)
Encoding	-	(bs, nf)
Dec1	Linear($nf, dim3$) ReLU	$(bs, dim3)$
Dec2	Linear($dim3, dim2$) ReLU	$(bs, dim2)$
Dec3	Linear($dim2, dim1$) Sigmoid	$(bs, dim1)$
Output	-	$(bs, dim1)$

324 The encoder Enc1 and Enc2 are blocks made up of pairs of Linear and ReLU activation layers. The
325 input encodings (bottleneck with dimension nf) are the result of Enc3. As a mirror of the encoder,
326 the decoder has also two blocks (Dec1 and Dec2) both combining Linear and ReLU layers. The last
327 decoding block (Dec3) is made up of a Linear layer with a Sigmoid activation as FCAE outputs are
328 expected to be in the range of 0 to 1 (features are min-max scaled). A ReLU rather than a Sigmoid
329 activation is used in all the other layers as it presents theoretical advantages such as sparsity and a
330 minor likelihood of vanishing gradient resulting in faster learning.

331 5.2. Convolutional autoencoder (CAE)

332 Table 3 shows the CAE architecture proposed in this paper. It is inspired from the one in [23],
333 which has been applied with good results for anomaly detection in time series data. The proposed
334 CAE is a variant of the FCAE where fully-connected and convolutional layers are mixed in the encoder
335 and decoder. Like the FCAE, the decoder structure is kind of a mirrored version of the encoder.

Table 3. Convolutional autoencoder (CAE) architecture

Block	Layer	Output shape
Input	-	(bs, nf, 30)
Conv1	Conv1D(30, 32, 5, 2, 1)	(bs, 32, 14)
	BatchNorm	
	ReLU	
Conv2	Conv1D(32, 64, 5, 2, 1)	(bs, 64, 6)
	BatchNorm	
	ReLU	
Conv3	Conv1D(64, 128, 3, 2, 2)	(bs, 128, 4)
	BatchNorm	
	ReLU	
Flatten	Flatten	(bs, 512)
Dense E1	Linear(512, 64)	(bs, 64)
	ReLU	
Dense E2	Linear(64, 32)	(bs, 32)
	ReLU	
Encoding	Linear(32, 9)	(bs, 9)
Dense D1	Linear(9, 32)	(bs, 32)
	ReLU	
Dense D2	Linear(32, 64)	(bs, 64)
	ReLU	
Reshape	Linear(64, 512)	(bs, 512)
	Reshape(bs, 128, 4)	(bs, 128, 4)
ConvTransp1	Conv1dTransp(128, 64, 3, 2, 2, 1)	(bs, 64, 6)
	BatchNorm	
	ReLU	
	Conv1dTransp(64, 32, 5, 2, 1, 1)	(bs, 32, 14)
ConvTransp2	BatchNorm	
	ReLU	
ConvTransp3	Conv1dTransp(32, nf, 5, 2, 1, 1)	(bs, nf, 30)
	Sigmoid	
	-	(bs, nf, 30)

336 Table 3 shows also the shape of the input and output tensors in each layer, where bs and nf
337 refers to batch size and number of features respectively. The input is three-dimensional, where the
338 last dimension is the (sliding) window size (30) and the second dimension is the number of channels
339 which in our case are the features (nf). Unlike a FCAE, a change in the size of the input window may
340 involve substantial parameter adaptation, because of the symmetric structure of the autoencoder and
341 the nature of convolutional layers.

342 The first three layers of the encoder (Conv1 to Conv3) and the last three ones of the decoder
343 (ConvTransp1 to ConvTransp3) are convolutional, whereas the innermost layers are fully-connected.
344 Conv1d(ic , oc , k , s , p) (and its transpose Conv1DTransp(ic , oc , k , s , p , op)) denotes a
345 convolutional (deconvolutional) layer with ic and oc channels, kernel size k , stride s and padding p
346 and output padding op . Please note that dimensionality reduction is achieved by using a $s > 1$ rather
347 than by using pooling layers [35].

348 All convolutional layers are followed by BatchNorm layers [36] to normalise the inputs to the
349 activation layers. ReLU is used as the activation layer everywhere except for the last one in the decoder
350 (ConvTransp3) where a Sigmoid has been used instead for the same reasons stated with the FCAE.

351 We tested other alternative options for the activation functions, such as using SELU everywhere
 352 (with appropriate LeCun normal initialisation of weights), or Sigmoid in the dense layers and ReLU in
 353 the convolutional ones as in [23]. However, the results were worse and these options were abandoned.

354 5.3. LSTM autoencoder (LSTMAE)

355 The blocks of the two autoencoders previously introduced are made up of either dense or
 356 convolutional layers, which makes necessary the use of sliding windows to provide them with
 357 fixed-sized input vectors. In this subsection, we introduce the LSTM autoencoders (LSTMAE), where
 358 layers consist of Long Short-Term Memory (LSTM) neural networks allowing for input sequences to
 359 be of variable length.

360 A LSTM is a type of RNN widely used for sequential data processing, since it can learn temporal
 361 dependencies over longer sequences than a simple RNN. For our particular purpose, a LSTMAE
 362 presents a theoretical advantage: samples can be entire flights instead of sliding windows, and
 363 anomaly scores can be calculated for the flight as a whole rather than by aggregating window anomaly
 364 scores.

365 In the literature, there have been a few efforts to implement a LSTMAE. For instance, Malhotra et
 366 al. [37] introduced the model in Figure 3 for anomaly detection in time series data.

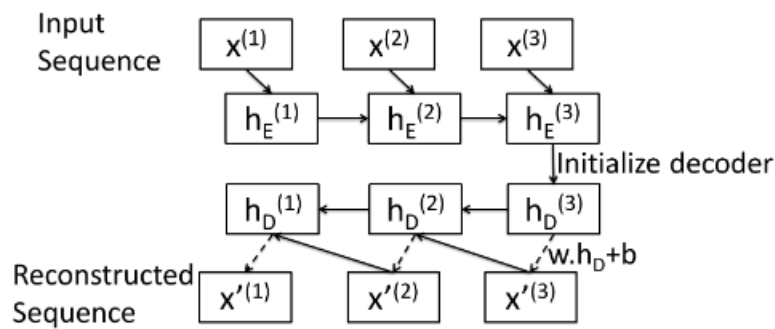


Figure 3. LSTM autoencoder by Malhotra et al.[37]

367 In this architecture, the encoder and the decoder are both LSTM networks. The last hidden state
 368 of the encoder is passed as the initial hidden state to the decoder. Then, at every instant t , the decoder
 369 takes as input the reconstruction of the previous instant $x'^{(t-1)}$ obtained by linear transformation of
 370 the hidden state h_{t-1} computed by the decoder cell.

371 Srivastava et al. [38] introduced the LSTMAE architecture shown in Figure 4, which combines
 372 input reconstruction with future prediction in order to achieve a better hidden representation.
 373 Reconstruction is operated in reversed order as the last hidden state of the input sequence contains
 374 better information about the short-term correlations which should improve the reconstruction of the
 375 beginning of the sequence in reversed order.

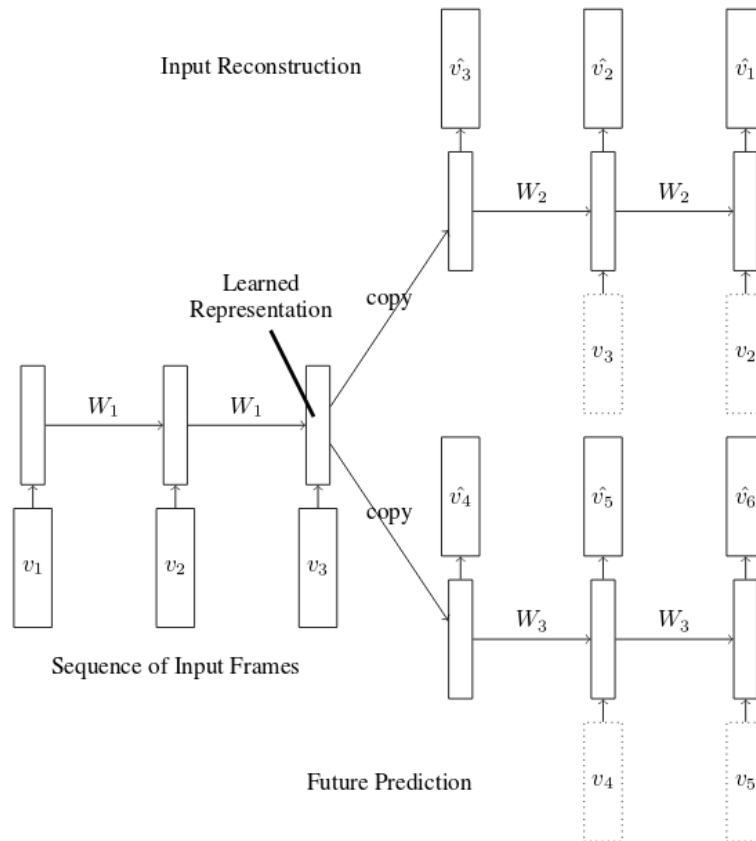


Figure 4. LSTM autoencoder by Srivastava et al.[38]

376 The authors reported good results in both papers. However, Malhotra's LSTMMAE is mostly
 377 applied to short univariate time series of around 30 points, or several hundred points in the case of
 378 periodic series. As for Srivastava's, good results are also reported with short video sequences (although
 379 of very high dimensions) by using high-dimensional hidden state vectors of size 2048 or even 4096.

380 We tested both LSTMMAE architectures with our data and reconstructions were in general of
 381 poor quality, even with large latent spaces and after dimensionality reduction with severe signal
 382 downsampling and PCA. The intuition is these methods do not scale well when applied to our time
 383 series, which can reach several thousand points in long-haul flights. This could be explained by the
 384 fact that the hidden representation $h^{(T)}$ has not enough capacity to allow for a good reconstruction of a
 385 long and multivariate input sequence.

386 Pereira et al. [39] mentioned this issue with long sequences and propose an attention-based
 387 mechanism to help the decoder. The resulting architecture is however highly complex, so we decided
 388 instead to work on a variant of the LSTMMAE architectures by Malhotra and Srivastava. The goal is to
 389 test whether such an architecture could be good enough for our data when combined with appropriate
 390 dimensionality reduction.

391 The proposed LSTMMAE architecture is depicted in Figure 5. Two encoders and two decoders are
 392 introduced to improve the reconstruction of long sequences, in particular the beginning and the end of
 393 the input sequence, where signal variation is very significant.

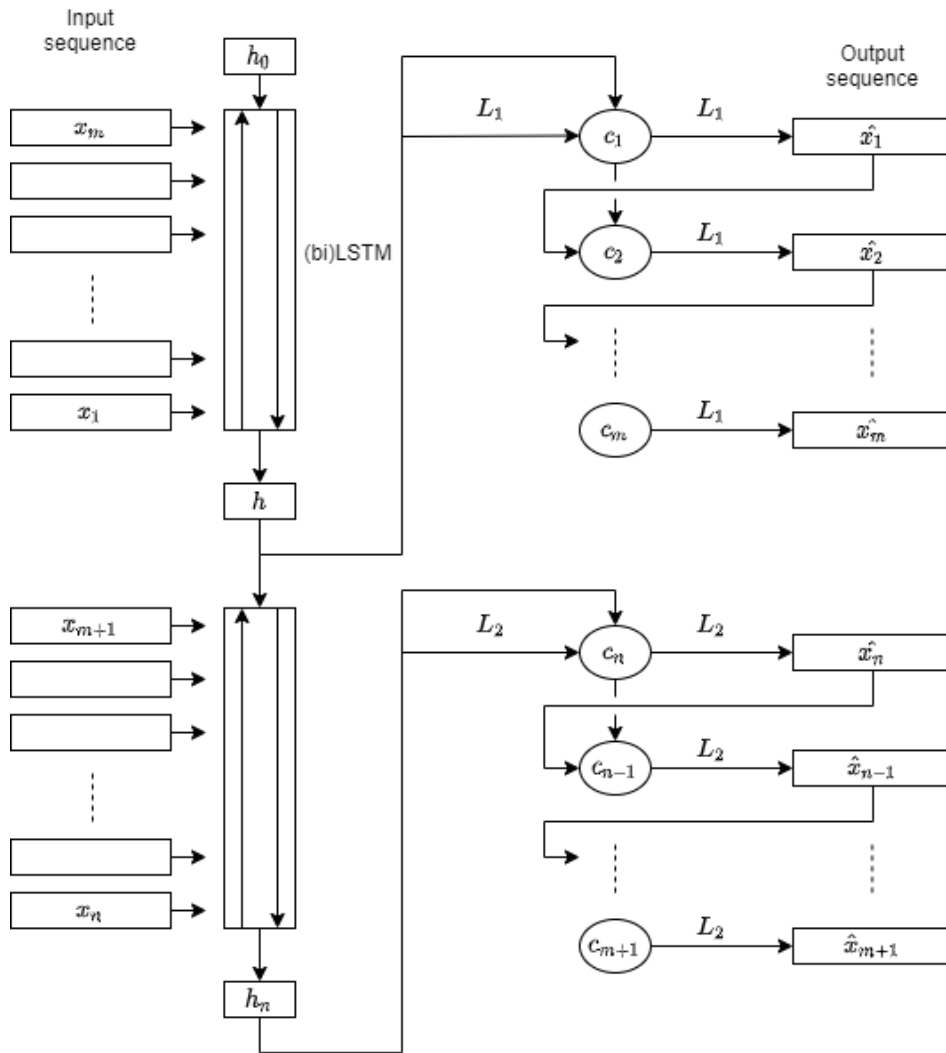


Figure 5. LSTM autoencoder proposal.

394 The input sequence of length n is split in half, where each x_i ($i = 1..n$) is a vector of size nf
 395 (the number of features). In Figure 5, each encoder is represented by a rectangular block of multiple
 396 LSTM cells, where h and h_n are the last hidden state vectors output respectively by the top and bottom
 397 encoder. As for the decoders, they are both represented on the right side. The two sequences of small
 398 circles are the LSTM cells of the decoders. Each c_i outputs a hidden state vector computed from the
 399 reconstruction and the hidden state of the previous decoder step. The dimensions of all hidden states
 400 of the two encoders and decoders have been set to 80.

401 The top LSTM encoder receives the first half of the input in reversed order and computes latent
 402 representation h which is fed to the top decoder. The bottom encoder receives h and the second half of
 403 the input sequence in the original order and outputs the hidden representation h_n . The top decoder
 404 takes h and linear transformation $L1(h)$ to feed cell state c_1 . The rest of the cell states c_i ($i = 2..m$)
 405 hidden states are computed from previous c_{i-1} hidden state and reconstruction \hat{x}_{i-1} . The first half of
 406 the output sequence is obtained by linear transformation $L1$ of the cell hidden states.

407 The inputs of the bottom LSTM encoder block are h and the second half of the input sequence
 408 in the original order, and its output the hidden state h_n . The bottom decoder decodes the sequence
 409 in reversed order, and takes as inputs h_n and linear transformation $L2(h)$ to output c_n hidden state.
 410 The rest of the cell hidden states for c_i ($i = n - 1..m + 1$) are computed from c_{i+1} hidden state and \hat{x}_{i+1} .
 411 The second half of the output sequence is obtained by linear transformation $L2$ of the cell hidden states
 412 and needs to be reversed.

413 6. Model training and testing

414 6.1. Dataset preparation

415 After studying the distribution of flight durations, we excluded from the dataset the flights with
 416 a duration of less than two hours (952 flights). These flights correspond to shorter legs in case the
 417 aircraft flies multiple legs in a single flight-segment. The goal was to reduce variability on operational
 418 conditions and focus the study on medium and long-haul flights, i.e. the main operating mode for the
 419 fleet under study. 139 flights with NaN values in sensor data were also removed. The NaN were present
 420 in the readings of sensors 2, 7 and 9 for the whole duration of the flights, since these sensors were not
 421 operational yet for these flights according to the airline.

422 Also, we noticed in some cases the presence of -9999 values recorded for practically the whole
 423 duration of a flight. These outlier values appeared simultaneously in several sensors and often for
 424 the four system units. This occurred independently of the real health status of the concerned cooling
 425 units due to a failure in a higher level system upstream of the cooling unit. In total, 207 flights were
 426 identified and removed from the dataset.

427 In order to reduce the dimensionality of sensor data, we generated several datasets by
 428 downsampling the original 1 Hz sensor data to lower rates such as 1/10 Hz, 1/30 Hz and 1/60 Hz. The
 429 last two sampling rates were tested only with the LSTMAE as a way to mitigate the LSTM limitations
 430 with long sequences. Also, like Malhotra et al. [37], we apply an additional linear dimensionality
 431 reduction technique (PCA), as an optional pre-processing step when using LSTMAE.

432 Figure 6 shows the variance ratio explained by the three first components, which reaches 99% no
 433 matter the sampling rate used to generate the dataset.

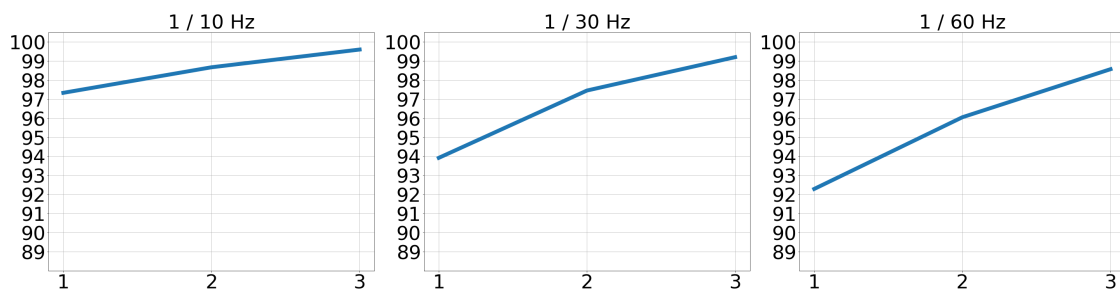


Figure 6. PCA - Variance ratio explained vs number of features.

434 Sliding windows were also generated for the FCAE and CAE models which need fixed-length
 435 sequences as inputs. After a few preliminary tests were done with different windows lengths and
 436 steps, we finally chose to apply 30-point-long and 20-step sliding windows (10 overlapping points).

437 Following the cleaning and re-sampling, we create a test set (te_{hf}) by selecting a subset of six tails
 438 with a significant number of failures, which accounts for about 20% of data. The 80% of data left is split
 439 into what we assume to be healthy and faulty data. Around 80% of healthy data is used as training set
 440 (tr_h), and the 20% left as validation set (v_h) for early-stopping. A second validation set (v_f) is created
 441 with the remaining faulty data, which is combined with v_h to set the anomaly threshold (see 7.2).

442 Finally, we calculate the max and min values for each feature in the training set tr_h in order to
 443 perform feature max-min scaling in all of the datasets.

444 In Table 4 we show the sample distribution per dataset type as well as in terms of health status
 445 (healthy vs faulty). Please note that there are four samples for each flight in the dataset, i.e. one for
 446 each system unit. When using sliding windows, the number of samples is much higher (see Table 5)
 447 and sample distribution can be slightly different as NaN filtering is performed at window rather than at
 448 flight level.

Table 4. Cooling unit dataset.

	faulty	healthy	Total	faulty	healthy	Total
test	2542	20850	23392	3.7%	30.1%	33.8%
train	0	31632	31632	0.0%	45.7%	45.7%
val	1652	12596	14248	2.4%	18.2%	20.6%
Total	4194	65078	69272	6.1%	93.9%	100.0%

Table 5. Cooling unit dataset with 30-20 sliding windows.

	faulty	healthy	Total	faulty	healthy	Total
test	420672	3578908	3999580	3.6%	30.7%	34.3%
train	0	5259420	5259420	0.0%	45.1%	45.1%
val	274398	2123922	2398320	2.4%	18.2%	20.6%
Total	695070	10962250	11657320	6.0%	94.0%	100.0%

449 6.2. Model training

450 Several runs were planned to test some combinations of models, training datasets and
 451 hyper-parameters. In particular, several experiments are defined to test the LSTMAE with different
 452 sampling rates as well as with and without PCA.

453 The batch size is set to 200 samples for LSTMAE and 20 000 samples for the FCAE and CAE. This
 454 is to take into account the fact that LSTMAE takes as input long time series corresponding with full
 455 flights (it can be several thousand points), whereas the other two models take windows of reduced
 456 size (30 points).

457 LSTM can be hard to train [40], especially with long input sequences of more of a thousand points
 458 for a long-haul flight with 1/10 Hz sampling rate. To avoid the issue of unstable training linked
 459 to exploding gradients, we apply gradient norm clipping set to 1.0 during the training of LSTMAE
 460 models. In addition, we use the truncated back-propagation through time (TBTT) technique [41]: input
 461 sequences are split into chunks of 200 time-steps for both the forward and backward-pass.

462 To avoid overfitting, the number of training epochs is controlled via an early-stopping mechanism
 463 and the validation v_h dataset. Early-stopping is setup to monitor the validation loss and halts the
 464 training process after 10 consecutive training epochs with no improvement of the validation loss.

465 The model parameters are optimised with Adam algorithm [42] to minimise the MSE loss with a
 466 learning rate set to 1×10^{-3} .

467 Table 6 lists the subset of the runs for further analysis in Section 7. Other runs were performed to
 468 test a few architectural options (see Section 5), but resulted in overall worse performance in terms of
 469 the metrics defined in Section 7.2. Hence, we dropped them to limit the study to the best candidate
 470 model found per autoencoder category. In the case of LSTMAE models, we included several runs to
 471 test the impact of several pre-processing options (the use of PCA and sampling rate).

Table 6. Runs performed with the number of epochs.

run	epochs
0 lstmae_pca_10	101
1 lstmae_10	101
2 lstmae_30	101
3 lstmae_60	8
4 lstmae_pca_30	29
5 lstmae_pca_60	54
6 cae_30_20_10	23
7 fcae_30_20_10	117

472 The name of each run indicates: 1) the model used (*fcae*, *cae*, *lstmae*), 2) whether PCA features
 473 (*pca*) are used instead of the original ones, 3) the length and step used for the sliding windows (30_20)
 474 for FCAE and AE models, 4) the resampling rate (e.g. 10 means 1/10 Hz). The epochs shown are the
 475 effective number of epochs as controlled by the early stopping mechanism with the maximum number
 476 of epochs set to 120.

477 6.3. Model testing

478 During the testing phase, trained models are used to output the anomaly scores for all the samples.
 479 Then, we set an anomaly threshold based on the scores in the validation dataset, which allows for the
 480 calculation of the performance metrics on the test set.

481 Figure 7 plots the anomaly score distribution per dataset type as calculated with a CAE model.
 482 You can see as the anomaly scores for the datasets with faulty data are distributed differently from the
 483 ones with normal data: faulty data present a higher density for the highest anomaly scores.

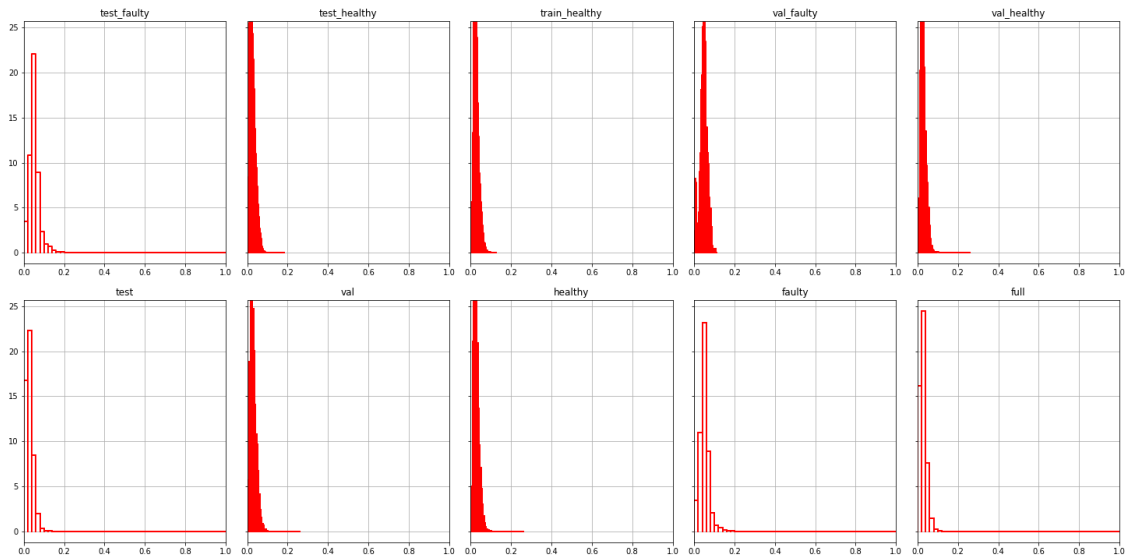


Figure 7. Anomaly score distribution with a LSTMAE obtained with a CAE model.

484 In order to calculate the metrics commonly used to evaluate the performance of binary classifiers,
 485 we need first to set a threshold τ . A flight is faulty if $\tau > a_i$ and healthy otherwise, where a_i is the
 486 anomaly score of a system unit.

τ is calculated by maximizing F_β -score:

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$$

487 according to the parameter β and the precision and recall values computed for a set of possible
 488 thresholds over the merge of the two validation sets v_h and v_f .

489 We can observe the trade-off between precision and recall in Figure 8, which plots the
 490 precision-recall curve for a run with a LSTMAE. For the reasons explained in Section 4.3,
 491 maximising the precision is the priority which means to choose a $\beta < 1$. For the sake of comparison,
 492 we will set $\beta = 0.05$ to compute the threshold as well as the related performance metrics.

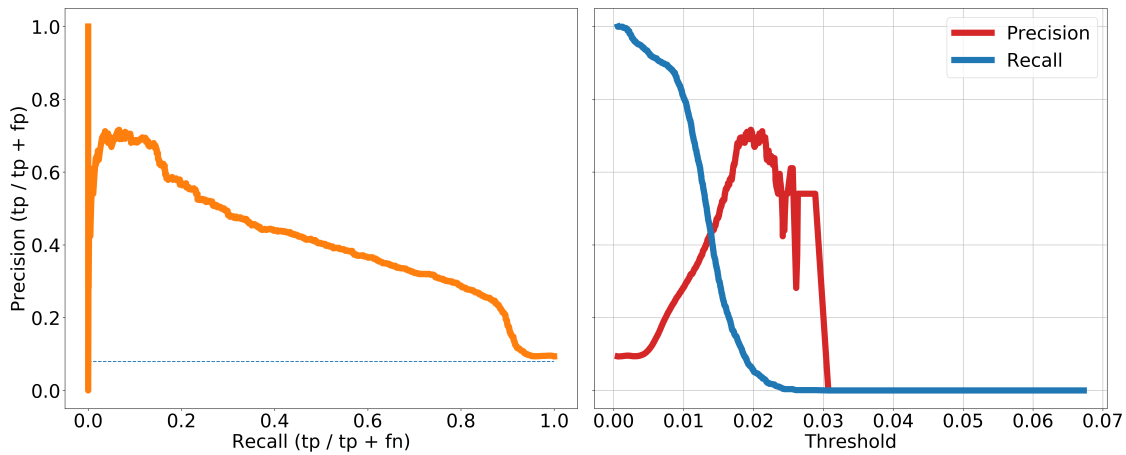


Figure 8. Precision-Recall curve for threshold setting.

493 Once the threshold is set, models can be evaluated with the test set by using classical performance
 494 metrics such as precision, recall and F_β -score. In addition to the classification performance metrics,
 495 we calculate the rate of flights predicted as faulty five days before the fault was actually identified
 496 in the maintenance checks ($pbfr$). This metric is to measure the predictive ability of the models to
 497 anticipate a fault, which highly depends on whether a "degradation signature" is actually present or
 498 not in the signals.

499 To take fault data uncertainty into account when calculating the performance metrics, we give a
 500 weight between 0 and 1 to each sample based on the level of confidence on its real label. Please note
 501 that by sample here we mean a system unit per flight (not a sliding window).

502 Table 7 describes more precisely the rules for sample allocation of health status label and
 503 uncertainty weights. We allocate zero weight to samples for the 20 days prior to the identification of a
 504 fault or for which we have an associated MMS, since their real health status is in fact unknown. On the
 505 other hand, we set a high weight for the flights where the fault has been labelled as TRUE or LIKELY
 506 by the airline. For the vast majority of cases (Rest category in Table 7), we assume they are mostly
 507 healthy with a weight of 0.85.

Table 7. Rules on sample allocation of health status label and weight.

Condition	Label	Weight
TRUE fault	FAULTY	1
LIKELY fault	FAULTY	0.7
DUBIOUS fault	FAULTY	0.2
20 flights before fault	HEALTHY	0
Associated MMS	HEALTHY	0
Rest	HEALTHY	0.85

508 7. Results and discussions

509 7.1. Training and validation losses

510 Before analysing the performance metrics in the next section, it is worth looking at what happened
 511 during the learning process in terms of the evolution of the reconstruction losses. This should help
 512 understand the models ability to learn to reconstruct what we consider normal data. By analysing
 513 the training and validation curves, we can also check whether there are cases of models overfitting or
 514 underfitting the data.

515 The x-axis in the loss plots represents the step in the training and validation set (i.e. an iteration
 516 with a forward and backward pass of a batch of samples) and y-axis is the MSE loss. The left subplots
 517 display the evolution of the loss for the training set (tr_h) and the right ones for the validation one (v_h).

518 Figure 9 shows the training and validation curves for the FCAE and CAE models. We can observe
 519 that convergence of both losses is reached pretty early by the two models. Loss decreasing is smooth
 520 in all cases except for a slight fluctuation of the validation loss of the CAE model. Based on these
 521 observations, there is no evidence of either model overfitting or underfitting.

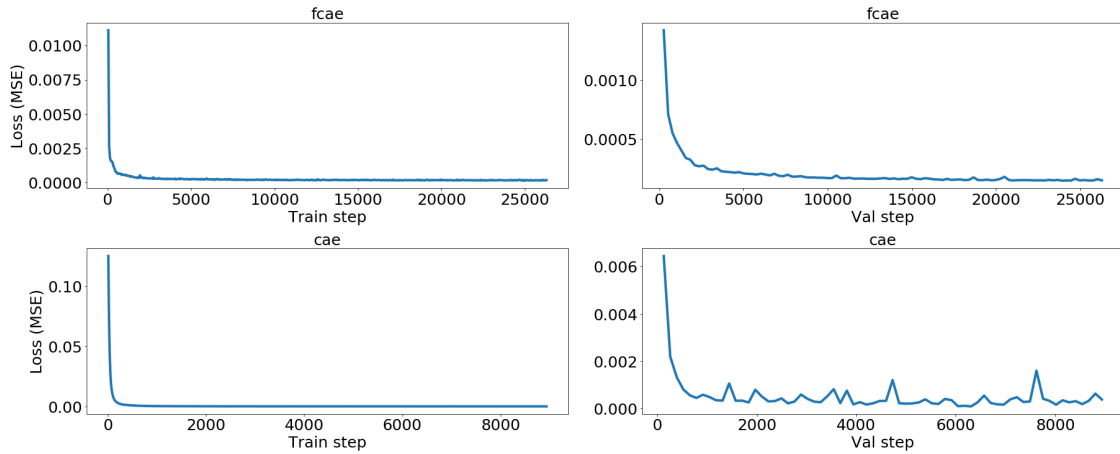


Figure 9. FCAE and CAE - Reconstruction loss (MSE) for the training (left) and validation (right) sets.

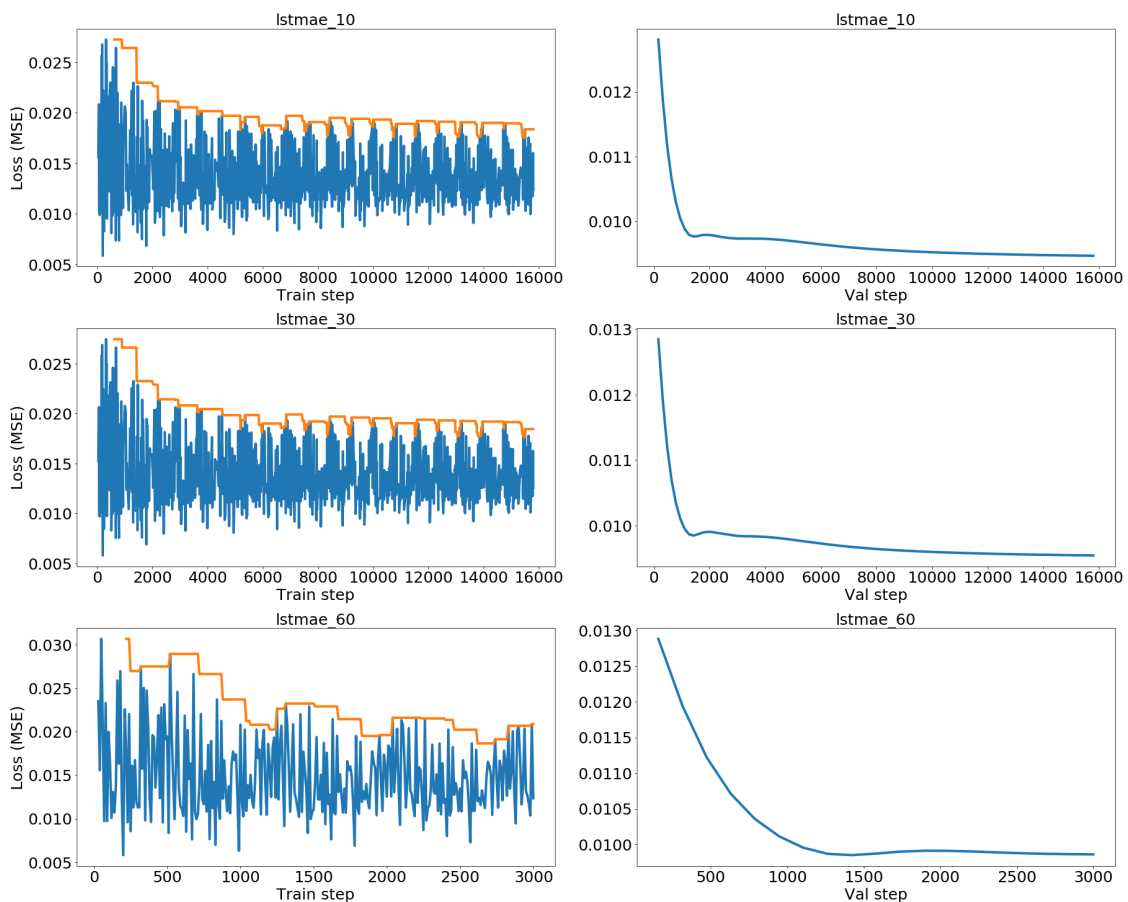


Figure 10. LSTMAE trained with the nine original features - Reconstruction loss (MSE) for the training (left) and validation (right) sets.

522 Figure 10 shows the training and validation curves for the LSTMAE models trained with the
 523 original nine features. The training loss is very noisy for the three runs, although a downwards trend
 524 (orange curve) can still be observed in the three cases. The periodic fluctuations in the training loss is a

525 sign of instability in the batch gradient descent caused by the variety of batches containing each one a
 526 different subset of flights. The high volatility is also symptomatic of the difficulty for our LSTM
 527 models to learn from long-sequences, in spite of our design with a double LSTM encoder/decoder
 528 and the application of training techniques such as gradient clipping to avoid gradient vanishing or
 529 exploding. A smoother downward trend can be observed in the three validation losses. In conclusion,
 530 there is no evidence of overfitting, but we cannot completely rule out a certain degree of model
 531 underfitting based on the shape of the training losses.

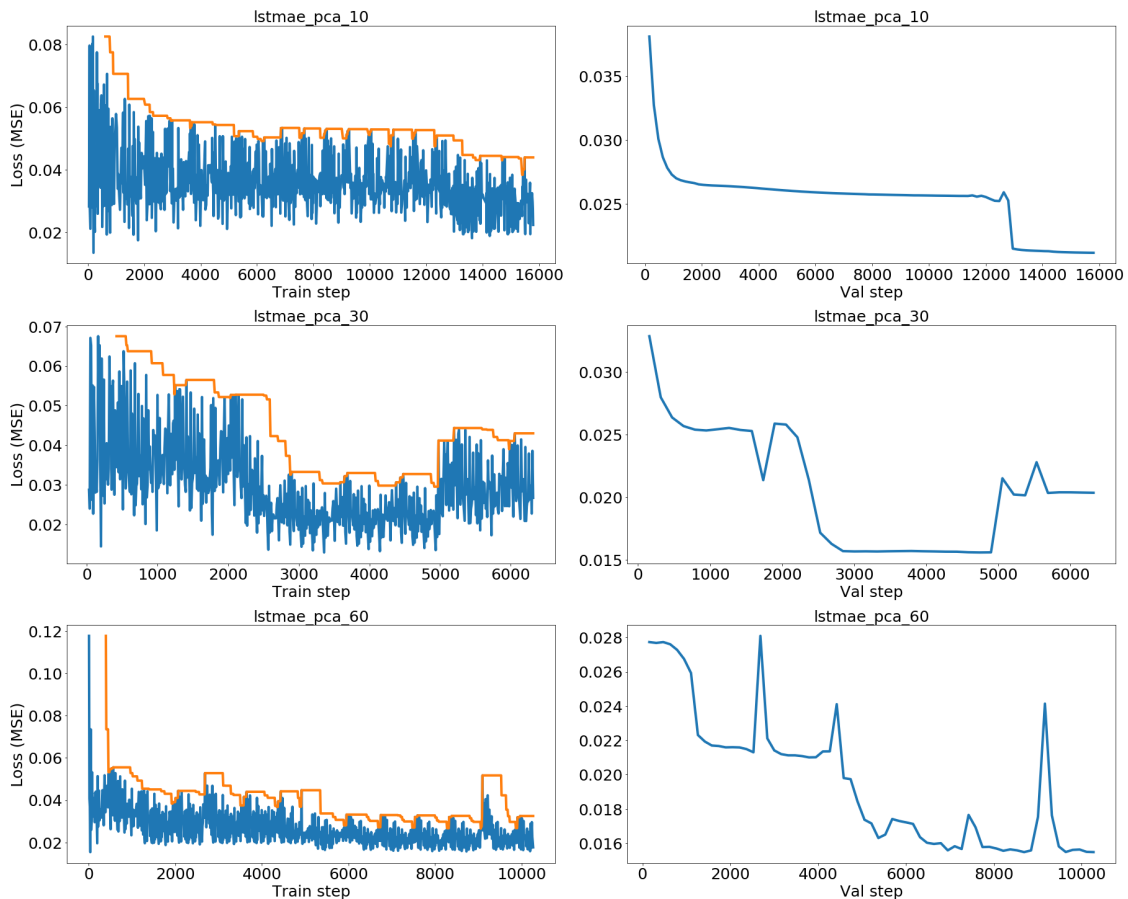


Figure 11. LSTMMAE trained with PCA - Reconstruction loss (MSE) for the training (left) and validation (right) sets.

532 The losses for the LSTMMAE trained with PCA are plotted in Figure 11. The training loss curve is
 533 again fluctuating but still headed downwards. However, in the last two plots there is a change of trend
 534 with an increase of the training and validation losses towards the end. The increases in the validation
 535 losses certainly triggered the early-stopping mechanism to halt the training process in order to avoid
 536 overfitting. The validation losses for the two last plots are also far less smooth when compared to the
 537 ones for the LSTMMAE models trained with the original features. If overfitting has been prevented by
 538 early-stopping even in the most evident case of `lstmae_pca_30`, we cannot rule out by looking at the
 539 training curve the possibility of a certain degree of underfitting.

540 7.2. Performance metrics

541 Table 8 lists the resulting performance metrics for the runs sorted out by the `auc_pr` (area under
 542 precision-recall curve) metric and F_β -score computed with $\beta = 0.05$.

Table 8. Performance metrics.

	run	auc_pr	precision	recall	fbeta	pbfr
0	lstmae_pca_10	0.471	0.805	0.126	0.794	0.124
1	lstmae_10	0.451	0.665	0.169	0.660	0.183
2	lstmae_30	0.450	0.661	0.169	0.656	0.185
3	lstmae_60	0.430	0.687	0.148	0.681	0.158
4	lstmae_pca_30	0.348	0.858	0.077	0.837	0.083
5	lstmae_pca_60	0.341	0.672	0.133	0.665	0.140
6	cae_30_20_10	0.231	0.934	0.033	0.875	0.054
7	fcae_30_20_10	0.218	0.971	0.031	0.902	0.049

543 In terms of F_β -score and precision only, CAE and FCAE models gets the top results. However,
 544 when considering other metrics like recall or pbfr, LSTMAE presents a more balanced overall
 545 performance in most of the runs, especially in lstmae_pca_10.

546 LSTMAE performance is highly sensitive to the combination of sampling rate and PCA. The best
 547 precision and F_β -score for LSTMAE models is practically always reached with PCA. On the other
 548 hand, the higher the sampling rate the better the auc_pr metric is when using PCA.

549 In general, but particularly with FCAE and CAE models, recall is low, even though it could
 550 be increased with a higher β by sacrificing part of the precision. Depending on the shape of the
 551 precision-recall curve, an increase of recall (even small) is not always possible without a significant
 552 drop of precision. In that sense, the relatively low auc_pr of FCAE and CAE models is not a good
 553 sign.

554 The evaluated autoencoders can correctly identify the health status of the flights in the test set
 555 with a level of precision higher than the goal of 65% discussed in Section 4.3. Although FCAE
 556 and CAE precision is the highest, both models tend to reconstruct some of the anomalies in the
 557 test set which results in a high rate of false negatives and low recall (further analysis provided
 558 in Section 7.4). The added complexity of a CAE compared to a FCAE does not translate into any
 559 performance improvement. LSTMAE models are more balanced in terms of precision and recall
 560 and overall superior when considering auc_pr and pbfr. Based on the auc_pr metric, which offers a
 561 good comparison between different models on a binary classification problems with an imbalanced
 562 dataset as in our case, we recommend using lstmae_pca_10 model.

563 As for the choice of sensors and features, apart from recommending the use of the PCA features
 564 and a sample rate of 1/10 Hz for the LSTMAE model, the analysis in Section 7.6 points out the
 565 relevance of sensors 2, 3 and 9 and the insignificance of the others as fault precursors. It would be
 566 interesting to evaluate the models again with only these three sensors. Concerning the parameters of
 567 the sliding windows for the FCAE and CAE, more tests are required to assess the sensitivity of the
 568 models to their length and step. It is however not straightforward in the case of the CAE to perform
 569 such tests as the design needs to be adapted each time we change the sliding window length.

570 7.3. Signal reconstruction

571 The main assumption when using autoencoders for anomaly detection is that after being trained
 572 with mostly normal data, they should be able to reconstruct healthy data better than faulty one.

573 FCAE and CAE models produce reconstructions of similar quality. Figure 12 illustrates this
 574 principle with an example of signal reconstruction for 4 sensors by a FCAE of both a healthy (left) and
 575 a faulty (right) cooling unit during two flights of aircraft tjjjdtaf. We can see that reconstruction is
 576 indeed better for the healthy than the faulty system, which results in a correct identification of their
 577 health status.

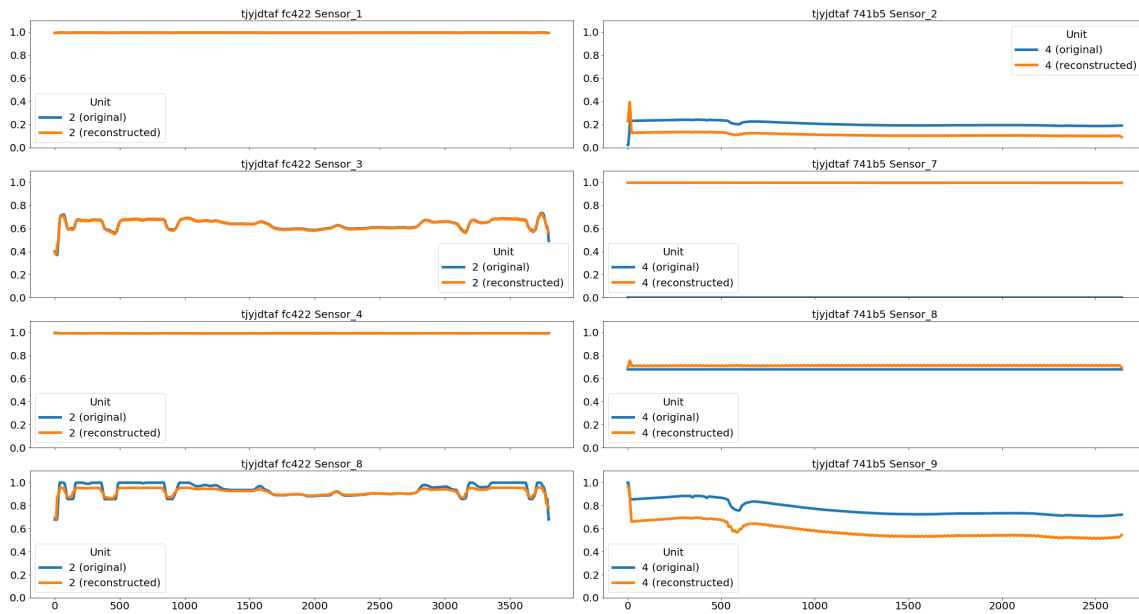


Figure 12. Signal reconstruction examples with a FCAE of two flights from tail tjyjdtaf. On the left, good reconstruction for a healthy cooling unit. On the right, bad reconstruction for a faulty cooling unit.

578 As for the LSTMAE (see Figure 13), in spite of reconstructions following the original signal trend,
 579 they are very smooth even with the dataset generated with a sample rate of 1/10 Hz. As high-frequency
 580 components of the signals are ignored, LSTMAE performance is worse than the one of the FCAE and
 581 CAE models in terms of F_{β} -score and precision (see Table 8). However, LSTMAE overall performance
 582 is more balanced in general as discussed in Section 7.2, and the HI produced correlates well with some
 583 of the failures (see Section 7.5).

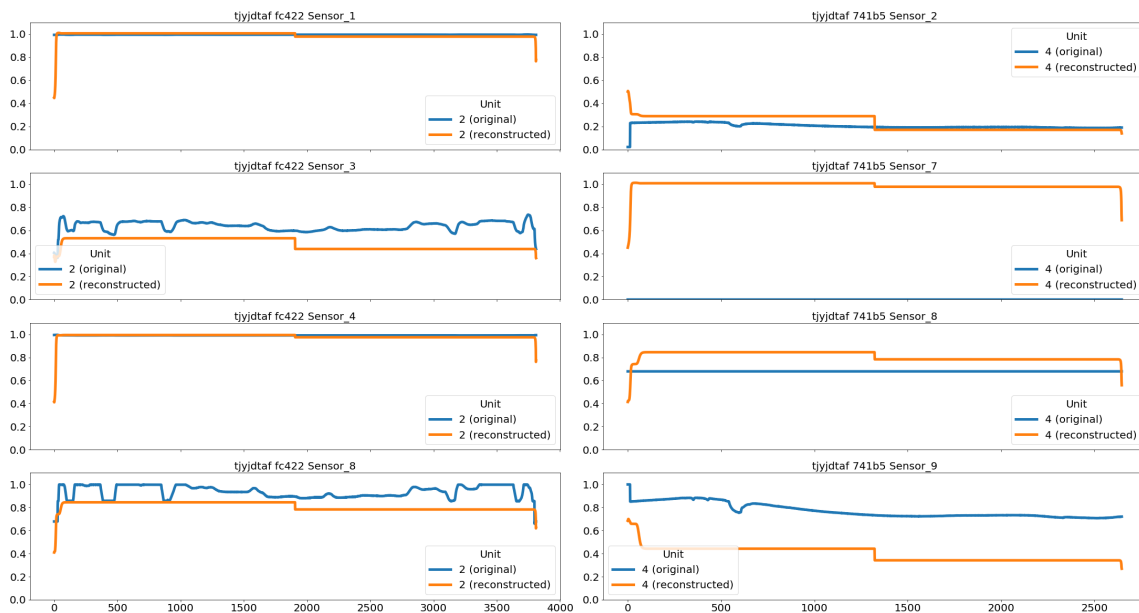


Figure 13. LSTMAE signal reconstruction examples (run lstmae_pca_10) of the same two flights and sensors from Figure 12.

584 7.4. False negative and false positive analysis

585 Unfortunately, we have cases where faulty signals are correctly reconstructed (false negatives)
 586 and healthy signals are not (false positives). Thus, it can be interesting to look at the loss signatures

587 to analyse which sensor contributes the most to the different cases. The spider plots represent the
 588 contribution to the reconstruction loss per sensor for the different cases and models.

589 Figure 14 shows the average loss per sensor as generated by a FCAE for each of the four cooling
 590 units. We plot on top the contribution of each sensor in the true positive cases, and in the bottom
 591 those of the false negatives. A first observation is that not all faulty cooling units (true positives)
 592 present exactly the same loss signature, so there are some differences in the way they fail. A second
 593 observation is that in the false negative cases, the model is able to reconstruct the signals better than it
 594 should have done, including sensor 9 (units 1, 3, 4) as well as sensors 5 and 6 (unit 2), which are the
 595 main contributors in the case of the true positives.

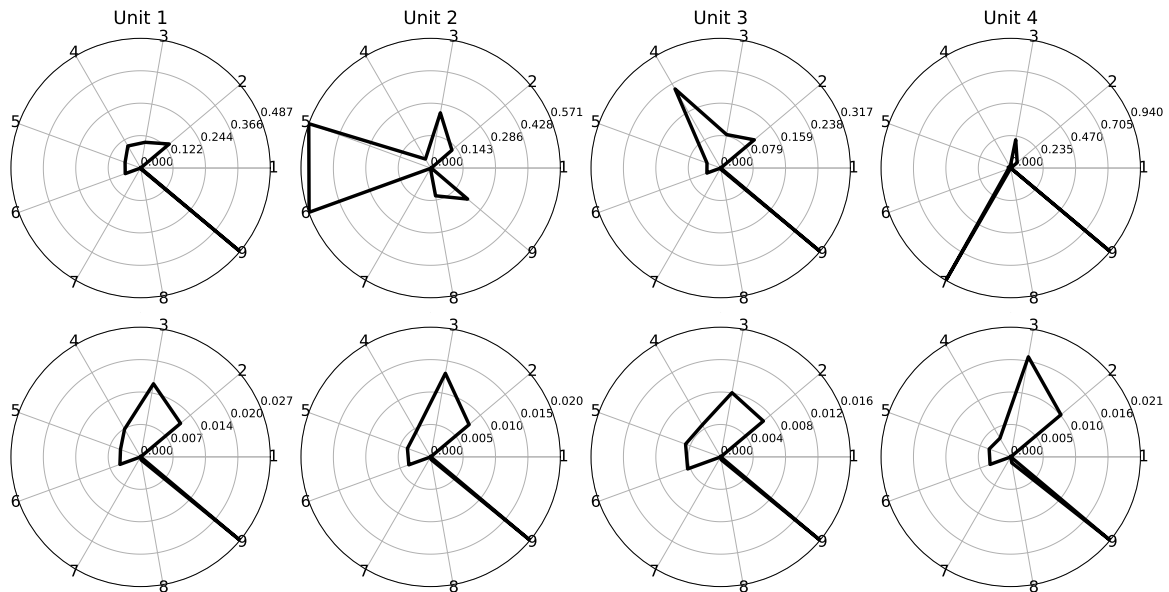


Figure 14. FCAE loss signatures per cooling unit for the true positive cases (top) and the false negatives (bottom).

596 Figure 15 shows the plots for the cases where healthy flights are correctly classified (true negatives)
 597 or wrongly classified as faulty (false positives). For the true negatives, sensor 9 signals are the
 598 ones being reconstructed the worst, although its loss value is still relatively low enough. For the
 599 false positives, the situation is more complex as several sensors contribute to the misclassification.
 600 Furthermore, each unit has a different false positive signature, although the ones in unit 1 and 3 are
 601 close. Sensor 9 is in all cases the major contributor to false positives.

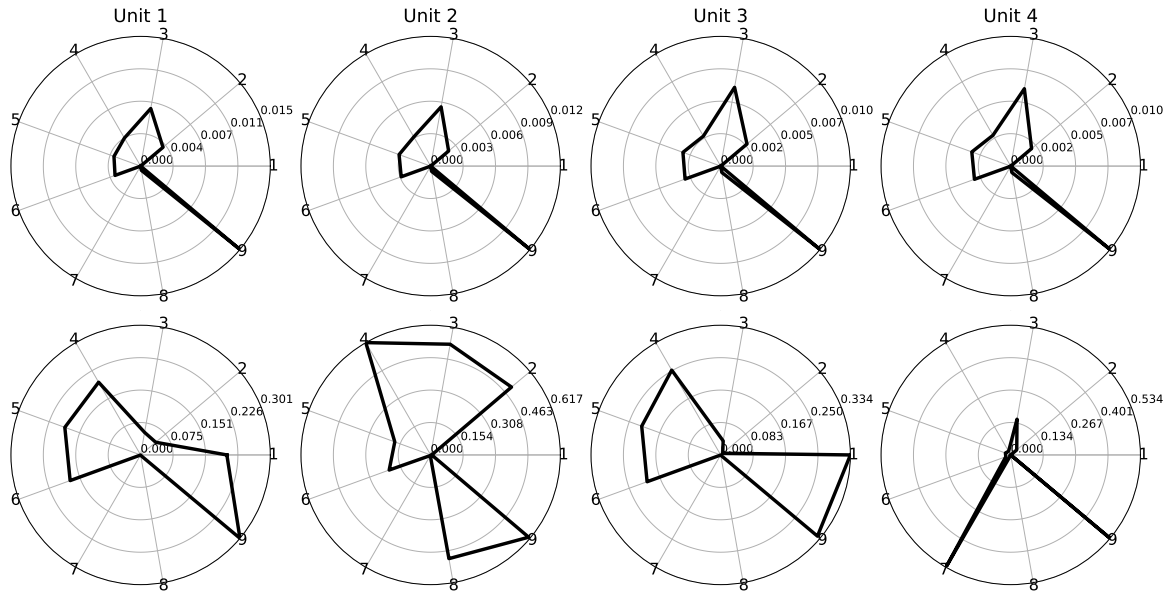


Figure 15. FCAE loss signatures per cooling unit for the true negative cases (top) and the false positives (bottom).

602 However, we noticed that loss signatures are model dependent. For instance, we can see in
 603 Figure 16 how the false negatives and false positive loss signatures for a LSTMMAE (run `lstmae_pca_10`)
 604 can be significantly different from the FCAE ones.

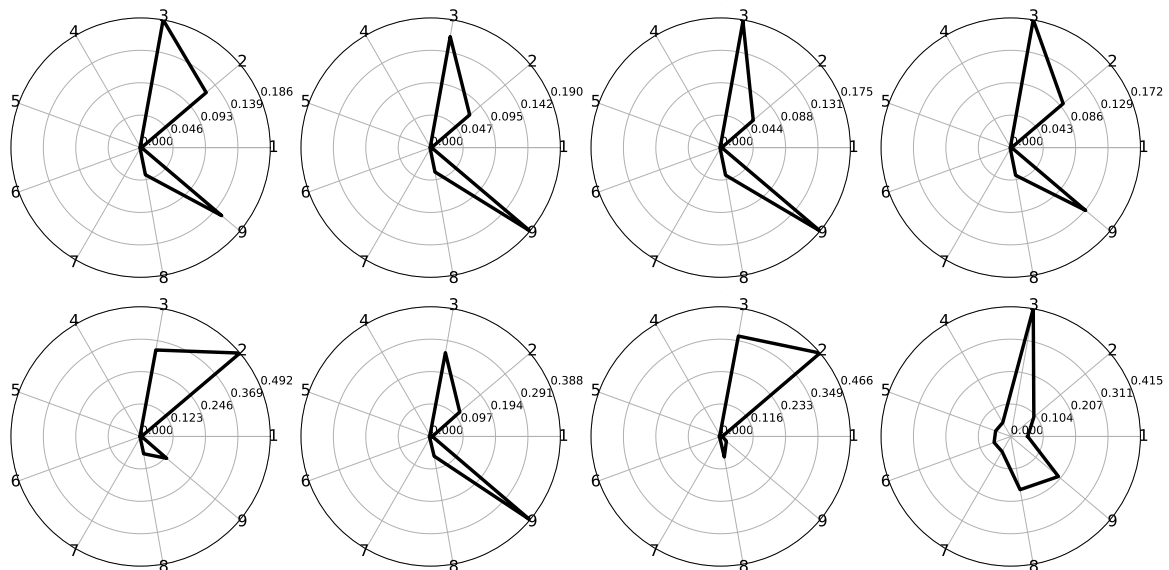


Figure 16. LSTMMAE loss signatures per cooling unit for false negatives (top) and false positives (bottom).

605 After analysis with the help of the airline, false negatives can be generally explained by weak or
 606 short-lived faulty signatures. Even for the airline experts, it was difficult sometimes to detect a clear
 607 faulty signature when looking at some of the signal plots. Another explanation could be the uncertainty
 608 on the labelling of health status, especially for the flights with incipient faults in the beginning of the
 609 failure period. This is however not always the case as it can be observed in the HI plots in Section 7.5.
 610 Finally, we noticed false negatives are less frequent for the flights where there is a corresponding MMS.

611 On the other hand, we have not found an enough degree of similarity in the signatures generated
 612 by the different models as to hold a specific subset of sensors responsible for the false positives. Noisy

613 data could have been then a clear cause, such as a malfunction in an external system impacting the
614 operation of the cooling system and changing signal correlations. Whereas noise can still be one of the
615 issues, false positives can also be linked to model shortcomings in discriminating certain cases. Also, a
616 subset of the false positives are repeating in almost every run, which indicates that for some reason
617 correlations in the sensor signals in these cases must be significantly different from the ones found
618 in healthy signals. Finally, some of the false positives are correlated with the presence of MMS (see
619 example in Section 7.5), which indicates the detection of some abnormal behaviour on the system.

620 7.5. Health indicator: correlation with PM faults and MMS

621 In this section, we analyse whether high anomaly scores and predicted failure health status
622 correspond well with the periods where a fault has been identified. For that, we plot the HI showing
623 the anomaly score per flight and system unit along with the failure periods. Due to the size of the
624 figures, we present only the plots for two of the tails in the test set. The anomaly scores in the plots has
625 been generated with a LSTMMAE model (run `lstmae_pca_10`).

626 Figure 17 shows the HI for the tail `enws1czm`. There are four plots, each one representing one of
627 the four system units. The blue horizontal line is the average HI for healthy data and the red horizontal
628 line is the anomaly threshold τ . On top of each plot, the coloured bars show the PM failure periods:
629 TRUE faults in red and LIKELY faults in orange. The blue little ticks underneath the bars represent the
630 MMS, whereas the vertical lines represent the anomaly scores for a system unit computed over a flight.
631 Health status is determined from the anomaly scores and threshold: red for the faulty flights ($\tau > HI$)
632 and green otherwise.

633 The long TRUE failure period in unit 1 has many flights properly identified as faulty. However, for
634 the LIKELY failure period that follows, no flights are identified as faulty. In unit 3, only four flights are
635 faulty for the LIKELY fault period. These two LIKELY cases are examples of fault uncertainty, which
636 makes it difficult to ascertain whether the model predictions are correct.

637 Another observation is that for the long failure period in unit 1, some of the anomaly scores can
638 be high in spite of the lack of an associated MMS. We have noticed this happening for other failure
639 periods in other tails, which drew the attention of the airline as it shows our anomaly detection can be
640 complementary to the one currently implemented in the aircraft.

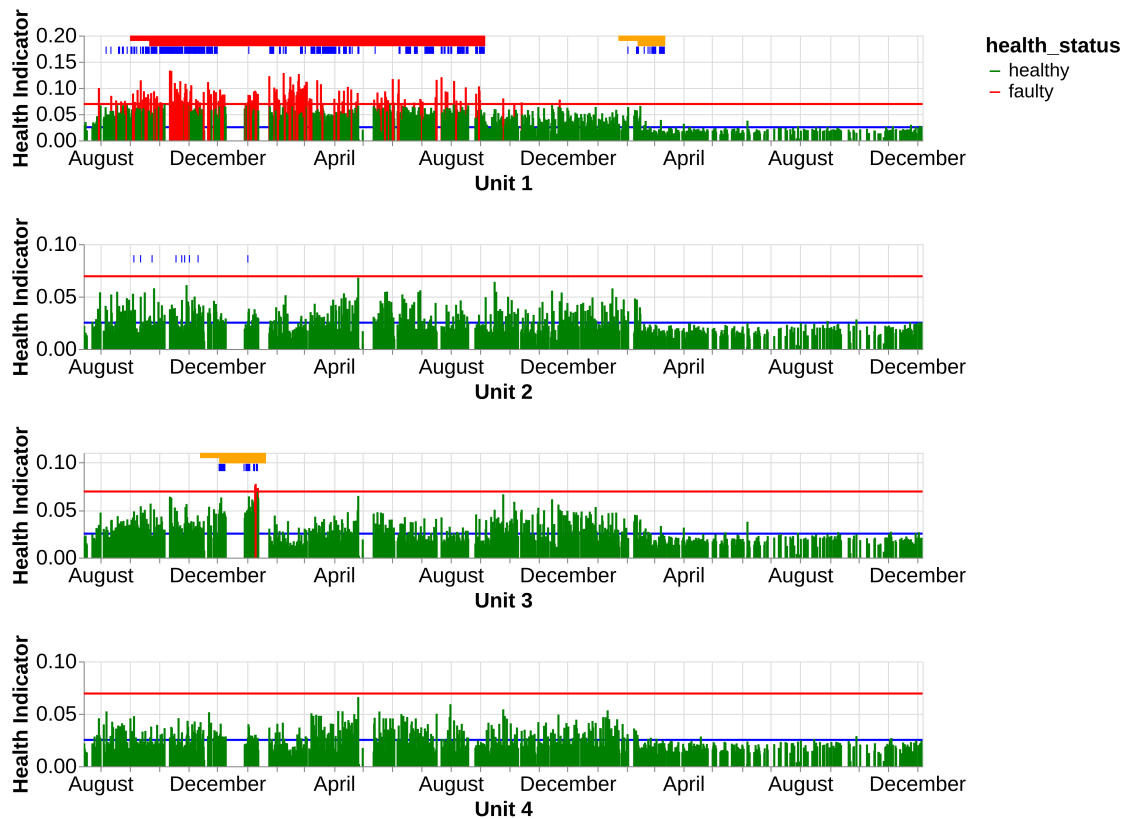


Figure 17. HI computed for the test tail enws1czm.

641 Figure 18 plots the HI for another tail in the test set ($\tau j y j d t a f$). The second fault in unit 4 is well
 642 identified, with all corresponding flights in the period being predicted as faulty. Unfortunately, this
 643 is not at all the case for the shorter failure periods in units 3 and 4, in spite of the anomaly scores
 644 being relatively high and close to the threshold. The shorter LIKELY fault period in unit 1 is also well
 645 identified. In unit 2, there are a few MMS and no confirmed PM faults. It can be observed that some of
 646 the MMS correlate sometimes with high anomaly scores, which seems to indicate there was some
 647 issue with the system identified by the aircraft and reflected in sensor data. Finally, we can see a few
 648 isolated red anomaly peaks in units 1, 2 and 3 with no corresponding MMS or fault, which are fault
 649 positives.



Figure 18. HI computed for the test tail `tjyjdtaf`.

650 7.6. Fault anticipation

651 In addition for our models to be able to properly discriminate healthy from faulty data, we would
 652 like them to anticipate the occurrence of a fault. Ideally, any incipient degradation present in the
 653 signals should translate into a progressively higher anomaly score for the few flights preceding the
 654 start of a failure period. Of course, such capability ultimately depends on whether a degradation
 655 signature actually exists in the sensor signals.

656 Figure 19 shows the evolution of the anomaly scores for the last 40 flights before a cooling unit
 657 runs into failure as computed in run `1stmae_pca_10`. We plot only the subset of faults tagged as `TRUE`
 658 and concerning the tails in the test set. Each plot shows the rolling average of both the anomaly score
 659 per sensor and the aggregated mean loss (thick blue line).

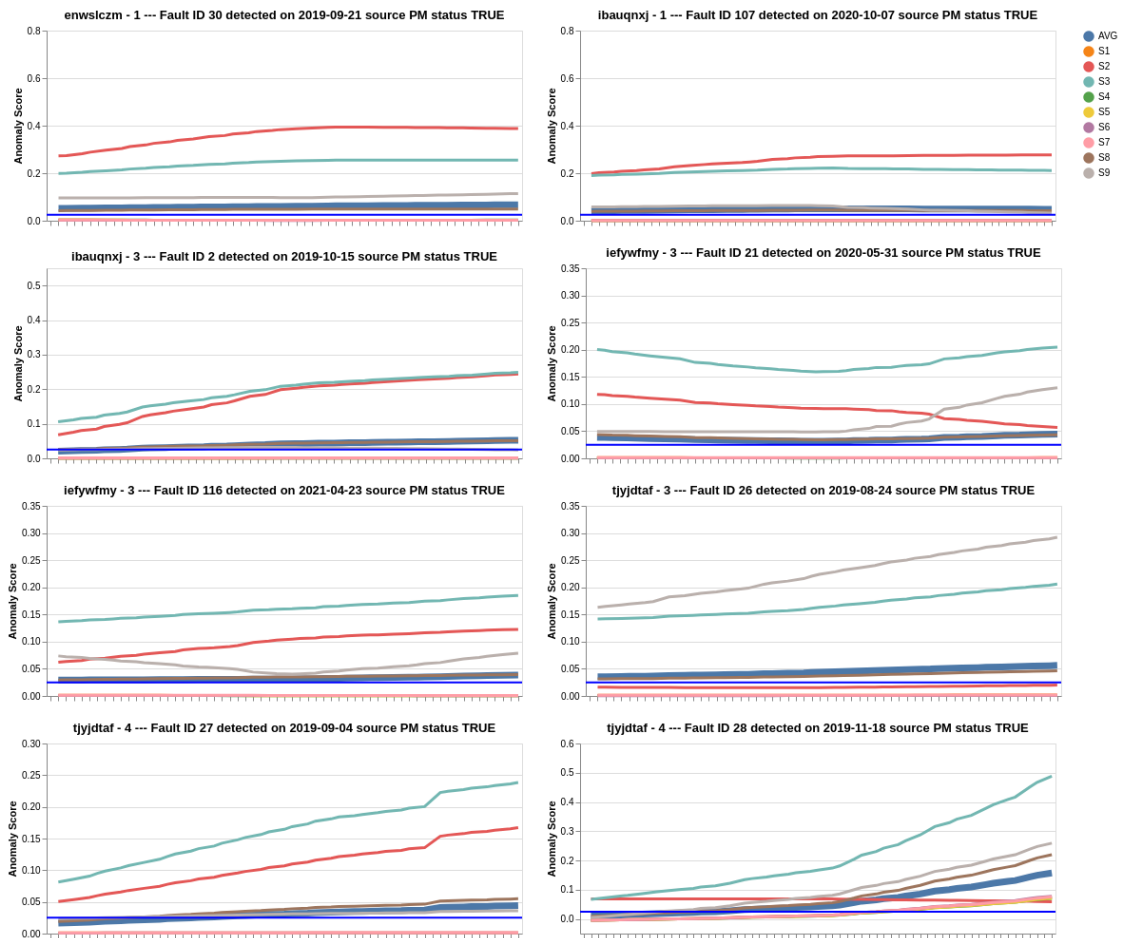


Figure 19. Run-to-failure computed in run `lstmae_pca_10` for the test tails.

660 In general, the average loss is rather flat except for the last plot (`fault_id=28`), where the upward
 661 trend is the most visible by far. Sensors 2, 3 and sometimes 9 are the main contributors to the increasing
 662 trend, whereas the rest of the sensors play a minor role and remain mostly flat except in the last plot.
 663 The characteristics of the anomaly score slopes are highly dependent on the fault and seem to indicate
 664 different degradation signatures probably corresponding to different failure modes.

665 Although an upward trend in the anomaly scores of some sensors can be observed for most of the
 666 faults, the prediction of a fault occurrence seems difficult as dynamics are different from one case to
 667 another. The limited number of faults in the dataset and the lack of knowledge on the failure modes
 668 make the task very challenging. However, what these plots do seem to reveal is the importance of
 669 sensors 2, 3 and 9 and the insignificance of the others as fault precursors when using a LSTMAE. These
 670 same subset of sensors are also the main contributors to the loss signatures of a LSTMAE (see Figure16).

671 8. Conclusions

672 In this study, we illustrated the difficulties in extracting meaningful information for health
 673 monitoring, by exploiting real raw sensor and maintenance data provided by an airline. We focused
 674 on a specific example with the cooling unit system of a modern wide body aircraft. Determining the
 675 ground truth on the real health status of the aircraft systems is an arduous task: the labelling of data is
 676 uncertain and the use of supervised ML techniques unfitted.

677 In this context, we investigated the potential of a semi-supervised anomaly detection framework:
 678 we evaluated three different autoencoder architectures to assess whether such approach is viable with
 679 a complex system such as the cooling unit system. The training and the proper evaluation of the
 680 models have proven to be challenging because of the uncertainty in the ground truth and the lack

681 of prior knowledge on the system. In the end, the proposed approach provided useful insights for
 682 condition monitoring in spite of uncertain and noisy data.

683 High anomaly scores and faulty status detected by our models matched periods where a fault
 684 had been identified. Further, our metrics (the health indicator) were able to correctly detect faulty
 685 flights which were not identified as such by the on-board detection systems, with a clear upward trend
 686 before a failure occurs in some situations. This approach opens promising perspectives, and would be
 687 appreciated as a complement to the existing certified aircraft fault detection systems. Alongside other
 688 information sources such as MMS, the health indicator can help the airline maintenance team with
 689 more efficient diagnostics. Although the different dynamics in the evolution of the anomaly curves do
 690 not make their use very straightforward for fault prediction, they can be of great value for monitoring
 691 the rate of degradation, anticipating an impending failure and scheduling a preventive replacement at
 692 a convenient time-slot.

693 Future research works include a thorough finetuning of hyper-parameters and autoencoder
 694 architectures and a better understanding of the trends in resulting anomaly scores depending on the
 695 physics of the system; this would bring the approach to a higher technology readiness level, which
 696 falls beyond the scope of our current study. Further, as LSTMAE and RNN-based autoencoders seems
 697 unfitted for such long time series, attention-based mechanisms [43] or transformer architectures [44]
 698 could be a powerful tool to cope with longer intra-flight or even inter-flight temporal dependencies.

699 In spite of the mentioned shortcomings pointed out in our study, we have shown the potential of
 700 our proposed approach and its value to monitor the health of an aircraft fleet.

701 **Author Contributions:** conceptualization, L.B.; methodology, L.B., P.B. and X.O.; software, L.B., P.B. and X.O.;
 702 validation, L.B., X.O. and F.F.; formal analysis, L.B., X.O. and P.B.; investigation, L.B., X.O. and P.B.; resources,
 703 L.B., X.O., P.B. and F.F.; data curation, L.B. and F.F.; writing—original draft preparation, L.B.; writing—review
 704 and editing, L.B., X.O., P.B. and F.F.; visualization, L.B.; supervision, L.B.; project administration, X.O.; funding
 705 acquisition, X.O.

706 **Funding:** This research has received funding from the European Union’s Horizon 2020 research and innovation
 707 programme under grant agreement No 769288.

708 **Conflicts of Interest:** The authors declare no conflict of interest.

709 Abbreviations

710 The following abbreviations are used in this manuscript:

711 ACARS	Aircraft Communications Addressing and Reporting System
AE	Autoencoder
CAE	Convolutional Autoencoder
CMCF	Central Maintenance Computing Function
CNN	Convolutional Neural Network
FDE	Flight Deck Effects
GRU	Gated Recurrent Unit
HI	Health Indicator
712 LSTM	Long-Short Term Memory
LSTMAE	Long-Short Term Memory Autoencoder
MMS	Maintenance Messages
MSE	Mean Square Error
PCA	Principal Component Analysis
PHM	Prognostics and Health Management
PM	Predictive Maintenance
RNN	Recurrent Neural Networks
RUL	Remaining Useful Life

713

- 714 1. Basora, L.; Olive, X.; Dubot, T. Recent Advances in Anomaly Detection Methods Applied to Aviation. *Aerospace* **2019**, *6*, 117. doi:10.3390/aerospace6110117.
- 715 2. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly detection: A survey. *ACM Computing Surveys* **2009**, *41*, 1–58.
- 716 717 doi:10.1145/1541880.1541882.
- 718 3. Pimentel, M.A.; Clifton, D.A.; Clifton, L.; Tarassenko, L. A review of novelty detection. *Signal Processing*
- 719 **2014**, *99*, 215–249. doi:10.1016/j.sigpro.2013.12.026.
- 720 4. Agrawal, S.; Agrawal, J. Survey on Anomaly Detection using Data Mining Techniques. *Procedia Computer*
- 721 *Science* **2015**, *60*, 708–713. doi:10.1016/j.procs.2015.08.220.
- 722 5. Chalapathy, R.; Chawla, S. Deep Learning for Anomaly Detection: A Survey. *arXiv:1901.03407* **2019**.
- 723 6. Das, S.; Matthews, B.L.; Srivastava, A.N.; Oza, N.C. Multiple kernel learning for heterogeneous anomaly
- 724 detection: algorithm and aviation safety case study. Proceedings of the 16th ACM SIGKDD international
- 725 conference on Knowledge discovery and data mining. ACM, 2010, pp. 47–56.
- 726 7. Schölkopf, B.; Williamson, R.C.; Smola, A.J.; Shawe-Taylor, J.; Platt, J.C. Support vector method for novelty
- 727 detection. *Advances in neural information processing systems*, 2000, pp. 582–588.
- 728 8. Puranik, T.G.; Mavris, D.N. Anomaly Detection in General-Aviation Operations Using Energy Metrics and
- 729 Flight-Data Records. *Journal of Aerospace Information Systems* **2017**, pp. 22–36.
- 730 9. Li, L.; Gariel, M.; Hansman, R.J.; Palacios, R. Anomaly detection in onboard-recorded flight data using
- 731 cluster analysis. 2011 IEEE/AIAA 30th Digital Avionics Systems Conference. IEEE, 2011, pp. 4A4–1.
- 732 10. Li, L.; Das, S.; John Hansman, R.; Palacios, R.; Srivastava, A.N. Analysis of flight data using clustering
- 733 techniques for detecting abnormal operations. *Journal of Aerospace information systems* **2015**, *12*, 587–598.
- 734 11. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A Density-based Algorithm for Discovering Clusters in Large
- 735 Spatial Databases with Noise. Proceedings of the Second International Conference on Knowledge Discovery
- 736 and Data Mining. AAAI Press, 1996, KDD'96, pp. 226–231.
- 737 12. Oehling, J.; Barry, D.J. Using machine learning methods in airline flight data monitoring to generate new
- 738 operational safety knowledge from existing data. *Safety science* **2019**, *114*, 89–104.
- 739 13. Kriegel, H.P.; Kröger, P.; Schubert, E.; Zimek, A. LoOP: local outlier probabilities. Proceedings of the 18th
- 740 ACM conference on Information and knowledge management. ACM, 2009, pp. 1649–1652.
- 741 14. Jolliffe, I.T. *Principal Component Analysis*; Springer Series in Statistics, Springer New York: New York, NY,
- 742 1986. doi:10.1007/978-1-4757-1904-8.
- 743 15. Baptista, M.; de Medeiros, I.P.; Malere, J.P.; Nascimento Jr, C.; Prendinger, H.; Henriques, E.M. Comparative
- 744 case study of life usage and data-driven prognostics techniques using aircraft fault messages. *Computers in*
- 745 *Industry* **2017**, *86*, 1–14.
- 746 16. Xiongzhi, C.; Jinsong, Y.; Diyin, T.; Yingxun, W. Remaining useful life prognostic estimation for aircraft
- 747 subsystems or components: A review. IEEE 2011 10th International Conference on Electronic Measurement
- 748 & Instruments. IEEE, 2011, Vol. 2, pp. 94–98.
- 749 17. Si, X.S.; Wang, W.; Hu, C.H.; Zhou, D.H. Remaining useful life estimation—a review on the statistical data
- 750 driven approaches. *European journal of operational research* **2011**, *213*, 1–14.
- 751 18. Zhao, R.; Yan, R.; Chen, Z.; Mao, K.; Wang, P.; Gao, R.X. Deep learning and its applications to machine
- 752 health monitoring. *Mechanical Systems and Signal Processing* **2019**, *115*, 213–237.
- 753 19. Tobon-Mejia, D.A.; Medjaher, K.; Zerhouni, N.; Tripot, G. A data-driven failure prognostics method based
- 754 on mixture of Gaussians hidden Markov models. *IEEE Transactions on reliability* **2012**, *61*, 491–503.
- 755 20. Zhao, P.; Kurihara, M.; Tanaka, J.; Noda, T.; Chikuma, S.; Suzuki, T. Advanced correlation-based anomaly
- 756 detection method for predictive maintenance. 2017 IEEE International Conference on Prognostics and
- 757 Health Management (ICPHM). IEEE, 2017, pp. 78–83.
- 758 21. Heimes, F.O. Recurrent neural networks for remaining useful life estimation. 2008 international conference
- 759 on prognostics and health management. IEEE, 2008, pp. 1–6.
- 760 22. Gugulothu, N.; Tv, V.; Malhotra, P.; Vig, L.; Agarwal, P.; Shroff, G. Predicting remaining useful life using
- 761 time series embeddings based on recurrent neural networks. *arXiv preprint arXiv:1709.01073* **2017**.
- 762 23. Husebø, A.B.; Kandukuri, S.T.; Klausen, A.; Robbersmyr, K.G.; others. Rapid Diagnosis of Induction Motor
- 763 Electrical Faults using Convolutional Autoencoder Feature Extraction. PHM Society European Conference,
- 764 2020, Vol. 5, pp. 10–10.

- 765 24. Lee, K.; Kim, J.K.; Kim, J.; Hur, K.; Kim, H. CNN and GRU combination scheme for bearing anomaly
766 detection in rotating machinery health monitoring. 2018 1st IEEE International Conference on Knowledge
767 Innovation and Invention (ICKII). IEEE, 2018, pp. 102–105.
- 768 25. Hundman, K.; Constantinou, V.; Laporte, C.; Colwell, I.; Soderstrom, T. Detecting spacecraft anomalies
769 using lstms and nonparametric dynamic thresholding. Proceedings of the 24th ACM SIGKDD international
770 conference on knowledge discovery & data mining, 2018, pp. 387–395.
- 771 26. Jianzhong, S.; Chaoyi, L.; Cui, L.; Ziwei, G.; Ronghui, W. A data-driven health indicator extraction method
772 for aircraft air conditioning system health monitoring. *Chinese Journal of Aeronautics* **2019**, *32*, 409–416.
- 773 27. Khumprom, P.; Grewell, D.; Yodo, N. Deep Neural Network Feature Selection Approaches for Data-Driven
774 Prognostic Model of Aircraft Engines. *Aerospace* **2020**, *7*, 132.
- 775 28. Schwartz, S.; Jimenez, J.J.M.; Salaün, M.; Vingerhoeds, R. A fault mode identification methodology based
776 on self-organizing map. *Neural Computing and Applications* **2020**, pp. 1–19.
- 777 29. Baptista, M.; Prendinger, H.; Henriques, E. Prognostics in Aeronautics with Deep Recurrent Neural
778 Networks. PHM Society European Conference, 2020, Vol. 5, pp. 11–11.
- 779 30. Reddy, K.K.; Sarkar, S.; Venugopalan, V.; Giering, M. Anomaly detection and fault disambiguation in large
780 flight data: a multi-modal deep auto-encoder approach. Annual Conference of the Prognostics and Health
781 Management Society, 2016.
- 782 31. Fu, X.; Luo, H.; Zhong, S.; Lin, L. Aircraft engine fault detection based on grouped convolutional denoising
783 autoencoders. *Chinese Journal of Aeronautics* **2019**, *32*, 296–307.
- 784 32. Balaban, E.; Saxena, A.; Bansal, P.; Goebel, K.F.; Curran, S. Modeling, Detection, and
785 Disambiguation of Sensor Faults for Aerospace Applications. *IEEE Sensors Journal* **2009**, *9*, 1907–1917.
786 doi:10.1109/JSEN.2009.2030284.
- 787 33. Denis, F. PAC learning from positive statistical queries. International Conference on Algorithmic Learning
788 Theory. Springer, 1998, pp. 112–126.
- 789 34. Zhang, B.; Zuo, W. Learning from positive and unlabeled examples: A survey. 2008 International
790 Symposiums on Information Processing. IEEE, 2008, pp. 650–654.
- 791 35. Springenberg, J.T.; Dosovitskiy, A.; Brox, T.; Riedmiller, M. Striving for simplicity: The all convolutional
792 net. *arXiv preprint arXiv:1412.6806* **2014**.
- 793 36. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal
794 covariate shift. *arXiv preprint arXiv:1502.03167* **2015**.
- 795 37. Malhotra, P.; Ramakrishnan, A.; Anand, G.; Vig, L.; Agarwal, P.; Shroff, G. LSTM-based encoder-decoder
796 for multi-sensor anomaly detection. *arXiv preprint arXiv:1607.00148* **2016**.
- 797 38. Srivastava, N.; Mansimov, E.; Salakhudinov, R. Unsupervised learning of video representations using
798 lstms. International conference on machine learning, 2015, pp. 843–852.
- 799 39. Pereira, J.; Silveira, M. Unsupervised anomaly detection in energy time series data using variational
800 recurrent autoencoders with attention. 2018 17th IEEE International Conference on Machine Learning and
801 Applications (ICMLA). IEEE, 2018, pp. 1275–1282.
- 802 40. Pascanu, R.; Mikolov, T.; Bengio, Y. On the difficulty of training recurrent neural networks. International
803 conference on machine learning. PMLR, 2013, pp. 1310–1318.
- 804 41. Williams, R.J.; Peng, J. An efficient gradient-based algorithm for on-line training of recurrent network
805 trajectories. *Neural computation* **1990**, *2*, 490–501.
- 806 42. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* **2014**.
- 807 43. Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhudinov, R.; Zemel, R.; Bengio, Y. Show, attend and
808 tell: Neural image caption generation with visual attention. International conference on machine learning.
809 PMLR, 2015, pp. 2048–2057.
- 810 44. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I.
811 Attention is all you need. *arXiv preprint arXiv:1706.03762* **2017**.