Appl Intell (2017) 46:227–239
DOI 10.1007/s10489-016-0829-4

Machine learning-based methods for TTF estimation with application to APU prognostics

Chunsheng Yang[1] · Sylvain Letourneau[1] · Jie Liu[2] · Qiangqiang Cheng[3] · Yubin Yang[4]

Published online: 16 August 2016

Abstract Machine learning-based predictive modeling is to develop machine learning-based or data-driven models to predict failures before they occur and estimate the remaining useful life or time to failure (TTF) accurately. Accurate TTF estimation plays a vital role in predictive maintenance or PHM (Prognostic and Health Management). Despite the availability of large amounts of data and a variety of powerful data analysis methods, predictive models developed for PHM still fail to provide accurate and precise TTF estimations. This paper addresses this problem by integrating machine learning algorithms such as classification, regression and clustering. A classification system is used to determine the likelihood of component failures such that rough indications of TTF are provided. Clustering and SVM-based local regression are then introduced to refine the time to failure estimations provided by the classification system. The paper illustrates the applicability of the proposed approach through a real world aerospace application with details on data pre-processing requirements. The results demonstrate that the proposed method can reduce uncertainty in estimating time to failure, which in turn helps augment the usefulness of predictive maintenance.

✉ Chunsheng Yang
Chunsheng.yang@nrc.gc.ca

1 National Research Council Canada, Ottawa, Ontario K1A 0R6, Canada

2 Carleton University, Ottawa, Canada

3 Nanchang University, Nanchang, China

4 The State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China

Keywords Machine learning · Prognostics and health management · Classification · Clustering · Regression · Support vector machines (SVMs)

1 Introduction

The needs for higher equipment availability, reliability, productivity, and lower maintenance costs are driving the development and integration of prognostic and health management (PHM) systems. Taking advantage of advances in sensor technologies, PHM systems favor a predictive maintenance strategy [1–4] through continuous monitoring of data gathered from equipment and maintenance staff reports in the event that there is a risk of a component failure. A PHM system may also supplement the component failure predictions with an estimation of the time to failure (TTF), which is defined as the expected remaining time before the given component stops fulfilling its function. In order to avoid disruption and minimize maintenance costs, these TTF estimates need to be as reliable and precise as possible.

Traditional methods to estimate TTF include reliability analysis [6] and knowledge-based approaches from physics and material sciences [7–10]. These approaches help in understanding the underlying physical mechanisms but they require enormous amounts of background information. Furthermore, applying these approaches may also be difficult as they tend to rely on vehicles and techniques to obtain data on component damage or material properties. With the development and integration of data acquisition devices into complex equipment, data mining approaches are now starting to complement the traditional methods for building prognostic models [5, 11, 14]. Recent results show the potential of classification systems in identifying the likelihood of component failures in a timely manner. However, none of the existing techniques can provide sufficiently precise time to failure estimates to optimize predictive maintenance which identify the optimal time for performing the next maintenance action [22–24].

For instance, the data mining methodology proposed in [11] can build classification models for predictive maintenance. These models continuously assess the probabilities of a component failure within a pre-specified alert target window (e.g., between 1 and 20 days in advance of a functional failure), but often fail to provide precise TTF estimates. When a classifier detects patterns in the data that are characteristic of an incipient failure, it generates an alert indicating that the suspected component is likely to fail within the alert target window and without being able to specify the exact number of days or hours of operation left. With this approach, the larger the alert target window, the higher the imprecision on the TTF estimates. In some specific applications, it is reasonable to try to increase precision by reducing the width of the target window. However, this is generally not suitable as it could prevent the end users from getting alerts as early as possible. In turn, this would reduce the opportunity for optimization and the benefits of prognostics. A too narrow target window may also have detrimental effects on the performance of the predictive models. For instance, when a component has various failure modes, each following their own time frame, there is a risk that a model specific to a narrow target window would only be able to detect a fraction of these failure modes.

Estimating TTF can be seen as a regression problem and as such, regression analysis and time-series forecasting methods could be used to build models that try to directly estimate TTF from sensory data. To be successful, such models need to accurately map all then subtle changes in the data to specific life reduction estimates. These models also need to account for the fact that with complex components, we often observe significant variations in actual time to failure. Obviously, building such models is a challenging task that requires ample amounts of high quality and relevant data. On top of this, data from real world equipment is typically characterized by issues such as irregular sampling intervals, small signal/noise ratio, and sensor measurement errors. It is therefore, generally hopeless to try to develop a global regression model for TTF from sensor data. On the other hand, it is plausible that regression could be successfully applied locally on well chosen portions of the real world sensor data. This paper investigates this hypothesis by trying to demonstrate that regression analysis can help improve the preciseness of TTF estimates.

This paper proposes an on-demand approach to estimating TTF by combining classification, clustering, and regression-based approaches. It relies on a comprehensive machine learning methodology to develop a classification system which is capable of identifying incipient component failures and providing rough TTF estimates. Clustering is used to partition the sensor data into multiple regions and a regression model is developed to estimate TTF within each region. When the classifier uncovers a potential component failure, a mapping function decides which regression model should be used to provide a TTF estimate. A final step produces the final TTF estimation based on the output from the classification and regression models. We name the proposed method "on-demand regression" as regression is only used once the classification system has identified the potential for a component failure.

To our best knowledge, this is first attempt to improve TTF estimation by combining classification and regression. The main idea behind this method is to improve the performance of regression models by filtering negative predictions generated from a classifier given system observations. In other words, only when a state observation is classified as a potential component failure (positive prediction) the regression models are used to estimate its TTF. For negative prediction, it is not necessary to estimate its TTF. The work related on-demand regression is model fusion such as stacking techniques [21, 27]. The stacking method fuses multiple baseline models to improve the model performance. For instance, the work in [21] is to stack the baseline classifiers for improving the performance of classification systems, and the work in [27] is to combine multiple baseline regression models to enhance the precision of regression. Our method is to select a best regression model from a set of baseline models to estimate TTF given a positive observation.

This paper extends a preliminary description of this work [12] by introducing a data mining methodology for developing predictive models from historical operation data, by providing updated results including the new results from other applications, and by discussing how the choice regarding the number of clusters affects performance in Discussion Section.

Prior to detailing the approach, the paper explains the challenges with support from real world data from an aerospace application. The same application is also used to illustrate the applicability and the usefulness of the proposed approach. The discussion section elaborates on parameter tuning, potential improvements, and on the use of the technique with other sources of data.

2 Challenges

Accurate and detailed health information on key systems and components is of utmost importance to help optimize the maintenance and management of complex systems. Ideally, powerful prognostic models, well integrated into the organization's information systems, would automatically combine sensor data, historical maintenance information, system configurations, and other sources of information to continuously provide accurate and precise TTF information. Regression methods, which are specifically designed to predict numerical values such as TTF, appear well suited to develop these models. Unfortunately, many typical issues of real world data from complex equipment severely constrain the applicability and power of regression modeling. To illustrate, let us consider an aerospace application in which the objective is to build a prognostic model for the starter motor of the Auxiliary Power Unit engine (APU).

The data for this application have been produced by a fleet of 73 commercial aircraft over a period of 10 years. Only ACARS (Aircraft Communications Addressing and Reporting System) APU starting reports have been made available. The dataset that has been created from these reports consists of 18 attributes (5 symbolic, 11 numeric, and 2 for date and time of the event). For 11 numeric attribute, there are six main variables related to APU performance: ambient air temperature, ambient air pressure peak value of exhaust gas temperature in starting process, rotational speed at the moment of occurrence, time duration of starting process, exhaust gas temperature when air conditioning is enable after starting with 100 % speed. The TTF is a responding variable and these six variables are used as independent variables in developing regression models, classifier, and model selector. More than 161000 observations are available for this task. Only a subset of these observations is relevant for learning the predictive models which have been collected around each occurrence of component failures. In this particular task, we use engine operating hours as the time unit. Our analysis has been based on data generated between 250 operating hours prior to the failure and 30 h after. A comprehensive search in the maintenance database revealed information on 83 occurrences of APU starter motor replacements. Access to information from further testing of the components following a replacement is not available and as such it is assumed that a replacement is equivalent to an actual failure. When an engine suffered consecutive failures in a short period of time, the aforementioned above interval is constrained to ensure that each observation is included only once. The data from 61 failures are used for learning and data from the remaining 22 failures are reserved for testing.

To evaluate the feasibility of regression as a direct way to predict TTF, the initial representation is augmented with a TTF attribute. This attribute is simply defined as the difference between the engine operating hours in the current observation and the operating hours of this engine at the next starter failure. Instances observed following a failure have been removed. An SVM (Support Vector Machine)-based regression model is built using the training dataset and then applied on the testing set. Figure 1 shows results from one

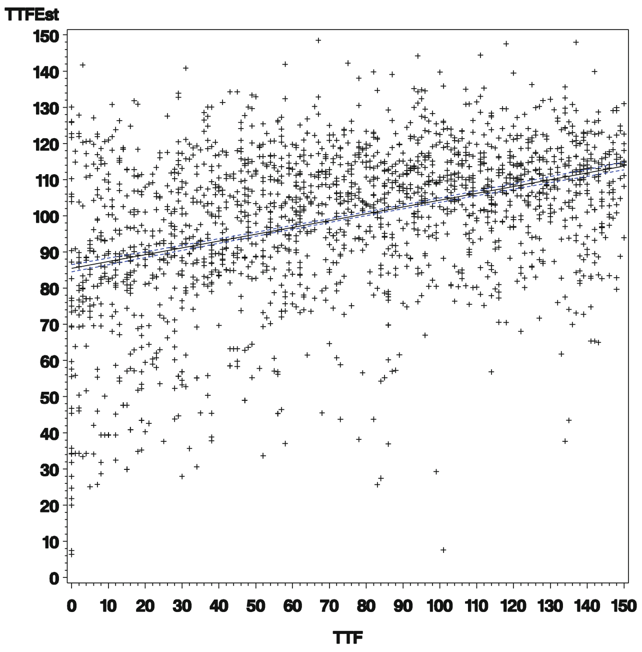


Fig. 1 TTF versus actual from a global SVM-based regression model

of the best SVM models developed using 61 failure time series data, testing with 22 failure time-series data. The axis ***TTF*** is the actual measurements, and the axis ***TTFEst*** is the estimate from the regression model. The straight line shows the best fit regression line while the dash lines represent the corresponding 95 % confidence interval. First, significant errors can be observed as the regression line is far from the desired 45 degree diagonal. Moreover, the scatter clearly illustrates the lack of fit of the model. The expected error from this model is 58.7 with a standard deviation of 43.8. Other regression methods such as NN (Neural Network) (for example, multilayer perception feedforward NN) and linear regression lead to similar performance. The following paragraphs discuss some of the reasons that explain the lack of success of global regression models.

2.1 Irregular sampling rate

Most traditional forecasting techniques require a fixed interval between measurements (e.g., every hour, every day, or every month). In this particular application, the data are systematically collected at every start of the APU. On the other hand, the data are not sent to the central system unless the on-board system decides to do so. Due to different configurations across the fleet and over time, APU data are recorded at every start for some of the aircraft and at every other start for other aircraft and so on up to every 8 starts. As a result, some forecasting methods can simply not be applied. Unequal sampling rate also means that the time-series corresponding to the different failures have different lengths. This may lead to an unbalanced representation of the failure cases. A possible solution is to re-sample the data through interpolation or smoothing but given the high variability in the original sampling rate and the low signal/noise ratio, such processing is likely to worsen the modeling for TTF estimation.

2.2 Lack of relevant information

In order to accurately predict remaining life, the model needs to be able to estimate the current life consumption. The information required to evaluate life consumption could come from highly informative core measurements that adequately account for the internal state of the component. Such high value information is typical from laboratory testing equipment but rarely available from sensors deployed on today's complex equipment. Information about life consumption can also be captured directly by means of a counter. For instance, in the APU starter motor application, we use the engine operating hours to approximate life consumption but it is far from perfect as the starter motor and the engine may consume their life differently. Moreover, engine overhauls which occur at regular intervals result in a reset of the engine operating hours counter. We do not have access to detailed information on work performed during the overhaul and as such, it is impossible to determine if the reset of the engine operating hours counter also corresponds to the repair of the starter motor. Consequently, these resets can possibly introduce cuts in the evaluation of life consumption and cause great difficulty for the regression models.

2.3 Large variance due to contextual effects

Equipment such as aircraft operates in a very dynamic environment. Changes in this environment affect the behavior of the system. In some cases, these changes also affect the measurements taken. For instance, all measurements related to temperature, flow, and pressure are likely to be affected by the altitude of the aircraft. The mode of operation and the status of internal sub-systems and components are also likely to affect the behavior of the performance parameters. All of these contextual effects need to be accounted for in order to understand the behavior of the key parameters and properly use them to infer reliable TTF estimates.

Notwithstanding all of the difficulties mentioned above, this paper argues that regression can still play a meaningful role in improving TTF estimates in prognostic applications. As explained in the following section, the main idea is to partition the data space into relatively homogeneous data subsets and then use different regression models for these subsets.

3 On-demand regression

As mentioned in the introduction, the objective of this paper is to augment the preciseness of TTF predictions by combining regression and an existing classification-based data mining approach for prognostics. Figure 2 illustrates the approach, which is named *On-demand regression*. This name stresses the fact that regression is only used after a need has been identified. First, the classification-based prognostic model analyzes the sensor data from the system to determine if there is a risk for component failure. When such a risk has been confirmed, then regression is applied. In other words, a potential failure is defined as a positive prediction from a classification-based prognostic model (classifier). To help mitigate the potential negative effects of the issues mentioned in the previous section, several regression models are built and only the most relevant one, which is selected based on evaluation performance of each individual model based on MSE metrics, is applied at any given point in time. As explained below, data clustering plays a key role while constructing the regression models and a simple classifier is used to select the most adequate regression model at run-time. The final step of the proposed approach

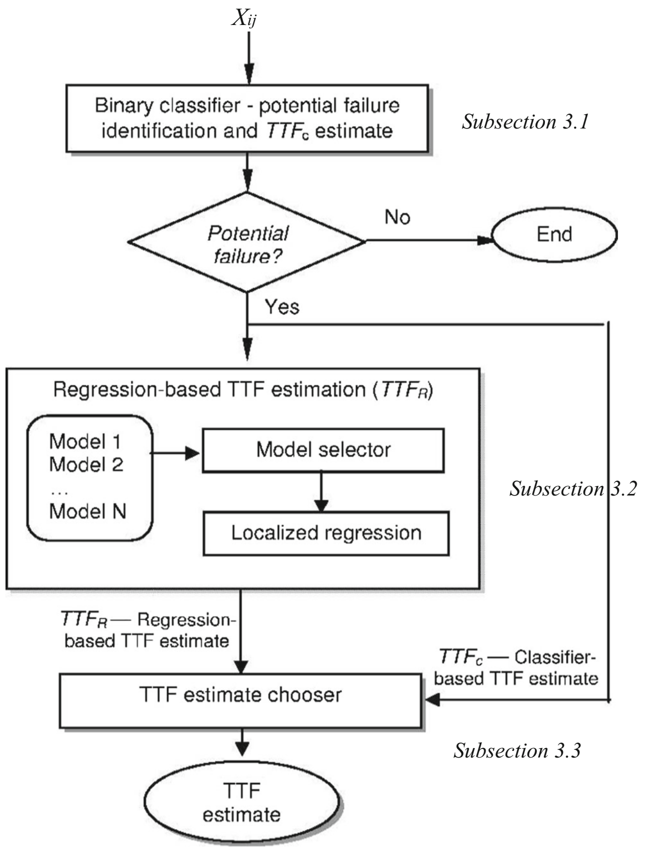

X_{ij}
Binary classifier - potential failure identification and TTF_c estimate
Subsection 3.1
Potential failure?
No
End
Yes
Regression-based TTF estimation (TTF_R)
Model 1
Model 2
...
Model N
Model selector
Localized regression
Subsection 3.2
TTF_R — Regression-based TTF estimate
TTF_c — Classifier-based TTF estimate
TTF estimate chooser
Subsection 3.3
TTF estimate

Fig. 2 Overview of the on-demand regression method

combines predictions from the classification and regression models. To perform the on-demand regression task for estimating the TTF, two types of the data-driven models, either classifiers or baseline regression models, have to be developed using data mining-based methodology described below from historic operational and maintenance data.

The following three sub-sections detail the approach. The first two sub-sections explain the construction of the classification-based prognostic model and the construction of all the models required for the regression step, respectively. The last sub-section discusses the strategy proposed to combine the various predictions.

3.1 Classification-based prognostic

The first step uses a binary classifier that can identify incipient component failures and provide a rough estimate of the remaining useful time. We build this classifier using the data mining methodology documented in [11–13]. As illustrated in Fig. 3, this methodology consists of four steps: data gathering, data representation, modeling and evaluation, and model fusion. These are now succinctly described.

3.1.1 Data gathering

Most data mining algorithms require, as input, a dataset containing examples consisting of vectors of attribute values. Modern machinery often generates many such datasets. For example, an Airbus A320 generates up to 19 different datasets reporting the status of the aircraft in different phases of operation. Our first problem is to select the dataset(s) to use to build models for a particular component. Expert advice and reliable documentation can greatly simplify this choice and help avoid a lengthy trial and error process in selecting appropriate data which are used as training and testing samples. In the case of the APU starter application, sensor measurements collected during start of the APU engine have been selected. This is followed by a selection of subsets of instances to use for the analysis. The datasets are typically so large; it is inefficient to build models using all instances. Simple solutions, such as random sampling, are also inappropriate. To build the desired predictive models, we must be much more focused. Our analysis is based on data generated around each occurrence of starter replacements. Firstly, replacement occurrences are identified and then instances around the time of these occurrences are retrieved.

Unfortunately, retrieving occurrences of failures is quite difficult. The difficulties are derived from errors in maintenance reports and the need to process free form texts. Many types of problems are observed with maintenance reports. Some reports mention a component although it did not fail. Some reports mention a component that failed but not in a way that we are trying to predict. Sometimes the part number entered is incorrect. Sometimes alternative part numbers are used. Sometimes the right information is entered but in the wrong field. Sometimes the right part number is used but the wrong component is identified, there are often multiple components of the same type within the machinery. Manual validation is therefore often required but it could be minimized using information retrieval techniques as explained in [16].

Once the date and the part identifier for each replacement of the APU starter are obtained, the relevant instances from the selected sensor datasets are retrieved. For each dataset and replacement, the data obtained between m operating hours prior to the replacement and n operating hours after are gathered. The numbers m and n depend on the dataset and the component; we usually select m so that we have at least 200 instances available for learning. As mentioned above, for the APU starter application, we use $m = 250$ and $n = 30$.

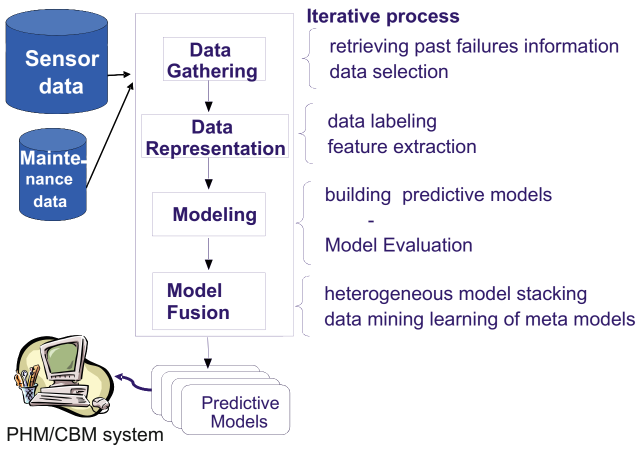


Fig. 3 The machine learning-based methodology for predictive modeling

3.1.2 Data representation

This step describes the data labeling and feature extraction processes. In order to use supervised learning algorithms, a class attribute (or label) is added to the selected sensor data. An automatic labeling approach is proposed which labels as positive ("1") all instances that fall in a pre-determined target window before the occurrence of a starter motor failure and as negative ("0") all other instances. This target window is determined with the requirements of failure mode or applications. For example, it is defined as 50 h for APU Starter prognostics. This labeling scheme allows a classifier to be built that generates an alert whenever the patterns in the data are similar to the ones observed near a failure. In practice, the width of the target window is decided by taking into account the optimal period for the end users to receive the alerts and the balance between positive and negative instances. As a rule of thumb, we try to keep a minimum of 15 % as positive instances to simplify the learning. After labeling, the raw data is transformed to improve data representation. This is done by augmenting the initial representation with new features created using methods from signal processing, time-series analysis, and constructive induction. For instance, a moving average in a given time frame helps to relieve the noise impact. Different techniques generate various new features for building high performance models. Feature selection is also applied on the augmented data representation to automatically remove redundant or irrelevant features. There are two kinds of feature selection methods: domain-oriented method and algorithm-based method [15, 16]. In this work, domain-oriented method, which selects the features based on domain knowledge, is used to select features for modeling.

3.1.3 Modeling and evaluating models

After updating the initial dataset with the class attribute and incorporating data representation enhancements, the required predictive models or classifiers can be built. Data is used from a subset of all failures for learning the models and keep the remaining data for testing. In early experiments, simple algorithms such as decision trees and naive-Bayes are preferred over more complex ones because of their efficiency and because they produce models that can be easily explained to end users. The same algorithm is applied several times with varying attribute subsets and cost information.

To compare the classifiers obtained, a score-based approach is applied that has been developed to evaluate classifiers for prognostic systems. This approach overcomes issues with other criteria (e.g., error-rate, recall, and precision) by taking into account two important aspects of prognostic applications.

The first aspect is that the usefulness of a prediction is a function of the time between the prediction and the actual replacement. Warning too early about a potential failure leads to non-optimal component use; warning too late makes proper repair planning difficult. An evaluation method is therefore required that takes alert timeliness into account. The second aspect relates to coverage of potential failures. Because the learned model classifies each report into one of two categories (likely to fail within target widow; not likely to fail within target window), a model might generate several alerts before the component is actually replaced. More alerts suggest a higher confidence in the prediction. However, a model that generates at least one alert for most component failures is clearly preferred over one that generates many alerts for just a few failures. That is, the model's coverage is very important to minimizing

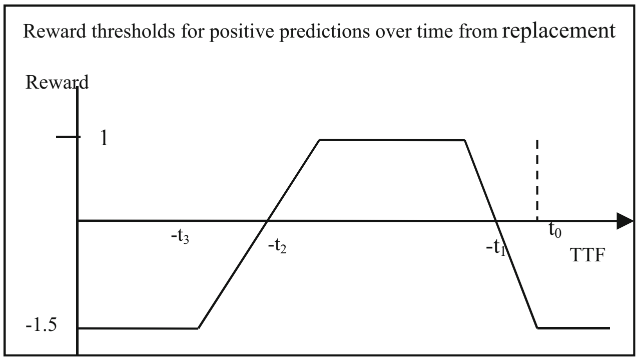


Fig. 4 A reward function for positive predictions

unexpected failures. Given this, an overall scoring metric is needed that considers alert distribution over the various failure cases.

We account for the first aspect (timeliness of the alerts) with a reward function that specifies the reward for each positive prediction (or alert) based on the number of operating hours between the prediction and the actual replacement. Figure 4 shows an example of a reward function. In this case, the optimal score is obtained when the system predicts the failure between t_1 and t_2 hours in advance. Outside that period, a smaller reward is provided to indicate a non-optimal prediction. If a prediction is too far from the target period, then the model gets a negative reward. The target periods and the reward values (maximal positive and negative values) are established with the end users based on operational requirements and resource constraints. Various components would typically require different reward functions.

To account for the second aspect (ability of the model to detect several failures), alert distribution are investigated over the different failure cases. Based on the accuracy, recall, and precision definition, we introduced an overall method to integrate these matrices into one formula. The overall performance metric we propose to evaluate a model is presented as (1).

$$Score = \left[\frac{NrDetected}{NrOfCase}\right]^{sign} \sum_{i=1}^{p} score_i \tag{1}$$

where:

- p is the number of positive predictions in the testing dataset;
- *NrDetected* is the number of failures, which contain at least one alert in the target interval;
- *NrofCase* is the total number of failures in a given testing dataset;
- *Sign* is the sign of $\sum_{i=1}^{p} score_i$. When *Sign* <0 and *NbrDetected*$=0$, *score* is set to zero; and

- *score$_i$* is calculated with the reward function above for each alert.

From (1), the total score for all positive predictions is found to be customized from accuracy and the first part (*Nrdetected/NrOfcase*) is relevant to the recall metric. In terms of process, the threshold of the reward function is determined based on the requirements of the prognostic application at hand (rewards, target period for predictions). Then, all models are run using test dataset(s) and their respective scores are calculated using (1). The model with the highest score is considered as the best model for the application.

3.1.4 Model fusion

Model fusion can be used for two reasons. First, when more than one data set is relevant for a given component, a model for each dataset can be built and then with model fusion, predictions are combined from the various models. Second, model fusion can be applied for performance optimization regardless of the number of datasets selected. In this case, various models are trained using various techniques or parameter settings and then combined to obtain increased performance over using any single model.

Bagging and boosting [17] are two popular techniques to combine models but they are only applicable when there is a single data set and one model type (i.e., a single learning algorithm). For heterogeneous models or multiple data sets, methods are applied based on a voting or stacking strategy [18, 19]. These techniques are globally referred to as multiple classifier systems. In the experiment reported in Section 4, we relied on a simple multiple classification system combining decision tree and rules.

As illustrated in Fig. 2, a classifier needs to provide a rough TTF estimate (TTF_C) whenever it predicts a potential component failure. This TTF estimate is defined based on the expected number of operating hours remaining between a positive prediction and the actual failure. We use only the training data to compute this expected value. Precisely,

$$TTF_c = \frac{1}{N}\sum\nolimits_{i=1}^{N} RemainingOPH_i \qquad (2)$$

where *RemainingOPHi* is the remaining life of the APU starter at the time of the i^{th} positive prediction from the training set, and N is the number of positive predictions made by the classifier on the training dataset. This value is constant for all positive predictions made by the classifier.

3.2 Regression-based TTF estimation

The objective of the second step is to try to improve the preciseness of TTF estimates provided by the classifier described above. This is done through localized regression models. Each model accounts for a specific area of the data space. Whenever the classifier makes a positive prediction, one of the local regression models is selected to compute a new TTF estimate. The construction of the required models is as follows.

First, clustering is used to partition the time-series associated to the various failures. Each individual time-series represents a failure, staring from installation to replacement. In terms of failure effect and mode analysis, the failures or times-series can be grouped based on failure mode or effect. In this work, we can't directly apply failure mode to group the failure time-series because we don't have this information. But we can try to use clustering techniques to group the times-series in which it reflects the similar failure effect. The intent is to obtain clusters of time-series as homogeneous as possible with respect to the performance of the core measurements. This is done by clustering based on the attributes that represent meaningful contexts for the component of interest. In other words, we use clustering to obtain subsets into which the potentially negative effect of contextual conditions is minimized. In the case of the APU starter motor application, the predominant contextual attribute is the age of the starter motor at the time of the failure. As explained above, we approximate this age using the engine operating hours at the time of the failure.

As it is often the case in clustering task, there is a need to pre-determine the number of clusters required for the given application. This number must be sufficiently large enough to obtain acceptable homogeneity within each cluster but not too large as to avoid over partitioning the data. Additionally, we also need to ensure that each cluster contains at least one test time-series for the evaluation purpose. With 22 occurrences of APU starter failures in the test data, at most 22 clusters are possible. As we will explain in the discussion section, it turns out that there was no benefit in using more than 16 clusters for this specific application.

Second, a model selector, which is N-class classifier, is developed in order to assign each positive prediction to a given data subset (Fig. 2). The clustering model built for partitioning the data cannot be deployed for this task as it relies on the operating hours at the failure time, which is unknown for yet to fail components. We resolve this issue with an N-class classifier, where N is the number of clusters. Once the clustering scheme has been established, each instance is tagged with a cluster ID. A classifier is then trained to differentiate each instance by cluster ID as accurately as possible using the measurements available. Based on our experiments, simple decision trees and naive-Bayes classifiers perform well for this task with a typical accuracy of 90 % on test data.

Finally, a regression model is built for each cluster. SVM/SMOReg, NN, and liner regression techniques can be used to build these models. After experiments, we use

SMO (Sequential Minimal Optimization) as the regression model in this work, because SMO outperformed NN and liner regression models. It is noticeable that only a subset of the training data available in each cluster is used to learn the models. This allows us to further limit the scope of the regression models to the areas with the greatest potential for enhancing the precision of TTF estimates. The evaluation procedure starts by running the classifier on the test data to identify positive predictions. For each of these positive predictions, the model selector chooses an adequate regression model to compute TTF estimate noted TTF_R.

3.3 Selecting which TTF estimate to use

Two TTF estimates are produced for each positive prediction: one from the classifier (TTF_C) and one from a local regression model (TTF_R). We now need to decide how to combine them into a single TTF estimate. Our approach is straight forward; it returns TTF_R if $TTF_R < TTF_C$. This is to avoid potentially significant errors that could come from an extrapolation of a regression model. TTF_C corresponds to the value used to limit the range of the output attribute while learning the regression models.

4 Experiments and results

This section reports experimental results on the application of the proposed methods to estimate TTF for prognostics of APU starter failures on commercial aircraft. Detailed information about the data has been discussed in the challenges Section. All models have been built using the WEKA package [20] following the data mining methodology described in Section 3.

To evaluate the performance, a 4-fold cross validation is conducted for each experiment. Table 1 summarizes the training and testing datasets for each fold. The experimental results are shown in Table 2. The last line in the table is the average performance over the cross-validation runs. In this work, the results are evaluated by analyzing the mean error (Err), standard deviation (std), mean squared error (MSE) based on the estimations from the models given the test dataset. These analysis results are summarized in Table 2.

The classifier for identifying potential failures is a multiple classifier system [21] which combines two binary classifiers that are built using the J48.PART and J48 algorithms using the default options. This classification system can detect 95 % failures with 5 % false alert rate. The automatic labeling step is configured such that all observations are tagged with remaining engine operating hours less than 50 hours as positive and all others as negative. This provides sufficient time for the maintenance staff to plan the repair of the starter prior to the actual failure. In this experiment, only the raw measurements without any data representation enhancement are used in addition to default cost information. The expected TTF estimate from this classifier is an average 22.5 hours. As reported in the last line of Table 2, the average error of the TTF estimates from this model alone on the test data is 14.5 with a standard deviation of 14.4. Figure 5 shows the graph of the TTF estimates versus actual

Table 1 Summary for each of the fold in the cross-validation experiments

Fold #	Test dataset		Train dataset	
	Failure cases	Instances	Failure cases	Instances
1	22	2307	61	8814
2	20	2704	63	8560
3	20	2735	63	8243
4	21	3375	62	2746

Table 2 Error for TTF estimation on test data for using regression-only, classifier-only, and the on-demand regression approaches

Fold #	Regression		Classifier		On-demand	
	Err±std	MSE	Err±std	MSE	Err±std	MSE
1	20.7±30.8	1373	14±14.1	393	5.9±16.7	311
2	20.5±36.7	1762	12.9±8.5	238	4.2±7.4	72
3	22.9±30.1	1433	14.9±17.2	517	7.9±12.4	216
4	26.9±35.3	1964	16.2±17.9	581	9.5±25.1	714
Avg	22.7.9±33.2	1639	14.5±14.4	432	4.9±5.4	228

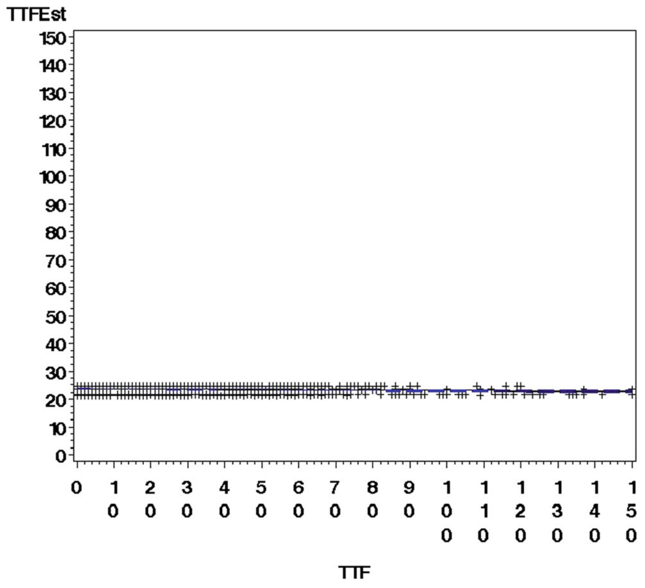


Fig. 5 Predicted TTF versus actual TTF using only the binary classification

TTF when using only the binary classification system. We notice that all the points are around 22.7 which is the estimate that this model returns for all positive instances from the test dataset.

As mentioned earlier, the operating hours at the failure time is used to partition the 83 time-series (one for each failure case) into 16 clusters. Results reported are based on K-means clustering. Experiments with EM-based clustering produced similar results. The model selector is built using J48. Its accuracy on test data is slightly above 85 %. We used SMOReg with a linear polynomial kernel to construct the 16 local regression models. If we assume that these models would be used to generate all TTF estimates, then the results would be as illustrated in Fig. 6. The scatter in the graph shows the lack of fit between many of the estimates and the actual TTF values. This is also confirmed by the second column on the last line in Table 2, which reports an average error of 22.7 ± 33.2 h.

The results from the on-demand regression approach are presented in the last two columns in Table 2 and shown in Fig. 7. With this approach, we observed a much better fit between the estimates and the actual TTF values. With an average error of 4.9±5.4 hours and a minimal squared error (MSE) of 228, the proposed approach clearly outperforms the initial classification-based approach. The large reductions in the average error and in the standard deviation suggest an improvement in the preciseness of TTF estimates by a factor of 5. Of important note is that the TTF estimation from on-demand regression shows the predictions are targeted in the given window. In other words, the only positive predictions from binary classifier are used as input for regression baseline mode based on the selection result of model selector. Particularly, the TTF estimation precision is greatly improved compared to the one global model described in Section 2. The average error is improved by a factor of 8. The errors for on-demand and one global mode are 4.9 to 43.8 for the same test dataset. From Figs. 5 to 7, it is obvious that the fusion rule used in on-demand regression helped greatly filtered some TTF estimations which are located outside of the target windows. In other words, the TTF_R which is greater than 50 is recapped with TTF_C.

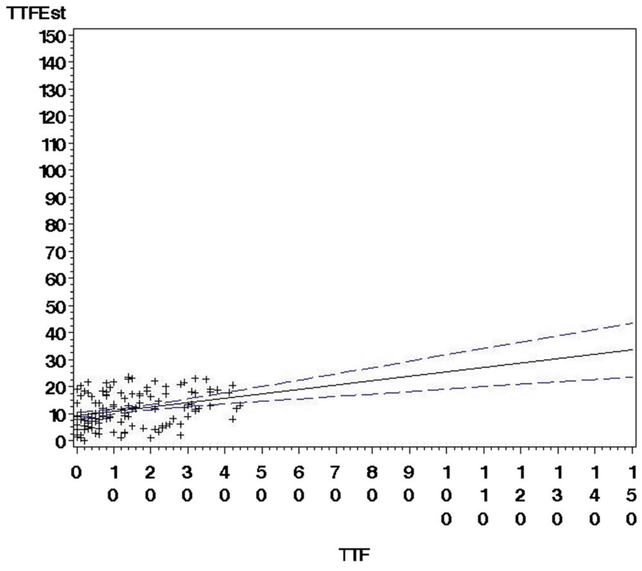


Fig. 7 Predicted TTF versus actual TTF from the proposed on-demand regression

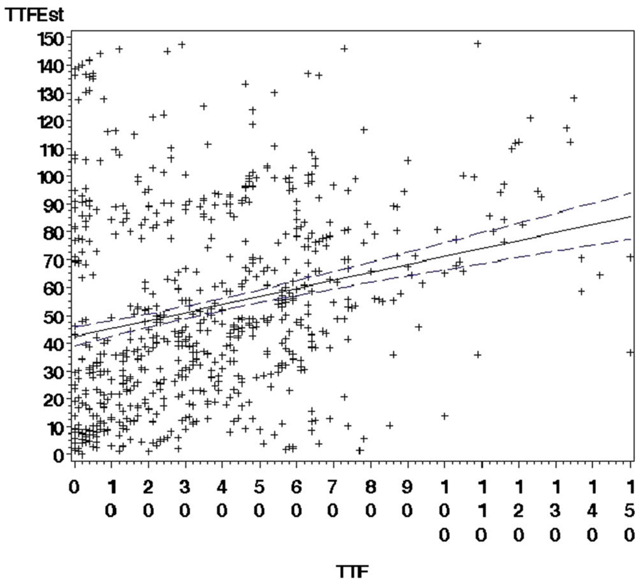


Fig. 6 Predicted TTF versus actual TTF using estimates from the regression based models only

5 Discussion

The APU prognostic application has demonstrated the applicability of the proposed on-demand regression approach to improve the precision of TTF estimations. On the other hand, a number of aspects deserve further attention.

One of the key parameters of the proposed approach is the number of clusters. In this application, we could have selected as many as 22 clusters as we had 22 occurrence of failures in the test set and that we needed a minimum of one test set in each cluster to evaluate the corresponding regression model. To decide on the optimal number of clusters for the given application and to better understand the impact of this choice, we experimented with 1, 4, 7, 10, 13, 16, 19, and 22 clusters. In each experiment, we built a regression model for each cluster, a model selector, and then ran the test data to evaluate the expected performance. As descripted in on-demand regression Section, model selector (N-class classifier) is used to select one of regression model to estimate TTF based given observation (X_{ij}). From these experiments, we observed that increasing the number of clusters consistently improves the accuracy of the local regression models but also decreases the performance of the model selector. This later observation is explained by the fact that the probability of selecting the right local regression model decreases as the number of cluster increases. The challenge is to find a trade off that would minimize the overall average error on TTF predictions. As shown in Fig. 8, 16 clusters appear to be the optimal choice in the case of the APU starter failure application. More than 16 clusters would not produce any significant benefit while less could reduce the expected precision of the estimates.

The construction of the classification-based prognostic model also involves a number of parameters including: the length of the time window around each failure used in data gathering (parameters m and n), the labeling period, and the various parameters used in the evaluation function. For the time window around each failure, we need to ensure that m, the number of observations kept before each failure, is sufficiently large to include the expected period of observable symptoms plus an equally long healthy period. Background knowledge about the various failure modes could be used to determine the expected period of observable symptoms within the sensor data. Alternatively, data visualization of key sensor measurements could be used. As for n, which determines the number of observations used after the replacement, our experiments show that it is good practice to keep it at around 10 % of m. To avoid false alerts, the labeling period (Section 3.1) should correspond to the expected period of observable symptoms. Finally, it is best to let the end users decide on the values for the parameters used in the reward function. This would help ensure that the selected models would deliver the expected benefits.

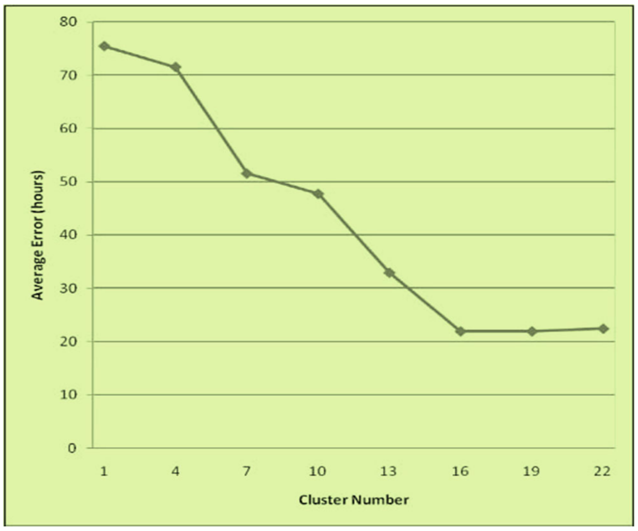


Fig. 8 Average error on TTF estimation for different cluster numbers

There are many ways to combine the results from a binary classification system and from one or more regression models. In this paper, we presented a single combination rule. We only keep the TTF_R that is less than TTF_C. Although this simple rule appears useful in improving the TTF estimations, it could negatively affect the fault detection rate. In a real-world setting, the combination rule could be adapted by taking into account, the intended usage of the alerts from the on-demand regression model, the operational constraints, and the various costs involved. For instance, if one would prefer to avoid false negative, a decision rule that replaces TTF_R by TTF_C when TTF_R is greater than TTF_C could be more appropriate. Such a rule would actually ensure that the overall approach has the same failure detection ability as a binary classification system.

It is also worth noting that the performance of the on-demand regression method could be improved by enhancing the fault detection algorithm. In this paper, we use an N-class classifier to perform this task but alternative methods could also be investigated. Actually, this should lead further research as a simple reduction in the rate of false alerts at the detection stage could have great impact on the preciseness of final TTF estimation.

In order to further validate the applicability and potential benefits of the proposed approach, we tried to apply the proposed method to estimate TTF for F404 engine No.4 Bearing prognostics. We performed preliminary experiments with the data provided from our other project, F404 engine No.4 Bearing prognostics [25, 26]. The No.4 Bearing is a critical component on the engine since its failure can result in delays, cancellation of missions, and even possible loss of an engine or aircraft. Each of these outcomes has a negative effect on the operation of the CF-18 within the Canadian Air Force. In No.4 Bearing prognostic project, our goal is to develop predictive models to predict the replacement of component from 10-year operational database collected in flight data record system and maintenance records. For this purpose we have obtained 54 replacement events and retrieved the relevant operational data for these replacements. As described in Section 3, we first built a binary classifier to identify incipient failures. We configured the automatic labeling approach so that all instances with remaining life time smaller than 50 hours are labeled as positive ("1") and all other instances as negative ("0"). As we did in PHM 2008 Challenge problem, using the positive instances, we constructed a single SMO regression model for estimating TTFs. Finally, we combined the results from the classifier and the regression model as specified. During evaluation, we simply ran the binary classifier and SMO-regression models on the testing dataset to predict the TTF for each No.4 Bearing replacement based on the given test dataset. The Table 3 shows the experimental results from two different SMO regression models.

Table 3 TTF estimation for No.4 Bearing on test data for using regression-only, classifier-only, and the on-demand regression method

Model #	Regression		Classifier		On-demand	
	mean±std	MSE	mean±std	MSE	mean±std	MSE
Model1	79.1±58.4	9670	49.2±44.4	6781	43.7±52.7	4672
Model2	74.2±43.4	6758	48.9±45.3	6749	43.6±52.6	4648

Similarly to what we did in Section 3, the results were compared in three different modes: binary classifier only, regression model only and on-demand regression. From the results, it is obvious that the TTF estimations of No.4 Bearing replacements are improved by applying the on-demand method even though we only developed one single SMO regression model. We are working on to build the multiple SMO models by performing clustering to determine the numbers of regression models. The results will be reported in the later reporter.

6 Conclusion

This paper presents a machine learning-based method developed to estimate time to failure for PHM. The paper describes the difficulties limiting the usefulness of regression for TTF estimation. In spite of these difficulties, the paper argues that regression can help improve TTF estimates using on-demand regression' which carefully integrates classification, clustering, and local regression. The paper fully describes the process to build the various predictive models involved. And it reported the experimental results from the APU prognostic application. These results demonstrate the potential of the approach for improving the preciseness of TTF estimates. Future work includes additional experiments with data from additional applications. And we will continue applying the on-demand methods to No.4 Bearing prognostics to estimate TTF for replacements by determining the numbers of local regression models through clustering experiments.

Acknowledgments Many people at the National Research Council of Canada have contributed to this work. We would also like to thank Air Canada for providing the data used in this research. This work is supported by the Natural Science Foundation (Grant No.61463031).

Acronym

ACARS	Aircraft Communications Addressing & Reporting System
APU	Auxiliary Power Unit
EM	Expectation Maximization Algorithm for clustering
J48	Decision Tree Classification Algorithm
MSE	Mean Squared Error
NN	Neural Network
PHM	Prognostic and Health Management
SMO	Sequential Minimal Optimization
SVM	Support Vector Machine
TTF	Time to Failure
WEKA	Waikato Environment for Knowledge Analysis

Notation

p	number of positive predictions
N	total number of positives made by classifiers in training dataset
$score_i$	score from the reward function for the i^{th} instance classified as positive
NbrDetected	number of detected failures
NbrofCase	total number of failures
M	number of observations kept before each failure
N	number of observations kept after each failure
Sign	sign of $\sum_{i=1}^{p} score_i$
TTF_C	TTF estimate from the classifier
TTF_R	TTF estimate from the regression model
RemainingOPH	remaining operational life of a component (in hours)
X_{ij}	The system state observation: the j^{th} instance in i^{th} time-series.

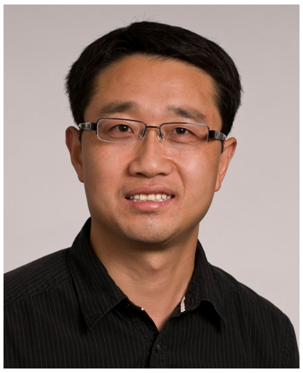
References

1. Liu K, Gebraeel NZ, Shi J (2013) A data-level fusion model for developing composite health indices for degradation modeling and prognostic analysis. IEEE Trans Autom Sci Eng 10(3):652–664
2. You MY, Li L, Meng G, Ni J (2010) Cost-effective updated sequential predictive maintenance policy for continuously monitored degrading systems. IEEE Trans Autom Sci Eng 7(2):581–265
3. Compare M, Zio E (2014) Predictive maintenance by risk sensitive particle filtering. IEEE Trans Reliab 63(1):134–143
4. Grall A et al. (2002) Continuous time predictive maintenance scheduling for a deteriorating system. IEEE Trans Reliab 51(2):141–150
5. Feng L, Wang H, Si X, Zou H (2013) A State-Space-Based prognostic model for hidden and Age-Dependent nonlinear degradation process. IEEE Trans Autom Sci Eng 10(4):1072–1085
6. Schouten FA, Wartenhorst P Time to failure, time to repair and availability of a two unit standby system with markovian degrading units. Technical report BS-R9007, Center for Mathematics and Computer
7. Gebraeel N, Lawley M, Liu R, Parmeshwaran V Residual life prediction from vibration-based degradation signals: a neural network approach. IEEE Trans Ind Electron 51(3)

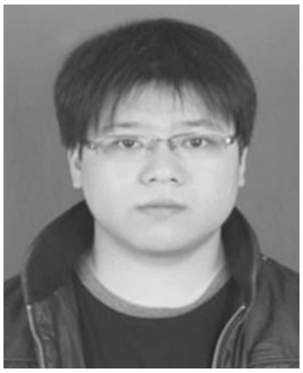
8. Grottke M, Trivedi K (2005) On a method for mending time to failure distributions. In: Proceedings of international conference on dependable systems and networks, Los Alamitos, USA, pp 560–569
9. Promellec B, Riant I, Tessier-Lescourret C (2006) Precise lifetime prediction using demarcation energy approximation for distributed activation energy reaction. J Phys Condens Matter 18(2006):2199–2216
10. Sornette D, Andersen JV (2006) Optimal prediction of time-to-failure from information revealed by damage. Europhys Lett 74(5):778–784
11. L'etourneau S, Yang C, Drummond C, Scarlett E, Valds J, Zaluski M (2005) A domain independent data mining methodology for prognostics. In: Proceedings of the 59th meeting of the society for machine failure prevention technology, MFPT '05, Virginia beach, Virginia, USA
12. L'etourneau S, Yang C, Liu Z (2008) Improving preciseness of time to failure predictions: application to APU starter. In: Proceedings of the 1st international conference on prognostics and health management, Denver, Colorado, USA
13. Yang C, L'etourneau S (2005) Learning to predict train wheel failures. In: Proceedings of the 11th ACM SIGKDD international conference on knowledge discovery and data mining (KDD2005), Chicago, USA, pp 516–525
14. L'etourneau S, Famili AF, Matwin S (1999) Data mining for prediction of aircraft component replacement. IEEE Intelligent Systems and their Applications 14(6):59–66
15. Hall M (2000) Correlation-based feature selection for discrete and numeric class machine learning. In: Proceedings of the 17th international conference on machine learning, A, pp 359–366
16. Kira K, Rendell L (1992) A practical approach to feature selection. In: Proceedings of the 9th international conference on machine learning, pp 249–256
17. Dietterich T (2000) An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting and randomization. IEEE Intelligent Systems and their Applications 40:139–158
18. Tsoumakas IKG, Blahavas I (2004) Effective voting of heterogeneous classifiers. In: Proceedings of the 15th european conference on machine learning (MCML2004), pp 465–476
19. Dzeroski S, Zenko B (2002) Stacking with multi-response model trees. In: The proceeding of international workshop in multiple classifier systems (MCS2002), pp 201–211
20. Witten IH, Frank E (1999) Data mining: practical machine learning tools and techniques with java implementations. Morgan Kaufmann
21. Dzeroski S, Zenko B (2004) Is combining classifiers with stacking better than selecting the best one? Mach Learn 54:254–273
22. Compare M, Zio E (2014) Predictive maintenance by risk sensitive particle filtering. IEEE Trans Reliab 63(1):134–143
23. You MY, Liu F, Wang W, Meng G (2010) Statistically planned and individually monitored predictive maintenance management for continuously monitored degrading systems. IEEE Trans Reliab 59(4):744–753
24. Park KS (1993) Condition-based predictive maintenance by multiple logistic function. IEEE Trans Reliab 42(4):556–560
25. Zaluski M, Létourneau S, Bird J, Yang C (2011) Developing data mining-based prognostic models for Cf-18 aircraft. Journal of Engine Gas Turbines Power 133(10)
26. Zaluski M, Létourneau S, Yang C (2009) Data Mining-based Prognostics for CF18 Aircraft Components. In: 13th CASI Aeronautics Conference, Kanata, Canada, May 5–7
27. Leblanca M, Tibshiranib R (1996) Combining estimates in regression and classification. J Am Stat Assoc 91(436):1641–1650

Dr. Chunsheng Yang is a Senior Research Officer at the National Research Council Canada. He is interested in data mining, machine learning, prognostic health management(PHM), reasoning technologies such as case-based reasoning, rule-based reasoning and hybrid reasoning, multi-agent systems, and distributed computing. He received an Hons. B.Sc. in Electronic Engineering from Harbin Engineering University, China, an M.Sc. in computer science from Shanghai Jiao Tong University, China, and a Ph.D. from National Hiroshima University, Japan. He worked with Fujitsu Inc., Japan, as a Senior Engineer and engaged on the development of ATM Network Management Systems. He was an Assistant Professor at Shanghai Jiao Tong University from 1986 to 1990 working on Hypercube Distributed Computer Systems. Dr. Yang has been the author for 144 papers, book chapters, and invited talks published in the referred journals and conference proceedings he was a Program Co-Chair for the 17th International Conference on Industry and Engineering Applications of Artificial Intelligence and Expert Systems, and the 20th IEEE International Conference on Computer Supported Cooperative Work in Design (CSCWD 2016). Dr. Yang is a guest editor for the International Journal of Applied Intelligence. He has served Program Committees for many conferences and institutions, and has been a reviewer for many conferences, journals, and organizations, including Applied Intelligence, NSERC, IEEE Trans., ACM KDD, PAKDD, AAMAS, IEA/AIE.

Dr. Sylvain Letourneau was a Research Officer at the National Research Council Canada, Ottawa, Ontario, Canada.



Dr. Jie Liu obtained his B.Eng. in Electronics and Precision Engineering from Tianjin University (China) in 1998, his M.Sc. in Control Engineering from Lakehead University (Canada) in 2005, and his Ph.D. in Mechanical Engineering from the University of Waterloo (Canada) in 2008. Prior to his graduate studies, he worked at Invensys Controls Inc. as a Product Engineer and then Production Manager for three years. Before joining Carleton, he worked as an Postdoctoral Fellow in the Dept. of Mechanical Engineering at UC Berkeley for one and half years. Dr. Liu is currently an Associate Professor in the Dept. of Mechanical & Aerospace Engineering at Carleton University, Ottawa, Canada. He has been engaged in interdisciplinary research in the areas of machine condition monitoring and failure prognostics, intelligent mechatronic systems, vibration control, intelligent systems, and power storage for about thirteen years. His research results have been disseminated through over 30 journal publications and 20 conference papers. Dr. Liu is a Steering Committee Member of Annual IEEE PHM Conferences, an IEEE Senior Member, and a registered Professional Engineer in Ontario, Canada.



Qiangqiang Cheng is an Assistant Professor with Nanchang University of Aeronautics and Astronautics, Jiangxi, China. He is also a Ph.D. Candidate at School of Information Technology, Nanchang University. He received the M.Sc. degrees in Precision Instrument and Machinery from Nanchang Hangkong University, Nanchang, China, 2011. His research interests include computer graphics, virtual reality, and surgery simulation.

Dr. Yubin Yang is currently a Professor with the State Key Laboratory for Novel Software Technology, Nanjing University. His current research interests include machine learning, artificial intelligence, media computing and large-scale data mining. He received the B.Sc. degree in computer science from Wuhan Technical University of Surveying and Mapping, Wuhan, China, in 1997, and the M.Sc. and Ph.D. degrees in computer science from Nanjing University, Nanjing, China, in 2000 and 2003, respectively. He participated in collaborative research at The Chinese University of Hong Kong (CUHK) in 2003-2005, and at University of New South Wales at Australian Defenses Force Academy (UNSW@ADFA) in 2005-2006.