Research Article

Integrating Feature Engineering with Deep Learning to Conduct Diagnostic and Predictive Analytics for Turbofan Engines

Chih-Hsuan Wang and Ji-Yu Liu

Department of Industrial Engineering & Management, National Yang Ming Chiao Tung University, 1001 University Rd., Hsinchu 30013, Taiwan

Correspondence should be addressed to Chih-Hsuan Wang; chihswang@nycu.edu.tw

Received 5 August 2022; Revised 19 August 2022; Accepted 23 August 2022; Published 26 September 2022

Copyright © 2022 Chih-Hsuan Wang and Ji-Yu Liu. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The prediction of remaining useful life (RUL) is a critical issue in many areas, such as aircrafts, ships, automobile, and facility equipment. Although numerous methods have been presented to address this issue, most of them do not consider the impacts of feature engineering. Typical techniques include the wrapper approach (using metaheuristics), the embedded approach (using machine learning), and the extraction approach (using component analysis). For simplicity, this research considers feature selection and feature extraction. In particular, principal component analysis (PCA) and sliced inverse regression (SIR) are adopted in feature extraction while stepwise regression (SR), multivariate adaptive regression splines (MARS), random forest (RF), and extreme gradient boosting (XGB) are used in feature selection. In feature selection, the original 15 sensors can be reduced to only four sensors that accumulate more than 80% degrees of importance and not seriously decrease the predictive performances. In feature extraction, only the top three principal components can account for more than 80% variances of original 15 sensors. Further, PCA combined with RF is more recommended than PCA and CNN (convolutional neural network) because it can achieve satisfactory performances without incurring tedious computation.

1. Introduction

Fast developments in IT technologies including wireless sensors, big data, artificial intelligence, and cloud computing have boosted as well as reshaped numerous areas in smart manufacturing, such as product-service system (PSS), predictive maintenance, cyber-physical system (CPS), and digital twins [1, 2]. One of the most famous examples is Rolls-Royce; it has successfully shifted from an engine manufacturer to an engine service provider and now, it claims its core business is engine health management (EHM). In the perspective of product lifecycle, Rolls-Royce provides a total solution and full diagnosis starting from design, manufacturing, and operation, to maintenance. In practice, EHM describes and transfers sensor signals from an engine on an aircraft to an operational center on the ground, which can be used to record and monitor the performance of an engine to ensure its reliability [3]. Although EHM brings about a new era in PSS, it still has thousands of parameters to monitor and needs to respond to requests from an operational center by sending back hundreds of hours of information specifically tailored to that request. These procedures inspire the origin of this research: the impacts and benefits of feature engineering on predictive maintenance.

Clearly, IT-enabled PSS helps EHM build an integrated interface between “product” and “service” to put these two terms together rather than separately. Typical benefits include high value-added business models, fast customer responses, and operational efficiencies. To achieve successful EHM, both preventive maintenance (PvM) and predictive maintenance (PdM) are designed to increase the reliability of the equipment. In reality, PvM is similar to a form of “regular” as well as “scheduled” maintenance without considering equipment conditions. In contrast, PdM relies on IT to monitor equipment condition and focus on the critical parameters. Clearly, PdM is performed only when
“needed” and hence it can significantly reduce labor and material costs. As mentioned, the integration of IT tools, such as the collection of sensor signals, data analytics, diagnosis, computing, prediction, and prescription, determine the eventual performances in PdM [4, 5]. The most important technique is the prediction of remaining useful life (RUL) because it assists practitioners in analyzing engine conditions and scheduling maintenance or replacement in advance [6, 7].

Recently, the fourth industrial revolution, Industry 4.0, brings about a new trend in smart manufacturing by extending computer automation to a cyber-physical environment composed of Internet of things (IOT), big data analytics, cloud computing, cognitive reasoning, and wireless sensors. Clearly, Industry 4.0 creates the paradigm of “smart factory,” in which IT-based cyber systems can collect data in production or quality control and monitor physical processes to make data-driven and decentralized decisions [8, 9]. Further, this paradigm shift significantly impacts on the developments of EHM and predictive maintenance. Despite numerous studies that have been presented, most of them did not address the impacts of feature engineering on the prediction of the RUL.

To the best of our knowledge, typical ways in feature engineering include filtering, extraction, wrapper, and embedded techniques [10, 11]. Specifically, filtering achieves dimension reduction without considering the interrelationships between input features. However, input features like sensors are usually interdependent. Thus, extraction focuses on component analysis of input features without considering their dependences on the response. In contrast, the wrapper and embedded approaches simultaneously conduct dimension reduction as well as predict the response (forecasting). This research attempts to present an integrated framework to highlight the impacts of feature engineering. For instance, what sensors should be firstly monitored and why? Which algorithms should be adopted to balance the trade-offs between forecasting performances and computational time? Based on the recognized key performance indicators (KPIs) or extracted principal components (PCs), the prediction of the RUL can be thoroughly assessed to help practitioners accomplish the following goals:

(i) What sensors are most effective to predict the RUL of turbofan engines (using feature selection)?

(ii) What components are most effective to predict the RUL of turbofan engines (using feature extraction)?

(iii) What are the best combinations of feature-engineering techniques with machine learning or deep learning algorithms to conduct RUL forecasting?

The rest of the study is organized as follows. Section 2 overviews feature engineering and predictive maintenance. Section 3 details the proposed framework. Experimental results are presented in Section 4. Discussions and insights are in Section 5. Conclusions and future work are shown in Section 6.

2. Overview

Rapid advances in information technologies (IT), such as big data and artificial intelligence, have brought about numerous applications in smart manufacturing and predictive maintenance. In practice, IT needs to be integrated with domain knowledge to conduct data-driven decision-making. For example, the following questions are frequently asked: which sensors (input features) should be selected? What algorithm (clustering, association, anomaly detection, classification, regression, and dimension reduction) is suitable to solve specific problems? What values or benefits can be obtained after implementation? Generally, clustering, association, and anomaly detection are unsupervised schemes (only related to input features), while classification and regression are supervised schemes (related to the response). The most interesting part is dimension reduction or the so-called feature engineering. It can be achieved in an unsupervised manner (component extraction) or a supervised way (filtering, wrapper, and embedded). All the above-mentioned schemes have been well integrated to solve complex problems in smart manufacturing (SM) and predictive maintenance (PdM).

SM is a systematic approach, which applies software to bring data together from a firm’s manufacturing execution systems and pass data between firm-level systems and shop-floor plants. Most manufacturing data are well structured and can be collected from multiple sensors to conduct data-driven decision-making. In simple words, huge amounts of manufacturing data can be transformed into real knowledge to achieve cost reduction, yield improvement, productivity enhancement, and fast delivery. In contrast, PdM is designed to help practitioners assess health conditions of in-service machinery equipment and prevent them from fatal failures. Today, cheap wireless sensors are ubiquitous to collect process data and signals at anytime and anywhere. Hence, SM and PdM become more feasible, easier, and cheaper than before. In reality, the following issues are critical but rarely addressed together: (1) how to identify the causality between performance indicators and the response (yield, productivity, cycle time, delivery, and RUL)? (2) How to apply dimension reduction to select key performance indicators or extract representative components? (3) How to assess the degree of improvement considering the impacts of feature engineering? To highlight research contributions, Table 1 compares this research to past studies. As indicated, lots of publications did not consider feature selection or feature extraction. Li et al. [16], Zhang et al. [9], Li et al. [10], Xiang et al. [20], and Li et al. [21] slightly outperform our presented approaches in terms of forecasting errors (RMSE or MAE in equations (1) and (2)). However, most of them did not show the details of hyperparameters for various models. Some authors [9, 10, 21] adopted two-stage deep-learning methods to achieve better performances. We do not consider these hybrid architectures because it takes tedious computation and not appropriate for fast diagnosis. In brief, the presented approach performs effectively but its computation is relatively efficient. In particular, this research aims at the best combination of feature engineering (PCA) with machine
learning (RF) or deep learning (CNN) and making a comprehensive comparison in feature selection and feature extraction.

%AB: adaptive boosting, ANOVA: analysis of variance, ARIMA: autoregressive integrated moving average, BPN: backpropagation network, CAE: convolutional autoencoder, CART: classification and regression tree, CNN: convolutional neural network, DNN: deep neural network, DWT: discrete wavelet transform, GB: gradient boosting, GRU: gate recurrent unit, KNN: K nearest neighbor, LSTM: long-short term memory, MARS: multivariate adaptive regression splines, MLR: multiple linear regression, PCA:
principal component analysis, RF: random forest, RNN: recurrent neural network, SIR: sliced inverse regression, SVM: support vector machine, TCN: temporal convolutional neural network, XGB: extreme gradient boosting.

2.1. Feature Engineering. This research particularly addressed the impacts of feature engineering on the prediction of the RUL for turbo engines. As mentioned, feature engineering consists of filtering, wrapper, embedded, and extraction schemes. Among them, principal component analysis (PCA) extracts the most representative components that can sufficiently account for the variances of the original features. In contrast, sliced inverse regression (SIR) integrates the concept of PCA considering the impacts of the predictors on the response. Filtering methods consider the interrelationships between individual features and the outcome, such as correlation coefficient, mutual information, and F score. Mutual information originates from information entropy and uses conditional probabilities to assess the discriminant power of input features. F score originates from analysis of variance (ANOVA) and calculates the between-group variances divided by the within-group variances. Filtering approach is excluded in this research because it does not consider the interrelationships between the predictors (sensor signals).

In feature engineering, both wrapper and embedded methods consider the classifiers or the regressors to perform feature selection. The wrapper approach, such as metaheuristics algorithms, needs to search for the nearly optimal subsets that can achieve satisfactory performances. Typical schemes include genetic algorithm (GA), particle swarm optimization (PSO), ant colony optimization (ACO), whale optimization algorithm (WHA), etc. In contrast, the embedded approach directly prioritizes the degrees of importance of input features. Due to limited lengths, this research considers the embedded approach in feature selection. Specifically, stepwise regression (SR), multivariate adaptive regression splines (MARS), random forest (RF), and extreme gradient boosting (XGB) are adopted to prioritize the degrees of importance of input features.

2.2. Predictive Maintenance. The prediction of remaining useful life (RUL) plays a key role in PdM because it can detect anomaly (internal failure) in advance to avoid huge losses (external failure). In reality, it can be viewed as a general regression problem. Based on experienced domain experts, representative sensor signals are collected as input features to conduct anomaly detection, real-time monitoring, and predictive analytics. However, it is difficult to know the “real” RUL because it is infeasible to observe a machinery equipment from the start to the end. Thus, distribution-based probability models (Wiener process and Weibull distribution) are simulated to estimate the true RUL [22]. A classical burn-in test requires an equipment to operate in an extremely tough environment (high temperature, high pressure, etc.) to speed up the process of degradation and effectively shorten its lifespan. Based on the shortened life in a burn-in test, the actual life operated in a normal environment can be estimated accordingly. Heimes [23] used the threshold to separate the RUL into the level segment (before the threshold) and the decline segment (after the threshold). Recently, machine learning and deep learning are applied to construct causal models between sensors and the RUL.

Typical machine learning includes multivariate adaptive regression splines (MARS), random forest (RF), extreme gradient boosting (XGB), and support vector machine (SVM). Among them, MARS, RF, and XGB can derive the importance of input features. Deep-learning regressors include deep neural network (DNN), convolutional neural network (CNN), recurrent neural network (RNN), long-short term memory (LSTM), and gated recurrent unit (GRU). DNN is the most classical but it has problems like gradient vanishing or gradient explosion. CNN is very popular in pattern recognition (image processing), while RNN is powerful in speech recognition (temporal signals). LSTM can be viewed as an improved version of RNN while GRU is a simplified version of LSTM. Although deep-learning regressors are powerful, they cannot identify the degrees of importance of input features. Besides, they require lots of samples to optimize network topologies and associated hyperparameters.

3. Methodologies

To combine feature engineering with the prediction of RUL, Figure 1 details the presented techniques: (1) turbofan engines with degradation are collected from the NASA dataset (FD001), in which sensors are treated as input features to predict the RUL, (2) feature extraction including principal component analysis (PCA) and sliced inverse regression (SIR) and feature selection including stepwise regression (SR), multivariate adaptive regression splines (MARS) random forest (RF), and extreme gradient boosting (XGB) are applied to conduct dimension reduction, (3) various regressors are tested with feature-engineering techniques to find the best combination, and (4) the impacts of feature engineering are thoroughly assessed to generate managerial insights.

Without loss of generality, machine learning includes multivariate regression splines (MARS), random forest (RF), extreme gradient boosting (XGB), support vector machine (SVM), and deep neural network (DNN), while deep learning contains recurrent neural network (RNN), gated recurrent unit (GRU), long-short term memory (LSTM), and convolutional neural network (CNN). In addition, three quantitative metrics including root mean square error (RMSE), mean absolute error (MAE), and mean absolute arctangent percentage error (MAAPE) are used to measure forecasting errors [24]:

\[
\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} e_i^2},
\]

\[
\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |e_i|,
\]
Gathering data samples of turbofan engines with degradation from the NASA turbofan dataset (FD001)

Applying feature selection and feature extraction to conduct dimension reduction

Using machine learning (RF, XGB, SVM, DNN) and deep learning (RNN, LSTM, GRU, CNN) to predict

Assessing the impacts of feature engineering on the prediction of RUL to justify research validity

Figure 1: Proposed research techniques.

\[ \text{MAAPE} = \frac{1}{n} \sum_{i=1}^{n} \arctan \left( \frac{e_i}{|y_i|} \right), \]

where \( n \) denotes the number of observations and \( e_i = F_i - y_i \) is an error between a predicted value \( (F_i) \) and the real data \((y_i)\).

### 3.1. Feature Extraction

Principal component analysis (PCA) is a dimension-reduction technique that can be used to reduce a large set of variables to a small set that still contains most of the information in the large set. PCA is a mathematical procedure that transforms a big number of possibly correlated variables into a small number of uncorrelated variables called principal components (PCs). The first principal component accounts for the most information (the largest proportion of variance in the original data) and the succeeding components sequentially account for as much of the remaining variability as possible.

Suppose the original variables, \( X = [x_1, x_2, \ldots, x_p]^T \), is a \( p \)-dimensional vector, its covariance matrix,

\[
\Sigma = \begin{bmatrix}
\sigma_1^2 & \sigma_1 \sigma_2 & \cdots & \sigma_1 \sigma_p \\
\sigma_2 \sigma_1 & \sigma_2^2 & \cdots & \sigma_2 \sigma_p \\
\vdots & \ddots & \ddots & \vdots \\
\sigma_p \sigma_1 & \sigma_p \sigma_2 & \cdots & \sigma_p^2
\end{bmatrix},
\]

has the diagonal elements meaning the variances of the original variables. If we linearly transform the original variables to the new components with the weights, \( w_i = [w_{i1}, w_{i2}, \ldots, w_{ip}]^T \), the following characteristics:

\[ \text{Var}(y_i) = w_i^T \Sigma w_i \]

\[ \text{Cov}(y_i, y_j) = w_i^T \Sigma w_j \]

can be obtained. Mathematically, we need to find a set of normalized weights \( (w_1^T w_1 = 1) \) to generate the principal components. By using linear algebra, the variances of new components are the eigenvalues of the covariance matrix and the normalized weights are just the corresponding eigenvectors. Therefore, the following property is found:

\[ \text{Var}(y_1) = \lambda_1 > \text{Var}(y_2) = \lambda_2 > \cdots \text{Var}(y_p) = \lambda_p, \]

where \( \lambda_1, \lambda_2, \ldots, \lambda_p \) represents the eigenvalues in a descending order. Intuitively, we can extract the top \( m \) principal components \( \sum_{i=1}^{m} \lambda_i / (\lambda_1 + \lambda_2 + \cdots + \lambda_p) \) that can account for most of the original information to accomplish dimension reduction.

Clearly, PCA performs feature extraction in an unsupervised manner without considering the relationships between the input variables and the response outcome. Thus, this research also considers a supervised version of PCA, and it is sliced inverse regression (SIR). Following similar definitions in PCA, we consider a univariate response variable \( Y \) with the original \( p \)-dimensional variables \( X = [x_1, x_2, \ldots, x_p]^T \) with \( \mu = E(X) \) and \( \Sigma = V(X) \). Let \( \beta = [\beta_1, \beta_2, \ldots, \beta_k]^T \) be a \( p \times K \) matrix (with \( K \leq p \)), where \( \beta_k \) are unknown \( p \)-dimensional vectors assumed to be linearly independent \([25]\). The semiparametric regression model, \( Y = g(\beta^T X) \), is an attractive dimension reduction approach to model the effect of \( X \) on \( Y \) with an unknown arbitrary link function \( g \) and given \( Y \) is independent of \( X \). Thus, we can replace \( X \in R^p \) by \( \beta^T X \in R^k \) without loss of information on the regression of \( Y \) on \( X \).

To estimate the subspace of effective dimension reduction (EDR), the basic concept of SIR is to reverse the role of \( Y \) and \( X \), the covariate \( X \) is regressed on the response variable \( Y \), and the linearity assumption on the distribution of \( X \), \( E[\beta^T X|Y] \). Here, \( B = [b_1, b_2, \ldots, b_k]^T \) denotes an estimated basis of the EDR subspace and it is used to estimate the mapping function \( g \). Specifically, the centered inverse regression curve is spanned by the columns of the \( p \times K \) matrix \( \Sigma \beta \). The eigenvectors associated with the largest \( K \) eigenvalues of the symmetric matrix \( \Sigma^{-1} M, M = \text{Var}[E[X|S(Y)]] \), are EDR directions (S is a slicing transformation to discretize the response \( Y \)).

### 3.2. Machine Learning

Without loss of generality, typical regressors include multivariate adaptive regression splines (MARS), random forest (RF), extreme gradient boosting (XGB), and support vector machine (SVM). Among them, MARS, RF, and XGB are used in feature selection because they can identify key performance indicators (KPIs) as follows:

\[
f(x) = a_0 + \sum_{m=1}^{M} a_m \prod_{k=1}^{K_m} [S_{km}(x_{km} - t_{km})]
\]

where \( a_0 \) and \( a_m \) are intercept and regression coefficients, \( M \) is the number of bases, \( K_m \) is the number of splits for the \( m \)th basis, \( S_{km} \) takes values of either 1 or 0 to indicate the
right or the left step function, \( x_{k,m} \) are input variables, and \( t_{k,m} \) are “knot” locations in each interval [26].

In MARS, general cross validation (GCV) can be used to prioritize the degrees of importance of the predictors. Based on quadratic programming, SVM uses “kernel” functions to convert the regression problem from the original space into the high-dimensional feature space. Therefore, the “nonlinear” fitting in the input space has been transformed into “linear” fitting in the high-dimensional space and the tolerance, epsilon, means limited “outliers” are allowed to be outside the cylinder to avoid overfitting (good training performance but poor test performance). The most common kernel is radial basis function (Gaussian kernel).

RF and XGB originating from decision trees like classification and regression tree (CART) considers “Gini index” as information impurity to recursively split tree nodes, grow a tree from the top root to the bottom leaves, and characterize the entire system. This process is called ensemble learning because it has been theoretically justified to significantly outperform a single tree. In RF, the meaning of “random” includes random selection of samples and partial features (predictors) to fit a single tree. The objective is keeping all the trees sufficiently dissimilar to conduct aggregate decision making. In contrast, “boosting” implies different probability distributions for different samples. Generally, data samples that are hard to predict will increase their probabilities to appear and vice versa. XGB is an enhanced version of gradient boosting. Based on the so-called loss functions, XGB searches for the best moves (gradient decent) to minimize forecasting errors and achieve the goal of curve fitting (nonlinear approximation).

3.3. Deep Learning. Deep neural network (DNN) is one of the most classical artificial neural networks. It has a serious problem, vanishing gradients, when feeding back an error signal to adjust the links (weights) between the neurons. Based on gradient boosting, specific parameters need to be optimized, such as the number of hidden layers and associated neurons, drop-out rate (some paths are randomly selected to remove), learning rate, activation function (hyperbolic tangent, sigmoid, and ReLu), and optimizer (stochastic decent, gradient decent, AdaDelta, AdaGrad,). For clarity, Table 2 gives a brief description for associated hyper-parameters in machine learning and deep learning [29]. The whole process is like trial-and-error and no “optimal” combinations can be guaranteed. Thus, past publications [18, 23, 32] considered metaheuristics algorithms to fast search for “relatively good” solutions for these parameters.

4. Illustrated Examples

The NASA “C-MAPSS (Commercial Modular Aero-Propulsion System Simulation)” dataset, FD001, composed of 33,727 simulated samples with 21 sensors is used. It contains lifetime data of 100 turbofan engines assessed in the same operation condition. All these engines were operated in a normal condition at the beginning with different degrees of deterioration and then gradually degraded with the operation cycles. Following the cited references, the calculation of the RUL is based on the maximal cycle minus the elapsed time since engine running. Intuitively, the longer the usage, the shorter the remaining useful life (RUL). Table 3 illustrates the details of the dataset. Specifically, temperature is measured in Rankine’s scale. Note that sensors 1, 5, 10, 16, 18, and 19 are removed because they do not vary with different engines. Possible reasons are due to the fact that these sensors represent the same test condition.

In this research, the impacts of feature engineering are thoroughly explored: feature selection using stepwise regression and machine learning and feature extraction using PCA and SIR. Meanwhile, various regressors consisting of machine learning (RF, XGB, SVM, DNN) and deep learning (RNN, LSTM, GRU, CNN) are combined with PCA and SIR to find the best performed combination. The prediction of the RUL is actually a regression problem: sensors or reduced features are treated as input features while the RUL is the response. Since the original dataset does not define the true RUL, Heimes [23] and the most cited references [9, 14–18, 20, 21, 33] used the threshold, 130, to separate the RUL into two segments: level (before cycle 130) and decline (after cycle 130). Mathematically, the RUL for engine \( n \) at time \( t \) is defined as, Max\((\text{Cycle}_{130}) – \text{Cycle}_{ct}\), the maximal cycle minus the elapsed time since engine running. The maximal cycle is set by 130 in this research.

For clarity, Table 4 details the hyperparameters needed to optimize for machine learning and deep learning. In reality, it is a trial-and-error process heavily reliant on experienced users. Grid search and Bayesian search are common ways to shorten computational time. Prior to applying feature engineering, all sensors used to predict the RUL and its
performances are shown in Table 5. Specifically, 20,631 samples are used for training while the remaining 13,096 samples are for testing. For clarity, computer architecture is reported as follows: CPU is AMD Ryzen 9 3900X 12-Core Processor, GPU is NVIDIA GeForce GTX1660, and RAM is 32 GB. In terms of MAAPE, the best performed regressors in the training set include RF and XGB while deep learning algorithms (DNN, RNN, LSTM, GRU, and CNN) are comparable in the test set. In terms of running time, deep-learning algorithms generally require much longer computation in the training set because they need to optimize network topologies and associated hyperparameters (batch size, learning rate, drop-out, hidden layers, hidden neurons, optimizer, activation function, etc.). However, the most surprising regressor is XGB because it also requires tedious computation to search for appropriate hyperparameters. Thus, it is not recommended in this study. The predictive performances for machine learning and deep learning do not differ a lot in the test set. Due to the so-called black-box property and tedious computation, bidirection deep learning or two-stage architectures are not considered in this research.

4.1. Feature Selection Using Stepwise Regression and Machine Learning. To conduct supervised embedded feature selection, four methods are adopted: stepwise regression, MARS, RF, and XGB. These methods can simultaneously perform feature selection and conduct forecasting. As indicated in Table 6, sensor 11 (ratio of fuel flow to static pressure at HPC outlet), sensor 9 (engine pressure ratio), sensor 12 (corrected fan speed), and sensor 4 (pressure at fan inlet) are commonly identified as key performance indicators (KPIs). Here, SR uses AIC (Akaike information criterion) or BIC (Bayesian information criterion), MARS uses general cross validation \cite{26}, RF and XGB use Gini index defined in information theory to derive the degrees of importance of input sensors. Note that, the top four sensors account for almost 85% cumulative degrees of importance and thus these sensors should be monitored in a real-time way. To test the validity of the identified sensors, their performances in the prediction of the RUL are shown in Table 7. Machine learning (RF, XGB, SVM, and DNN) performs better in the training set while deep learning (RNN, LSTM, GRU, and CNN) performs better in the test set. Compared to Table 5, the performances based on the identified four “KPIs” are on average worse than considering all 15 sensors.

4.2. Feature Extraction Using Component Analysis and Sliced Inverse Regression. To justify research validity, principal component analysis (PCA) and sliced inverse regression (SIR) are used in feature extraction. PCA is implemented in an unsupervised manner without considering the RUL, while SIR is a supervised version because it simultaneously conducts feature extraction and regression. To determine the appropriate number of extracted components, cumulative proportions are listed in Table 8. The eigenvalue of each principal component (PC) explains the variances of the original sensors. Since the top three PCs, PC1, PC2, and PC3, can account for more than 80% cumulative variances, they are extracted as the predictors.

| Table 2: Brief comparison in machine learning and deep learning. |
|---------------|-----------------------------------------------|
| **Machine learning** | **Main adjustable hyper-parameters** |
| MARS | Degree of nonlinearity (number of knots), cost penalty |
| RF, XGB | Minimal samples to split a node, maximal depth, number of trees, learning rate, split ratio between training and testing |
| SVM | Cost penalty, spread of Gaussian kernels (gamma), width of tolerance (epsilon) |
| CNN, DNN, RNN, LSTM, GRU, CNN | Number of hidden layers, number of neurons, activation functions, optimizers, drop-out rates, learning rates, dense layers, initializers |

| Table 3: Definitions of input features and RUL. |
|----------------|-----------------|-------|
| **Features** | **Definitions** | **Unit** |
| Sensor_2 | Total temperature at high-pressure compressor (HPC) outlet | °R |
| Sensor_3 | Total temperature at low-pressure turbine (LPT) outlet | °R |
| Sensor_4 | Pressure at fan inlet | psia |
| Sensor_6 | Total pressure at HPC outlet | psia |
| Sensor_7 | Physical fan speed | rpm |
| Sensor_8 | Physical core speed | rpm |
| Sensor_9 | Engine pressure ratio (P50/P2) | — |
| Sensor_11 | Ratio of fuel flow to Ps30 | pps/psi |
| Sensor_12 | Corrected fan speed | rpm |
| Sensor_13 | Corrected core speed | rpm |
| Sensor_14 | Bypass ratio | — |
| Sensor_15 | Burner fuel-air ratio | — |
| Sensor_17 | Total temperature at fan inlet | °R |
| Sensor_20 | High-pressure turbine (HPT) coolant bleed | lbm/s |
| Sensor_21 | LPT coolant bleed | lbm/s |
| Response | Remaining useful life (RUL) | Cycle |
Further, the relationships between the extracted components and the original sensors are indicated in Table 9. The dependences of the first component (PC1) that accounts for more than 60% total variances seem to be quite diverse: it is closely related to sensor 4, sensor 11, and sensor 12 that are three of the four KPIs shown in Table 6. In contrast, the second component (PC2) is highly related to sensor 9 (engine pressure ratio) and sensor 11 (ratio of fuel flow to static pressure at HPC outlet) and the third component (PC3) is strongly associated with sensor 6 (total pressure at HPC outlet). Similarly, the top three directions (DIR1, DIR2, and DIR3) in the SIR can account for almost 99% variances and thus they are treated as the predictors.

Finally, based on PCA and SIR, Tables 10 and 11 demonstrate the predictive performances for the RUL. Apparently, PCA outperforms SIR in both training set and test set. For PCA, deep learning performs better than machine learning while machine learning and deep learning are comparable in SIR. Moreover, the performances based on PCA are almost equivalent to using all sensors (Table 5). Hence, in predicting the RUL, PCA is recommended in dimension reduction. To visualize the predictive results using PCA, Figure 2 (PCA and machine learning) and Figure 3 (PCA and deep learning) demonstrate all 100 engines that are mixed together. For simplicity, all engines are operated in the last cycle. The horizontal axis represents the IDs of engines.
testing engines while the vertical axis is the RUL. It is clear to see the RUL for all engines ranging between cycle 0 and cycle 140. Intuitively, the higher degrees of finesses between the actual RUL and the predictive values, the smaller errors are listed in Table 10. Similar to Table 5, Table 10 demonstrates that RF requires the least computation in the training set while its predictive performance is comparable to other schemes in the test set.

Without loss of generality, representative engines, such as engine 62 (beginning phase), engine 49 (medium phase), and engine 81 (declining phase), are selected and tested at different phases for a single-sample test. Due to limited lengths, only engine 49 is visualized. Figures 4 and 5 respectively demonstrate the predictive results combining machine learning (RF, XGB, SVM, and DNN) and combining deep learning (RNN, LSTM, GRU, and CNN) for a
single sample. Clearly, the true RUL presents a two-stage linear pattern (level + decline) because it is assumed to be a constant (no decline) before cycle 130 and linearly declining after cycle 130. Here, the horizontal axis represents the elapsed time in running while the vertical axis is the RUL. Surprisingly, machine learning seems to perform better than deep learning in the prediction of engine 49. The true RUL begins to decline since cycle 199 but all models predict engine deterioration at earlier cycles. This implies warning alerts may be sent earlier than engine deterioration really starts. In terms of quality control, type 1 error (false alarms) can be tolerated because it does not incur huge losses in production lines. In contrast, type 2 error (miss alarms), in which the models send alert signals later than the engine starts to deteriorate, is not accepted because it can result in fatal damage.

5. Discussions and Insights

Equipment maintenance is a classical and critical issue because it significantly affects production schedules, product quality, and on-time deliveries [36]. In the past, PvM (occurs in regular cycles) is designed to keep parts in good repair but it does not take the state of a component or equipment into account. Today, PdM (occurs as needed) conducts real-time data collection and analysis to identify machine problems at:

| Table 9: Explained proportions of variances for the top three principal components. |
|-----------------|-----|-----|-----|
| Features        | PC1 | PC2 | PC3 |
| Sensor_2        | 0.273 | 0.018 | 0.022 |
| Sensor_3        | 0.26 | 0.077 | 0.031 |
| Sensor_4        | 0.301 | 0.01 | 0.018 |
| Sensor_6        | 0.062 | -0.048 | -0.997 |
| Sensor_7        | -0.298 | 0.052 | -0.016 |
| Sensor_8        | 0.281 | -0.243 | 0.032 |
| Sensor_9        | 0.105 | 0.643 | -0.026 |
| Sensor_11       | 0.309 | -0.015 | 0.015 |
| Sensor_12       | -0.304 | 0.062 | -0.022 |
| Sensor_13       | 0.281 | -0.245 | 0.024 |
| Sensor_14       | 0.066 | 0.666 | -0.029 |
| Sensor_15       | 0.287 | 0.022 | 0.007 |
| Sensor_17       | 0.269 | 0.078 | 0.012 |
| Sensor_20       | -0.282 | -0.019 | -0.015 |
| Sensor_21       | -0.283 | -0.024 | -0.022 |

| Table 10: Forecasting errors using PCA (principal component analysis). |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Training        | RF  | XGB | SVM | DNN | RNN | LSTM | GRU | CNN |
| RMSE            | 18.68 | 18.47 | 20.13 | 20.44 | 20.25 | 20.74 | 19.9 | 21.47 |
| MAE             | 13.66 | 14.06 | 14.24 | 16.16 | 15.93 | 16.16 | 15.51 | 17.42 |
| MAAPE           | 20.9 | 20.69 | 22.76 | 22.81 | 22.96 | 23.31 | 23.2 | 23.07 |
| Time (s)        | 59  | 6564 | 190 | 989 | 3461 | 9897 | 9344 | 2306 |
| Testing         | RF  | XGB | SVM | DNN | RNN | LSTM | GRU | CNN |
| RMSE            | 17.58 | 17.14 | 20.2 | 17.8 | 17.72 | 18.71 | 17.76 | 17.6 |
| MAE             | 12.36 | 12.05 | 14.09 | 12.46 | 12.3 | 13.09 | 12.25 | 12.45 |
| MAAPE           | 20.64 | 19.93 | 23.54 | 19.59 | 19.32 | 20.92 | 19.68 | 19.92 |
| Time (s)        | 0.47 | 3.14 | 2.22 | 0.29 | 0.33 | 0.58 | 0.34 | 0.31 |

| Table 11: Forecasting errors using SIR (sliced inverse regression). |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Training        | RF  | XGB | SVM | DNN | RNN | LSTM | GRU | CNN |
| RMSE            | 19.35 | 19.55 | 21.16 | 21.76 | 21.72 | 22.09 | 21.9 | 19.35 |
| MAE             | 14.58 | 15.27 | 14.31 | 17.81 | 17.75 | 18.38 | 17.57 | 14.58 |
| MAAPE           | 23.14 | 23.22 | 23.09 | 26.18 | 26.01 | 26.41 | 25.63 | 23.14 |
| Testing         | RF  | XGB | SVM | DNN | RNN | LSTM | GRU | CNN |
| RMSE            | 19.43 | 18.97 | 21.13 | 19.22 | 19.17 | 19.04 | 18.98 | 18.35 |
| MAE             | 13.57 | 13.36 | 14.69 | 14.21 | 14.14 | 14.15 | 14.03 | 13.51 |
| MAAPE           | 23.8 | 23.19 | 24.68 | 24.01 | 24.46 | 23.77 | 23.74 | 22.64 |
Figure 2: Predictive RUL using PCA and machine learning for all engines.

Figure 3: Predictive RUL using PCA and deep learning for all engines.

Figure 4: Predictive RUL using PCA and machine learning for engine 49.
the nascent stage before they can interrupt production processes. PdM simultaneously avoids excessive maintenance in PvM and prevents unexpected equipment breakdown. Clearly, the prediction of the RUL is the core to accomplish successful PdM. Following statistical probability distributions [22, 34–37], conventional burn-in tests derive the actual RUL at a normal condition given an object is operated in a rigorous condition like high temperature. However, it is difficult to justify its validity because we cannot wait for the equipment or devices to really fail. Therefore, machine learning or deep learning algorithms attempt to predict the actual RUL by identifying critical sensors. These sensors can alert warnings before machine failures happen. In the prediction of the RUL, practitioners want to know what sensors should be firstly monitored and when to send alerts?

In this research, feature selection is achieved using machine learning (SR, MARS, RF, and XGB), while feature extraction is achieved by unsupervised PCA and supervised SIR. To prioritize the degrees of importance for input features (sensors), our presented feature-selection methods follow the embedded approach. As expected, the performances using feature selection are slightly worse than the use of all sensors because only partial information is considered. Future extension in feature selection can incorporate metaheuristics [18, 32] to consider the wrapper approach. Typical metaheuristics algorithms include genetic algorithm, particle swarm optimization, ant colony optimization, bee colony optimization, grey wolf optimization, etc. Basically, they try to find *relatively* good combinations of input features to achieve satisfactory performances. In feature extraction, it is surprising to find supervised SIR does not outperform unsupervised PCA. Possible reasons are due to skewed distributions of the RUL. As indicated by Figure 6, the histogram of the RUL is biased toward 120–130 cycles. SIR groups data samples into distinct categories according to the sequence of the RUL. However, it cannot predict the RUL very well assuming the response is not fairly distributed (small between-group differences).

Not surprisingly, we observe the performances using deep learning are on average better than other methods. In particular, some publications use the CAE (convolutional autoencoder) features to achieve very small errors. However, these features are automatically synthesized in deep learning. They are hard to explain and apply to real industries. An architecture composed of two-stage deep-learning algorithms is also popular in recent years [3, 21]. This concept is very similar to stacking in ensemble learning. However, a
hybrid architecture incurs tedious computation and it cannot achieve fast response. In practice, the trade-offs between predictive performance and computational complexity needs to be carefully considered. Response speed and data privacy also inspires edge computing that can improve the drawbacks of cloud computing.

6. Conclusions

Industry 4.0 has been recognized as a radical innovation, which creates numerous technology dynamics and market opportunities in smart manufacturing and predictive maintenance. In the past, the "high-volume and low-variety" paradigm emphasizes mass production and preventive maintenance. Today, the "low-volume but high variety" paradigm focuses on firm's agility to respond and capability to convert huge amounts of data into decision analytics. To achieve better predictive maintenance, this research particularly addresses the impacts of feature engineering on the prediction of the RUL for turbofan engines. Experimental results show that feature selection using machine learning can reduce original 15 sensors to only 4 representative sensors but without declining the predictive performance too much (decreasing 4% MAAPE). Besides, feature extraction using unsupervised PCA performs better than supervised SIR. In particular, PCA’s performances are comparable with less than 20% MAAPE) for both machine learning and deep learning and better than using all sensors. More importantly, the presented framework can be extended to other areas, such as productivity enhancement in smart manufacturing and yield improvement in predictive maintenance. In summary, research contributions are outlined as follows:

(i) In feature extraction, PCA and SIR are used to generate sufficiently representative components or directions that can account for most variances of original sensors.

(ii) In feature selection, SR, MARS, RF, and XGB are used to identify significant sensors by prioritizing the degrees of importance of original sensors.

(iii) PCA + RF is more recommended than PCA + CNN because it can balance the trade-offs between predictive performances and computational time.

In future work, several directions are specifically indicated: (1) this research only considers feature extraction and feature selection. Feature transformation like discrete wavelet transform or metaheuristics based wrapper approaches can be included. (2) The prediction of the RUL is basically a regression problem (numerical outcome): the dependent outcome is assumed to be initially a level constant and then follow a linear decline (piecewise linear pattern). Statistical simulation-based models may be included to obtain precise estimates and increase generalization of this research. (3) In addition to predicting the RUL, diagnostic analytics for hidden states (healthy, risky, and sick) of turbine engines deserves to be modeled and explored in engine health management.

Data Availability
All data used in this research were extracted from https://www.kaggle.com/datasets/behrad3d/nasa-cmaps.

Conflicts of Interest
The authors declare that they have no conflicts of interest.

Acknowledgments
This research was under the support of Taiwan Ministry of Science and Technology.

References

[1] P. Aivaliotis, K. Georgoulas, and G. Chryssolouris, “The use of Digital Twin for predictive maintenance in manufacturing,” International Journal of Computer Integrated Manufacturing, vol. 32, no. 11, pp. 1067–1080, 2019.

[2] C. Gutsch, N. Furian, J. Suschini, D. Neubacher, and S. V. Voessner, “Log-based predictive maintenance in discrete parts manufacturing,” Procedia CIRP, vol. 79, pp. 528–533, 2019.

[3] I. Remadna, S. L. Terrissa, R. Zemouri, S. Ayad, and N. Zerhouni, “Leveraging the power of the combination of CNN and Bi-directional LSTM networks for aircraft engine RUL estimation,” in Proceedings of the 2020 Prognostics and Health Management Conference (PHM-Besancon), Besancon, France, May 2020.

[4] C.-Y. Lin, Y.-M. Hsieh, F.-T. Cheng, H.-C. Huang, and M. Adnan, “Time series prediction algorithm for intelligent predictive maintenance,” IEEE Robotics and Automation Letters, vol. 4, no. 3, pp. 2807–2814, 2019.

[5] K. T. P. Nguyen and K. Medjaher, “A new dynamic predictive maintenance framework using deep learning for failure prognostics,” Reliability Engineering & System Safety, vol. 188, pp. 251–262, 2019.

[6] V. Mathew, T. Toby, V. Singh, B. M. Rao, and M. G. Kumar, “Prediction of remaining useful lifetime (RUL) of turbofan engine using machine learning,” in Proceedings of the 2017 IEEE International Conference on Circuits and Systems (ICCS), Thiruvananthapuram, India, December 2017.

[7] H. Soni, A. Kansara, and T. Joshi, “Predictive maintenance of gas turbine using prognosis approach,” International Research Journal of Engineering and Technology, vol. 7, no. 6, pp. 4683–4691, 2020.

[8] K. Deng, X. Zhang, Y. Cheng et al., “A remaining useful life prediction method with long-short term feature processing for aircraft engines,” Applied Soft Computing, vol. 93, Article ID 106344, 2020.

[9] X. Zhang, P. Xiao, Y. Yang et al., “Remaining useful life estimation using CNN-XGB with extended time window,” IEEE Access, vol. 7, pp. 154386–154397, 2019.

[10] H. Li, W. Zhao, Y. Zhang, and E. Zio, “Remaining useful life prediction using multi-scale deep convolutional neural network,” Applied Soft Computing, vol. 89, Article ID 106113, 2020.

[11] O. E. Yurek and D. Birant, “Remaining useful life estimation for predictive maintenance using feature engineering,” in Proceedings of the 2019 Innovations in Intelligent Systems and Applications Conference (ASYU), Izmir, Turkey, November 2019.
A. M. Raid, H. K. Elminir, and H. M. Elattar, "Evaluation of neural networks in the subject of prognostics as compared to linear regression model," International Journal of Engineering & Technology IJET-IJENS, vol. 10, no. 6, p. 50, 2010.

K. Le Son, M. Fouladirad, A. Barros, E. Levrat, and B. Jang, "Remaining useful life estimation based on stochastic deterioration models: a comparative study," Reliability Engineering & System Safety, vol. 112, pp. 165–175, 2013.

P. Lim, C. K. Goh, and K. C. Tan, "A time window neural network based framework for Remaining Usefull Life estimation," in Proceedings of the 2016 International Joint Conference on Neural Networks (IJCNN), Vancouver, BC, Canada, July 2016.

G. Sateesh Babu, P. Zhao, and X.-L. Li, Deep Convolutional Neural Network Based Regression Approach for Estimation of Remaining Usefull Life, pp. 214–228, Springer International Publishing, Berlin/Heidelberg, Germany, 2016.

X. Li, Q. Ding, and J.-Q. Sun, "Remaining useful life estimation in prognostics using deep convolutional neural networks," Reliability Engineering & System Safety, vol. 172, pp. 1–11, 2018.

J. Li, X. Li, and D. He, "A directed acyclic graph network combined with CNN and LSTM for remaining useful life prediction," IEEE Access, vol. 7, pp. 75464–75475, 2019.

C. Chen, N. Lu, B. Jiang, and C. Wang, "A risk-averse remaining useful life estimation for predictive maintenance," IEEE/CAA Journal of Automatica Sinica, vol. 8, no. 2, pp. 412–422, 2021.

Z. Kang, C. Catal, and B. Tekinerdogan, "Remaining useful life (RUL) prediction of equipment in production lines using artificial neural networks," Sensors, vol. 21, no. 3, p. 932, 2021.

S. Xiang, Y. Qin, J. Luo, H. Pu, and B. Tang, "Multicellular LSTM-based deep learning model for aero-engine remaining useful life prediction," Reliability Engineering & System Safety, vol. 216, Article ID 107927, 2021.

X. Li, H. Jiang, Y. Liu, T. Wang, and Z. Li, "An integrated deep multiscale feature fusion network for aeroengine remaining useful life prediction with multisensor data," Knowledge-Based Systems, vol. 235, Article ID 107652, 2022.

J. Hu and P. Chen, "Predictive maintenance of systems subject to hard failure based on proportional hazards model," Reliability Engineering & System Safety, vol. 196, 2020.

F. O. Heimes, "Recurrent neural networks for remaining useful life estimation," in Proceedings of the 2008 International Conference on Prognostics and Health Management, Denver, CO, USA, October 2008.

S. Kim and H. Kim, "A new metric of absolute percentage error for intermittent demand forecasts," International Journal of Forecasting, vol. 32, no. 3, pp. 669–679, 2016.

S. Girard, H. Lorenzo, and J. Saracco, "Advanced topics in sliced inverse regression," Journal of Multivariate Analysis, vol. 188, Article ID 104852, 2022.

T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer Series in Statistics, New York, 2001.

K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: encoder–decoder approaches. SSST–8," in Proceedings of the Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, October 2014.

X. Yuan, L. Li, and Y. Wang, "Nonlinear dynamic soft sensor modeling with supervised long short-term memory network," IEEE Transactions on Industrial Informatics, vol. 16, no. 5, pp. 3168–3176, 2020.

C. H. Wang, "Considering economic indicators and dynamic channel interactions to conduct sales forecasting for retail sectors," Computers & Industrial Engineering, vol. 165, Article ID 107965, 2022.

H. Wang, M. Peng, R. Xu, A. Ayodeji, and H. Xia, "Remaining useful life prediction based on improved temporal convolutional network for nuclear power plant valves," Frontiers in Energy Research, vol. 8, 2020.

H. Wang, M. J. Peng, Z. Miao, Y. K. Liu, A. Ayodeji, and C. Hao, "Remaining useful life prediction techniques for electric valves based on convolution auto encoder and long short term memory," ISA Transactions, vol. 108, pp. 333–342, 2021.

A. Delmas, M. Sallak, W. Schon, and L. Zhao, "Remaining useful life estimation methods for predictive maintenance models: defining intervals and strategies for incomplete data," in Proceedings of the 10th IMA International Conference on Modelling in Industrial Maintenance and Reliability, Manchester, UK, June 2018.

Y.-W. Lu, C.-Y. Hsu, and K.-C. Huang, "An autoencoder gated recurrent unit for remaining useful life prediction," Processes, vol. 8, no. 9, p. 1155, 2020.

X.-S. Si, W. Wang, C.-H. Hu, and D.-H. Zhou, "Remaining useful life estimation – a review on the statistical data driven approaches," European Journal of Operational Research, vol. 213, no. 1, pp. 1–14, 2011.

C. Okoh, R. Roy, J. Mehnen, and L. Redding, "Overview of remaining useful life prediction techniques in through-life engineering services," Procedia CIRP, vol. 16, pp. 158–163, 2014.

A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in Proceedings of the International Conference on Prognostics and Health Management, IEEE, Denver, CO, USA, October 2008.

T.-B.-L. Nguyen, M. Djeziri, B. Ananou, M. Ouladsine, and J. Pinatou, "Remaining Useful Life estimation for noisy degradation trends," IFAC-PapersOnLine, vol. 48, no. 21, pp. 85–90, 2015.