Multiobjective Deep Belief Networks Ensemble for Remaining Useful Life Estimation in Prognostics

Chong Zhang, Pin Lim, A. K. Qin, *Senior Member, IEEE*, and Kay Chen Tan, *Fellow, IEEE*

***Abstract*— In numerous industrial applications where safety, efficiency, and reliability are among primary concerns, condition-based maintenance (CBM) is often the most effective and reliable maintenance policy. Prognostics, as one of the key enablers of CBM, involves the core task of estimating the remaining useful life (RUL) of the system. Neural networks-based approaches have produced promising results on RUL estimation, although their performances are influenced by handcrafted features and manually specified parameters. In this paper, we propose a multiobjective deep belief networks ensemble (MODBNE) method. MODBNE employs a multiobjective evolutionary algorithm integrated with the traditional DBN training technique to evolve multiple DBNs simultaneously subject to accuracy and diversity as two conflicting objectives. The eventually evolved DBNs are combined to establish an ensemble model used for RUL estimation, where combination weights are optimized via a single-objective differential evolution algorithm using a task-oriented objective function. We evaluate the proposed method on several prognostic benchmarking data sets and also compare it with some existing approaches. Experimental results demonstrate the superiority of our proposed method.**

***Index Terms*— Deep belief network (DBN), ensemble learning, evolutionary algorithm (EA), multiobjective, prognostics.**

I. Introduction

AN EFFICIENT prognostic health management (PHM) system can provide early warning of system failure based on condition monitoring data, domain knowledge about equipments, and environmental information, which enables effective maintenance plans. The benefits of PHM systems include the reduction of maintenance costs and the improvement of the safety and reliability of the monitored system. Fatal failures may occur in engineering systems (or their units) due to aging or unexpected incidents. To reduce such risks, making the reliable estimation of remaining useful life (RUL) is crucial especially for systems requiring infallible performances. Accordingly, RUL estimation has become one of the indispensable tasks in PHM systems. By leveraging RUL estimation, industries, such as civil aerospace, automobile, and manufacturing, can improve maintenance schedules to avoid catastrophic failures and consequently save the resultant costs.

Most engineering systems are complicated. As a result, it is difficult to model their degradation processes. Current prognostic approaches have shown limitations when applied to such systems [1], [2]. In real-world applications, costly system failures often result from multiple incidents involving tenable causalities and unintentional interactions among multiple system units. There are three popular categories of prognostic approaches, i.e., physics-of-failure (PoF)-based approaches [3], data-driven approaches [4], and their hybridization [5]. Although PoF-based approaches are typically more accurate if well designed, they rely on a large amount of knowledge about physical systems which is often unavailable in practice. As a result, PoF approaches become incapable when failure mechanism is unknown. On the other hand, data-driven approaches model the degradation characteristics of the system based on historical run-to-failure sensor data. They can infer correlations and causalities hidden in data while learning underlying trends from data. Hybrid approaches combine PoF with data-driven approaches, aiming to take advantages of both approaches while avoiding their disadvantages.

Although various prognostics approaches have been proposed for different applications, no existing approaches can achieve consistently good performances across different applications. Insufficient domain knowledge about common causes leading to system degradation makes it impossible to formulate precise physical models. As a result, data-driven approaches are more suitable and easier to use without relying on any domain knowledge. Implementation of traditional data-driven methods [4], [6]–[12], such as neural networks (NNs) [7] [83], support vector machines (SVMs) [6], local Gaussian regression [12], fuzzy similarity-based methods [11], [13], and Kalman filter (KF) [10], usually involves a grid search to seek for the optimal parameters of these methods. Such approaches have two major disadvantages as follows.

1) It is difficult to choose the most appropriate model to handle a specific situation. Furthermore, seeking the optimal parameter setting for a chosen model via grid search is typically computationally expensive. Moreover, many existing models take as an input a set of handcrafted features, and leave as an open question how to obtain the most useful features relevant to the estimation task. Deep belief networks (DBNs) offer a promising solution to this problem, which can learn powerful hierarchical feature representations from data.

Manuscript received February 9, 2016; revised June 2, 2016; accepted June 8, 2016. Date of publication February 9, 2016; date of current version September 15, 2017. This work was supported by the Ministry of Education, Singapore, under Grant R-263-000-A12-112.

C. Zhang and K. C. Tan are with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117583 (e-mail: zhangchong@u.nus.edu; eletankc@nus.edu.sg).

P. Lim is with the Advance Technology Center of Rolls Royce Singapore, Singapore 797575 (e-mail: pin.lim@rolls-royce.com).

A. K. Qin is with the School of Science, RMIT University, Melbourne, VIC 3000, Australia (e-mail: kai.qin@rmit.edu.au).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2016.2582798

2) The performance of a data-driven method could be influenced by various factors [14], e.g., capricious environmental uncertainties, considerable variability in the operational conditions of different system units, varying linear or nonlinear degradation patterns, the available number of sensors used to monitor the system, and the number of training samples. Therefore, a single model, after well configured based on the data collected under certain situations, may not generalize well to the other situations. As a potential solution, ensemble learning approaches [14]–[16] have shown superiority over single-model-based methods. The selection, training, and combination of individual learning models that constitute an ensemble are crucial to the success of such approaches. Evolutionary algorithms (EAs) [17]–[20], as an efficient metaheuristic stochastic search technique, may provide an effective way to train and combine individual learning models.

To overcome the above-mentioned deficiencies of traditional data-driven approaches, we propose a multiobjective deep belief networks ensemble (MODBNE) method. MODBNE employs a powerful multiobjective EA (MOEA) named MOEA based on decomposition (MOEA/D) integrated with the traditional DBN training technique to evolve multiple DBNs simultaneously subject to accuracy and diversity as two conflicting objectives. In this paper, each DBN has three hidden layers. Decision variables in MOEA/D are composed of DBN's structure parameters (i.e., the number of hidden neurons per hidden layer) and the parameters (i.e., the weight cost and learning rates) used by the traditional DBN training technique (i.e., contrastive divergence [21] followed by backpropagation (BP) [22], [23]). Therefore, any candidate solution generated by MOEA/D will lead to a specific DBN trained via contrastive divergence followed by BP using specific parameters. MOEA/D evolves a population of candidate solutions (and accordingly their associated DBNs) for a specified maximum number of generations. Eventually, all of the DBNs with respect to the final population of MOEA/D are combined to establish an ensemble model, where combination weights are optimized via single-objective differential evolution (DE) [17], [18], [24] using the average training error as its objective. Experimental results on several prognostic benchmarking data sets demonstrate the outstanding performance of MODBNE in comparison with some existing methods.

The rest of this paper is organized as follows. Section II reviews the existing literature related to data-driven prognostics, ensemble learning, and DBNs. The proposed MODBNE is detailed in Section III. Section IV evaluates the proposed MODBNE on several prognostic benchmarking data sets and compares its performance with those of some existing approaches. Section V concludes this paper and briefs our future research plan.

II. Literature Review and Related Works

A. Data-Driven Prognostics

Condition-based maintenance (CBM) has been widely used in many industries. By monitoring and analyzing the system's health conditions, CBM makes maintenance schedules that can not only reduce maintenance costs but also decrease the overall downtime of the system. This has led to increasing research attention on prognostics which is one of the key enablers of CBM. Niu *et al.* [25] presented a CBM system that uses reliability-centered maintenance mechanism to optimize maintenance costs, and employs data fusion strategy to improve condition monitoring, health assessment, and prognostics. The method they proposed involves degradation prediction and prognostics assessment. Based on the features extracted in time and frequency domains, Dempster–Shafer regression (DSR) and least square-SVM are used to estimate the RUL.

Modeling the relation between a set of feature vectors and their corresponding RUL values is the mainstream research in data-driven prognostics. For this kind of methods, each multidimensional feature vector is associated with an RUL value. Different machine-learning techniques can be used to learn the mapping between feature vectors and their associated RUL values, e.g., feed-forward NNs [26]–[32], recurrent NN (RNN) [4], KF [15], [30], particle filter [33], fuzzy rules [11], and so on.

Many approaches had been evaluated on the benchmarking prognostic problems studied in this paper. In [31], a Multilayer Perceptron (MLP) was utilized to predict the RUL of aero turbofan engines. Heimes [4] showed that RNN could learn complex nonlinear dynamic mappings, which considers the sequential states of the system. However, their approach ignored the discrepancies between independent engines, which require different networks applied to different engines. In [7], a feed-forward NN was utilized to predict the RUL. The age and condition monitoring data of the suspension system were used as an input, and the life percentage was generated as an output. Lim *et al.* [15] proposed an approach to deal with nonlinear problems, which can switch among different prediction models according to the health state detected by switching KF (SKF). This approach is sensitive to the initialization of the MLP. In [34], decision trees with fuzzy rules optimized by genetic algorithm were proposed and evaluated on some benchmarking prognostic problems. The generated fuzzy rules can be easily understood by engineers. Extreme learning machine (ELM) had also been applied to prognostic problems [32], which can model the degradation states of the system without assuming a homogeneous pattern.

Garvey and Hines [13] proposed a similarity-based approach using fuzzy similarity analysis for RUL estimation. Based on failure dynamic scenarios, this approach estimates the RUL based on the similarity between the test degradation pattern and the archived reference patterns.

Hu *et al.* [14] proposed an ensemble data-driven prognostic approach, which employs three weighting schemes to combine multiple base models. By minimizing the k-fold cross-validation error of the ensemble, their proposed optimization-based weighting scheme achieved robust RUL estimation. In [35], a semisupervised cotraining data-driven prognostic algorithm was proposed to deal with RUL estimation in a partially labeled scenario. The feed-forward NN and radial basis network are used therein as two regressors that are cotrained with suspension data on rolling-element

bearing and electric cooling fan. Fink *et al.* [26] proposed a multilayer feed-forward NN with multivalued neurons approach to deal with the reliability and degradation time-series prediction problem, and conducted a case study on predicting the degradation of a railway turnout system.

Many existing data-driven prognostic approaches involve one single model which can hardly maintain good generalization performance across various prognostic scenarios, especially when this model is well configured for a certain scenario. Moreover, most approaches employ handcrafted features which can hardly adapt to different prognostic applications. In this paper, an ensemble of DBNs is used to address these two issues. We propose a multiobjective evolutionary ensemble learning framework incorporated with the traditional DBN training technique to evolve multiple DBNs subject to accuracy and diversity as two conflicting objectives. All of the eventually evolved DBNs are combined to establish an ensemble model for RUL prediction, where combination weights are optimized via DE using a task-oriented objective. Two prerequisites required by our proposed method include the following:

1) the maximally allowed system degradation level (i.e., the threshold of system failure);
2) the relevant historical data (sensor data, environmental and/or operational parameters, and so on) of the system.

Based on historical data, prognostic models can be established to estimate the RUL.

B. Ensemble Learning

Many works [36]–[39] have demonstrated that a single method can seldom perform consistently well across a variety of applications. An ensemble of multiple models (i.e., base learners) with an appropriate combination of individual models' outputs can take advantage of each ensemble member so as to improve generalization. Although an ensemble may perform better than any of its members, it is not easy to construct a good ensemble. The most crucial task is to obtain base learners which individually outperform random guessing and meanwhile are less correlated with each other in terms of training errors [40]. The previous works [41], [42] indicated that an ideal ensemble should consist of base learners which can generate smaller errors and meanwhile have good diversity among themselves. Increasing diversity among base learners is a way to pursue better generalization. Finding an appropriate combination of diversified ensemble members is another key task in ensemble learning.

Ensemble methods have also been applied to improve the ability of handling noisy sensor data and complex relations underlying the data. Opitz and Maclin [43] had shown that good accuracy and diversity of a base learner are sufficient conditions to enable an ensemble to outperform any of its members. However, these two objectives are often conflicting [44]. As a result, MOEAs [45]–[50] can provide a solution. Existing works on MOEAs-based ensembles include negative correlation learning (NCL) [51], memetic Pareto artificial NN [52], [53], and diverse and accurate ensemble learning algorithm [44]. In fact, many existing ensemble prognostic approaches only rely on one single objective to train their ensemble members, e.g., mean square error [30], [31], [54], root mean square error (RMSE) [15], and PHM08 score [4], [8], [15], [31], which may lead to undesirable generalization performance. In this paper, we use accuracy and diversity as two conflicting objectives to train the base learners of the ensemble.

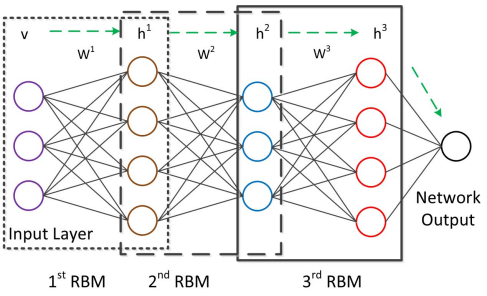


Fig. 1. Architecture of a DBN composed of three stacked RBMs, where $\mathbf{v}$ represents the input layer, $\mathbf{h}^i$ $(i = 1, \ldots, 3)$ denote three hidden layers, and $\mathbf{w}^i$ $(i = 1, \ldots, 3)$ are connection weight matrices in three RBMs.

C. Deep Belief Networks

In recent years, DBNs have become one of the most active research topics [55] in machine learning and been successfully applied in a variety of fields, such as computer vision [56], acoustic classification [57], phoneme recognition [58], and natural language processing [59]. These success stories reveal that DBNs are capable of learning the most suitable feature representations from data. This paper aims to explore the application of DBNs in prognostics which has been seldom studied in the field of prognostics.

A majority of traditional data-driven prognostic approaches need to first extract (and select) a set of handcrafted features and then utilize these features (in the form of a feature vector) as the input of the learning model. However, such approaches are usually unable to capture more relevant and useful feature representations. For complex engineering systems, features may have some hierarchical relations. A single-hidden-layer NN may not be able to capture such relations. Furthermore, traditional NNs are very inefficient to be trained when having a large amount of hidden neurons. Moreover, many hidden layers may not necessarily lead to better performance due to the dilution of the correction signal when using BP [22], [23]. Therefore, traditional NNs-based approaches usually constrain the network to contain one or two hidden layers.

The DBN is capable of extracting a hierarchy of features, where features at higher network levels are usually more relevant to the ultimate task. It is composed of stacked restricted Boltzmann machines (RBMs) which can be trained via contrastive divergence [21] in an unsupervised manner.

1) Restricted Boltzmann Machine: The RBM [60] is a two-layer undirected graph model, which is typically trained in an unsupervised manner. It is composed of one visible layer and one hidden layer with no connections allowed between nodes at the same layer.

2) Deep Belief Network: The DBN was proposed by Hinton *et al.* [21], which is composed of several stacked RBMs [60], as shown in Fig. 1. It typically involves layerwise

TABLE I
PROGNOSTICS BASED ON MODBNE

Step 1:	Define a prognostic problem and pre-process its associated training data.
Step 2:	Use an MOEA integrated with the traditional DBN training technique to evolve multiple DBNs based on training data.
Step 3:	Combine the eventually evolved DBNs to establish an ensemble model where combination weights are optimized via single-objective differential evolution using a certain task-oriented objective function.
Step 4:	Evaluate the ensemble model on test data.

pretraining by applying contrastive divergence [60] to each component RBM, followed by the fine-tuning of the entire network with respect to a certain criterion of interest. The unsupervised pretraining process may produce good initialization of network parameters, which facilitates the subsequent fine-tuning process. DBNs do not require handcrafted features. They usually take in raw data and automatically learn suitable feature representations. Unless otherwise mentioned, all DBNs considered for the rest of this paper consist of three hidden layers.

III. MULTIOBJECTIVE DEEP BELIEF NETWORKS ENSEMBLE

We propose an MODBNE method, which employs an MOEA integrated with the traditional DBN training technique to evolve multiple DBNs simultaneously subject to accuracy and diversity as two conflicting objectives. Each DBN has a fixed number of hidden layers (e.g., three in this paper) and is trained via the traditional DBN training technique, i.e., contrastive divergence followed by BP. For any DBN, its number of hidden neurons per hidden layer as well as the weight cost and learning rates used by contrastive divergence and BP to train it are encoded as decision variables that are optimized by the MOEA. This means that any candidate solution generated by the MOEA will lead to a DBN with the specific structure (i.e., the specific number of hidden neurons for each of three hidden layers) and trained via contrastive divergence followed by BP using the specific weight cost and learning rates. The MOEA evolves a population of candidate solutions (and accordingly their associated DBNs) for a specified maximum number of generations by considering several conflicting objectives that measure the quality of a candidate solution from different aspects (e.g., accuracy and diversity are used as two objectives in this paper). The eventually evolved DBNs are combined to establish an ensemble model for RUL estimation, where combination weights are optimized via single-objective DE using a task-oriented criterion as its objective.

Table I describes the procedure of prognostics based on the proposed MODBNE. First, a prognostic problem with the known maximally allowable degradation level is defined with the associated training data being preprocessed. Second, an MOEA integrated with the traditional DBN training technique is applied to evolve multiple DBNs simultaneously. Each candidate solution generated in the MOEA corresponds to a specific DBN, and solution quality is measured by the performance of this DBN on the training set after being trained via contrast divergence followed BP using the parameters encoded in this candidate solution. The MOEA evolves a population of candidate solutions (and accordingly their associated DBNs) subject to accuracy and diversity as two conflicting objectives. The finally evolved DBNs are combined to establish an ensemble model used for RUL estimation. The combination weights of this ensemble model are obtained via a single-objective DE algorithm using the average training error as its objective.

Typically, two key factors are considered in ensemble learning to train the base learners of an ensemble, i.e., accuracy and diversity. Accuracy quantifies the similarity between the output of a base learner and the ground-truth, while diversity measures the discrepancy between the output of a base learner and the outputs of other base learners. It has been demonstrated that an ensemble model with larger diversity among its base learners tends to achieve better generalization performance [36]. However, overly increasing diversity might drive the outputs of base learners away from the ground-truth. Meanwhile, merely increasing accuracy tends to equalize the outputs of base learners. For example, if all base learners achieve 100% accuracy, there is no diversity among them. MOEAs provide an effective way to balance accuracy and diversity when training the base learners of the ensemble. They can treat accuracy and diversity as two conflicting objectives, and evolve a population of base learners subject to these two objectives.

Assume an ensemble contains M DBNs, where the ensemble output is denoted by $P_w = \sum_{m=1}^{M} w_m p_m$. The weight coefficient w_m defines how much the mth DBN's output p_m contributes to the ensemble output P_w. The weight coefficients are constrained by $\sum_{m=1}^{M} w_m = 1$. Given a training set Φ_{train} with N different training samples, the following two objectives are optimized by the MOEA in the proposed MODBNE.

Accuracy: The first objective is to maximize the average prediction accuracy of an ensemble member (i.e., DBN) on the training set, which is equivalent to minimizing the average prediction error on the training set defined as follows:

$$\text{Minimize: } \text{Err}_m = \frac{1}{N}\sum_{i=1}^{N}\left(p_m^i - \text{RUL}^i\right)^2 \tag{1}$$

where RUL^i is the actual RUL value of the ith training sample, and the p_m^i represents the estimated RUL value obtained by the mth DBN for the ith training sample.

Diversity: The second objective is to maximize the diversity between the outputs of different ensemble members (i.e., DBNs). Here, we employ the idea from NCL [61], which uses the correlation between the outputs of ensemble members to define the diversity [44] as follows:

$$\text{Minimize: } D_m = \sum_{i=1}^{N}\left(p_m^i - P^i\right)\sum_{j\neq m, j=1}^{M}\left(p_j^i - P^i\right) \tag{2}$$

where p_m^i and p_j^i represent the outputs of the mth and jth DBNs for the ith training sample, respectively. P^i denotes the average output of all DBNs in the ensemble.

Equation (2) defines the total correlation between the output of one DBN and the output of each of the other DBNs. It has been demonstrated [42], [62] that good diversity among ensemble members (if unbiased) can be achieved when the outputs of ensemble members are negatively correlated with each other which can be achieved by minimizing (2).

As discussed earlier, maximizing accuracy (i.e., minimizing error) and meanwhile maximizing diversity (i.e., minimizing correlation) are somewhat conflicting. In the proposed MODBNE, we employ MOEA/D [63]—one of the most popular and powerful MOEAs—to evolve a population of candidate solutions, aiming at minimizing error (1) and correlation (2) simultaneously. The solution space is defined in terms of DBN's structural parameters (i.e., the number of hidden neurons per hidden layer), the contrastive divergence parameter (i.e., weight cost), and BP parameters (i.e., learning rates for weights and biases). Each candidate solution in this solution space will lead to a trained DBN with its prediction error and output correlation calculated by (1) and (2) as the values of two objectives in MOEA/D. Moreover, the DE operator and Gaussian mutation are incorporated into MOEA/D to generate candidate solutions.

In a multiobjective optimization problem (MOP), multiple objectives are maximized or minimized simultaneously. Therefore, any candidate solution $\mathbf{x}$ in solution space $\mathbf{\Omega}$ (possibly constrained) is associated with an objective vector $\mathbf{Z}(\mathbf{x}) = \{z_1(\mathbf{x}), \ldots, z_F(\mathbf{x})\}$, where $z_f(\mathbf{x})$ stands for the value of the fth objective function on $\mathbf{x}$. In this paper, two objectives to be minimized are considered, i.e., $F = 2$.

In MOEAs, the superiority of a candidate solution is evaluated in terms of multiple objectives. One candidate solution is superior to another (also known as one candidate solution dominates another) only if it has better quality for at least one objective while having the same quality for all the remaining objectives (if any). Otherwise, these two candidate solutions are regarded as nondominated solutions, i.e., one cannot dominate another. MOEAs can produce a set of nondominated candidate solutions which cannot dominate each other but dominate the other candidate solutions generated during evolution. Such a set is termed Pareto set. Its involved nondominated candidate solutions and their corresponding objective functions' values are called Pareto optimal solutions and Pareto front (PF), respectively. Among numerous existing MOEAs, MOEA/D features the decomposition of the approximation of the PF into a number of scalar optimization problems by using the Tchebycheff approach [64]. Simply speaking, it transforms an MOP into multiple computationally efficient single-objective optimization tasks. Such a decomposition is achieved via a prespecified set of evenly distributed weight vectors of dimensionality equal to the total number of objectives. Each weight vector corresponds to a single-objective optimization task. Assuming the similarity of two weight vectors indicates the closeness of their corresponding Pareto optimal solutions (motivation behind MOEA/D), each weight vector can form a neighborhood defined by some other weight vectors similar to it. Then, solving one single-objective problem with respect to one weight vector can benefit from and to solving other single-objective problems with respect to those weight vectors within its neighborhood. Assuming all of the multiple objectives are to be minimized (which is the case in this paper), the algorithmic details of MOEA/D-based DBNs ensemble learning are shown in Algorithm 1.

In real-world applications, the values of different objective functions may have different magnitudes, where some magnitudes could be much larger than others. Consequently, some objectives may undesirably bias the search direction. To address this issue, we introduce an adaptive normalization scheme to adjust the values of F ($F = 2$ in our case) objective functions z_f ($f = 1, \ldots, F$) at each generation g ($g \geq 1$) as follows:

$$z_f(\mathbf{x}) = \frac{z_f(\mathbf{x})}{\tilde{\mathrm{z}}_f} \tag{3}$$

where $\tilde{\mathrm{z}}_f$ ($f = 1, \ldots, F$) denote the normalization factors that are updated at the very beginning of each generation g ($g \geq 1$) by $\tilde{\mathrm{z}}_f = \max_{i \in \{1,\ldots,M\}} z_f(\mathbf{x}_g^i)$. For the first generation ($g = 1$), $\tilde{\mathrm{z}}_f$ ($f = 1, \ldots, F$) are set to uniformly distributed random numbers.

In the proposed MODBNE, the trained DBNs corresponding to each candidate solution in the final population of MOEA/D are combined to generate an ensemble model. The combination weights (summed to 1) are obtained via a single-objective DE algorithm using the average training error of the ensemble model as its objective. Finally, the ensemble model armed with such optimized combination weights is applied to test data. The algorithmic description of MODBNE is given in Table II.

IV. Experiments

A. Data Description

The proposed method is evaluated and compared with some existing approaches on a prognostic benchmarking problem, i.e., NASA's turbofan engines degradation problem. The corresponding data set was produced from the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) test bed, which was developed by NASA and available in NASA's data repository [65]. The C-MAPSS data set includes four sub-datasets. Each sub-dataset contains one training set and one test set and is composed of multivariate temporal data coming from 21 sensors. The operational status of each engine is healthy in the early stage and begins to degrade as time progresses until a failure occurs. Different degradation patterns had been discussed in [66]. In each sub-dataset, both training and test sets contain a certain number of engines. Each engine corresponds to a number of data points obtained by sampling its multivariate sensor data over a period of operational cycles. During training, all of the available data points from all engines are used as training samples. For testing, only one data point corresponding to the last recorded operational cycle for each engine is used as the test sample. Therefore, the number of training samples depends on both the number of engines and the number of data points per engine, while the number of test samples only depends on the number of engines. The summary of the C-MAPSS data set is described in Table III. In addition to sensor measurement, each data point (training or test sample) is associated with its corresponding engine

Algorithm 1 MOEA/D Based DBNs Ensemble Learning

Input:

An MOP with solution space Ω and F objectives to be minimized simultaneously

M: Population size (i.e. sub-problems in MOEA/D)

T: Neighbourhood size

G: Maximum number of generations (i.e. stopping criterion)

Output:

Trained DBNs $\{DBN_G^1, \cdots, DBN_G^M\}$ which correspond to the last-generation population $\{\mathbf{x}_G^1, \cdots, \mathbf{x}_G^M\}$

Step 1) Initialization:

Step 1.1) Generate an initial population $\{\mathbf{x}_1^1, \ldots, \mathbf{x}_1^M\}$ via uniformly random sampling in solution space Ω, and evaluate each candidate solution $\mathbf{x}_1^i$ $(i = 1, \cdots, M)$ in the initial population via its corresponding trained DBN DBN_1^i (where structural and learning parameters are encoded in $\mathbf{x}_1^i$) to obtain a vector of its objective functions' values $\mathbf{Z}(\mathbf{x}_1^i)$.

Step 1.2) Initialize reference point $\mathbf{Z}^* = \{z_1^*, \ldots, z_F^*\}$ where $z_f^* = \min_{i \in \{1, \cdots, M\}} z_f(\mathbf{x}_1^i)$ $(f = 1, \cdots, F)$.

Step 1.3) Randomly generate M uniformly distributed weight vectors $\{\lambda^1, \ldots, \lambda^M\}$ where $\lambda^i = \{\lambda_1^i, \ldots, \lambda_F^i\}$ corresponds to the ith population member. For each weight vector λ^i, its closest T weight vectors $\{\lambda^{i_1}, \cdots, \lambda^{i_T}\}$ are found to form its neighborhood $B(i) = \{i_1, \cdots, i_T\}$.

Step 1.4) Randomly generate F uniformly distributed numbers $\tilde{\mathrm{z}}_f$ $(f = 1, \cdots, F)$.

Step 2) Evolution:

for $g = 1, \cdots, G$ **do**

Step 2.1) Adaptive normalization: Apply adaptive normalization to each member $\mathbf{x}_g^i$ $(i = 1, \cdots, M)$ in the current population, i.e. $z_f(\mathbf{x}_g^i) = \frac{z_f(\mathbf{x}_g^i)}{\tilde{\mathrm{z}}_f}$ $(f = 1, \cdots, F)$, to make the values of each objective to have similar magnitudes.

for $i = 1, \cdots, M$ **do**

Step 2.2) Reproduction: Randomly select two indices j and k from $B(i)$, and then generate a new candidate solution $\mathbf{x}_g^{i'}$ from $\mathbf{x}_g^i$, $\mathbf{x}_g^j$ and $\mathbf{x}_g^k$ by using the DE operator followed by a Gaussian mutation[1] applied under probability of 0.5:

$$\mathbf{x}_g^{i'} = \begin{cases} \mathbf{x}_g^i + F \cdot (\mathbf{x}_g^j - \mathbf{x}_g^k) + rnd_G(\mathbf{0}, \sigma) & \text{if } rnd_U(0,1) \leq 0.5 \\ \mathbf{x}_g^i + F \cdot (\mathbf{x}_g^j - \mathbf{x}_g^k) & \text{otherwise} \end{cases}$$

In this work, F is set to 0.5 and each element in vector σ is set to one twentieth of the corresponding decision variable's range.

Step 2.3) Repairing: If any element in $\mathbf{x}_g^{i'}$ exceeds its corresponding decision variable's upper (lower) bound, set this element to the upper (lower) bound.

Step 2.4) Evaluation: Train a DBN of three hidden layers (with the number of hidden neurons per hidden layer encoded in $\mathbf{x}_g^{i'}$) via contrastive divergence (with weight cost encoded in $\mathbf{x}_g^{i'}$) followed by BP (with learning rates for weights and biases encoded in $\mathbf{x}_g^{i'}$), and use the performance of this trained DBN on the training set to determine $\mathbf{Z}(\mathbf{x}_g^{i'}) = \{z_1(\mathbf{x}_g^{i'}), \cdots, z_F(\mathbf{x}_g^{i'})\}$.

Step 2.5) Adaptive normalization: Apply adaptive normalization to $\mathbf{x}_g^{i'}$, i.e. $z_f(\mathbf{x}_g^{i'}) = \frac{z_f(\mathbf{x}_g^{i'})}{\tilde{\mathrm{z}}_f}$ $(f = 1, \cdots, F)$.

Step 2.6) Replacement: For each $i_s \in B(i)$, if $\max_{f \in \{1, \cdots, F\}} \lambda_f^{i_s} \cdot |z_f(\mathbf{x}_g^{i'}) - z_f^*| \leq \max_{f \in \{1, \cdots, F\}} \lambda_f^{i_s} \cdot |z_f(\mathbf{x}_g^{i_s}) - z_f^*|$, then replace $\mathbf{x}_g^{i_s}$ and $\mathbf{Z}(\mathbf{x}_g^{i_s})$ with $\mathbf{x}_g^{i'}$ and $\mathbf{Z}(\mathbf{x}_g^{i'})$.

Step 2.7) Updating: For each $f \in \{1, \cdots, F\}$, if $z_f(\mathbf{x}_g^{i'}) < z_f^*$, then set $z_f^* = z_f(\mathbf{x}_g^{i'})$.

end for

Step 2.8) Updating: $\tilde{\mathrm{z}}_f = \max_{i \in \{1, \cdots, M\}} z_f(\mathbf{x}_g^i)$, $\mathbf{x}_{g+1}^i = \mathbf{x}_g^i$, $\mathbf{Z}(\mathbf{x}_{g+1}^i) = \mathbf{Z}(\mathbf{x}_g^i)$ $(i = 1, \cdots, M)$.

end for

[1]NOTE: (1) $rnd_G(\mathbf{a}, \mathbf{b})$ is a Gaussian random vector generator with mean vector $\mathbf{a}$ and standard deviation vector $\mathbf{b}$; (2) $rnd_U(a, b)$ is a uniform random number generator sampling in $[a, b]$.

TABLE II

ALGORITHMIC DESCRIPTION OF MODBNE

Step 1:	Randomly initialize a population of M candidate solutions where M is the total number of DBNs used to establish the ensemble model. Suppose each DBN has H hidden layers (H is three in this work). Each candidate solution $\mathbf{x}^i$ $(i = 1, \cdots, M)$ involves, as its decision variables, the number of hidden neurons in each of H hidden layers (l_h^i, $h = 1, \cdots, H$), the weight cost (ω^i) used by contrastive divergence, and the learning rates for weights and biases (η_w^i and η_b^i) used by BP. Each decision variable will be randomly generated in its valid range.
Step 2:	For each candidate solution in the initial population of MOEA/D, train a DBN with the number of hidden neurons at each of H hidden layers specified by this candidate solution using the traditional DBN training technique in which the parameters involved in contrastive divergence and BP (i.e. weight cost and learning rates) are specified by this candidate solution. The values of multiple objectives with respect to this candidate solution can be obtained from the trained DBN.
Step 3:	Apply MOEA/D to evolve the population as detailed in Algorithm 1. Whenever a new candidate solution is generated, a DBN is trained as Step 2 does so as to derive the values of multiple objectives with respect to this candidate solution.
Step 4:	Combine all of the DBNs with respect to the final population of MOEA/D to establish an ensemble model for RUL estimation where combination weights are optimized via single-objective differential evolution using the average training error of the ensemble model as its objective.

TABLE III

DESCRIPTION OF C-MAPSS DATA SET

Sub-datasets	FD001	FD002	FD003	FD004
# Engines in the training set	100	260	100	248
# Engines in the test set	100	259	100	248
# Training samples	20631	53759	24720	61249
# Test samples	100	259	100	248

TABLE IV

FORMAT OF C-MAPSS DATA SET

Column No.	Contents
1	Engine Unit No.
2	Time in cycles
3	Operational Setting 1
4	Operational Setting 2
5	Operational Setting 3
6	Sensor Measurement 1
⋮	⋮
26	Sensor Measurement 21

unit number, operational cycle, and operational settings. The format of the C-MAPSS data set is detailed in Table IV.

In the training set of each sub-dataset, a linear or piecewise linear degradation model [4], [15] is used to estimate the ground-truth RUL value with respect to each training sample. In the test set of each sub-dataset, the actual ground-truth RUL value is available for each test sample.

B. Data Preprocessing

1) Sensor Selection: The C-MAPSS data set is composed of multivariate temporal data from 21 sensors. Details of these 21 sensors can be found in [67]. Among 21 sensors, some sensors have constant outputs throughout the life time of the engine, which cannot provide any useful information to

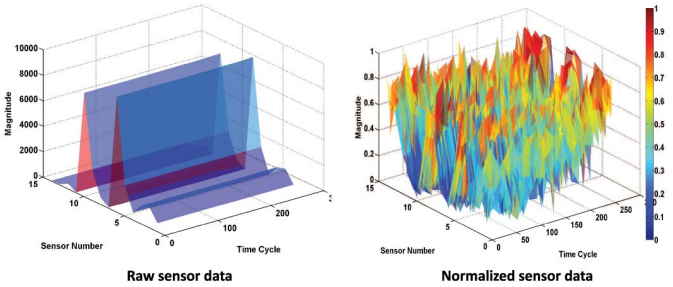


Fig. 2. Illustration of the data from the 14 selected sensors during the entire life cycle of a single engine in the training set of FD001 before and after normalization.

facilitate prediction. Therefore, we eliminate the outputs of these sensors from the C-MAPSS data set as did in [4] and [15]. As a result, each data point in the C-MAPSS data set involves 14 features corresponding to the outputs of 14 sensors with indices 2, 3, 4, 7, 8, 9, 11, 12, 13, 14, 15, 17, 20, and 21.

2) Normalization: For each sub-dataset in C-MAPSS, the value range of each sensor is normalized as follows to enable the unbiased contribution from the output of each sensor [30]:

$$x_i^{j'} = \frac{2\left(x_i^j - x_{\min}^j\right)}{x_{\max}^j - x_{\min}^j} - 1 \quad \forall i, j \tag{4}$$

where $x_i^{j'}$ and x_i^j represent the ith data point from the jth sensor after and before normalization, respectively. $x_{\min}^j$ and $x_{\max}^j$ denote the minimal and maximal values over all data points from the jth sensor before normalization, respectively.

Fig. 2 shows the data of the 14 selected sensors during the entire life cycle of a single engine in the training set of FD001 before and after normalization.

3) Time Window Processing: Many multivariate temporal data-based prediction models take as an input a multivariate data point sampled at a single time stamp. This way neglects some useful temporal information that may much improve prediction performance. To address such an issue, this paper utilizes a fixed-size time window (TW) to enclose multivariate data points sampled at consecutive time stamps. Specifically, at any specific time stamp, multivariate data points within the TW that covers the current time stamp and its several preceding time stamps are concatenated into a high-dimensional feature vector which is then fed as an input into the prediction model. In practice, a suitable TW size can be chosen via cross-validation. In this paper, the TW size is set to 30. Fig. 3 shows the data from the 14 selected sensors within a TW of size 30 for the initial five time instances with respect to a single engine in the training set of FD001. For each time instance, a feature vector of dimension size 14×30 is obtained as the input of the prediction model.

C. Evaluation Metrics

1) Scoring Function: The scoring function method used in this paper is the same to those used in [4], [8], [68], and [69] and that employed by the International Conference on Prognostics and Health Management (PHM08) Data Challenge.

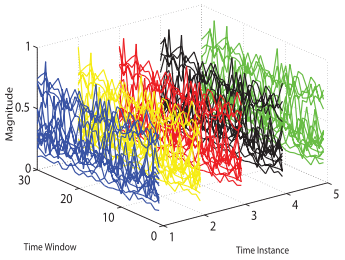


Fig. 3. Illustration of the data from the 14 selected sensors within a TW of size 30 for the initial five time instances with respect to a single engine in the training set of FD001.

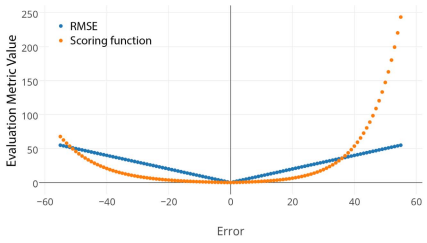


Fig. 4. Comparison between scoring function and RMSE with respect to different errors.

The scoring function is asymmetric, which penalizes late prediction (i.e., the estimated RUL value is larger than the actual RUL value) more heavily than early prediction (i.e., the estimated RUL value is smaller than the actual RUL value) since late prediction may result in more severe consequences. Furthermore, larger prediction errors are more severely penalized. The formulation of this scoring function [70] is as follows:

$$s = \sum_{i=1}^{N} s_i, \quad s_i = \begin{cases} e^{-\frac{d_i}{13}} - 1, & \text{for } d_i < 0 \\ e^{\frac{d_i}{10}} - 1, & \text{for } d_i \geq 0 \end{cases} \tag{5}$$

where N is the total number of data points, and d_i denotes the $\text{RUL}_i' - \text{RUL}_i$ (i.e., the estimated RUL value minus the actual RUL value with respect to the ith data point).

2) RMSE: Besides the scoring function, RMSE is another commonly used measure to evaluate prediction accuracy [15], [30], [71], although it equally penalizes both early and late predictions. Fig. 4 shows the comparison between scoring function and RMSE. The formulation of RMSE is as follows:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} d_i^2}. \tag{6}$$

D. Experimental Setup

In the proposed MODBNE, the structural parameters (i.e., the number of hidden layers and the number of hidden neurons per hidden layer) of a DBN strongly depend on problem complexity and the number of available training samples. Given the complexity of our studied problems and the limited number of available training samples, the number of hidden layers and the number of hidden neurons per hidden layer in a DBN are not set to be very large in our experiments. Specifically, the number of hidden layers for each DBN is

set to three. The number of hidden neurons per hidden layer is encoded as a decision variable optimized by MOEA/D, which has range [10, 50]. The ranges of the other decision variables in MOEA/D are: [0.001, 0.005] for weight cost in contrastive divergence, and [0.01, 0.5] for learning rates of weights and biases in BP. We set population size to 20 and neighborhood size to four. The DE-based optimization of combination weights is run for 50 generations. For comparison, we employ a multilayer perceptron (MLP) and a single DBN. Both have three hidden layers, where the number of hidden neurons per hidden layer is randomly chosen within range [10, 50] for each experimental run. Moreover, we also compare the proposed method with some other existing approaches which are introduced as follows. ELM is a single-hidden-layer feed-forward NN with randomized connection weights between the input and hidden layers and analytically determined connection weights between the hidden and output layers [72]. Hierarchical ELM (HELM) is a multilayer ELM. SVM [73] is one of the most popular supervised learning techniques which constructs a class separation hyperplane in a high-dimensional space implicitly defined via a certain kernel function. Least absolute shrinkage and selection operator (LASSO) [74] is a linear regression model with an L_1 regularizer. Extra tree regressor (ETR) is a metaestimator with randomized decision trees. KNeighbors regressor (KNR) is a regression model based on the k-nearest neighbors algorithm. Gradient boosting (GB) [75] is an ensemble of decision trees with a boosting-based combination scheme. Random forest (RF) [76] is a powerful ensemble learning method based on random decision forests. SKF [15], [16] is a forward pass state estimator with the backward recursive prediction process, which can approximate nonlinear functions.

Each method in comparison is run ten times with the average performance over ten experimental runs being reported.

E. Experimental Results

We investigate several factors which may impact the performance of MODBNE, i.e., the number of hidden layers, the adaptive normalization scheme, and TW processing. Furthermore, we evaluate the performance of MODBNE in comparison of many existing approaches on C-MAPSS data set.

1) Analysis of the Number of Hidden Layers: The proposed MODBNE uses DBNs to establish the ensemble. Each DBN has the same number of hidden layers with the number of hidden neurons per hidden layer being optimized. For a single DBN, its performance is influenced by its number of hidden layers. To specify the appropriate number of hidden layers for DBNs in the ensemble, we study the performance of a single DBN with the varying number of hidden layers in comparison with an MLP with the corresponding number of hidden layers. Figs. 5 and 6 show the average RMSEs and the average scoring function values (also known as scores) over ten runs obtained by MLP and DBN on FD001, respectively, as the number of hidden layers increases from one to seven. At each hidden layer number, MLP and DBN use the same specified number of hidden neurons per hidden layer. It can be observed that the

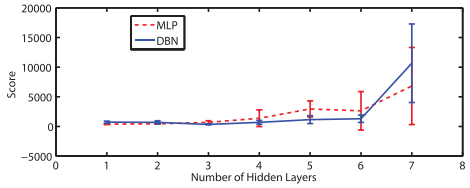


Fig. 5. Average scoring function values (scores) over ten runs achieved by MLP and DBN with respect to different numbers of hidden layers on FD001.

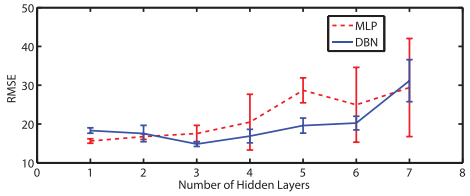


Fig. 6. Average RMSEs over ten runs achieved by MLP and DBN with respect to different numbers of hidden layers on FD001.

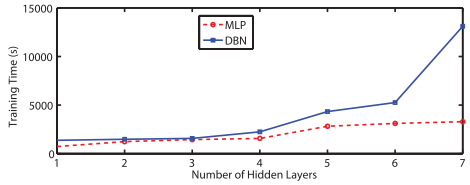


Fig. 7. Average training time over ten runs obtained by MLP and DBN with respect to different numbers of hidden layers on FD001.

performances of both DBN and MLP degrade as the number of hidden layers increases. Comparatively, the degradation of DBN is slower than that of MLP. Fig. 7 shows the average training time over ten runs obtained by DBN and MLP on FD001, as the number of hidden layers increases from one to seven. It can be observed that DBN has similar computation time to MLP when the number of hidden layers is relatively small, although DBN's computation time rapidly goes up to be higher than that of MLP, as the number of hidden layers exceeds four. In fact, many existing works [66], [77]–[79] have used three hidden layer DBNs for failure diagnosis, health states classification, and time-series forecasting. Considering the overfitting risk due to the limited number of training samples in our studied problems, the number of hidden layers is set to three for DBNs in MODBNE as well as MLP and DBN used for comparison in this paper. Meanwhile, the number of hidden neurons per hidden layer is constrained between 10 and 50.

2) Analysis of the Adaptive Normalization Scheme: Fig. 8 shows the evolved population of MOEA/D after 500 generations without and with applying normalization in the objective space. It can be observed from Fig. 8(a) that without normalization, many candidate solutions in the population bias toward one objective. In contrast, Fig. 8(b) compares the evolved populations (after 500 generations) under two normalization schemes: constant normalization and adaptive normalization. Different from adaptive normalization, as described in Section III, normalization factors in constant normalization are initialized as uniformly distributed random numbers and keep fixed throughout evolution. It can be observed from Fig. 8(b)

TABLE V
PERFORMANCE COMPARISON OF 11 METHODS ON EACH OF FOUR C-MAPSS SUB-DATASETS WHEN TW PROCESSING IS NOT APPLIED, i.e., TW SIZE IS 1

TW=1	Average	MODBNE	DBN	SKF	MLP	ELM	HELM	SVM	LASSO	ETR	KNR	GB	RF
FD001	Score	640.27	1001.44	762.85	959.63	740.52	835.53	852.07	894.21	1359.38	604.26	**575.04**	802.23
	RMSE	**17.96**	18.48	19.24	18.48	19.40	20.02	20.58	22.43	22.05	19.73	18.80	20.23
FD002	Score	**10851**	15633	23834	13018	53634	46691	521461	231995	231030	76373	44213	84068
	RMSE	28.06	30.05	**27.45**	29.78	28.89	28.79	36.27	39.43	33.01	29.69	28.67	30.01
FD003	Score	**683.41**	2074.57	925.78	1442.70	809.84	897.77	1108.68	1144.55	1757.60	1129.82	863.66	1000.51
	RMSE	19.41	20.99	**18.29**	19.64	21.48	21.83	23.30	23.72	24.52	23.38	22.25	22.34
FD004	Score	**7210**	8411	7544	30717	24853	25084	46611	100321	69771	26847	20538	22250
	RMSE	29.45	30.02	**26.76**	34.41	30.04	30.02	40.77	43.71	32.42	33.42	29.46	29.62

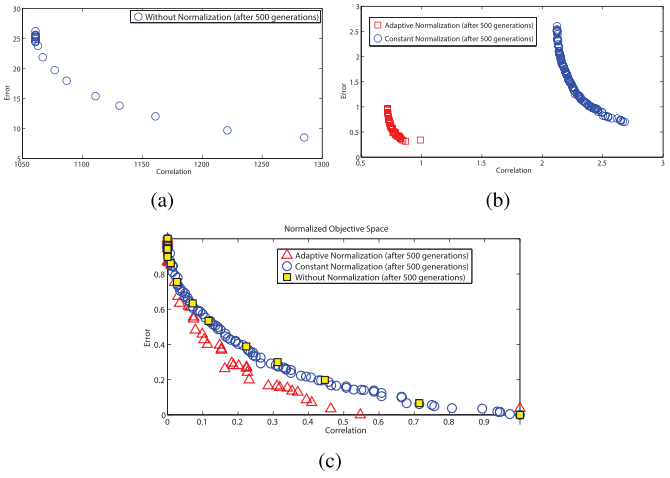


Fig. 8. Evolved population of MOEA/D obtained after 500 generations. (a) Without normalization. (b) With constant normalization and adaptive normalization. (c) Combination of (a) and (b) into one plot after normalizing each of two objectives in (a) and (b) into range [0, 1], respectively.

that the obtained population under adaptive normalization is better than that obtained under constant normalization. Fig. 8(c) compares final populations obtained under no normalization, adaptive normalization, and constant normalization. It combines Fig. 8(a) and (b) into one plot after normalizing each of two objectives in Fig. 8(a) and (b) into range [0, 1], respectively. It is obvious that the population obtained under adaptive normalization is superior to the other two.

3) Performance Evaluation and Comparison Without Time Window Processing: This section studies MODBNE in comparison of 11 other methods when TW processing is not applied, i.e., TW = 1.

Table V reports the average performances of MODBNE, DBN, SKF MLP, ELM, HELM, SVM, LASSO, ETR, KNR, GB, and RF over ten runs on each of four C-MAPSS sub-datasets. It can be observed that MODBNE is always among the top three in terms of both RMSE and score on every sub-dataset. Fig. 9 shows the box plot of the RMSEs and the scoring function values (scores) over ten runs obtained by 12 methods in comparison with each of four C-MAPSS sub-datasets. It can be observed that MODBNE consistently demonstrates robust performance evidenced by small variance. Moreover, the scores obtained by many methods over ten runs result in large variance due to large penalty applied by the scoring function to late prediction.

4) Analysis of Time Window Processing: As discussed in Section IV-B3, applying TW processing to obtain feature vectors by concatenating data points sampled at several consecutive time stamps can bring useful temporal information that may much improve prediction performance. In practice, the TW size usually impacts performance. This section investigates the influence of different TW sizes on prediction performance in the context of a DBN of three hidden layers with the number of hidden neurons per hidden layers randomly chosen within range [10, 50] at each experimental run.

Fig. 10 shows the box plot of the scoring function values over ten runs obtained by DBN under five different TW sizes on each of four C-MAPSS sub-datasets. Fig. 11 shows the box plot with respect to RMSE. It can be observed that in most cases, the performance first improves as the TW size increases, and then degrades when the TW size becomes too large (e.g., TW = 40). The performance improvement is due to the utilization of useful temporal information contained in the TW. On the other hand, the performance degradation is because of irrelevant information introduced by samples at too early time stamps. Moreover, a large window size will lead to high-dimensional feature vectors which increase computational costs. Given these facts, the TW size used in our studies is set to 30.

5) Performance Evaluation and Comparison With Time Window Processing: This section studies MODBNE in comparison of ten other methods when TW processing with the window size of 30 is applied. SKF is not considered here, because its performance results are directly taken from the literature [16] which does not involve TW processing.

Table VI reports the average performances of MODBNE, DBN, MLP, ELM, HELM, SVM, LASSO, ETR, KNR, GB, and RF over ten runs on each of four C-MAPSS sub-datasets. It can be observed that MODBNE consistently outperforms all of the other compared methods in terms of both RMSE and score on every sub-dataset. Fig. 12 shows the box plot of the RMSEs and the scoring function values (scores) over ten runs obtained by 11 methods in comparison with each of four C-MAPSS sub-datasets. It can be observed that the performance of MODBNE is both superior and robust, evidenced by small median and variance.

6) Computation Time Analysis: Most of computation time in MODBNE comes from the training of DBNs. In fact, whenever a new candidate solution is generated in MOEA/D, its corresponding DBN needs to be trained by using the traditional DBN training technique, i.e., contrastive divergence

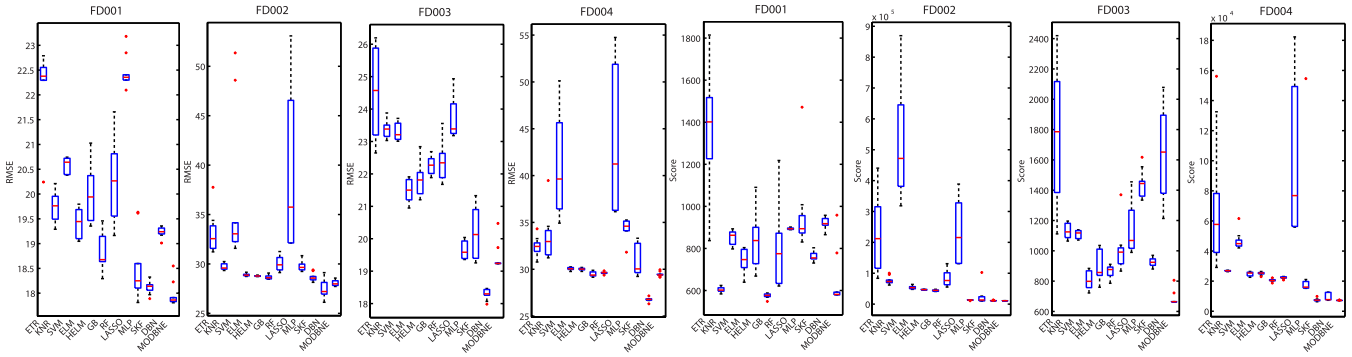


Fig. 9. Comparison of MODBNE, DBN, SKF, MLP, ELM, HELM, SVM, LASSO, ETR, KNR, GB, and RF in terms of the box plot of the RMSEs and the scoring function values (scores) over ten runs on four C-MAPSS sub-datasets, respectively, when TW processing is not applied.

TABLE VI

PERFORMANCE COMPARISON OF 11 METHODS ON EACH OF FOUR C-MAPSS SUB-DATASETS WHEN TW PROCESSING WITH THE TW SIZE OF 30 IS APPLIED

TW=30	Average	MODBNE	DBN	MLP	ELM	HELM	SVM	LASSO	ETR	KNR	GB	RF
FD001	Score	**334.23**	417.59	560.59	523.00	562.96	7703.33	653.85	1667.86	729.32	474.01	479.75
	RMSE	**15.04**	15.21	16.78	17.27	17.43	40.72	19.74	23.76	20.46	15.67	17.91
FD002	Score	**5585.34**	9031.64	14026.72	498149.97	434056.50	316483.31	276923.89	803203.04	450094.09	87280.06	70456.86
	RMSE	**25.05**	27.12	28.78	37.28	34.84	52.99	37.13	36.05	53.57	29.09	29.59
FD003	Score	**421.91**	442.43	479.85	573.78	456.32	22541.58	1058.36	2240.70	1030.29	576.72	711.13
	RMSE	**12.51**	14.71	18.47	18.90	17.69	46.32	21.38	25.66	22.59	16.84	20.27
FD004	Score	**6557.62**	7954.51	10444.35	121414.47	105492.77	141122.19	125297.19	345956.76	234396.56	17817.92	46567.63
	RMSE	**28.66**	29.88	30.96	38.43	37.98	59.96	40.70	40.01	54.44	29.01	31.12

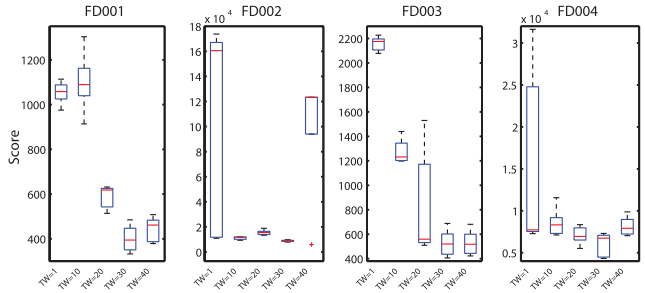


Fig. 10. Box plot of scoring function values (scores) over ten runs obtained by DBN under five different TW sizes on each of four C-MAPSS sub-datasets.

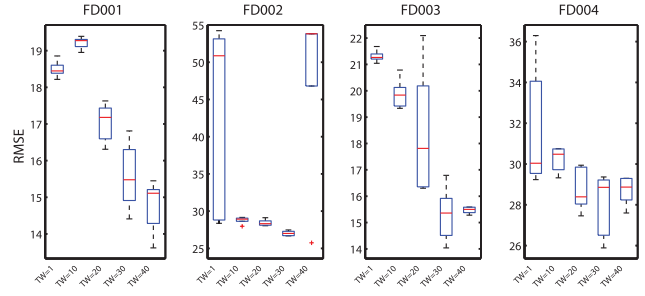


Fig. 11. Box plot of RMSEs over ten runs obtained by DBN under five different TW sizes on each of four C-MAPSS sub-datasets.

followed by BP, so as to evaluate the quality of this candidate solution based on the trained DBN's performance on the training set. The training time for a DBN depends on its network size.

TABLE VII

AVERAGE COMPUTATION TIME OF DBN, MLP, ELM, HELM, SVM, LASSO, ETR, AND KNR OVER TEN RUNS ON FD001 UNDER DIFFERENT TW SIZES

Time(s)	DBN	MLP	ELM	HELM
TW=1	227.6	187.1	6.0	23.0
TW=30	1134.5	367.1	10.8	39.6
Time(s)	**SVM**	**LASSO**	**ETR**	**KNR**
TW=1	26.9	1.3	1.3	1.7
TW=30	2258.2	312.4	50.6	5.7

Table VII reports the average computation time of DBN, MLP, ELM, HELM, SVM, LASSO, ETR, and KNR over ten runs on FD001 without and with TW processing. These eight methods in comparison belong to nonensemble methods. It can be observed that TW processing may much increase computation time due to its resultant high-dimensional feature vectors. As to ensemble methods, the average computation time of MODBNE over ten runs is compared with that of GB and RF with grid search of optimal parameters, where TW processing (TW = 30) is applied. The comparison results are reported in Table VIII. It can be observed that the computation time of MODBNE is shorter than GB but a bit longer than RF.

Our experimental platform is a desktop PC with Intel Core i7-3770 3.40-GHz CPU and 16-GB RAM. The operating system is Windows 7.

7) Comparison With Publicly Available Results: Some existing works on RUL estimation also used C-MAPSS data sets for performance evaluation and comparison. Among four C-MAPSS sub-dataset, FD001 has been most often used. Therefore, this section compares the performances of

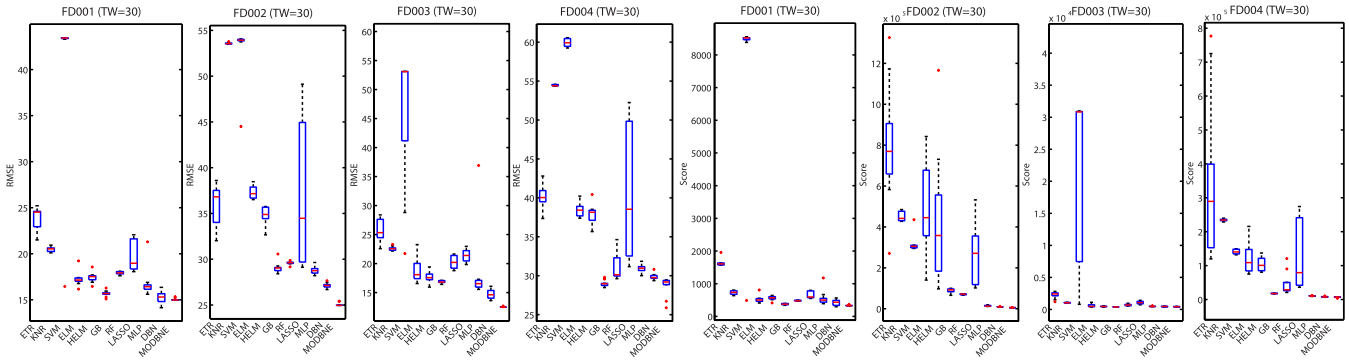


Fig. 12. Comparison of MODBNE, DBN, MLP, ELM, HELM, SVM, LASSO, ETR, KNR, GB, and RF in terms of the box plot of the RMSEs and the scoring function values (scores) over ten runs on four C-MAPSS sub-datasets, respectively, when TW processing with the TW size of 30 is applied.

TABLE VIII

AVERAGE COMPUTATION TIME OF MODBNE OVER TEN RUNS ON FD001 IN COMPARISON OF GB AND RF WITH GRID SEARCH OF OPTIMAL PARAMETERS

Algorithm	Computation time(s)
MODBNE	1153760.67
GB (grid search)	1917717.68
RF (grid search)	915425.13

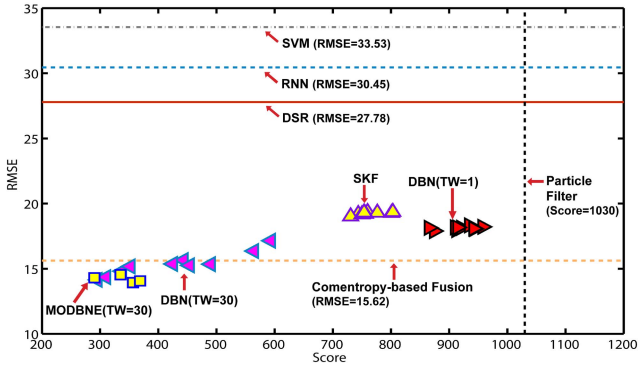


Fig. 13. RMSEs and scores over ten runs obtained by MODBNE (yellow squares), DBN without TW processing (red right triangles), and DBN with TW processing (purple left triangles) compared with the publicly available results achieved by some existing methods on FD001.

MODBNE and a single DBN without and with TW processing to some results reported in the literature [16], [80], [81] on FD001. Specifically, we use the results of DSR, SVM, RNN, and comentropy-based fusion from [80], particle filter from [81], and SKF from [16]. DSR is a nonprobabilistic transferable belief model by adopting the subjectivist. Comentropy-based fusion is a PHM-oriented fusion prognostics framework that takes advantage of individual algorithms. The other methods have been described in Section II. In [80], the performance measure is defined by $e_{\text{MSE}} = (1/N)(\sum_{i=1}^{N}(p^i - \text{RUL}^i)^2)^{1/2}$, where RUL^i is the actual RUL value of the ith data sample and the p^i represents the estimated RUL value for the ith data sample. To enable performance comparison under the same measure, e_{MSE} is transformed into RMSE via $\text{RMSE} = \sqrt{N} \times e_{\text{MSE}}$. Fig. 13 compares the RMSE and scores over ten runs obtained by MODBNE, DBN without TW processing, and DBN with TW processing to the publicly available results achieved by DSR, SVM, RNN, comentropy-based fusion, particle filter, and SKF. In fact, some results are only available in terms of either RMSEs or scores. Such cases correspond to straight lines in Fig. 13. It can be observed from Fig. 13 that the proposed MODBNE demonstrates the overall best performance.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed a multiobjective evolutionary ensemble learning method named MODBNE to evolve multiple DBNs simultaneously subject to accuracy and diversity as two conflicting objectives. The eventually evolved DBNs are combined to establish an ensemble model for RUL estimation, where combination weights are optimized via single-objective DE using the average training error as its objective. The proposed method was evaluated on NASA's C-MAPSS aeroengine data set which contains four sub-datasets. Experimental results demonstrated the outstanding performance of MODBNE in comparison with some existing approaches.

In our future studies, the proposed method will be extended to address multiple prognostic health states classification problems. Furthermore, more useful yet conflicting objectives will be considered (instead of mere accuracy and diversity) in the proposed method, which extends multiobjective optimization involved in MODBNE into many-objective optimization. Moreover, graphics processing unit computing-based [82] implementations of MODBNE will be investigated to accelerate its computational speed.

REFERENCES

[1] Z. S. Chen, Y. M. Yang, and Z. Hu, "A technical framework and roadmap of embedded diagnostics and prognostics for complex mechanical systems in prognostics and health management systems," *IEEE Trans. Rel.*, vol. 61, no. 2, pp. 314–322, Jun. 2012.

[2] B. D. Youn, C. Hu, and P. Wang, "Resilience-driven system design of complex engineered systems," *J. Mech. Design*, vol. 133, no. 10, p. 101011, 2011.

[3] M. Pecht and J. Gu, "Physics-of-failure-based prognostics for electronic products," *Trans. Inst. Meas. Control*, vol. 31, nos. 3–4, pp. 309–322, Jun./Aug. 2009.

[4] F. O. Heimes, "Recurrent neural networks for remaining useful life estimation," in *Proc. Int. Conf. Prognostics Health Manage. (PHM)*, Oct. 2008, pp. 1–6.

[5] A. Heng, S. Zhang, A. C. C. Tan, and J. Mathew, "Rotating machinery prognostics: State of the art, challenges and opportunities," *Mech. Syst. Signal Process.*, vol. 23, no. 3, pp. 724–739, 2009.

[6] B. Saha, K. Goebel, and J. Christophersen, "Comparison of prognostic algorithms for estimating remaining useful life of batteries," *Trans. Inst. Meas. Control*, vol. 31, nos. 3–4, pp. 293–308, Jun./Aug. 2009.

[7] Z. Tian, L. Wong, and N. Safaei, "A neural network approach for remaining useful life prediction utilizing both failure and suspension histories," *Mech. Syst. Signal Process.*, vol. 24, no. 5, pp. 1542–1555, Jul. 2010.

[8] T. Wang, J. Yu, D. Siegel, and J. Lee, "A similarity-based prognostics approach for remaining useful life estimation of engineered systems," in *Proc. Int. Conf. Prognostics Health Manage. (PHM)*, Oct. 2008, pp. 1–6.

[9] M. Schwabacher, "A survey of data-driven prognostics," in *Proc. AIAA Inf. Aerosp. Conf.*, 2005, pp. 1–5.

[10] G. L. Plett, "Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs: Part 3. State and parameter estimation," *J. Power Sour.*, vol. 134, no. 2, pp. 277–292, 2004.

[11] E. Zio and F. Di Maio, "A data-driven fuzzy approach for predicting the remaining useful life in dynamic failure scenarios of a nuclear system," *Rel. Eng. Syst. Safety*, vol. 95, no. 1, pp. 49–57, Jan. 2010.

[12] S. Saha, B. Saha, A. Saxena, and K. Goebel, "Distributed prognostic health management with Gaussian process regression," in *Proc. Aerosp. Conf.*, Mar. 2010, pp. 1–8.

[13] D. R. Garvey and J. W. Hines, "An integrated fuzzy inference-based monitoring, diagnostic, and prognostic system for intelligent control and maintenance," in *Foundations of Generic Optimization*. The Netherlands: Springer, 2008, pp. 203–222.

[14] C. Hu, B. D. Youn, P. Wang, and J. T. Yoon, "Ensemble of data-driven prognostic algorithms for robust prediction of remaining useful life," *Rel. Eng. Syst. Safety*, vol. 103, pp. 120–135, Jul. 2012.

[15] P. Lim, C. K. Goh, K. C. Tan, and P. Dutta, "Estimation of remaining useful life based on switching Kalman filter neural network ensemble," in *Proc. Int. Conf. Prognostics Health Manage. (PHM)*, 2014, pp. 2–9.

[16] P. Lim, C. K. Goh, K. C. Tan, and P. Dutta, "Multimodal degradation prognostics based on switching Kalman filter ensemble," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. PP, no. 99, pp. 1–13, 2016.

[17] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[18] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009.

[19] X. Qiu, J.-X. Xu, K. C. Tan, and H. A. Abbass, "Adaptive cross-generation differential evolution operators for multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 232–244, Apr. 2016.

[20] J. J. Liang, S. Baskar, P. N. Suganthan, and A. K. Qin, "Performance evaluation of multiagent genetic algorithm," *Natural Comput.*, vol. 5, no. 1, pp. 83–96, Mar. 2006.

[21] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[22] F. Rosenblatt, "Principles of neurodynamics: Perceptrons and the theory of brain mechanisms," Cornell Aeronautical Lab Inc., Buffalo, NY, USA, Tech. Rep. VG-1196-G-8, 1961.

[23] R. S. Sutton, "Two problems with backpropagation and other steepest-descent learning procedures for networks," in *Proc. 8th Annu. Conf. Cognit. Sci. Soc.*, 1986, pp. 823–831.

[24] A. K. Qin and X. Li, "Differential evolution on the CEC-2013 single-objective continuous optimization testbed," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2013, pp. 1099–1106.

[25] G. Niu, B.-S. Yang, and M. Pecht, "Development of an optimized condition-based maintenance system by data fusion and reliability-centered maintenance," *Rel. Eng. Syst. Safety*, vol. 95, no. 7, pp. 786–796, Jul. 2010.

[26] O. Fink, E. Zio, and U. Weidmann, "Predicting component reliability and level of degradation with complex-valued neural networks," *Rel. Eng. Syst. Safety*, vol. 121, pp. 198–206, Jan. 2014.

[27] N. Gebraeel, M. Lawley, R. Liu, and V. Parmeshwaran, "Residual life predictions from vibration-based degradation signals: A neural network approach," *IEEE Trans. Ind. Electron.*, vol. 51, no. 3, pp. 694–700, Jun. 2004.

[28] R. Huang, L. Xi, X. Li, C. R. Liu, H. Qiu, and J. Lee, "Residual life predictions for ball bearings based on self-organizing map and back propagation neural network methods," *Mech. Syst. Signal Process.*, vol. 21, no. 1, pp. 193–207, Jan. 2007.

[29] S. G. Pierce, K. Worden, and A. Bezazi, "Uncertainty analysis of a neural network used for fatigue lifetime prediction," *Mech. Syst. Signal Process.*, vol. 22, no. 6, pp. 1395–1411, Aug. 2008.

[30] L. Peel, "Data driven prognostics using a Kalman filter ensemble of neural network models," in *Proc. Int. Conf. Prognostics Health Manage. (PHM)*, Oct. 2008, pp. 1–6.

[31] A. M. Riad, H. K. Elminir, and H. M. Elattar, "Evaluation of neural networks in the subject of prognostics as compared to linear regression model," *Int. J. Eng. Technol.*, vol. 10, no. 6, pp. 50–56, 2010.

[32] K. Javed, R. Gouriveau, and N. Zerhouni, "A new multivariate approach for prognostics based on extreme learning machine and fuzzy clustering," *IEEE Trans. Cybern.*, vol. 45, no. 12, pp. 2626–2639, Dec. 2015.

[33] C. Chen, B. Zhang, G. Vachtsevanos, and M. Orchard, "Machine condition prediction based on adaptive neuro–fuzzy and high-order particle filtering," *IEEE Trans. Ind. Electron.*, vol. 58, no. 9, pp. 4353–4364, Sep. 2011.

[34] R. Ishibashi and C. L. N. Júnior, "GFRBS-PHM: A genetic fuzzy rule-based system for PHM with improved interpretability," in *Proc. Int. Conf. Prognostics Health Manage. (PHM)*, Jun. 2013, pp. 1–7.

[35] C. Hu, B. D. Youn, and T. Kim, "Semi-supervised learning with co-training for data-driven prognostics," in *Proc. IEEE Conf. Prognostics Health Manage. (PHM)*, Jun. 2012, pp. 1–10.

[36] G. Brown, J. Wyatt, R. Harris, and X. Yao, "Diversity creation methods: A survey and categorisation," *Inf. Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

[37] F. Fernández-Navarro, P. A. Gutiérrez, C. Hervás-Martánez, and X. Yao, "Negative correlation ensemble learning for ordinal regression," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 11, pp. 1836–1849, Nov. 2013.

[38] J. Yang, X. Zeng, S. Zhong, and S. Wu, "Effective neural network ensemble approach for improving generalization performance," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 6, pp. 878–887, Jun. 2013.

[39] R. Coop, A. Mishtal, and I. Arel, "Ensemble learning in fixed expansion layer networks for mitigating catastrophic forgetting," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 10, pp. 1623–1634, Oct. 2013.

[40] T. G. Dietterich, "Machine-learning research," *AI Mag.*, vol. 18, no. 4, p. 97, 1997.

[41] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 10, pp. 993–1001, Oct. 1990.

[42] A. Krogh and J. Vedelsby, "Neural network ensembles, cross validation, and active learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 1995, pp. 231–238.

[43] D. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *J. Artif. Intell. Res.*, vol. 11, pp. 169–198, Aug. 1999.

[44] A. Chandra and X. Yao, "Ensemble learning using multi-objective evolutionary algorithms," *J. Math. Model. Algorithms*, vol. 5, no. 4, pp. 417–445, Dec. 2006.

[45] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "An efficient approach to nondominated sorting for evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 201–213, Apr. 2015.

[46] S. B. Gee, K. C. Tan, V. A. Shim, and N. R. Pal, "Online diversity assessment in evolutionary multiobjective optimization: A geometrical perspective," *IEEE Trans. Evol. Comput.*, vol. 19, no. 4, pp. 542–559, Aug. 2015.

[47] B. Chen, W. Zeng, Y. Lin, and D. Zhang, "A new local search-based multiobjective optimization algorithm," *IEEE Trans. Evol. Comput.*, vol. 19, no. 1, pp. 50–73, Feb. 2015.

[48] M. Z. Ali, P. N. Suganthan, R. G. Reynolds, and A. F. Al-Badarneh, "Leveraged neighborhood restructuring in cultural algorithms for solving real-world numerical optimization problems," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 218–231, Apr. 2016.

[49] X. Ma *et al.*, "A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 275–298, Apr. 2016.

[50] Y. Yuan, H. Xu, B. Wang, B. Zhang and X. Yao, "Balancing convergence and diversity in decomposition-based many-objective optimizers," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 180–198, Apr. 2016.

[51] Y. Liu and X. Yao, "Ensemble learning via negative correlation," *Neural Netw.*, vol. 12, no. 10, pp. 1399–1404, Dec. 1999.

[52] H. A. Abbass, "A memetic Pareto evolutionary approach to artificial neural networks," in *AI 2001: Advances in Artificial Intelligence*. Berlin, Germany: Springer, 2001, pp. 1–12.

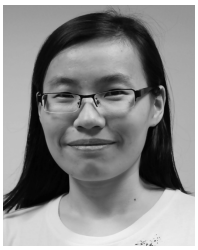
[53] H. A. Abbass, "Pareto neuro-evolution: Constructing ensemble of neural networks using multi-objective optimization," in *Proc. Congr. Evol. Comput. (CEC)*, vol. 3. Dec. 2003, pp. 2074–2080.

[54] O. W. Laslett, A. R. Mills, M. A. Zaidan, and R. F. Harrison, "Fusing an ensemble of diverse prognostic life predictions," in *Proc. IEEE Aerosp. Conf.*, Mar. 2014, pp. 1–10.

[55] L. Szymanski and B. McCane, "Deep networks are effective encoders of periodicity," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 10, pp. 1816–1827, Oct. 2014.

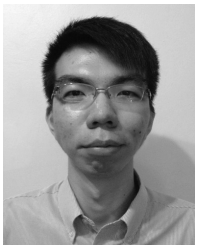
[56] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proc. 26th Int. Conf. Mach. Learn.*, 2009, pp. 609–616.

[57] A.-R. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 20, no. 1, pp. 14–22, Jan. 2012.
[58] A.-R. Mohamed, T. N. Sainath, G. Dahl, B. Ramabhadran, G. E. Hinton, and M. A. Picheny, "Deep belief networks using discriminative features for phone recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2011, pp. 5060–5063.
[59] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
[60] G. Hinton, "A practical guide to training restricted Boltzmann machines," *Momentum*, vol. 9, no. 1, p. 926, 2010.
[61] Y. Liu, X. Yao, and T. Higuchi, "Evolutionary ensembles with negative correlation learning," *IEEE Trans. Evol. Comput.*, vol. 4, no. 4, pp. 380–387, Nov. 2000.
[62] H. Chen and X. Yao, "Multiobjective neural network ensembles based on regularized negative correlation learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 12, pp. 1738–1751, Dec. 2010.
[63] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 284–302, Apr. 2009.
[64] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.
[65] A. Saxena and K. Goebel. (2008). *Turbofan Engine Degradation Simulation Data Set*, NASA Ames Prognostics Data Repository. [Online]. Available: http://ti.arc.nasa.gov/project/prognostic-data-repository
[66] C. Zhang, J. H. Sun, and K. C. Tan, "Deep belief networks ensemble with multi-objective optimization for failure diagnosis," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2015, pp. 32–37.
[67] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *Proc. Int. Conf. Prognostics Health Manage. (PHM)*, Oct. 2008, pp. 1–9.
[68] J. B. Coble and J. W. Hines, "Prognostic algorithm categorization with PHM Challenge application," in *Proc. Int. Conf. Prognostics Health Manage. (PHM)*, Oct. 2008, pp. 1–11.
[69] P. Wang, B. D. Youn, and C. Hu, "A generic probabilistic framework for structural health prognostics and uncertainty management," *Mech. Syst. Signal Process.*, vol. 28, pp. 622–637, Apr. 2012.
[70] A. Saxena *et al.*, "Metrics for evaluating performance of prognostic techniques," in *Proc. Int. Conf. Prognostics Health Manage. (PHM)*, Oct. 2008, pp. 1–17.
[71] E. Ramasso and A. Saxena, "Review and analysis of algorithmic approaches developed for prognostics on CMAPSS dataset," in *Proc. Annu. Conf. Prognostics Health Manage. Soc. (PHM)*, 2014, pp. 612–622.
[72] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, nos. 1–3, pp. 489–501, 2006.
[73] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statist. Comput.*, vol. 14, no. 3, pp. 199–222, Aug. 2004.
[74] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc. B (Methodol.)*, vol. 58, no. 1, pp. 267–288, 1996.
[75] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001.
[76] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, Aug. 1998.
[77] P. Tamilselvan, Y. Wang, and P. Wang, "Deep belief network based state classification for structural health diagnosis," in *Proc. IEEE Aerosp. Conf.*, Mar. 2012, pp. 1–11.
[78] P. Tamilselvan and P. Wang, "Failure diagnosis using deep belief learning based health state classification," *Rel. Eng. Syst. Safety*, vol. 115, pp. 124–135, Jul. 2013.
[79] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi, "Time series forecasting using a deep belief network with restricted Boltzmann machines," *Neurocomputing*, vol. 137, pp. 47–56, Aug. 2014.
[80] J. Xu, Y. Wang, and L. Xu, "PHM-oriented integrated fusion prognostics for aircraft engines based on sensor data," *IEEE Sensors J.*, vol. 14, no. 4, pp. 1124–1132, Apr. 2014.
[81] P. Baraldi, M. Compare, S. Sauco, and E. Zio, "Ensemble neural network-based particle filtering for prognostics," *Mech. Syst. Signal Process.*, vol. 41, nos. 1–2, pp. 288–300, 2013.
[82] A. K. Qin, F. Raimondo, F. Forbes, and Y. S. Ong, "An improved CUDA-based implementation of differential evolution on GPU," in *Proc. 14th Annu. Conf. Genet. Evol. Comput. (GECCO)*, New York, NY, USA, 2012, pp. 991–998.
[83] J. Hu, H. Tang, K. C. Tan, and H. Li, "How the brain formulates memory: A spatio-temporal model research frontier," *IEEE Comput. Intell. Mag.*, vol. 11, no. 2, pp. 56–68, May 2016.



Chong Zhang received the B.Eng. degree in engineering from the Harbin Institute of Technology, Harbin, China, in 2011, and the M.Sc. degree from the National University of Singapore, Singapore, in 2012, where she is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering.

She is a Research Engineer with the National University of Singapore, Singapore. Her current research interests include computational intelligence in data analytics, multiobjective optimization, deep learning, machine learning, and applications in prognostics and diagnostics.



Pin Lim received the B.Eng. degree (Hons.) in engineering science from the National University of Singapore, Singapore, in 2012.

He is currently enrolled under the Industrial Post-Graduate Program in electrical and computer engineering with the National University of Singapore, and being part of the Computational Engineering Team with the Advanced Technology Centre, Rolls-Royce Singapore, Singapore. His current research interests include machine learning and its applications in condition-based maintenance and prognostic systems for aerospace applications.

A. K. Qin (S'06–M'07–SM'12) received the B.Eng. degree from Southeast University, Nanjing, China, in 2001, and the Ph.D. degree from Nanyang Technological University, Singapore, in 2007.

He was with the University of Waterloo, Waterloo, ON, Canada, and the French Institute for Research in Computer Science and Automation, France, from 2007 to 2012. He is currently a Lecturer with RMIT University, Melbourne, VIC, Australia. He has authored over 60 publications. His current research interests include evolutionary computation, machine learning, image processing, GPU computing, and service computing.

Dr. Qin received the 2012 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award, the 2012 Genetic and Evolutionary Computation Conference Best Paper Nominee, and the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems overall best paper award. Two of his co-authored journal papers are the most cited ones (Thomson Reuters) among papers published in the 2006 and 2009 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION. He is currently the Chair of the IEEE Computational Intelligence Society task force on collaborative learning and optimization.

Kay Chen Tan (SM'08–F'14) received the B.Eng. degree (Hons.) in electronics and electrical engineering and the Ph.D. degree from the University of Glasgow, Glasgow, U.K., in 1994 and 1997, respectively.

He is currently an Associate Professor with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore. He has authored over 100 journal papers and over 100 papers in conference proceedings, and co-authored five books. His current research interests include computational and artificial intelligence, with applications to multi-objective optimization, scheduling, automation, data mining, and games.

Dr. Tan was a recipient of the 2012 IEEE Computational Intelligence Society Outstanding Early Career Award for his contributions to evolutionary computation in multiobjective optimization. He was the Editor-in-Chief of the *IEEE Computational Intelligence Magazine* from 2010 to 2013. He is currently the Editor-in-Chief of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION. He serves as an Associate Editor/Editorial Board Member of over 15 international journals, such as the IEEE TRANSACTIONS ON CYBERNETICS, the IEEE TRANSACTIONS ON COMPUTATIONAL INTELLIGENCE AND AI IN GAMES, *Evolutionary Computation*, the *European Journal of Operational Research*, the *Journal of Scheduling*, and the *International Journal of Systems Science*.