An Optimal Stacking Ensemble for Remaining Useful Life Estimation of Systems Under Multi-Operating Conditions

FEI LI, LI ZHANG, BIN CHEN, DIANZHU GAO, YIJUN CHENG, XIAOYONG ZHANG, YINGZE YANG, KAI GAO, AND ZHIWU HUANG, (Member, IEEE)

1School of Automation, Central South University, Changsha 410083, China
2School of Computer Science and Engineering, Central South University, Changsha 410083, China
3School of Automotive and Mechanical Engineering, Changsha University of Science and Technology, Changsha 410076, China

Corresponding authors: Dianzhu Gao (gaodianzhu@csu.edu.cn) and Zhiwu Huang (hzw@csu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61672537, Grant 61873353, and Grant 61672539.

ABSTRACT Remaining useful life (RUL) estimation is expected to provide appropriate maintenance for components or systems in industry to improve the reliability of the systems. Most data-based methods are limited to a single model, which is susceptible to various factors like environmental variability and diversity of operating conditions. In this paper, we propose an optimal stacking ensemble method combining different learning algorithms as meta-learners to mitigate the impact of multi-operating conditions. The selection of meta-learners follows a multi-objective evolutionary algorithm named non-dominated sorting genetic algorithms-II to balance the two conflicting objectives in terms of accuracy and diversity. Then the eventually evolved meta-learners are integrated by the meta-classifier for RUL estimation. In addition, a long-short-term feature extraction strategy is proposed to capture more degradation information from lifecycle data dynamically. Extensive experiments are performed on aero-engine dataset and battery dataset provided by NASA, which achieves the higher prognostic accuracy compared with the single models and existing methods.

INDEX TERMS Remaining useful life (RUL), reliability, optimal stacking ensemble, meta-learners.

I. INTRODUCTION

Intelligent prognostic and health management (PHM) has been widely used in aircrafts, automobiles, nuclear power plants, dams, and other industrial fields [1]. It comprehensively utilizes the latest research of modern information and artificial intelligence technologies to provide a powerful tool to health management, which is significant in industries [2]. PHM includes fault detection and isolation [3], fault diagnosis [4], fault prediction and health management, etc. The remaining useful life (RUL) estimation is one of the main tasks for health management. It is carried out by monitoring and analyzing the health of a system or components, and is used to predict the remaining time before breaking down in a real-time fashion. This not only reduces maintenance costs but also improves the stability of a system.

The associate editor coordinating the review of this manuscript and approving it for publication was Cristian Zambelli.

In general, the methods of RUL estimation for engineered systems are divided into three categories: model-based methods, data-based methods, and hybrid methods [5]. Both the model-based [6] and hybrid methods [7], [8] usually use a mathematical model of a system based on its physical characteristics, but it is difficult for a complex system due to the lack of sufficient knowledge about physical mechanism in the process. On the contrary, data-based methods first build a mathematical input-output model, and then train it using historical run-to-failure data. Since run-to-failure data are easily acquired by various sensors, this kind of method can be implemented conveniently. The data-based methods are mainly classified into neural network-based methods, decision tree-based methods and other traditional methods (support vector machine, logic regression, etc.).

In the category of data-based methods, neural network-based methods have been widely used in the last
decade due to their strong function-approximation ability. In [9], the multilayer perceptron was used to predict the RUL of turbofan engines. Gugulothu et al. [10] utilized a recurrent neural network to generate embeddings for time series subsequences. In the same year, the variant of recurrent neural network have also been utilized in RUL prediction [11], [12]. It was shown that a recurrent neural network was able to learn complex nonlinear degradation modes and also to take into consideration of the characteristics of a time series in degenerate data. Similarly, a convolution neural network was used to predict RUL in [13] by combining a time window proposed in [14]. The convolution kernel of a convolution neural network effectively extracted the useful features of time-window data. However, sometimes the fitting ability of neural network-based methods are too strong to overfit the noise.

Decision tree-based methods are another major class of data-based methods. They feature interpretability and effectiveness in solving prediction problems. In contrast to that neural network-based methods are prone to overfitting the noise, they are very robust to outliers due to their tree structure and robust loss function. Gradient boosting and random forest algorithms were used to predict the RUL of a turbofan engine and yielded good prediction accuracy [15]. A variant of the gradient boosting method, the light gradient boosting machine was used to estimate RUL, which is characterized by the rapidity of the method and the redundancy of noise [16]. But the tree structure can be insensitive to the slight degradation in the time series data.

As for other traditional methods, they can be called shallow learning since they always mine shallow relationships between features and targets. Khelif et al. [17] adopted support vector machine to predict the RUL of the engine system, which has a positive impact on the accuracy of the prediction while reducing the calculation time. Yu [18] utilized an liner regression model with varying moving window to fit the global degradation trend of lithium-ion batteries. But the traditional methods can’t catch the relationship among features and targets as deep as neural network-based methods.

Although a single model is easy, it usually suffers a setback for its one-sidedness, domain unity, and intrinsic property. Consider the range fluctuation in time series data caused by unstable environmental variability (e.g., temperature, relative humidity, pressure) [19], diversity of operating modes (e.g., loading, usage, rotatory speed) [20], nonlinear degradation modes with noise [21], it is difficult to use a single model mapping from signal features to RUL values in a multi-noise and time-varying environment [22]. While the combination of multiple models has the potential for reducing the influence of range fluctuation in degradation data. The combining method is referred to as Ensemble.

Ensemble can be divided into homogeneous ensemble and heterogeneous ensemble according to whether it is composed of the same algorithm [23]. Homogeneous ensemble includes bagging and boosting. It consists of the same models with different structures, resulting in it’s dependence on models that are heavily influenced by parameters. In contrast, stacking [24] is a heterogeneous ensemble method. It combines meta-learners (level-0) by a meta-classifier (level-1). Meta-learners can be any number of different algorithms and meta-classifier is a algorithm to combine meta-learners. Note that the application of different models enables us to observe data from different data spaces and structures. Even though different models may have similar error rates, they tend to make different mistakes, since they get different professions [25]. In this paper, stacking ensemble is utilized. Singh et al. [26] simply combined the gradient boosting decision tree and multilayer perceptron methods by stacking and obtained a better result. Hu et al. [21] selected five algorithms to make an ensemble to predict the RUL of engines. However, they don’t give the sound reason for choosing these meta-learners, thus the RUL estimation in other cases may not have the same performance. And the utilization of the stacking method for RUL estimation is still in early phase.

There are many researches devoted to the selection of meta-learners. Literature [27] adopted prediction accuracy as objective, optimized by artificial bee colony algorithm to collect meta-learners. In [28], ant colony algorithm was applied to optimize Local information, which represented the precisions of the base-level classifiers to configure stacking ensembles. But the single-objective optimization algorithms usually adopt a greedy search strategy that easily leads to local minimum. It doesn’t take much accuracy improvement but excess meta-learners. Literature [29] adopted a multi-objective optimization algorithm named non-dominated sorting genetic algorithms-II (NSGA-II) to evolve a ensemble and the result is averaged by each individual. It maximizes the generalization capacity of the ensemble and minimize its structural complexity simultaneously to get a better ensemble.

While literature [30] and [31] describe that the ideal ensemble is constructed using learners of small error and good diversity. However, rich diversity may cause the predicted value of meta-learners to deviate from the true values, and the improvement of individual accuracy often reduces the diversity of meta-learners, that is, accuracy and diversity are usually conflicting to each other. Thus in this paper, in order to get the ideal meta-learners for stacking, The NSGA-II is adopted to maximize accuracy and diversity simultaneously. NSGA-II is regarded as one of the most excellent multi-objective evolutionary algorithms and has been widely used in many fields so far [32]. It features for it’s fast and accurate search performance. The main contributions of this paper are as follows:

- The NSGA-II selects meta-learners. It features maximizing both accuracy and diversity and selects the most suitable models as meta-learners, avoiding falling in a local minimum and reducing computational expense.
- In the framework of optimal stacking, different models observe data from different data spaces and structures, reducing the influence of multi-noise and
time-varying environments and improving the accuracy and generalization of the global model.
• More effective features are extracted including short-term features and long-term features by a long-short-term feature extraction strategy.

The rest of this paper is organized as follows. The framework of the optimal stacking method for RUL estimation is presented in Section II. C-MAPSS dataset and battery dataset provided by NASA are applied to verified the effectiveness of the proposed method in Section III and Section IV, respectively. Conclusions are drawn in Section V.

II. RUL ESTIMATION METHOD

The framework of the optimal stacking ensemble method for remaining useful life estimation of systems under multi-operating conditions is introduced in this section, which is illustrated in Fig. 1. It has four major steps: (i) Data processing. (ii) Model selecting using NSGA-II. (iii) Model fusing using stacking. (iv) RUL estimating. Data processing includes sensor signals selection, sensor signals normalization and feature extraction, which are prepared for training and testing dataset. After data processing, the NSGA-II algorithm is applied to select the most suitable meta-learners from candidate models. Finally, the selected meta-learners constitute a stacking to get the final RUL estimation.

A. DATA PROCESSING

Sensor signals are the direct way to obtain aging data. As system degrades, sensor signals have different performance, including ascending, descending, irregular, unchanged, etc. Only sensor signals that can reflect the degradation characteristics of the systems can be selected.

The range of these selected signals may be very different, which will have a great impact on the model training. Therefore, signals need to be normalized. Let selected sensor signals be denoted as $V_{N \times K} = [v_{ij}]_{N \times M}, i = 1, 2, \ldots, N, j = 1, 2, \ldots, K$. $N$ is the total number of observations and $K$ is the total number of selected sensor signals. In our research, standard normalization is used, which can convert it to standard normal distribution, as follows:

$$z_{ij} = \frac{v_{ij} - \bar{v}_j}{\sigma_j}$$  \hspace{1cm} (1)

$$\bar{v}_j = \frac{1}{N} \sum_{i=1}^{N} v_{ij}$$  \hspace{1cm} (2)

$$\sigma_j = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (v_{ij} - \bar{v}_j)^2}$$  \hspace{1cm} (3)

where $v_{ij}$ indicates the $j^{th}$ sensor signals of the observation $i^{th}$.

However, systems are always working under numbers of operating conditions, which means that operating conditions will switch back and forth. Fig. 2 shows the value of a sensor signal in a degradation process under different operating conditions. It seems that sensor signals no longer have obvious trends either upward or downward. It is found that these sensor signals are largely affected by the operating conditions, which make the changes caused by degradation over time unapparent. Therefore, sensor signals under each operating condition need to be separated and standardized. In addition, useful features are necessary to be extracted.

In feature extraction, we proposed a long-short-term feature extraction strategy to catch features that can reflect long-term and short-term trends in time series data. In health management case, raw data is usually from sensor signals and it is typical time series data. For most of machine learning methods, the time window strategy can be introduced to capture the degradation trend of the historical data instead of a single time point [14]. When a system or component operates in a constant environment, time window data may be sufficient to estimate health status. However, when the object operates in complex time-varying operating conditions and...
the failure mode is not uniform, modeling the degradation process of the system using only the time window data will lose the accuracy of the prediction. Expanding the time window sounds feasible, but the computation cost will increase. While the time window strategy can extract features of the limited time series, useful information in earlier historical data is discarded. Thus the features that can reflect long-term characteristic is extracted including the cumulation feature as well as the dynamic mean and variance of the raw data of all historical data under the same conditions. In this paper, four parts of features are extracted as follows:

- Time window data, which can well capture the short-term trend and improve the prediction accuracy.
- The cumulative number of times for each operating mode. There are different operating conditions in the datasets, which can be grouped by K-means algorithm. The first part of the new features is the cumulative number of times for different operating modes as long-term features.
- The current operation mode label (given by the trained K-means algorithm), which is a multi-dimensional vector after one-hot encoding.
- The dynamic mean and variance of all historical data under the current operating condition, which can well capture the long-term degraded information.

### B. MODEL SELECTING USING NSGA-II

In this section, the framework of model selecting using NSGA-II algorithm are introduced. We employ the NSGA-II to select the most suitable meta-learners by maximizing accuracy and diversity as two conflicting objectives. NSGA-II is regarded as one of the most excellent multiobjective evolutionary algorithms and has been widely used in many fields so far. It features for it’s fast and accurate search performance. The schematic diagram of NSGA-II for model selecting is illustrated in Fig. 3. The fivefold cross-validation is to get the predicted RUL of the training set. The evaluate objective functions are accuracy and diversity based on the predicted RUL and the actual RUL, while diversity measures the differences between meta-learners. Suppose there are $M$ individual algorithms which are selected by NSGA-II, for each individual $m$, the cost function $Cost_m$ is defined as:

$$Cost_m = E_m + \lambda D_m$$  \hspace{1cm} (4)

where $E_m$ represents the accuracy, and $D_m$ represents the diversity. $\lambda$ is the weight. Here Mean Square Error is adopted as an indicator of accuracy:

$$E_m = \frac{1}{N} \sum_{i=1}^{N} \left( p_i^m - r^i \right)^2$$  \hspace{1cm} (5)

where $r^i$ is the actual RUL of the $i$th training sample, and the $p_i^m$ is the predicted RUL obtained by the $m$th meta-learner for the $i$th training sample. $N$ is the number of samples. The correlation $D_m$ is defined as:

$$D_m = \frac{1}{N} \sum_{i=1}^{N} \left( p_i^m - p_i \right) \sum_{k \in M, k \neq m} \left( p_k^i - p_i \right)$$  \hspace{1cm} (6)

where $p_i^m$ and $p_k^i$ represent the predicted RUL values of the $m$th and $k$th meta-learners for the $i$th training instance, respectively. $p^i$ is the average predicted RUL value of the models in the ensemble. Reference [31] proves that good diversity can be achieved (if there is no bias) when the individual models are negatively correlated, which means the lower the $D_m$ is, the larger the diversity is. Since the two objective functions
have different magnitudes, normalization is required so that the algorithm does not favor a larger magnitudes. In this paper, we normalize the value of objective functions \( f, f \in \{E, D\} \), at each generation \( g \) as follows:

\[
\tilde{f}(X) = \frac{f(X)}{Z_f}
\]

where \( X \) represents candidate solution. \( Z_f \) is the normalization factor of each objective function which is calculated at each generation \( g \) by \( Z_f = \max_{\exists f \in \{1...2M\}} |f(X_f)| \).

The cost function for the ensemble can be the average of these individual algorithms’ cost:

\[
C = \frac{1}{M} \sum_{m=1}^{M} Cost_m
\]

where \( M \) is the number of models in the ensemble.

As literatures [29] explain, minimize \( C \) equals to minimize each individual \( Cost_m \). If we take \( \lambda = 1 \) then the training of an individual learner in stacking involves minimization of two terms: mean square error term and correlation term. However, minimizing the two terms simultaneously is conflicting as discussed earlier. Therefore, non-dominated sorting and crowding distance sorting are employed to find a suitable set of candidate solutions by balancing equations (5) and (6) after being normalized by (7). The solution space is built up by different algorithms. Each candidate solution in solution space includes mean square error of RUL and correlation of the trained model with the rest models in the ensemble in NSGA-II.

The non-dominated sorting sorts the population including parent and child into different non-dominated fronts, e.g., Front 1, Front 2, Front 3 and Front 4 in Fig 3. If \( X_f \) satisfies the following conditions, it belongs to the current front and is removed from the rest population. \[
\exists X_i (i = 1, 2, \ldots, M) \notin X^*
\]

\[
\forall f_1 \in \{E, D\}, \quad f_1(X_i) \leq f_1(X^*)
\]

\[
\land \exists f_2 \in \{E, D\}, \quad f_2(X_i) < f_2(X^*)
\]

where \( X^* \) is the population that does not include \( X_f \).

The crowding distance sorting sorts the population in the same front. The crowding distance of the sorting edge is set to infinity. For the rest individuals, can be caculated as:

\[
n_d[i] = n_d[i] + \frac{\tilde{f}[i + 1] - \tilde{f}[i - 1]}{\tilde{f}_{\text{max}} - \tilde{f}_{\text{min}}}
\]

where \( \tilde{f}[i] \) is the corresponding objective function value of the \( i \)th individual in \( L_{\text{rank}} \), \( \tilde{f}_{\text{max}} \) and \( \tilde{f}_{\text{min}} \) are the maximum and minimum values of the corresponding objective function.

The solutions of the NSGA-II algorithm are represented by the arrays of different algorithms. Each candidate algorithm has a position, therefore, to generate the offspring, genetic operator of mutation is adopted. The position of offspring is computed using

\[
x = x_{\text{min}} + \text{rand}(0, 1) \times (x_{\text{max}} - x_{\text{min}})
\]

where \( x_{\text{max}} \) and \( x_{\text{min}} \) represent the maximum and minimum positions.

In conclusion, the basic idea of the NSGA-II based model selecting for RUL estimation is: firstly, the initial population \( P_0 \) of size \( M \) is randomly generated. By genetic operators of mutation, the offspring population \( Q \) is generated, and the two populations are combined to form a population of size \( 2M \). Calculate the objective functions of accuracy \( E \) and diversity \( D \) for each individual in the population according to their predicted remaining useful life and actual remaining useful life. Secondly, after normalizing the value of two objective functions, do non-dominated sort to the combined population, and calculate the crowding distance of individuals in the same non-dominated layer. Then select the appropriate individuals to form a new parent population according to the non-dominated relationship and the crowding distance of the individual. Finally, randomly generate a new offspring populations by mutation. Repeats the above operations until the last generation. The details are shown in Algorithm 1.

It is an NP-hard problem if the best ensemble is searched by brute force. However, NSGA-II can reduce the complexity to \( O(nm^2) \) per iteration [32], where \( n \) is the number of objective function and \( m \) is the size of ensemble. It can produce a set of non-dominated candidate solutions during the iteration. This group is called the Pareto set. The set of objective function values corresponding to these non-dominated solutions is Pareto Front. All solutions located in the Pareto Front are not dominated by solutions other than Pareto Front, thus these non-dominated solutions have the least conflict than other solutions and can provide a better choice for decision makers.

C. MODEL FUSING USING STACKING

Methods for different model fusion include bagging, boosting and stacking. For problems of regression, the final prediction can be obtained by average [33] in bagging and weight in boosting. They simply linearizes the results of the individual models without introducing nonlinearities. The stacking method itself is not an independent machine learning method, but it is accomplished by combining multiple machine learners. It generates an ensemble based on two levels. At the basic level, different meta-learners are trained by K-fold cross-validation. Since different meta-learners can produce different predicted RUL in the same dataset, diversity is introduced. The final predicted RUL is generated by a meta-classifier, which uses the dataset generated by K-fold cross-validation at the basic level as the training instances.

For better fusion, in this paper, stacking is used, which can be seen in Fig. 1. Meta-learners are selected by the NSGA-II algorithm introduced in the last subsection. The meta-classifier uses a neural network with a hidden layer because the neural network model can introduce nonlinearity and one hidden layer can reduce time consumptions.

III. EXPERIMENTAL STUDY I

In this section, the experiment based on the proposed method is carried out on the C-MAPSS dataset provided by NASA.

VOLUME 8, 2020

31858
Algorithm 1 The NSGA-II Based Model Selecting for RUL Estimation

Input:
- Candidate model space: \( \{Model_1^G, Model_2^G, \ldots, Model_M^G\} \)
- \( M \): solution space (popSize), \( M \leq S \).
- \( E \) and \( D \): objectives to be minimized simultaneously.
- \( G \): Maximum number of generations.
- \( O \): the number of offspring.

Output:
- Selected models \( \{Model_1^G, Model_2^G, \ldots, Model_M^G\} \) in the ensemble.

1. Initialize:
   - Randomly initialize population \( P_0 = \{X_1, X_2, \ldots, X_M\} \) and initialize each individual’s rank value \( i_{\text{rank}} = 1 \)
2. Repeat for each episode \( t \leq G \):
   3. Generate a new generation of offspring \( Q_t \) by computing the number of model using \( x_i = x_{\text{min}} + \text{rand}(0, 1) \times (x_{\text{max}} - x_{\text{min}}) \), \( x_{\text{max}} \) and \( x_{\text{min}} \) represent the maximum and minimum number, \( i = 1, 2, \ldots, O \).
   4. Combining \( P_t \) and \( Q_t \) to produce a combined population \( R_t = P_t \cup Q_t \). Calculate the \( \tilde{f} \) for each function \( f \in (E, D) \), \( Z_t = \text{max} \{[1, ..., 2M] \cdot f(X^t_i) \} \).
   5. Non-dominated sorting for \( R_t \):
   6. Repeat if \( R_t \neq \emptyset \):
      7. Find the non-dominated solution set in the \( R_t \).
      8. Record it to the non-dominated front \( L_{\text{rank}} \) and remove it from the \( R_t, i_{\text{rank}} = i_{\text{rank}} + 1 \)
9. Calculate the individual distance \( n_d \):
   - Initialize \( n_d[i] = 0 \).
   - Set the individual on the sorting edge infinity.
   - Calculate the \( n_d[i] \) for the rest of the sorted \( \tilde{f} \):
     \[ n_d[i] = n_d[i] + \frac{\tilde{f}[i] - \tilde{f}[i-1]}{(\tilde{f}_{\text{max}} - \tilde{f}_{\text{min}})} \]
   where \( \tilde{f}[i] \) is the corresponding objective function value of the \( i \)th individual in \( L_{\text{rank}} \), \( \tilde{f}_{\text{max}} \) and \( \tilde{f}_{\text{min}} \) are the maximum and minimum values of the corresponding objective function.
10. Generate a new population \( P_{t+1} \) with size of \( M \) according to \( i_{\text{rank}} \) and \( n_d \).
11. end for

By comparing with various single methods, the superiority of the proposed method is illustrated.

### A. C-MAPSS DATASET

C-MAPSS dataset is the turbofan engine degradation simulation dataset provided by NASA [34]. It has four sub-datasets. Each sub-dataset contains a training dataset and a testing dataset. In order to simulate the real environment, Gaussian noise is added to each sub-dataset. Each sub-dataset either from the train or test file has 26 columns which include the engine number, the time cycle, 3 environment variables and 21 sensor signal variables. More sensor information can be found in [34]. The training dataset includes run-to-failure sensor records of multiple turbofan engines under different operating conditions and failure modes. Each engine unit has different degrees of wear from the initial start. As time progresses, the engine units begin to deteriorate, until they reach a system fault which is declared as an unhealthy time cycle. At the same time, the sensor records in testing dataset end before the system fault, and the purpose of this paper is to predict the RUL of these turbofan engines. For verification, the actual RUL value of the test engines is also given. In this study, the four sub-datasets are evaluated. The detailed information of the C-MAPSS dataset is presented in Table 1, and the four sub-datasets are denoted as FD001, FD002, FD003 and FD004 respectively.

| Trend       | Sensor data number |
|-------------|--------------------|
| Ascending   | 2,3,4,8,9,11,13,15,17 |
| Descending  | 7,12,20,21         |
| Irregular/unchanged | 1,5,6,10,14,16,18,19 |

### B. DATA PROCESSING

The dataset provided by NASA has 21 sensor signal variables. As system degrades, sensor signals have different performance. Reference [12] divided the trend into three categories listed in Table 2. 14 sensor measurements are selected with normalization and the irregular/unchanged sensor data is abandoned the same as [13], [15].

In this paper, the feature extracted by long-short term feature extraction strategy mentioned in section II serves as inputs, which contains time window data and other three additional parts. The cumulative value of operating conditions \( N_{\text{cum}} \) and the current operation mode label \( N_{\text{var}} \) are both six-dimensional vectors, and the mean and variance of the sensor signals in the same operating condition are both 14-dimensional vectors, represented by \( N_{\text{mean}} \) and \( N_{\text{var}} \) respectively. Assume that the time window size is \( N_{\text{tw}} \), and a time point has 14 sensor signals, then the size of the input feature is \( 14 \times N_{\text{tw}} + N_{\text{cum}} + N_{\text{var}} + N_{\text{mean}} + N_{\text{var}} \). Considering the minimum operating time cycle given by the test dataset, the maximum of time window size is 19 time cycles, which is also the default set of time window size. The influence of the time window size on the prediction result will be discussed in sub-section E.

For the label of remaining useful life, the way of piece-wise linear degradation model has been adopted since it has been verified to be feasible in [13]. As Eq. (12) shows, at the beginning of degradation, RUL value is limited at a maximum constant. With aging, RUL linear model starts to degrade at a

## Table 1. Information of the C-MAPSS datasets.

| Dataset      | FD001 | FD002 | FD003 | FD004 |
|--------------|-------|-------|-------|-------|
| Engine units for training | 100   | 260   | 100   | 249   |
| Engine units for testing  | 100   | 259   | 100   | 248   |
| Operating conditions    | 1     | 6     | 1     | 6     |
| Fault modes             | 1     | 1     | 2     | 2     |

## Table 2. Sensor data trends in first dataset.

| Trend       | Sensor data number |
|-------------|--------------------|
| Ascending   | 2,3,4,8,9,11,13,15,17 |
| Descending  | 7,12,20,21         |
| Irregular/unchanged | 1,5,6,10,14,16,18,19 |
certain point $R_{early}$. 

$$
\text{label} = \begin{cases} 
R_{early}, & \text{label}_r > R_{early} \\
\text{label}_r, & \text{label}_r \leq R_{early} 
\end{cases}
$$

(12)

where label$_r$ is the actual label of training sample for remaining useful life. $R_{early}$ is set to 125 for that it is proved to be a time cycle to get the best performance [13].

C. PERFORMANCE EVALUATION

Two metrics are used in this paper. First, Root Mean Square Error (RMSE) is used to evaluate the performance of the proposed method as follows:

$$
\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (r^i - p^i)^2}
$$

(13)

where $r$ is the actual RUL value, $p$ is the predict RUL value. $N$ is the number of testing instance. At the same time, the scoring function is also used as a performance metric. The function is presented in Eq. (14):

$$
s = \sum_{i=1}^{N} s_i
$$

$$
s_i = \begin{cases} 
e^{-\frac{d_i}{\alpha}} - 1, & d_i < 0 \\
e^{\frac{d_i}{\beta}} - 1, & d_i \geq 0.0 
\end{cases}
$$

(14)

where $d_i = p^i - r^i$. The scoring function has more penalty on late prediction than early prediction, since a late prediction may lead to severer disasters.

D. EXPERIMENTAL SETUP

In order to obtain optimal meta-learners, twelve different machine learning algorithms are prepared as candidates. The twelve algorithms are all state-of-the-art machine learning methods and get different professions. They are mainly divided into three kinds and are introduced as follows.

Neural network-based methods: Multilayer perceptron (MLP) is a forward-structured artificial neural network that uses back propagation techniques [14]. Convolution neural network (CNN) is a deep feed-forward artificial neural network that has been successfully applied to image recognition [13] Recurrent neural network (RNN) is an artificial neural network with internal feedback connections and feedforward connections between layers [35]. It applies more on time series issues. Extreme learning machine (ELM) is a generalized single-hidden layer feedforward neural network [36].

Decision tree-based methods: Random forest (RF) is a regression or classification algorithm based on multiple decision trees. Gradient boosting decision tree (GBDT) is an integrated algorithm based on residual iterative tree [26], which has been widely applied to click-through rate prediction. Extreme gradient boosting (XGBoost) [22] and Light gradient boosting machine (LightGBM) [16] are variants of GBDT. they have achieved good results on other prediction problems. Extra tree regression (ETR) is an algorithm randomly selects features for segmentation in comparison with RF.

Other machine learning methods: Support vector machine (SVM) can map the sample space into a high-dimensional feature space, to transform the problem of nonlinear separability in the original sample space to linear separability in feature space [17]. Least absolute shrinkage and selection operator (LASSO) performs variable selection and regularization while fitting a generalized linear model [37]. KNeighbors regression (KNR) [38] determines the regression values of the test samples by the values of the surrounding K training samples.

The parameter adjustment range of each candidate model is set to a commonly used range and the final setting of parameters is carried out by using grid searching. The value of popsize $M$, generation $G$, offspring $O$ and mutation rate for NSGA-II are 4, 20, 4, 0.5 respectively. The generation is set to 2 because NSGA-II basically converged after 20 iterations. The mutation rate does not have much effect on the result because it only affects the number of iterations to converge. Offspring is set to double the popsize in this paper, considering the number of candidate models. And the effect of the number of popsize on the experimental result is presented in Fig. 5. By the way, we allow at most two same models in an ensemble. Since the combination of datasets with multiple environments and single environments will lead to unbalanced datasets, two meta-classifiers are prepared for FD001, FD003 and FD002, FD004 respectively. The meta-classifier is a one-hidden-layer neural network with 12 neurons, the learning rate of which is 0.01. The result of each candidate individual is used for comparison. The results in experiment are taken the average of 10 trials to exclude the effects of random disturbances. All the experiments are conducted on a PC with Intel Core i7 CPU, 8-GB RAM and in the environment of Python and the operating system is Windows 10.

E. PERFORMANCE ANALYSIS

First, the single-objective artificial bee colony (ABC) algorithm proposed in the [27] will be used for comparison. The factor of time window size which may impact the performance of the proposed method is investigated. The impact of the feature extraction strategy proposed in this paper will also be discussed. Furthermore, the performance of the proposed method in comparison with many existing approaches on C-MAPSS dataset is evaluated.

1) COMPARISON WITH THE ARTIFICIAL BEE COLONY (ABC) ALGORITHM

To ensure fairness, the proposed method and ABC method are performed in the same batch of candidate models, the same stacking structure and the same hyper-parameter set for each meta-learner. We adopt one-hidden-layer neural network with 12 neurons as the second layer in stacking, but the number of hidden neurons will affect the performance. Fig. 4 shows the performance of proposed method with the varying number
F. Li et al.: Optimal Stacking Ensemble for RUL Estimation of Systems Under Multi-Operating Conditions

The prognostic performance and training time of proposed method and ABC algorithm with the varying number of hidden neurons for FD004 dataset.

(a) The effect of the number of hidden neurons on RMSE. (b) The effect of the number of hidden neurons on training time.

The effect of popsize on the prognostic performance.

FIGURE 5.

Table 3. Performance comparison with the ABC algorithm.

| Method                        | Number of meta-learners | FD001 | FD002 | FD003 | FD004 |
|-------------------------------|-------------------------|-------|-------|-------|-------|
| ABC algorithm                 |                         | 6     | 23.83 | 12.02 | 23.46 |
| Proposed method with popsize  |                         | 4     | 12.81 | 23.08 | 11.01 | 22.69 |
| Proposed method with pareto   |                         | 2     | 13.32 | 23.44 | 11.89 | 22.96 |

The effect of time window size on the prognostic performance.

FIGURE 6.

Another influential factor is the meta-learners. Increasing the number of meta-learners may improve global generalization, but excess may lead to overfitting. At the same time, the more the meta-learner, the higher the computing cost. In practice, the number of meta-learner is determined by the popsize. Fig. 5 shows the effect of popsize on the prognostic performance. It can be seen that the most suitable popsize is four with highest RMSE. At the same time, the computing time increases monotonously with the number of hidden neurons increases, but the impact on the ABC algorithm is greater, because the ABC algorithm uses the meta-classifier for each solution while NSGA-II algorithm dose not.

Another influential factor is the meta-learners. Increasing the number of meta-learners may improve global generalization, but excess may lead to overfitting. At the same time, the more the meta-learner, the higher the computing cost. In practice, the number of meta-learner is determined by the popsize. Fig. 5 shows the effect of popsize on the prognostic performance. It can be seen that the most suitable popsize is four with highest RMSE. At the same time, the computing time increases monotonously with the number of hidden neurons increases, but the impact on the ABC algorithm is greater, because the ABC algorithm uses the meta-classifier for each solution while NSGA-II algorithm dose not.

The effect of time window size is presented in Fig. 6. Four different sizes of time window are evaluated on the four sub-datasets. It can be seen from the figure that the larger the time window size becomes, the more accurate the prediction result will be, since more degradation information from historical data will be involved within each training. A continuous amount of data can overcome the interference caused by noise and many papers [13]–[15] have proved its practicality. But at the same time, the computing load increases from 683s to 1110s as the time window size grows.

2) Analysis of the Influence of Time Window Size

The effect of time window size is presented in Fig. 6. Four different sizes of time window are evaluated on the four sub-datasets. It can be seen from the figure that the larger the time window size becomes, the more accurate the prediction result will be, since more degradation information from historical data will be involved within each training. A continuous amount of data can overcome the interference caused by noise and many papers [13]–[15] have proved its practicality. But at the same time, the computing load increases from 683s to 1110s as the time window size grows.
TABLE 4. Performance comparison of 12 methods and proposed method without feature extraction on each of four C-MAPSS sub-datasets.

| TW=19 | Op-stacking | GBDT | XGBoost | LightGBM | MLP | RNN | CNN | SVM | RF | Lasso | ETR | KNR | ELM |
|-------|-------------|------|---------|----------|-----|-----|-----|-----|----|-------|-----|-----|-----|
| FD001 | RMSE: 15.50 | 16.91| 16.58   | 16.25    | 16.55| 17.23| 16.09| 18.91| 18.39| 19.81| 18.21| 21.64| 21.31|
| Score | 420.2       | 474.0| 415.7   | 387.4    | 455.6| 623.9| 440.1| 894.5| 574.7| 685.5| 574.3| 1402.8| 964.1|
| FD002 | RMSE: 25.69 | 26.77| 26.52   | 26.49    | 26.94| 27.62| 27.92| 29.26| 27.38| 31.26| 27.72| 28.88| 30.45|
| Score | 7280.2      | 9439.8| 8293.0 | 8987.1   | 7814.5| 8903.7| 11524.1| 17224.8| 8587.0| 16692.3| 6956.2| 10270.8| 13216.7|
| FD003 | RMSE: 15.63 | 16.39| 16.97   | 16.57    | 17.80| 17.96| 17.36| 20.95| 20.58| 21.36| 20.38| 23.16| 22.41|
| Score | 588.3       | 576.7| 603.5   | 665.4    | 887.6| 942.3| 788.7| 2401.1| 2128.1| 1775.17| 1721.13| 3207.9| 1782.5|
| FD004 | RMSE: 26.49 | 27.65| 27.37   | 27.27    | 28.64| 28.81| 28.90| 30.03| 29.89| 32.07| 29.95| 31.54| 33.13|
| Score | 4644.2      | 5486.6| 4943.0 | 5133.4   | 6582.1| 6947.5| 7185.2| 7369.0| 6706.6| 9830.0| 6821.2| 8861.5| 10428.6|

FIGURE 7. Comparison of Op-stacking, GBDT, XGBoost, LightGBM, MLP, RNN, CNN, SVM, RF, Lasso, ETR, KNR and ELM in terms of the box plot of the RMSEs over ten runs on four C-MAPSS sub-datasets, respectively.

from 1 to 19, that is because the dimension of the vectors inputted to models increases.

3) PERFORMANCE EVALUATION WITH TIME WINDOW STRATEGY
This part studies the proposed method in comparison with other twelve methods when the time window size of 19 is applied. Table 4 reports the average performances of optimal stacking (Op-stacking) with other 12 algorithms over ten runs on each of four C-MAPSS sub-datasets. The parameters of these models are obtained through grid search. It can be seen from the Table 4 that the RMSEs of the proposed method on four sub-datasets are lowest. Sometimes the scoring is higher when the RMSE is lower, since the scoring function has a greater penalty for the late prediction. Fig. 7 shows box plots of RMSE for 10 runs of 13 methods on four C-MAPSS sub-datasets. It can be observed that the proposed method has smaller median and variance.

4) THE EFFECTS OF PROPOSED FEATURE EXTRACTION STRATEGY
Fig. 8 shows the effect of the long-short-term feature extraction strategy proposed in this paper and only the time window strategy on the accuracy of the experiment. (a), (b), (c) and (d) are test units randomly selected in test sets FD001, FD002, FD003 and FD004, respectively. The abscissa is the working cycle, and the ordinate is the RUL. The black line represents the actual RUL of test engine unit. According to the definition of RUL described as eq.(12), when the actual RUL is less than the threshold labelr, the cycle increases by 1 then the actual RUL decreases by 1, and the label decreases linearly. It can be seen from the figure that the feature extraction strategy proposed in this paper is obviously better than merely using the time window feature, because with the cumulative sum of current operation and the average and the variance features, historical data outside the time window are taken into account and can provide more degradation information for the model. Furthermore, it can be seen that the predicted value becomes increasingly accurate as the engine degrades, since the fault signals become more obvious as the degree of degradation increases. The accurate prediction in the later period can greatly improve the security and reliability of the system. The RMSE and scoring of stacking with the long-short term feature extraction strategy are given in Table 5.

5) PERFORMANCE EVALUATION WITH PROPOSED FEATURE EXTRACTION STRATEGY
This part studies the proposed method in comparison with other twelve methods when the long-short-term feature extraction strategy is applied. Table 5 reports the average performances of Op-stacking with other 12 algorithms over ten runs on each of four C-MAPSS sub-datasets. It can be observed that LightGBM performs worse than MLP on the first sub-dataset while better than other three sub-datasets, indicating that LightGBM predicts better in multiple operating environments and multiple fault modes, while MLP performs better in single operating environment and single fault mode. By combining two methods through stacking, the accuracy of the global model is better than the single model in terms of RMSE and scores for each sub-dataset. The RMSE and scoring of KNR and ELM are both high, meaning
that they are not suitable for the turbofan engine with strong nonlinear and time-dependent characteristics, which cannot capture the complex relationship between the sensor data and RUL within a comparatively plain structure. Fig. 9 shows box plots of RMSE for 10 runs of 13 methods on four C-MAPSS sub-datasets. It can be observed that the proposed method has smaller median and variance and has significant improvement on FD002 and FD004 under multi-operating environments.

To better analyze the prognostic results, Fig. 10 presents the sorted actual RUL of testing engine instances in four testing datasets with predicted RUL of four selected meta-learners and proposed method correspondingly. The abscissa is the index of test unit which is sorted in ascending order by actual RUL. Hence on the right of the figure, the test units run from comparable health status, while close to the left of the figure, the test units degenerate gradually. It can be seen from the Fig. 10 that each model has different advantages of performance, such as LightGBM and Xgboost are more accurate in the later period of degradation, while MLP is more accurate in earlier period of degradation, and Lasso has greater fluctuation but introduces diversity into ensemble. It is because that the proposed NSGA-II operator selects the meta-learners on optimal non-dominated candidate solutions considering their accuracy and diversity. And as box plots shows, LightGBM predicts better in multiple operating environments and multiple fault modes, while MLP performs better in single operating environment and single fault mode. Op-stacking is adopted to further
FIGURE 9. Comparison of Op-stacking, GBDT, XGBoost, LightGBM, MLP, RNN, CNN, SVM, RF, Lasso, ETR, KNR and ELM in terms of the box plot of the RMSEs over ten runs on four C-MAPSS sub-datasets, respectively.

FIGURE 10. The sorted RUL prediction of four selected meta-learners and proposed method for four testing datasets.

obtain nonlinear features for better prediction results. And the predicted RUL of Op-stacking is closer to the actual RUL than four meta-learners in four datasets. Furthermore, the prediction biases of MLP, LASSO, and LightGBM are different in some places, and at these places, the prediction of Op-stacking is always more accurate than meta-learners. For example, when the prediction of MLP is below actual RUL while the prediction of Lasso is above actual RUL, the results of Op-stacking are better than any meta-learners, indicating that different prediction biases of different models are also helpful for stacking.

6) COMPARISON WITH PUBLICLY AVAILABLE RESULTS
Some state-of-the-art works on RUL estimation also used C-MAPSS datasets for performance evaluation and comparison. It can be seen in Table 6 that, in overall, ensemble methods perform better than single model. The literature [26] used the stacking method without choosing the meta-learner, which merely combined a GBDT and a MLP by the

| Method                | RMSE FD001 | FD002 | FD003 | FD004 |
|-----------------------|------------|-------|-------|-------|
| Echo state network with PSO [39] | 37.12 | N/A   | N/A   | N/A   |
| First attempt of deep CNN [40] | 18.45 | 30.29 | 19.82 | 29.16 |
| Random forest [15] | 17.91 | 29.59 | 20.27 | 31.12 |
| ELM [15] | 17.27 | 37.28 | 18.90 | 38.43 |
| Deep LSTM [11] | 16.14 | 24.49 | 16.18 | 28.17 |
| Stacking ensemble method [26] | 16.67 | 25.57 | 18.44 | 26.76 |
| GBDT [15] | 15.67 | 29.09 | 16.84 | 29.01 |
| Time window based NN [14] | 15.16 | N/A   | N/A   | N/A   |
| MODBNE [15] | 15.04 | 25.05 | 12.51 | 28.66 |
| Proposed method | **12.81** | **23.08** | **11.01** | **22.69** |
meta-classifier, so there is little improvement in accuracy. The literature [15] also finds base learners for ensemble through multiobjective evolutionary algorithm, but its base learners only limit to deep belief networks and cannot model the data from different perspectives, which is not good on FD002 and FD004. The proposed method can select the most suitable meta-learners with compromised accuracy and diversity, which has the highest accuracy on four sub-datasets.

IV. EXPERIMENTAL STUDY II

In this section, the experiment based on the proposed method is carried out on the battery dataset provided by NASA to verify the behavior of proposed method.

A. LITHIUM-ION BATTERY DATASET

The lithium-ion battery degradation simulation dataset is carried out by the NASA Ames PCoE [41]. It has performed accelerated life tests under different experimental conditions for each group of batteries. One of the groups is selected for the validation, which contains four batteries (B0005, B0006, B0007 and B0018). Experimental conditions include charge current, charge voltage, discharge current, cut-off voltage and ambient temperature. The experimental conditions of the four batteries are shown in the Table 7. The experiments end when the batteries reached 30% fade in rated capacity, which is deemed invalid (from 2 Ah to 1.4 Ah). The data collected from the experiments include battery surface temperature, load voltage and battery capacity.

B. DATA PROCESSING

Fig. 11(a) shows the terminal voltage of the lithium battery B0005 during discharge in different cycles. In this work, mean voltage falloff (MVF) [42] and energy of voltage signal (VSE) [43] are extracted as features. MVF is calculated according to the following Equation.

$$MVF_i = \frac{1}{N} \sum_{j=1}^{N} |V_j - V_r|$$

(15)

where $V_j$ is the $j$th voltage value within a specified time. $N$ is the number of sampling points in a specified time. $V_r$ is the rated voltage. The specified time is from 500 s to 1500 s where the voltage drop is flat. And VSE is calculated as:

$$VSE_i = \int_{-\infty}^{+\infty} |V(t)|^2 dt$$

(16)

where $V(t)$ is the voltage at time $t$. The features of battery B0005 changing with cycles are shown in the Fig. 11(b) and (c). It can be seen that they both have clear monotonous trends with degradation.

C. PERFORMANCE ANALYSIS

The performance evaluation and experimental setup are the same as the last section. The size of time window is 10. The popsize and evolved models of different training set are listed in Table 8.

Fig. 12 shows box plots of RMSE for 10 runs of 13 methods on four batteries. It can be observed that Op-stacking has the best performance on each test battery. The variance of decision tree-based methods is smaller, while the variance of neural network-based methods is larger. Decision tree-based methods perform better than other single models. The large RMSE of the neural network-based methods may be due to
FIGURE 12. Comparison of Op-stacking, GBDT, XGBoost, LightGBM, MLP, RNN, CNN, SVM, RF, Lasso, ETR, KNR and ELM in terms of the box plot of the RMSEs over ten runs on four batteries.

FIGURE 13. The prognostic performance of meta-learners and Op-stacking on B0018.

the over-fitting of the models since the number of training samples is small.

Test battery B0018 is used for further analysis. Fig. 13 shows the performance comparison between the proposed method and selected meta-learners. It can be seen that among the meta-learners, the prediction of ETR is closer to actual RUL. The overall prediction of MLP is below actual RUL, and SVM is above actual RUL, indicating that MLP and SVM can introduce diversity into ensemble. Through stacking, the final prediction is more accurate, especially in the middle and later periods of degradation. The proposed method can combine different prediction results of single models to get better performance. At the same time, it indicates that different prediction biases of different models are helpful for stacking once again.

V. CONCLUSION

In this paper, the optimal stacking method is proposed for the RUL prediction of engineered systems by evolving the most suitable meta-learners simultaneously subject to accuracy and diversity as two conflicting objectives. The selected meta-learners are combined through the meta-classifier of stacking. The C-MAPSS dataset and lithium-ion battery dataset provided by NASA are used to verify the proposed method. Sensor signals are selected, normalized in both datasets and processed by long-short-term feature extraction strategy in the C-MAPSS dataset before input. Good performance has been achieved on both training and testing datasets in comparison with a single-objective optimization algorithm and single models. The RMSE of C-MAPSS dataset under multi-operating conditions is improved by 2.215, and the Scoring is increased by 1617.6 on average. And the RMSE of lithium-ion battery dataset is increased by 5.45 on average. However, the proposed method needs to train all candidate models offline, so the computing cost for offline is expensive, and the algorithm convergence performance may be affected by some initial settings such as the size of population. Our future research will focus on multiple RUL estimation problems, and we will continue to improve the objective function, not limited to accuracy and correlation coefficient.

REFERENCES

[1] T. H. Loutas, D. Roulias, and G. Georgoulas, “Remaining useful life estimation in rolling bearings utilizing data-driven probabilistic E-support vectors regression,” *IEEE Trans. Rel.*, vol. 62, no. 4, pp. 821–832, Dec. 2013.

[2] Z. Zhao, B. Liang, X. Wang, and W. Lu, “Remaining useful life prediction of aircraft engine based on degradation pattern learning,” *Rel. Eng. Syst. Saf.*, vol. 164, pp. 74–83, Aug. 2017.

[3] B. Huang, K. Cohen, and Q. Zhao, “Sequential active detection of anomalies in heterogeneous processes,” 2017, arXiv:1704.00766. [Online]. Available: http://arxiv.org/abs/1704.00766v1

[4] B. Huang, K. Cohen, and Q. Zhao, “Active anomaly detection in heterogeneous processes,” *IEEE Trans. Inf. Theory*, vol. 65, no. 4, pp. 2284–2301, Apr. 2019.

[5] J. Lee, F. Wu, W. Zhao, M. Ghaffari, L. Liao, and D. Siegel, “Prognostics and health management design for rotary machinery systems—Reviews, methodology and applications,” *Mech. Syst. Signal Process.*, vol. 42, nos. 1–2, pp. 314–334, Jan. 2014.

[6] S. Li, H. Ma, T. K. Saha, Y. Yang, and G. Wu, “On particle filtering for power transformer remaining useful life estimation,” *IEEE Trans. Power Del.*, vol. 33, no. 6, pp. 2643–2653, Dec. 2018.

[7] K. T. Huynh, Y. Langeron, and A. Grall, “Degradation modeling and RUL estimation of deteriorating systems in S-plane,” *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 12249–12254, Jul. 2017.

[8] L. Liao and F. Kottig, “Review of hybrid prognostics approaches for remaining useful life prediction of engineered systems, and an application to battery life prediction,” *IEEE Trans. Rel.*, vol. 63, no. 1, pp. 191–207, Mar. 2014.

[9] 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. 06, pp. 52–58, 2010.
[10] N. Gugulothu, V. TV, P. Malhotra, L. Vig, P. Agarwal, and G. Shroff, “Predicting remaining useful life using time series embeddings based on recurrent neural networks,” 2017, arXiv:1709.01073. [Online]. Available: http://arxiv.org/abs/1709.01073.

[11] S. Zheng, K. Rostovski, A. Farahat, and C. Gupta, “Long short-term memory network for remaining useful life estimation,” in Proc. IEEE Int. Conf. Prognostics Health Manage., Jun. 2017, pp. 88–95.

[12] Y. Wu, M. Yuan, S. Dong, L. Lin, and Y. Liu, “Remaining useful life estimation of engineered systems using vanilla LSTM neural networks,” Neurocomputing, vol. 275, pp. 167–179, Jan. 2018.

[13] X. Li, Q. Ding, and J.-Q. Sun, “Remaining useful life estimation in prognostics using deep convolution neural networks,” Rel. Eng. Syst. Saf., vol. 172, pp. 1–11, Apr. 2018.

[14] P. Lim, K. G. Chi, and K. C. Tan, “A time window neural network based framework for remaining useful life estimation,” in Proc. Int. Joint Conf. Neural Netw., 2016, pp. 1746–1753.

[15] C. Zhang, P. Lim, A. K. Qin, and K. C. Tan, “Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics,” IEEE Trans. Neural Netw. Learn. Syst., vol. 28, no. 10, pp. 2306–2318, Oct. 2017.

[16] F. Li, L. Zhang, C. Bin, and G. Dianzhu, “A light gradient boosting machine for remaining useful life estimation of aircraft engines,” in Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC), Nov. 2018, pp. 3562–3567.

[17] R. Kheif, B. Chebel-Morello, S. Malinowski, E. Lajafl, F. Fnaiech, and N. Zerhouni, “Direct remaining useful life estimation based on support vector regression,” IEEE Trans. Ind. Electron., vol. 64, no. 3, pp. 2276–2285, Mar. 2017.

[18] Y. V. A. State of health prediction of lithium-ion batteries: Multiscale logic regression and Gaussian process regression ensemble,” Rel. Eng. Syst. Saf., vol. 174, pp. 82–95, Jun. 2018.

[19] K. Gao, F. Han, P. Dong, N. Xiong, and R. Du, “Connected vehicle as a mobile sensor for real time queue length at signalized intersections,” Sensors, vol. 19, no. 9, p. 509, May 2019.

[20] H. Li, X. Zhang, J. Peng, J. He, Z. Huang, and J. Wang, “Cooperative CC-CV charging of supercapacitors using multi-charger systems,” IEEE Trans. Ind. Electron., to be published, doi: 10.1109/tie.2019.2962485.

[21] C. Hu, B. D. Youn, P. Wang, and J. Taek Yoon, “Ensemble of data-driven prognostic algorithms for robust prediction of remaining useful life,” Rel. Eng. Syst. Saf., vol. 103, pp. 120–135, Jul. 2012.

[22] X. Zhang, P. Xiao, Y. Yang, Y. Cheng, B. Chen, D. Gao, W. Liu, and Z. Huang, “Remaining useful life estimation using CNN-XGB with extended time window,” IEEE Access, vol. 7, pp. 154386–154397, 2019.

[23] T. Nguyen, A. V. Luong, M. T. Dang, A. W.-C. Liew, and J. Mccall, “Ensemble selection based on classifier prediction confidence,” Pattern Recognit., vol. 100, Apr. 2020, Art. no. 107104.

[24] D. H. Wolpert, “Stacked generalization,” Neural Netw., vol. 5, no. 2, pp. 241–259, 1992.

[25] N. C. Oza and K. Tumer, “Classifier ensembles: Select real-world applications,” Inf. Fusion, vol. 9, no. 4, Jan. 2008.

[26] S. K. Singh, S. Kumar, and J. P. Dwivedi, “A novel soft computing method for engine RUL prediction,” Multimedia Tools Appl., vol. 78, no. 4, pp. 4065–4087, Feb. 2019.

[27] P. Shunmugapriya and S. Kannani, “Optimization of stacking ensemble configurations through artificial bee colony algorithm,” Swarm Evol. Comput., vol. 12, pp. 24–32, Oct. 2013.

[28] Y. Chen, M.-L. Wong, and H. Li, “Applying ant colony optimization to configuring stacking ensembles for data mining,” Expert Syst. Appl., vol. 41, no. 6, pp. 2688–2702, May 2014.

[29] R. Futurna, S. Curteanu, and F. Leon, “Multi-objective optimization of a stacked neural network using an evolutionary hyper-heuristic,” Appl. Soft Comput., vol. 12, no. 1, pp. 133–144, Jan. 2012.

[30] S. Mao, J.-W. Chen, L. Jiao, S. Gou, and R. Wang, “Maximizing diversity by transformed ensemble learning,” Appl. Soft Comput., vol. 82, Sep. 2019, Art. no. 105587.

[31] 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.

[32] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” IEEE Trans. Evol. Comput., vol. 6, no. 2, pp. 182–197, Apr. 2002.

[33] H. Erdal and I. Karahanoglu, “Bagging ensemble models for bank profitability: An empirical research on Turkish development and investment banks,” Appl. Soft Comput., vol. 49, pp. 861–867, Dec. 2016.
DIANZHU GAO received the B.Eng. degree from Southwest Jiaotong University, Chengtu, China, in 1999, and the M.S. degree from Central South University, Changsha, China, in 2009, where he is currently pursuing the Ph.D. degree with the School of Information Science and Engineering. His current research interests include intelligent control technology and intelligent transportation.

YIJUN CHENG received the B.S. degree from the School of Resources and Safety Engineering, Central South University, China, in 2015, where she is currently pursuing the Ph.D. degree with the School of Information Science and Engineering. Her current research interests include fault diagnosis, prognostics, and health management.

XIAOYONG ZHANG received the B.S., M.S., and Ph.D. degrees in computer application technology from Central South University, Changsha, China, in 2000, 2005, and 2009, respectively. He is currently an Associate Professor with the School of Information Science and Engineering, Central South University. His current research interests include intelligent transportation, train network control and prognostics, and health management.

YINGZE YANG received the B.S. and Ph.D. degrees in control theory and control engineering from Central South University, in 2006 and 2010, respectively. He is currently a Lecturer with the School of Information Science and Engineering, Central South University. His current research interests include communication performance optimization, distributed fault diagnosis, prognostics, and health management.

KAI GAO was born in Baoding, Hebei, China. He received the B.S. and Ph.D. degrees from Central South University, Changsha, China, in 2008 and 2014, respectively. He joined the Changsha University of Science and Technology, in 2015. His research interests include intelligent transportation system, vehicular networks, and cooperative control.

ZHIWU HUANG (Member, IEEE) received the B.S. degree from Xiangtan University, in 1987, the M.S. degree from the University of Science and Technology Beijing, in 1989, and the Ph.D. degree from Central South University, in 2006. From 2008 to 2009, he was with the School of Computer Science and Electronic Engineering, University of Essex, U.K., as a Visiting Scholar. He is currently a Professor with the School of Information Science and Engineering, Central South University, China. His research interests include fault diagnostic, co-operative control, and cloud computing.