Received: 31 August 2019 | Revised: 14 February 2020 | Accepted: 9 March 2020
DOI: 10.1002/cpe.5762

WILEY

SPECIAL ISSUE PAPER

Reliable machine prognostic health management in the presence of missing data

Yu Huang[1] | Yufei Tang[1] | James VanZwieten[2] | Jianxun Liu[3]

[1]Department of Computer & Electrical Engineering and Computer Science, Florida Atlantic University, Boca Raton, Florida

[2]Department of Civil, Environmental, and Geomatics Engineering, Florida Atlantic University, Boca Raton, Florida

[3]School of Computer Science and Engineering, Hunan University of Science and Technology, Hunan, China

Correspondence
Yufei Tang, Florida Atlantic University, Boca Raton, FL 33431.
Email: tangy@fau.edu

Summary

Prognostics and health management enables the prediction of future degradation and remaining useful life (RUL) for in-service systems based on historical and contemporary data, showing promise for many practical applications. One major challenge for prognostics is the common occurrence of missing values in time-series data, often caused by disruptions in sensor communication or hardware/software failures. Another major concern is that the sufficient prior knowledge of critical component degradation with a clear failure threshold is often not readily available in practice. These issues can significantly hinder the application of advanced signal and data analysis methods and consequently degrade the health management performance. In this article, we propose a novel data-driven framework that is capable of providing accurate and reliable predictions of degradation and RUL. In this approach, one-hot health state indicators are appended to the historical time series so that the model learns end-of-life automatically. A modified gate recurrent unit based variational autoencoder is employed in generative adversarial networks to model the temporal irregularity of the incomplete time series. Experiments on multivariate time-series datasets collected from real-world aeroengines verify that significant performance improvement can be achieved using the proposed model for robust long-term prognostics.

KEYWORDS

generative adversarial networks, missing data imputation, prognostics and health management, remaining useful life prediction

1 | INTRODUCTION

Performance deterioration over time of almost every system is contingent on various mechanisms. Therefore, maintenance should be exercised to guarantee the up-to-standard level of reliability during the system life-cycle. Traditional schemes, such as corrective maintenance (only acts when failure already occurs) and scheduled preventive maintenance (acts periodically regardless of system's heath states), are inadequate for meeting the increasing industrial demand for efficiency and reliability. As a consequence, machine condition monitoring and intelligent maintenance is becoming a crucial part of many industrial sectors including energy, manufacturing, aerospace, and heavy industries[1]. Prognostic and health management (PHM), rooted from aerospace industries, attempts to improve system reliability and safety, reduce costly unscheduled or excessively scheduled maintenance and maximize functionality, making it a promising technology for application in many industries[1]. The latest developments in PHM enable real-time system health condition assessment along with the prediction of its future status utilizing up-to-the-minute information[2]. Engineering systems are becoming more intricate with massive data been collected; data-driven approaches are regarded as powerful solutions for PHM to extract practical information and make suitable maintenance decisions from historical and online statistics. Machine learning, in which a model is

trained based on historical data to yield a desired output, such as degradation level or remaining useful life (RUL), is capable of harnessing industrial "big data" for PHM applications.[2]

Machine learning based data-driven approaches exhibited performance improvements in prognostics of given assets, such as long short-term memory (LSTM) networks[3,4], convolutional neural networks[5,6], variational autoencoders[7], generative adversarial networks[8,9], and so on. The prognostic in these methods is achieved in two ways: (i) using a linear relationship between the maximum degradation feature value associated with the end-of-life (EoL) or (ii) using the degradation trend with a threshold, both requiring prior domain knowledge associated with multiple failure scenarios. Without having a comprehensive understanding of the system, it is difficult to select an appropriate threshold that produces reliable prognostics. Consequently, this uncertainty in threshold selection hinders the model's ability to produce reliable outputs.

Another major factor affecting the reliability of prognostic performance is the uncertainty in collected data. Despite various machine learning based approaches exercised in PHM, it has been verified that the data representation determines the upper bound performance of the prognostic model.[10] Such representations are usually analyzed and extracted based on electronic data, which usually contain missing values due to occasional failures in communication of sensor networks, hardware/software failures, or mistakes by human operation. To deal with missing values, the simplest method is deleting the partially missing data[11]. However, such deletion adversely impacts further prognostic analyses since it decreases sample diversity and sample volume for machine learning. It also leads to poor performance when the missing rate is high[11]. To compensate the data loss, Yong et al proposed a semisupervised learning approach for asset RUL prediction with a limited fraction of health status information (ie, label) dropped[7]. While this method is effective in dealing with the problem of insufficient labels, it cannot be extended to use deep generative models for embedding and also cannot model the deleted situations.

Imputation method is of essence in data preprocessing for prognostics. Imputation can be categorized into statistic imputation and machine learning based imputation.[12] In statistic imputation, missing values are replaced by plausible estimates, which is simple but ignores temporal information. Machine learning based imputation, consisting of constructing a predictive model to estimate the absent, generally achieves high precision. However, the commonly used machine learning based methods, such as k-nearest neighbors (KNN)[13], also rarely reckon with the temporal dependencies and nature of complex distribution in multivariate time series[14].

Recently, Goodfellow et al[15] proposed a generative adversarial networks (GANs) approach, which learns the latent distribution of a dataset and is capable of originating realistic data following the learnt distribution from random noise. GANs have been successfully employed and used to achieve state-of-the-art performance in fields like face or sequences generation, such as SeqGAN[16] and MaskGAN.[17] However, these GANs, similar to the majority of machine learning based missing value imputations[18] that only use the set of complete records to train the network[19], suffer from serious decline in accuracy when the missing rate increases. In addition, most GANs were devised with the aspiration of generating various new samples apart from the existing ones. On the contrary, the missing data insertion requires that the imputed value be as analogous as possible to the original incomplete data. Although an efficient GAN-based method named GAIN has been proposed by Yoon et al[20] for imputing missing data, it does not attempt to accurately represent the correlation within time series.

Inspired by previous research,[8,14] this article proposes a novel robust data-driven prognostic model based on adversarial learning, with a focus on enhancing long-term degradation predictions and remaining useful life predictions. The main contributions of this research are 3-fold:

1. Time-series data are concatenated with one-hot health indicators to learn varied failure time, that is, end-of-life (EoL), which enables the model to bypass low-accuracy predictions generated from an imprecise predefined failure threshold.
2. A time decay matrix is developed to record the time lag between two consecutive valid observations, enabling the model to automatically learn the internal representations of time series observations determined by the time lag and thus reduce the influence of missing values.
3. The present-to-end degradation trajectory is able to be generated by the proposed model with prediction as a combination of multiple Gaussian distributions. Robustness of the proposed model, say imputation accuracy and prognostic accuracy, is evaluated and quantified by aeroengine health datasets generated from the Modular Aero-Propulsion System Simulation (MAPSS).

The remainder of this article is organized as follows: Section 2 gives a brief overview of related work. Section 3 explicitly formulates the problem and presents the proposed framework in detail. Section 4 presents experimental results from the proposed approach as applied to aircraft engine data. Finally, Section 5 contains concluding remarks and future research directions.

2 | RELATED WORK

2.1 | Machine health prognostics

2.1.1 | Prediction of degradation behavior

The performance of most mechanical and electrical systems undergoes gradual degradation, eventually progressing to failure under repeated usage conditions. Damage tolerant system can often continue to operate with some faults/damage, as long as this damage can be monitored and

controlled to avoid system-level failures. It is also possible to decrease the rate of degradation. The prediction of degradation behavior can be categorized into physics-based methods and data-driven methods. *Physics-based* approaches are developed based on understandings in physical phenomena through numerous test data. Many researchers have tried to model the degradation using physical models in fields like battery capacity degradation,[21] fatigue crack growth,[22] and mechanical joints.[23] The advantage of physics-based model is that the predicted degradation behavior tends to be intuitive and based on physical laws. However, since physical models are usually formulated under idealized assumptions and conditions, its applicability and flexibility are highly limited. *Data-driven* approaches use contemporary and historic data to identify the characteristics of the currently measured degradation status and to predict the future degradation trajectory. Data-driven approaches are usually employed when the degradation behavior is too complex to be simulated by a physical model. A variety of data-driven algorithms have been investigated, primarily including recurrent neural network (RNN),[3,4,24] convolutional neural networks,[5,6] fuzzy neural networks,[25] and so on. Furthermore, GAN-based approaches[8] and transfer learning-based algorithms[9] are also proposed to improve the prognostics of machine health in the case of insufficient training data.

2.1.2 | Prediction of remaining useful life

Remaining useful life (RUL) prediction of in-service systems is a key part of PHM, which is estimated by subtracting the current cycle from the EoL prediction. Researchers have proposed many RUL prediction methods in recent years, which are typically based on the following two mechanisms. *Label-based* methods predict EoL by labeling the training data auxiliary, with each sample using its RUL label as a target. The piecewise linear method[26] is usually adopted for this mechanism. This requires extra work and is generally very time-consuming. Moreover, if the available label information is relatively small, the model capability could be highly limited. To overcome this, a generative model[7] with limited health status information has been built for future asset reliability prediction. Furthermore, a GAN-based model is established in Reference 27 to cope with the insufficiency of meaningful data for gear health monitoring. *Threshold-based* methods used to predict EoL rely on a failure threshold defined in advance. The EoL is defined as being obtained when a degradation indicator exceeds that threshold. For example, Li et al[6] employ short-time Fourier transform to process the raw vibration data to the time-frequency domain for RUL prediction after a certain threshold is reached, which indicates the start of degradation. However, this EoL threshold is application specific and relies on significant domain experience. For example, when advanced machine learning method like support vector machine (SVM) is utilized, an appropriate threshold is required to separate the hyperplane of the high- dimensional fault-related features. However, sufficient prior knowledge of critical components is not always readily available.

2.2 | Missing value processing methods

The presence of missing values in time-series data significantly hinders the use of advanced data analysis techniques. Missing data imputation methods proposed in recent years can be categorized as statistic imputation and machine learning based imputation. *Statistic imputation* algorithms fill missing values with some reasonable statistical attributes,[12] such as replace missing value with median value, mean value, most common value, or last-observed valid value. Even though statistic imputation is simple and has low computation cost, it usually suffers from severe distribution distortion for the variable and lacks the utilization and analysis of temporal information. *Machine learning based* imputation methods include multivariate imputation by chained equations (MICE),[28] K-Nearest Neighbor (KNN),[13] Matrix Factorization (MF), and Neural Network (NN).[14,18] Recently, Cao et al[29] proposed a bidirectional recurrent dynamical system to directly learn the missing value without any specific assumption. Choi et al[30] applied medGAN to produce synthetic healthcare data represented by numerical and binary features. Later, Camino et al[31] extended the medGAN to a multicategorical GANs, where the outputs are divided into several parallel layers conditioned on the categorical variables size. Yoon et al[20] introduced a GAN-based model that adopts a hint vector conditioned on actual observations to replace missing data, but exhibits weakness in time series imputation. Later, they proposed an advanced two-stage GRU-GAN[14] aiming at time series imputation, which attempts to find a best matched noise vector of the generator to generate the most similar samples. While machine learning based imputation is computationally expensive, experimental studies show that it outperforms statistical methods based on sensitivity and accuracy in most cases.

3 | METHOD

3.1 | Problem formulation

In this problem, multivariate run-to-failure records from historical sensor measurements and contemporary online measurements are considered. Let $\boldsymbol{x}^{(i)} = [x^{(i;1)}, \dots, x^{(i;m)}]$ indicates a vector of multivariate sensor records at time i, where m is the number of recording sensors. In this way, a full-length sensor record with n total time steps can be presented as $\mathcal{X} = \{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n)}\}$, where $\boldsymbol{x}^{(i)} \subseteq \mathcal{R}^m$. In PHM, generally, the data-driven approach

is to learn the best predictor of run-to-failure degradation from historic data, namely, training data $\mathcal{D}^T \subseteq \mathcal{X}$. With a trained predictor, we can calculate the RUL or equivalent degradation indicators.

However, due to some unexpected accidents in reality, such as hardware failure or sensor communication error, the measurements $\mathcal{X}$ may have missing values. To represent the missing values in $\mathcal{X}$, a masking vector $\mathcal{M} = \{\boldsymbol{m}^{(1)}, \dots, \boldsymbol{m}^{(n)}\}$ is usually adopted.[14,29] The challenging problem is to find an appropriate transformation of the raw observations $\mathcal{X}$ with missing value to efficiently learn a reliable predictor. In this way, such problem can be worked out by establishing a nonlinear mapping function $\mathcal{F} : \{\mathcal{X}, \mathcal{M}\} \rightarrow \boldsymbol{z}$, with latent variable $\boldsymbol{z} \subseteq \mathcal{R}^k$ and $k < m$. Subsequently, the optimal predictor can be formulated based on $\boldsymbol{z}$ as follow:

$$f_\gamma(\boldsymbol{z}) = \underset{\gamma}{argmax} p(\boldsymbol{y}|\boldsymbol{z}, \gamma), \tag{1}$$

where $\boldsymbol{y}$ is the RUL associated with $x^{(i)} \subseteq \mathcal{X}$ and γ is the parameter of $f_\gamma(\boldsymbol{z})$ stands in need of optimization through training.

The primary objective of this article is to develop a GAN-based approach to find the optimal nonlinear mapping function f_γ to get a robust feature representation of $\mathcal{X}$ for degradation progress modeling and robust RUL prediction as accurate as possible from the available incomplete training dataset $\mathcal{D}^T$.

3.2 | Feature extraction and reconfiguration

Feature extraction and selection is the first and essential phase of prognostics. In this procedure, we attempt to identify the typical feature that contains sufficient degradation signatures from the original sensor measurements, wherefore we can prune the cost of feature measurement and reduce the necessitated dimensions of data to represent the degradation progress, consequently, increase the efficiency and reliability of prognostic results.[8]

Considering the physical existence that most typical features either decrease or increase monotonically during degradation, the correlative and monotonic metrics[32] are first adopted in this work to identify such features from initially sampled records. Afterward, we employ a novel feature arrangement, shown in Equation (2), to portray the machinery degradation progress instead of using these selected features directly:

$$\boldsymbol{S}_{t_i} = (\boldsymbol{f}_{t_i}, HI_{t_i}), \tag{2}$$

where $\boldsymbol{f}_{t_i} \subseteq \mathcal{R}^k$ is the collection of multivariate features at associated time step $T = \{t_0, \dots, t_n\}$, k is the total number of selected features, and $HI = (h_1, h_2)$ is the one-hot-encoded health indicator, from which the model acquires a knowledge of the end-of-life (EoL) cycle. In detail, $(1, 0)$ indicates the system is presently in action with good condition, whereas $(0, 1)$ indicates the system is malfunctioning and calls for corrective maintenance. The initial value of $\boldsymbol{S}_0$ is fixed to $(\boldsymbol{0}, 1, 0)$. It is noteworthy that the range of each time series may differ in this work, on that account, all sequences are fixed to a length of $T_{\max}$, where $T_{\max}$ is the longest length among all the time series in training datasets. As the length T_s of $\boldsymbol{f}$ is usually shorter than $T_{\max}$, we fill up S_{t_i} to be $(\boldsymbol{0}, 0, 1)$ for $i > T_s$. In principle, $T_{\max}$ can be considered as a hyperparameter.

As explained in Section 3.1, the collected historical time series $\boldsymbol{f}_{t_i}$ is incomplete. Since the locations of the missing values is of great importance, we introduce the mask matrix $\boldsymbol{M} \subseteq \mathcal{R}^{n \times k}$ to represent whether the values of $\boldsymbol{f}$ exist or not.[14,29] Specifically, $\boldsymbol{M}^k_{t_i} = 1$ if $\boldsymbol{f}^k_{t_i} = 1$ exists, and $\boldsymbol{M}^k_{t_i} = 0$ if not.

3.3 | GAN architecture

Generative adversarial networks (GANs),[15] proposed with the aspiration of generating realistic data $\tilde{\boldsymbol{x}}$ that follow the distribution of authentic data $\boldsymbol{x}$, is composed of two adversarial models: a generator G that apprehends the real data distribution and a discriminator D that assesses the probability of data being generated from G or originated from the real data. Both G and D customarily use multilayer perceptron (MLP) or other nonlinear mapping functions. Specifically, the generator formulates a mapping function that projects a prior noise distribution $p_z(\boldsymbol{z})$ to the data space $G(\boldsymbol{z}; \theta_g)$, in order to capture a generative distribution p_g analogous to the authentic data distribution p_{data}. The discriminator uses a single scalar to differentiate the $G(\boldsymbol{z}; \theta_g)$ from $D(\boldsymbol{x}; \theta_d)$, depicting the probability that $\tilde{\boldsymbol{x}}$ comes from the training data p_{data} rather than p_g.

To improve the stability of optimization of GAN model and avoid mode collapse, same as WGAN,[33] we use the Wassertein distance to measure the distance between two probability distributions, that is, p_g and p_{data}. The Wasserstein distance is informally interpreted as the minimum energy cost (mass times transport distance) of transporting mass in order to transform the distribution p_g into the distribution p_{data}.[33] The optimization of WGAN can be formulated as a minmax problem, with a global optimum for $p_g = p_{\text{data}}$ as follows:

$$\min_G \max_{D \subseteq \mathcal{D}} V(D, G) = E_{x \sim p_{\text{data}}(\boldsymbol{x})}\left[D(\boldsymbol{x})\right] - E_{\tilde{x} \sim p_g(\boldsymbol{z})}\left[D(G(\boldsymbol{z}))\right], \tag{3}$$

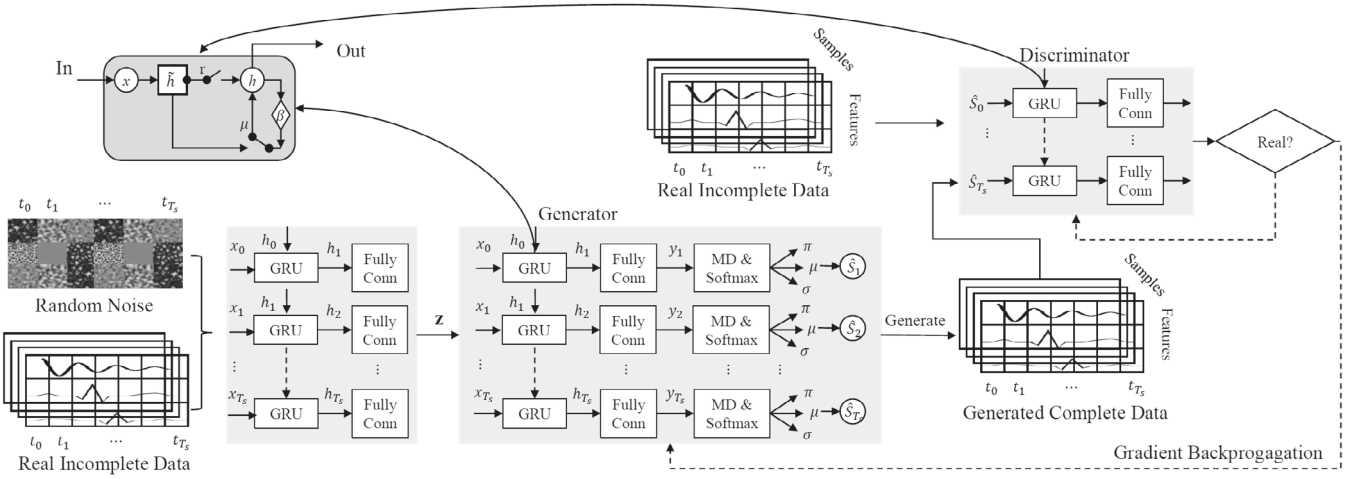


FIGURE 1 The architecture of the proposed model. The generator is a denoising autoencoder, where the encoder is composed by a GRU layer and a fully connected layer and the decoder is added with a mixture density (MD) layer, whose input is a noised incomplete time series and output is the complete time series. The discriminator is composed by a GRU layer and a fully connected output layer

in which, under an optimal discriminator, minimizes the value function (Equation (3)) with respect to the generator parameters minimizes Wasserstein distance $W(p_{\text{data}}, p_g)$. As the loss function converges during training, $W(p_{\text{data}}, p_g)$ decreases and the generated $\tilde{\boldsymbol{x}}$ grows closer to real $\boldsymbol{x}$. The framework of our proposed model is shown in Figure 1.

Gated recurrent unit (GRU): the core of the proposed model is a GRU-stacked neural network that is able to model the temporal irregularity of the incomplete time series. GRU was proposed by Cho et al[34] to make each recurrent unit to adaptively capture dependencies of different time scales. Specifically, inspired by the modified GRU cell for data imputation proposed by Luo et al,[14] we adopt the time lag matrix $\boldsymbol{\delta} \subseteq \mathcal{R}^{n\times k}$ (fixed to 0 at the initial time step) to record the irregular time lag between current observations and last valid observations as shown in Equation (4) as follows:

$$\delta_t^k = \begin{cases} t_i - t_{i-1} & M_{t_{i-1}}^k == 1 \\ t_i - t_{i-1} + \delta_{t_{i-1}}^k & M_{t_{i-1}}^k == 0 \;\; i > 0 \end{cases}. \tag{4}$$

Inside of the GRU cell, a time decay vector $\boldsymbol{\beta}$ is adopted to model the impact of the past observations. $\boldsymbol{\beta}$ is a combination of $\boldsymbol{\delta}$, calculated as follow:

$$\boldsymbol{\beta}_{t_i} = 1/e^{\max(0, \boldsymbol{W}_\beta \boldsymbol{\delta}_{t_i} + \boldsymbol{b}_\beta)}, \tag{5}$$

where $\boldsymbol{W}_\beta$ and $\boldsymbol{b}_\beta$ are parameters requiring optimization. The exponential formulation limits $\boldsymbol{\beta} \subseteq (0, 1]$. The hidden state $\boldsymbol{h}$ is updated by elementwise multiplying the decay factor $\boldsymbol{\beta}$ as:

$$\boldsymbol{h}'_{t_{i-1}} = \boldsymbol{\beta}_{t_i} \odot \boldsymbol{h}_{t_{i-1}}, \tag{6}$$

and the flow of information inside the GRU is redefined as the following steps.

The activation $\boldsymbol{h}_{t_i}$ of the GRU at time t_i is a linear interpolation between the previous activation $\boldsymbol{h}'_{t_{i-1}}$ and the candidate activation $\tilde{\boldsymbol{h}}_{t_i}$:

$$\boldsymbol{h}_{t_i} = (1 - \boldsymbol{\mu}_{t_i}) \odot \boldsymbol{h}'_{t_{i-1}} + \boldsymbol{\mu}_{t_i} \odot \tilde{\boldsymbol{h}}_{t_i}, \tag{7}$$

where $\boldsymbol{\mu}_{t_i}$ is an update gate deciding how much the unit updates its activation, or content. The update gate is computed by Equation (8) and the candidate activation $\tilde{\boldsymbol{h}}_{t_i}$ is computed by Equation (9) as follows:

$$\boldsymbol{\mu}_{t_i} = \sigma(\boldsymbol{W}_\mu S_{t_i} + \boldsymbol{U}_\mu \boldsymbol{h}'_{t_{i-1}}), \tag{8}$$

$$\tilde{\boldsymbol{h}}_{t_i} = \tanh(\boldsymbol{W}_h S_{t_i} + \boldsymbol{U}_h(\boldsymbol{r}_{t_i} \odot \boldsymbol{h}'_{t_{i-1}})), \tag{9}$$

where σ is the sigmoid activation function and tanh is the hyperbolic tangent activation function. $\boldsymbol{W}_\mu$, $\boldsymbol{U}_\mu$, $\boldsymbol{W}_h$, and $\boldsymbol{U}_h$ are weights parameters. r_{t_i} is a set of reset gates. When $\boldsymbol{r}$ off (r_{t_i} close to 0), the reset gate effectively makes the unit act as if it is reading the first symbol of an input sequence,

allowing it to forget the previously computed state. Similar to update gate $\boldsymbol{z}$, the reset gate $\boldsymbol{r}$ is computed by:

$$\boldsymbol{r}_{t_i} = \sigma(\boldsymbol{W}_r S_{t_i} + \boldsymbol{U}_r \boldsymbol{h}'_{t_{i-1}}). \tag{10}$$

Generator G: The Generator *G*, as shown in Figure 1, is first composed by the above modified GRU as an encoder to compress the noised incomplete time series input $\boldsymbol{S}_{\text{noise}}$ into a low-dimensional vector $\boldsymbol{z}$ with the help of recurrent neural network. Then, the decoder tries to learn a mapping $G(z) = \boldsymbol{z} \mapsto \boldsymbol{S}$ that maps the latent space vector $\boldsymbol{z}$ to a complete time series, which contains no missing value. The design of *G* in this article is inspired by the denoising autoencoder proposed by Vincent et al,[35] which reconstructs the original samples from the compressed destroyed samples.

Due to the reason that the training data in this work already contain high missing rate, it is not appropriate to destroy the original samples by dropping out some input values as traditional ways do. Instead, the low-dimensional vector $\boldsymbol{z}$ is compressed from the noised incomplete time series input $\boldsymbol{S}_{\text{noise}}$ (by adding a random noise sampled from a standard distribution $\mathcal{N}(0, 0.01)$). After feeding $\boldsymbol{S}_{\text{noise}}$ to the encoder, we take the final hidden state $\boldsymbol{h}$ and project it into two vectors $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. Under such encoding scheme, the latent vector $\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \mathcal{N}(0, I)$ is a random vector conditioned on the input series instead of a deterministic output for a given input.

The decoder samples output series conditioned on given encoded latent vector $\boldsymbol{z}$. At each step, we feed the previous data point $\boldsymbol{S}_{t_{i-1}}$ concatenated with the latent vector $\boldsymbol{z}$ as an input $\boldsymbol{x}_{t_i}$. To simplify mechanism of modified GRU from Equations (5) to (10), the computation of the GRU layer of the decoder can be interpreted as follows:

$$\boldsymbol{h}_{t_i} = GRU\left(\boldsymbol{x}_{t_i}, \boldsymbol{\delta}_{t_i}, \boldsymbol{h}_{t_{i-1}}\right), \tag{11}$$

and the initial hidden states $\boldsymbol{h_0}$ is worked out by a single layer network $\boldsymbol{h_0} = \tanh(\boldsymbol{W}_z \boldsymbol{z} + \boldsymbol{b}_z)$. The output is the hidden state $\boldsymbol{h}_{t_i}$, which are the initial parameters for a probability distribution of the consequent data $\boldsymbol{S}_{t_i}$.

Then, a fully connected layer is stacked on top of the last hidden state of GRU layer. To prevent overfit, the dropout is employed in fully connected layer. The fully connected layer is adopt to map the hidden state $\boldsymbol{h}_{t_i}$ to the yield vector $\boldsymbol{y}_{t_i}$ that could be divided into *M* mixed Gaussian distributions to describe f_{t_i} and one categorical (p_1, p_2) distribution to describe health state indicator *HI* as follows:

$$\boldsymbol{y}_{t_i} = \boldsymbol{W}_y \boldsymbol{h}_{t_i} + \boldsymbol{b}_y = \left[(\hat{\pi}_1, \mu_1, \hat{\sigma}_1), \ldots, (\hat{\pi}_M, \mu_M, \hat{\sigma}_M), (\hat{p}_1, \hat{p}_2)\right]. \tag{12}$$

The Gaussian components have two complementary roles: (i) separately modeling different stochastic events and (ii) different mixture components model different situations with different rules.[36] At each time step, the feature f_{t_i} in $\boldsymbol{S}_{t_i}$ described by the mixture Gaussian distributions with *M* normal distributions as:

$$p(f_{t_i}) = \sum_{m=1}^{M} \pi_i \mathcal{N}(f_{t_i} | \mu_m, \sigma_m), \tag{13}$$

where μ_m is the mean of the *m*th normal distribution with a standard deviation of σ_m. We should notice that $\sum_{m=1}^{M} \pi_m = 1$.

The *exp* and *softmax* operations are applied after the fully connection layer as the nonnegative of the standard deviations and the probability properties. The probabilities for the categorical distribution is determined by mapping the outputs to logit values as follow:

$$\sigma_m = exp(\hat{\sigma}_m) \tag{14}$$

$$\pi_k = \frac{exp(\hat{\pi}_k)}{\sum_{m=1}^{M} exp(\hat{\pi}_m)}, \quad k = 1, 2, \ldots, M, \tag{15}$$

$$p_k = \frac{exp(\hat{p_k})}{\sum_{i=1}^{2} exp(\hat{p}_i)}, \quad k = 1, 2. \tag{16}$$

Discriminator D: The Discriminator *D* is also composed by a GRU layer and a fully connected layer. In *D*, the output of each GRU block is fed into a fully connected layer whose weights are shared through time. The final decision for the sequence is a probability integer indicating the degree of authenticity, which is the average of all sigmoid yields from the GRU blocks.

The training procedure is to optimize the loss function. Given G, we first demonstrate the optimization of D. The training of the discriminator counts on minimizing the Wasserstein distance rather than crossentropy compared with the original GAN. The discriminator loss function $\mathcal{L}^D$ is shown as follow:

$$\mathcal{L}^D(\theta_d, \theta_g) = E_{S\sim p_{\text{data}}(\boldsymbol{S})}\left[D(\boldsymbol{S})\right] - E_{z\sim p_g(\boldsymbol{z})}\left[D(G(\boldsymbol{z}))\right], \tag{17}$$

where $G(\boldsymbol{z}) = \hat{\boldsymbol{S}}$ is the generated samples while its associated real degradation data are denoted as $\boldsymbol{S}$.

Next, G is optimized by minimizing the discrimination accuracy of D. Since the generator is supposed to produce new sample $\boldsymbol{S\prime}$ that is most similar to $\boldsymbol{S}$, we add log-likelihood loss term to the loss function of traditional WGAN, where the reconstructed loss function is formulated as the sum of three terms: $\mathcal{L}^G = \mathcal{L}^{D(G)} + \lambda(\mathcal{L}_f^G + \mathcal{L}_h^G)$, where:

$$\mathcal{L}_f^G = -\frac{1}{T_s}\sum_{t_i=1}^{T_s}\log\left(\sum_{m=1}^{M}\pi_{t_i,m}\mathcal{N}(f_{t_i}|\mu_{t_i,m}, \sigma_{t_i,m})\right), \tag{18}$$

$$\mathcal{L}_h^G = -\frac{1}{T_{\max}}\sum_{t_i=1}^{T_{\max}}\left[h_1^{t_i}\log(p_1^{t_i}) + h_2^{t_i}\log(p_2^{t_i})\right], \tag{19}$$

$$\mathcal{L}^{D(G)} = -D(G(\boldsymbol{z})), \tag{20}$$

in which λ is a hyperparameter controlling the weight of reconstructed loss $\mathcal{L}_f^G + \mathcal{L}_h^G$ and discriminative loss $\mathcal{L}^{D(G)}$. $\mathcal{L}_f^G$, and $\mathcal{L}_h^G$ correlate with the features and the health state indicator separately. The reconstruction loss terms maximize the log-likelihood of the generated probability distribution to explain the training data.[37] Note that we only calculate the nonmissing values of the sequences. In addition, we discard the mixture density parameters modeling the f points beyond T_s when calculating L_f^G, while L_h^G is calculated using all of the mixture density parameters modeling the (h_1, h_2) health indicators until $T_{\max}$.

3.4 | Imputation and prognostics

Missing value imputation: In order to replace the missing values in time series with reasonable values, we first train the proposed model to learn the distribution of the training dataset. For any incomplete time series $\boldsymbol{S}$, after the reconstruction loss L^G converging to the optimal equilibrium, it means that the model could find the best vector z from the latent input space so that the generated series $G(z)$ is most similar to $\boldsymbol{S}$. For each original incomplete time series, we fill in the missing value of $\boldsymbol{S}$ with the generated $G(\boldsymbol{z})$ as:

$$\boldsymbol{S}_{\text{imputed}} = \boldsymbol{S}\odot\boldsymbol{M} + (1-M)\odot G(\boldsymbol{z}). \tag{21}$$

Degradation prediction: After training, the proposed approach is capable of making a present-to-EoL prediction when given the current measurements. Specifically, at time t, the generator feeds in the reconstructed feature data $\boldsymbol{S}_t$ and then yields $\boldsymbol{y}_t$, which is the parameters of the mixture probability distributions of $\boldsymbol{S}_{t+1}$. Subsequently, $\hat{\boldsymbol{S}}_{t+1}$ is determined by sampling from such generated M Gaussian mixture distribution and a categorical distribution. Unlike the training process, the $\hat{\boldsymbol{S}}_{t+1}$ predicted at time step t is reused as the input at next time step $t+1$. When the health state indicator HI alters to $(0, 1)$ from $(1, 0)$, the recurrent prediction process is terminated, which means the system is failed and calls for corrective maintenance. Algorithm 1 shows the entire pseudo code of creating the present-to-EoL prediction given on-line measurements before t_p, where t_p is the present time step and t_{EoL} is the prediction horizon, that is, the predicted end of life.

RUL prediction: The policy of RUL prediction is analogous to that of present-to-EoL prediction. Assume that the RUL prediction begins at current time t_p, we recurrently calculate the prediction $\hat{\boldsymbol{S}}$ and its associated health state indicators HI until $HI = (0, 1)$ is derived at time step t_{EoL}. On that account, the predicted remaining useful life is attained by $\text{RUL} = t_{\text{EoL}} - t_p$.

4 | EXPERIMENT

4.1 | Experimental setup

Model layout details: In the generator, the GRU layer consists of 128 hidden units. The dimension of the low-dimensional vector $\boldsymbol{z}$ is 128. We used $M = 5$ components for the Gaussian mixture model.

Dataset description: The proposed method is evaluated on datasets created from the Modular Aero-Propulsion System Simulation (MAPSS), where a set of aeroengines with different levels of initial wear and unspecified manufacturing disparity are simulated. The datasets include multiple run-to-failure time indexed trajectories generated from 100 such aeroengines; each trajectory is a multivariate time-series with 24 sensor records and three operational parameters. However, a portion of the sensors' reading remain constant over the engine's whole life cycle and do not contribute useful information for prognostics. In order to extract the optimal sensor subset, criteria coefficients of selected sensors are evaluated by correlative and monotonic metrics[32]. Observing that the 11th, 12th, and 13th sensors have similar performance; therefore, we choose the normalized 11th sensor records as the input f_t. In addition, we down sample all the selected series to fix the length of the each sequences (T_s) within 100 ($T_{\max}$). At each time step $t < T_s$, the health state indicator $HI = (1, 0)$ is concatenated to f_t and $HI = (0, 1)$ is concatenated when $T_s \le t \le T_{\max}$. For imputation experiments, we randomly drop p percent of data points of all the time series, where $p \in \{10, 20, \dots, 80\}$. The imputation performance is compared by the mean squared error (MSE) between the original series and imputed series.

Implementation: In the training process, the back propagation through time (BPTT) is adopted. The batch size is 16 and the iteration steps fragment k of minibatch stochastic gradient descent (SGD) is set to 4. The recurrent dropout with a drop probability fixed with 10%. For all the experiments, 10% of the datasets is selected as validation set and another 10% datasets is selected as test set. Before adversarial training, the generator is pretrained for 100 epochs with loss function $\mathcal{L}^G = \mathcal{L}_f^G + \mathcal{L}_h^G$. The learning rate is 0.001 and the gradient clipping is 1.0. The experiment is exercised based on the open-source software library Tensorflow, and the neural networks are trained on NVIDIA Geforce GTX 1080 Ti and Titan Xp GPU equipped with 32 GB memory.

Baselines: For performance comparison, the baseline models for data imputation are (1) to (4) and for prognostic (4) and (5) are discussed as follow:

1. *Statistical imputation*: The missing values are simply filled out by the mean value and most frequent observed valid value.[12]

Algorithm 1. Present-to-EoL degradation prediction

Input: feature series $\boldsymbol{S} = \boldsymbol{S}_t,\ t = 1, 2, \dots, t_p$
initialize $h_0 = 0, \boldsymbol{S}_0 = (f_0, \mathcal{HI}) = (0, 1, 0), t = 0$;
while $\mathcal{HI} \ != (0, 1)$ **do**
  Generate $\boldsymbol{h}_{t+1}$ and $\boldsymbol{y}_{t+1}$ by $\boldsymbol{h}_t, \boldsymbol{S}_t$ and $\boldsymbol{z}$;
  Sample $\hat{\boldsymbol{S}}_{t+1}$ using $\boldsymbol{y}_{t+1}$;
  $t = t + 1$;
  if $t > t_p$ **then**
    $\boldsymbol{S}_t = \hat{\boldsymbol{S}}_t$;
  end
end
$t_{\text{EoL}} = t$;
Output: return $\boldsymbol{S} = \hat{\boldsymbol{S}}_t,\ t = t_p, \dots, t_{\text{EoL}}$

2. *MICE*: The multivariate imputation by chained equations (MICE) fills the missing data by adopting iterative regression model.[28] The implementation is based on the impute class from the open-source Scikit-learn library with default setting.

3. *KNN*: The missing data are replaced by K nearest neighbor sample,[13] with normalized Euclidean distance. The implementation is based on the "neighbors" class from the open-source Scikit-learn library. The number of neighbors K is set to 15 and other parameters remain default.

4. *GRU-GAN*: This imputation model is one of the state-of-the-art GAN-based model first proposed by Luo et al,[14] where G and D both consist of a GRU layer and a full-connection layer. The number of hidden units of GRU layer is 64. The dimension of z is 256, and the batch size is 16.

5. *VAE&MD*: The single generator of the proposed VAE&MD-GAN with the same parameter setting.

Evaluation: The time series imputation performance evaluation of the proposed model was conducted by using the mean squared error (MSE) of imputed output as:

$$\text{MSE}_{\text{imputation}} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{T_s}\sum_{t=1}^{T_s}\left|\boldsymbol{S}_t^{\text{imputed}} - \boldsymbol{S}_t\right|^2\right). \tag{22}$$

The machine health prognostics performance evaluation of the proposed model was conducted by using the mean absolute error (MAE) of generated output as:

$$\text{MAE}_{\text{decay}} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{T_s}\sum_{t=t_p}^{T_s}\left|\hat{\boldsymbol{S}}_t - \boldsymbol{S}_t\right|\right), \tag{23}$$

$$\mathrm{MAE}_{\mathrm{RUL}} = \frac{1}{N}\sum_{i=1}^{N}\left|\hat{\mathrm{RUL}}_t - \mathrm{RUL}_t\right|, \tag{24}$$

where T_s is the full length of the nominated test degradation series. $\hat{\boldsymbol{S}}_t$ represents the predicted value, while $\boldsymbol{S}_t$ stands for the true degradation value for comparison. N is the total number of the selected series.

4.2 | Performance comparison for time series imputation

Table 1 shows the comparison of the time series imputation results on MAPSS dataset between the proposed VAE&MD-GAN approach and baseline imputation approaches, which include statistical imputation and GRU-GAN based imputation. The first row of Table 1 is the missing rate which specifies the percentage of values that are dropped and other rows are the MSEs. As the result shows, GAN-based models boost the imputation accuracy. The proposed VAE&MD-GAN model has similar accuracy compared with the state-of-the-art GRU-GAN.[14] In this way, it verifies that the proposed model is capable of imputing the missing values with the most reasonable values, but it exhibits better imputation accuracy when the missing rate exceed 40%. It shows that the mixture density components help modeling different stochastic events better.[36]

4.3 | Performance comparison for machine health prognostics

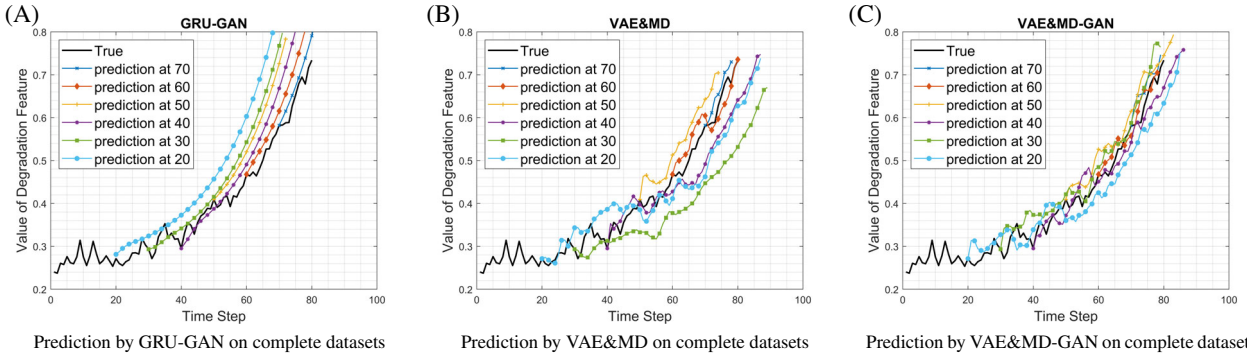
The statistical results of prognostic performance comparison on complete datasets is shown in Table 2. The performance is compared on two aspects: the present-to-EoL degradation prediction and the RUL prediction. For GRU-GAN imputation model,[14] we attach the *exp* and *softmax* operations to the fully connection layer in order to meet the one-hot-encoded constraints of *HI* outputs, and use Algorithm 1 to make the present-to-Eol prediction. As mentioned before, in all three prognostic models with *HI* integrated, they do not require any predefined failure threshold, that means no prior experiential knowledge is required in prediction, leading to a better generalization ability. As shown in Table 2, the overall prognostic performance of VAE&MD-GAN is better than GRU-GAN and VAE&MD, as the $\mathrm{MAE}_{\mathrm{Decay}}$ quickly converges to 0.02 and $\mathrm{MAE}_{\mathrm{RUL}}$ declines to a certain level nearby 2. In addition, VAE&MD-GAN outperforms VAE&MD in RUL prediction even at early stage, and once again verifies that the model could learn the EoL better by adversarial training.

TABLE 1 The imputation performance (MSE) comparison results of baselines and proposed VAE&MD-GAN

Missing rate	10%	20%	30%	40%	50%	60%	70%	80%
Mean filling	0.5996	0.5859	0.6067	0.6659	0.6869	0.6828	0.5640	0.6382
Most frequent	0.6288	0.6661	0.8217	0.8749	0.9891	0.9912	1.0226	1.0754
MICE	0.6965	0.6578	0.7713	0.6461	1.0011	0.7727	0.7743	0.8936
KNN	0.3144	0.3026	0.3338	0.3692	0.4199	0.4111	0.5363	0.6411
GRU-GAN	**0.0941**	**0.1484**	**0.1892**	0.2568	0.2624	0.3646	0.5156	0.5471
VAE&MD-GAN	0.1123	0.1596	0.2041	**0.2428**	**0.2454**	**0.2510**	**0.4025**	**0.5153**

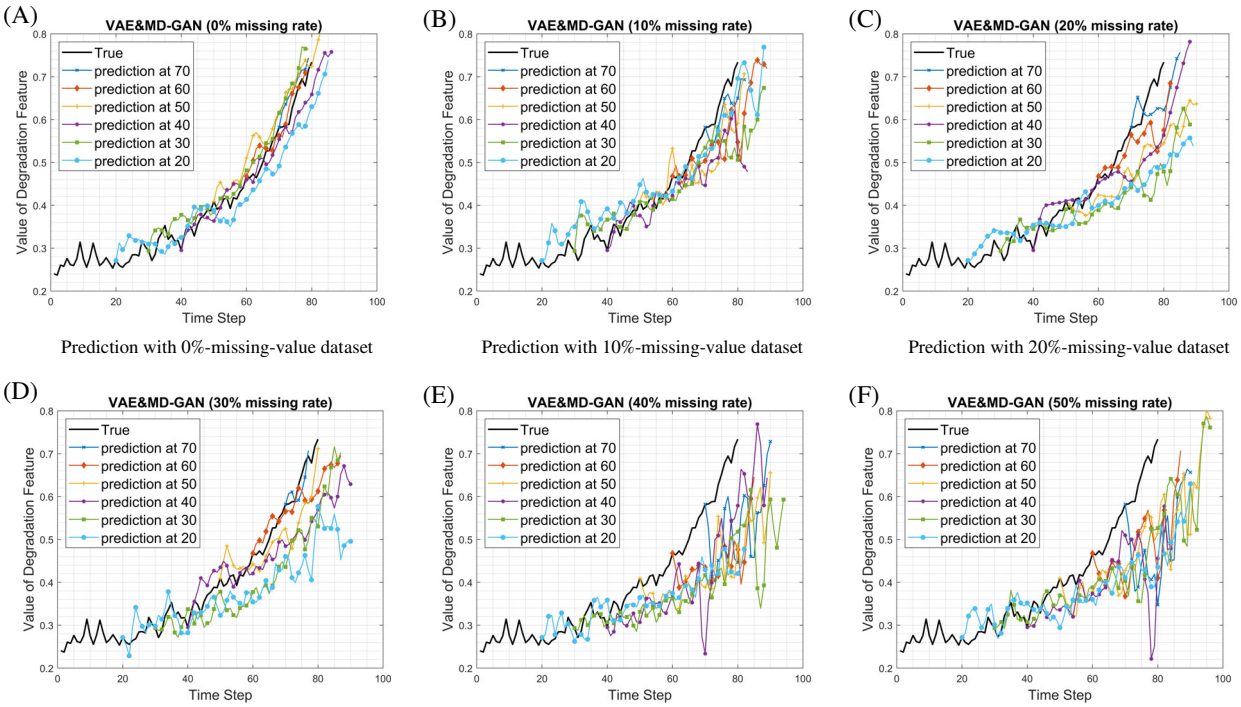
TABLE 2 The prognostics performance results of baseline and the proposed VAE&MD-GAN with complete datasets

Predict time		20	30	40	50	60	70
GRU-GAN	Decay	0.0828	0.0730	0.0729	0.0594	0.0269	0.0216
	RUL	9.72	9.09	7.36	6.18	2.90	3.12
VAE&MD	Decay	0.0852	0.0767	0.0650	0.0626	0.0477	0.0304
	RUL	7.37	6.02	6.15	4.98	4.26	2.65
VAE&MD-GAN	Decay	**0.0417**	**0.0337**	**0.0352**	**0.0310**	**0.0213**	**0.0208**
	RUL	**4.04**	**3.37**	**3.10**	**3.02**	**2.67**	**2.05**



Prediction by GRU-GAN on complete datasets | Prediction by VAE&MD on complete datasets | Prediction by VAE&MD-GAN on complete datasets

FIGURE 2 Visualization of the present-to-EoL prediction result conducted at different time steps for GRU-GAN, VAE&MD, and VAE&MD-GAN. In these figures, the prediction time steps t_p are chosen at 20, 30, 40, 50, 60, and 70



Prediction with 0%-missing-value dataset | Prediction with 10%-missing-value dataset | Prediction with 20%-missing-value dataset

Prediction with 30%-missing-value dataset | Prediction with 40%-missing-value dataset | Prediction with 50%-missing-value dataset

FIGURE 3 Visualization of the present-to-EoL prediction result with different data missing rate conducted at different time steps using VAE&MD-GAN. In these figures, the prediction time steps t_p are chosen at 20, 30, 40, 50, 60, and 70

In MAPSS datasets, each time series represents a specific degradation mode. In terms of the prediction results in Table 2, the VAE&MD-GAN outperforms GRU-GAN in both degradation and RUL prediction. This is mainly due to the reason that the mixture density layer networks can in principle present any conditional probability distribution,[36] especially when the modeled situation is not well represented by a single distribution. The mixture Gaussian distributions can help model the degradation with multiple possible modes, which should not be averaged.

Figure 2 illustrates the present-to-EoL prediction results produced by the proposed VAE&MD-GAN model and comparison models on original complete datasets. As an example, we visualized the prediction results of one typical test series at every 10 time steps. The starting circle of prediction t_p is 20 since the indicator remains stable at the beginning. The present-to-EoL degradation curves generated by VAE&MD and GRU-GAN model roughly represent the authentic tendency. Due to the reason that the degradation prediction is conditioned on the previous observations, the trajectory predicted by all three models becomes closer to real-degradation curve since more observations are available as time goes on. As shown in Figure 2, the VAE&MD-GAN is capable of generating more accurate present-to-EoL degradation curves, and it makes reliable prediction even at the early stage, which indicates that the adversarial training could help the model to learn the distribution of real data better.

TABLE 3 The prognostics performance results of the proposed VAE&MD-GAN with imputed datasets of various missing rate

Imputed rate:	0%		10%		20%		30%		40%		50%	
Predict time	Decay	RUL	Decay	RUL	Decay	RUL	Decay	RUL	Decay	RUL	Decay	RUL
20	0.0417	4.04	0.0899	7.17	0.0854	8.96	0.0863	9.85	0.0954	14.78	0.0941	18.47
30	0.0337	3.37	0.0671	5.62	0.0712	8.78	0.0813	9.34	0.0964	12.33	0.1018	18.36
40	0.0352	3.10	0.0617	5.45	0.0647	6.02	0.0710	9.53	0.1039	11.44	0.1228	14.17
50	0.0310	3.02	0.0596	4.20	0.0624	4.36	0.0591	8.12	0.1268	12.10	0.1272	15.34
60	0.0213	2.67	0.0414	3.82	0.0530	4.24	0.0633	6.58	0.1442	11.60	0.1831	15.44
70	0.0208	2.05	0.0323	2.76	0.0414	4.01	0.0652	6.05	0.1047	10.76	0.2075	16.03

Since the complete datasets is not always readily available, it is hard to directly compare the imputation accuracy for some cases. In this way, the downstream prognostic performance is used indirectly to evaluate the imputation accuracy. For the prognostics evaluation using p percent missed datasets, we first impute the missing values by the proposed model and then normalize the datasets. Then, we train the proposed model with the completely imputed and normalized datasets to perform the prognostic tasks, that is, present-to-EoL degradation prediction and remaining useful life prediction. Figure 3 presents the present-to-EoL prediction results generated by the proposed VAE&MD-GAN model using imputed dataset with 10%, 20%, 30%, 40%, and 50% missing rate separately.

As Figure 3 illustrated, the deviation between the authentic degradation curve and prediction curves enlarged and the trajectory lengthened. The reason behind this partly lies in that, when missing rate grows higher, more data near the EoL is dropped out causing the unknown length of degradation. And the imputed data scalar is smaller than the real one, so in order to follow the overall distribution of real degradation series, more data are imputed after the EoL (which is 0 in real data after EoL, discussed in Section 3.2). The diminished performance in present-to-EoL degradation and RUL prediction consistent with reduction of imputation accuracy with growing missing rate in training dataset. Furthermore, the fluctuation level of predictions indirectly reflects the imputation accuracy. When the missing rate is high in the original dataset, the imputed data points becomes inconsistent with the monotonic characteristic of real degradation data. The imputation seems to aim at minimizing the overall time-series data distribution loss, instead of effectively capturing the inherent monotonic signatures. The quantified results of present-to-EoL degradation and RUL prediction are shown in Table 3.

5 | CONCLUSION

In this article, we proposed a novel generative adversarial neural network model by taking the advantage of modified GRU recurrent cell and mixture density networks, which is capable of providing accurate and reliable predictions of degradation trajectory and RUL based on incomplete time series. In this model, the modified GRU that adopts the time lag matrix to record the missing value in combination with the MD that modulates the distribution of original data together. This approach is able to produce new complete samples that are closer to the original incomplete one. Since the recurrent part (ie, GRU) of the model allows the modeling and prediction of time-series data, the MD part enables the model to make predictions as a combination of multi-Gaussian distributions, making it creative in modeling different types of scenarios in a single model. In addition, we concatenate the time-series data with one-hot health indicator to learn the failure time automatically, bypassing the low accuracy prediction associated with predefined failure threshold. Experiments with real-word datasets validate that our proposed model has high missing data imputation accuracy and can achieve reliable prediction in machine prognostic and health management.

ACKNOWLEDGMENT

This work was supported in part by the US National Science Foundation (NSF) under grant ECCS-1809164 and the Key Research and Development Project of Hunan Province, China under grant no. 2018WK2022.

ORCID

Yufei Tang https://orcid.org/0000-0002-6915-4468

REFERENCES

1. Pecht MG, Kang M. *Machine Learning: Diagnostics and Prognostics*. Hoboken, NJ: John Wiley & Sons; 2018.
2. Kim N-H, An D, Choi J-H. *Prognostics and Health Management of Engineering Systems: An Introduction*. New York, NY: Springer; 2016.
3. Wu Y, Yuan M, Dong S, Lin L, Liu Y. Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. *Neurocomputing*. 2018;275:167-179.
4. Elsheikh A, Yacout S, Ouali M-S. Bidirectional handshaking LSTM for remaining useful life prediction. *Neurocomputing*. 2019;323:148-156.

5. Li X, Ding Q, Sun Jian- Q. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliab Eng Syst Saf*. 2018;172:1-11.
6. Li X, Zhang W, Ding Q. Deep learning-based remaining useful life estimation of bearings using multi-scale feature extraction. *Reliab Eng Syst Saf*. 2019;182:208-218.
7. Yoon AS, Lee T, Lim Y, et al. Semi-supervised learning with deep generative models for asset failure prediction; 2017. arXiv preprint arXiv:1709.00845.
8. Huang Y, Tang Y, Van Zwieten J, Liu J, Xiao X. An adversarial learning approach for machine prognostic health management. Paper presented at: Proceedings of the 2019 International Conference on High Performance Big Data and Intelligent Systems (HPBD\&IS); 2019:163-168; IEEE.
9. Li X, Zhang W, Ding Q, Li X. Diagnosing rotating machines with weakly supervised data using deep transfer learning. *IEEE Trans Ind Inform*. 2019;16(3):1688-1697.
10. Bengio Y, Courville A, Vincent P. Representation learning: a review and new perspectives. *IEEE Trans Pattern Anal Mach Intell*. 2013;35(8):1798-1828.
11. Graham JW. Missing data analysis: making it work in the real world. *Annu Rev Psychol*. 2009;60:549-576.
12. Lin W-C, Tsai C-F. Missing value imputation: a review and analysis of the literature (2006–2017). *Artif Intell Rev*. 2020;53(2):1487-1509.
13. Hudak AT, Crookston NL, Evans JS, Hall DE, Falkowski MJ. Nearest neighbor imputation of species-level, plot-scale forest structure attributes from LiDAR data. *Remote Sens Env*. 2008;112(5):2232-2245.
14. Luo Y, Cai X, Zhang Y, Xu J. Multivariate time series imputation with generative adversarial networks. Paper presented at: Proceedings of the Advances in Neural Information Processing Systems; 2018:1596-1607.
15. Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets. *Adv Neural Inf Process Syst*. 2014;2:2672-2680.
16. Yu L, Zhang W, Wang J, Yu Y. Seqgan: sequence generative adversarial nets with policy gradient. Paper presented at: Proceedings of the 31st AAAI Conference on Artificial Intelligence; 2017.
17. Fedus W, Goodfellow I, Dai AM. MaskGAN: better text generation via filling; 2018. arXiv preprint arXiv:1801.07736.
18. Nishanth KJ, Ravi V. Probabilistic neural network based categorical data imputation. *Neurocomputing*. 2016;218:17-25.
19. Lai X, Wu X, Zhang L, Lu W, Zhong C. Imputations of missing values using a tracking-removed autoencoder trained with incomplete data. *Neurocomputing*. 2019;366:54-65.
20. Yoon J, Jordon J, Vander Schaar M. Gain: missing data imputation using generative adversarial nets; 2018. arXiv preprint arXiv:1806.02920.
21. Tao L, Lu C, Yang C. Battery capacity degradation prediction using similarity recognition based on modified dynamic time warping. *Struct Control Health Monit*. 2018;25(1):e2024.
22. Zhao Y, Ma M, Qin R, et al. A fabrication history based strain-fatigue model for prediction of crack initiation in a radial loading wheel. *Fatigue Fract Eng Mater Struct*. 2017;40(11):1882-1892.
23. Giorgio M, Guida M, Postiglione F, Pulcini G. Bayesian estimation and prediction for the transformed gamma degradation process. *Qual Reliab Eng Int*. 2018;34(7):1315-1328.
24. Tian Z. An artificial neural network method for remaining useful life prediction of equipment subject to condition monitoring. *J Intell Manuf*. 2012;23(2):227-237.
25. Huang Y, Tang Y, Van Zwieten J, Jiang G, Ding T. Remaining useful life estimation of hydrokinetic turbine blades using power signal. Paper presented at: Proceedings of the 2019 IEEE Power \& Energy Society General Meeting (PESGM); 2019:1-5; IEEE.
26. Heimes FO Recurrent neural networks for remaining useful life estimation. Paper presented at: Proceedings of the 2008 International Conference on Prognostics and Health Management; 2008:1-6; IEEE.
27. Li J, Liu S, He H, Li L. A novel framework for gear safety factor prediction. *IEEE Trans Ind Inform*. 2018;15(4):1998-2007.
28. White IR, Royston P, Wood AM. Multiple imputation using chained equations: issues and guidance for practice. *Stat Med*. 2011;30(4):377-399.
29. Cao W, Wang D, Li J, Zhou H, Li L, Li Y. BRITS: bidirectional recurrent imputation for time series. *Adv Neural Inf Process Syst*. 2018;31:6775-6785.
30. Choi E, Biswal S, Malin B, Duke J, Stewart WF, Sun J. Generating multi-label discrete patient records using generative adversarial networks; 2017. arXiv preprint arXiv:1703.06490.
31. Camino R, Hammerschmidt C, State R. Generating multi-categorical samples with generative adversarial networks; 2018. arXiv preprint arXiv:1807.01202.
32. Guo L, Li N, Jia F, Lei Y, Lin J. A recurrent neural network based health indicator for remaining useful life prediction of bearings. *Neurocomputing*. 2017;240:98-109.
33. Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville AC. Improved training of wasserstein gans. Paper presented at: Proceedings of the Advances in Neural Information Processing Systems; 2017:5767-5777.
34. Cho K, Van Merriënboer B, Bahdanau D, Bengio Y. On the properties of neural machine translation: encoder-decoder approaches; 2014. arXiv preprint arXiv:1409.1259.
35. Vincent P, Larochelle H, Bengio Y, Manzagol PA. Extracting and composing robust features with denoising autoencoders. Paper presented at: Proceedings of the 25th International Conference on Machine Learning; 2008:1096-1103; ACM.
36. Ellefsen KO, Martin CP, Torresen J. How do mixture density RNNs predict the future?; 2019. arXiv preprint arXiv:1901.07859.
37. Wu Q, Ding K, Huang B. Approach for fault prognosis using recurrent neural network. *J Intell Manufact*. 2018;29:1-13.

How to cite this article: Huang Y, Tang Y, VanZwieten J, Liu J. Reliable machine prognostic health management in the presence of missing data. *Concurrency Computat Pract Exper*. 2020;e5762. https://doi.org/10.1002/cpe.5762