Image-Based Prognostics Using Deep Learning Approach

Gurkan Aydemir, *Student Member, IEEE*, and Kamran Paynabar

***Abstract*—This article proposes two methods based on deep learning for estimating time-to-failure (TTF) of an industrial system using its degradation image. This provides an effective tool for predictive maintenance practitioners toward digitization of maintenance processes in Industry 4.0 transformation. Both methods utilize the long short-term memory (LSTM) networks for capturing temporal information. First methodology consists of two convolutional layers preceding a single LSTM layer to extract compact information from the individual images and rescue LSTM network from curse of dimensionality. Then, an LSTM layer estimates the TTF value from the extracted features. In the second approach, the dimension of the individual images are decreased by a fully connected neural network, which is trained as an autoencoder. A separate LSTM network is trained and run over this lower dimensional space. The strength of suggested architectures is shown using simulation data and a dataset of infrared image streams collected from rotating machinery. The performance comparison of proposed methods and other methods is also provided.**

***Index Terms*—Deep learning, image prognostics, Industry 4.0, predictive maintenance, remaining useful life (RUL) estimation.**

I. Introduction

PREDICTIVE maintenance is attracting considerable interest due to the transformation of manufacturing with Industry 4.0 revolution [1], [2]. To achieve a profitable predictive maintenance, a smart machine requires efficient predictive analytics to estimate the remaining life of its components in order to take self-corrective actions [3]. Deep learning is a promising tool in predictive maintenance domain, and thus, it has been widely studied with various types of sensors such as vibration, temperature, current, voltage sensors, and imaging devices [4]. Among all these sensors, image analytics have especially attracted researchers' attention in the areas of prognostics, condition monitoring, and structure health management due to rich information that they provide. The main advantage of imaging is the ease of implementation because cameras are mainly noncontact and do not require permanent installation. In addition, image data provide key information about the health status of the photographed object. Several types of imaging are employed for various industrial monitoring applications. Charge-coupled device images are used for assessing surface quality of the products [5]. X-ray imaging is studied to model the development of the damages in composite materials [6]. Infrared thermography is utilized for diverse applications such as civil structure monitoring [7], fatigue dissipation in steel sheets [8], tool condition monitoring [9], and machinery inspection [10], [11]. Although there are numerous works that study imaging for condition monitoring/diagnostics, there is only few researches that focus on prognostics utilizing degradation image streams.

An intuitive approach for image-based prognostics is to model the degradation image streams as a spatiotemporal process [12], and use its parameters to estimate the time-to-failure (TTF) of a system. However, the connection between the proposed spatiotemporal process and physical degradation is defined only under special conditions, and moreover, the model involves a predefined threshold for the failure, which is hard to determine for such image streams. In a recent work [13], the authors propose a penalized tensor regression method for estimating the TTF of a system from their degradation image streams. Even though they solve the curse of dimensionality by tensor decomposition techniques and obtain good results in a real application, their approach is not scalable since tensor decomposition techniques such as CANDECOMP/PARAFAC (CP) and Tucker that have nonlinear computational time with dimensions of the image streams are used. To address this issue, in this article, we utilize deep neural networks to obtain scalable image prognostics models.

Deep learning has been extensively studied for prognostics and diagnostics in the past decade. Stacked denoising autoencoders are employed to extract features from the raw monitoring data for motor fault diagnosis [14], [15]. Convolutional neural networks (CNNs) in one-dimensional [16] and two-dimensional (2-D) [17] are proposed for motor fault detection and classification. However, these studies focus on the monitoring and diagnostics and do not consider the gradual development of the faults, which is crucial in our study. In [18], the authors propose an enhanced version of restricted Boltzmann machine to predict remaining useful life (RUL) of rotatory machinery.

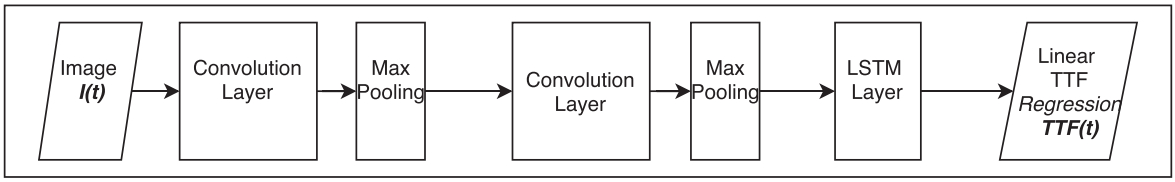
Manuscript received July 31, 2019; revised October 3, 2019; accepted November 9, 2019. Date of publication November 27, 2019; date of current version May 26, 2020. The work of G. Aydemir is supported in part by Bursa Technical University, and the work of K. Paynabar was supported in part by the National Science Foundation under Grant CMMI-1839591. Paper no. TII-19-3528. *(Corresponding author: Gurkan Aydemir.)*

G. Aydemir is with the Institute of Graduate Studies in Science and Engineering, Bogazici University, 34342 Istanbul, Turkey (e-mail: gurkan.aydemir@boun.edu.tr).

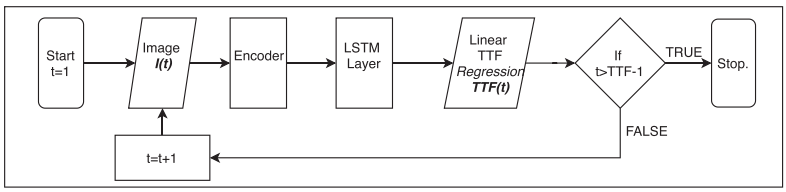
K. Paynabar is with the H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: kamran.paynabar@isye.gatech.edu).

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

Digital Object Identifier 10.1109/TII.2019.2956220



(a)



(b)

Fig. 1. Two methods for image prognostics. (a) Method 1: The convolution layers extracts useful information from the individual images, and the preceding LSTM layer obtains the temporal information. Finally, a fully connected layer with linear activation is employed for calculating the TTF value. (b) Method 2: Images in the streams are encoded to a lower dimensional space with a feed-forward neural network that is trained as autoencoder. Then, the LSTM layer with a linearly activated fully connected layer predicts the TTF value of the corresponding component.

Recurrent neural networks (RNN) that are capable of capturing temporal information from time-series data, are investigated for aero engine RUL estimation in [19], and it is shown that long short-term memory (LSTM) networks outperform gated recurrent unit (GRU) networks and Vanilla RNNs. However, the sensor data that are used for the experiments are very low dimensional as compared to image prognostics problem. In [20], an architecture that combines convolutional and LSTM neural networks is proposed. In this model, a convolution layer extracts robust localized features from raw time-series data and bidirectional LSTM layer captures temporal information. However, the architecture is used for processing individual profile data and characterizing the current status of the machine. An LSTM-based encoder–decoder architecture is utilized for modeling multisensor data in [21] and the reconstruction error is used as a health indicator. However, high dimensionality is not a problem in this study, and moreover, the LSTM network is not used for capturing temporal information. In a recent work [22], the authors proposed a deep CNN for estimating remaining useful life of turbofan engines, and sliding window approach is adopted for making CNN operational for variable length sequences. The main disadvantage of the deep CNN is that the sliding window approach prevents complete utilization of input sequence by disregarding input data older than the length of the sliding window.

This article proposes two different models for expanding utilization of deep learning in prognostics to image data. In the first model, an integrated CNN-LSTM architecture is designed to predict TTF. In the second model, an autoencoder is constructed to reduce the dimension of the individual image in the streams, and a separate LSTM network is employed to predict the failure time. The main contributions of the proposed algorithms are as follows:

1) combining deep learning approaches to separate learning of spatial and temporal content of the image streams to improve prediction accuracy in the context of prognostics;
2) decreasing the computational complexity by reducing the dimension of individual images in the first layers of models.

The rest of this article is organized as follows. Section II presents high-level review of the proposed methodologies. In Section III, mathematical preliminaries of the artificial neural networks that we utilize in our methods are briefly reviewed. Section IV provides our methodology for TTF prediction in detail. The efficiency of the proposed frameworks are validated using a simulation data and a case study in Section V. Finally, Section VI concludes this article.

II. Overview of the Methodology

The proposed image prognostics architectures are illustrated in Fig. 1. As demonstrated in Fig. 1(a), in the first design, two convolutional layers with two pooling layers extract the spatial information from the individual images in the degradation streams. It is followed by an LSTM layer that extracts the temporal information from those features. Then, for each individual image, a TTF value is generated as a linear function of the output state of LSTM cells. This approach addresses the challenge of the high dimensionality and the limited number of samples in image prognostics. The convolution layers provide the compact summary of the individual images that contain a lot of redundancy because of the correlation between neighbor pixels, and let LSTM network capture short- and long-term temporal information from this lower dimensional space.

Second model attacks the high dimensionality of the problem in a different way. A separate deep autoencoder is designed for decreasing the dimension of the input images in an unsupervised manner. Encoder part of the autoencoder projects the individual image to a lower dimensional space.

An LSTM network is trained on this lower dimensional space as in Fig. 1(b). Note that although CNNs are by far the best in

object recognition in images [23], [24], they are not as effective in analysis of degradation image streams. For instance, in a typical infrared degradation image stream, effect of the fault starts at the boundaries of the image, and propagates to the center. Thus, a fully connected neural network also constitutes an efficient compact representation of the degradation images.

III. Preliminaries

In this section, a brief overview of the autoencoders, CNNs, and LSTM networks are provided.

A. Autoencoders

Autoencoder is a special type of neural network that is in general used to learn representation from a dataset. It is trained to copy its input to output. The whole architecture may be divided into two parts: an encoder function that maps input to a code space $\mathbf{h}$ as

$$\mathbf{h} = f(\mathbf{x}) \tag{1}$$

and a decoder function that reconstructs the input from the mapping or code as

$$\hat{\mathbf{x}} = g(\mathbf{h}). \tag{2}$$

A feed-forward neural network model with several hidden layers is employed for approximating these functions, instead of directly finding analytic functions. Each node in the hidden layers is calculated as a nonlinear activation of linear combination of nodes in the preceding layer as

$$h_{ij} = \sigma\left(\sum_{k=1}^{n} W_{ijk} h_{(i-1)k}\right) \tag{3}$$

where h_{ij} represents jth hidden node value in the layer i, W_{ijk} represents trainable weight parameter between nodes h_{ij} and $h_{(i-1)k}$, n represents the number of nodes in the layer $i-1$. $\sigma(.)$ is the nonlinear activation function. Rectified linear unit (ReLU) is mostly recommended as an initial candidate for the activation function in modern neural networks, which is defined as $\sigma(x) = \max(0, x)$ [24]. However, an ReLU has a drawback when it is learned via gradient-based methods. When a neuron equals to 0, it cannot be recovered again from this 0 state because the slope of the activation function is 0 when $x < 0$. One alternative that is proposed to overcome this problem is Leaky ReLU [25]. Leaky ReLU is defined as

$$\sigma(x) = \begin{cases} x, & \text{if } x > 0 \\ \alpha x, & \text{else} \end{cases} \tag{4}$$

where α is a constant much smaller than 1. Autoencoders serve as an effective nonlinear extension of principal component analysis (PCA) when both of the encoder function f and the decoder function g are nonlinear. For this reason, deep autoencoders are mostly designed as a set of hidden layers with nonlinear activation both in encoder and decoder part.

However, if the capacity of the network, i.e., the number of parameters, is high and the training set consists of limited number of samples, the autoencoder may easily overfit the training data [24]. Therefore, in some applications, it is more reasonable to employ a nonlinear deep encoder and linear one-layer decoder. In this case, overall network can be seen as kernel PCA since x is reconstructed as a linear combination of h, which is a nonlinear function of input [24].

Autoencoders are predominantly trained with gradient descent algorithms with mean-squared-error cost function that is defined as

$$C(\mathbf{X}, \hat{\mathbf{X}}) = \frac{1}{n} \sum_{i=1}^{n} ||\mathbf{x}_i - \hat{\mathbf{x}}_i||_2^2. \tag{5}$$

Backpropagation is employed for the computation of the gradients of the parameters [26].

B. Convolutional Neural Networks (CNNs)

The CNN, that is first suggested in [27] is by far the most preferred approach in machine vision. It is named as convolutional because of the *convolution* operation that is adopted instead of general matrix multiplication in standard feed-forward neural networks. The convolution operation for discrete functions is defined as

$$s[n] = (x * w)[n] = \sum_{k=-\infty}^{\infty} x[n-k] w[k] \tag{6}$$

where x is input and w is weighting function or kernel. The convolution may be expanded to 2-D by using the 2-D kernel K as

$$S[i,j] = (I * K)[i,j] = \sum_{m} \sum_{n} I[i-m, j-n] K[i,j]. \tag{7}$$

There are two key ideas behind the CNNs, which are as follows.

1) *Sparse interactions:* Each output unit only interacts with a subset of input units as opposed to standard fully connected networks.
2) *Parameter sharing:* Same kernels are used for different parts of the input(image) that decrease the number of parameters to be trained.

The convolution operation is followed by a nonlinear activation, which is mostly selected as Leaky ReLU that is defined in (4) in image processing applications. This nonlinear activation is named as detector, because when the convolution output of a kernel for a region in the input generates a nonzero value on this activation function, it is interpreted as something is detected in this region.

Another key part of the CNN is pooling function. The pooling layer in a CNN replaces the output of the neurons at a certain location of the preceding layer with the summary statistics of these outputs. Maximum pooling is mostly preferred in machine vision because the exact location of the object is not of interest and this provides strong computational efficiency for the training. Moreover, it makes the network invariant to small translations in the input. However, in some image processing tasks, each activation is important, and thus, average pooling is preferred to obtain better summary of the preceding convolutional layers.

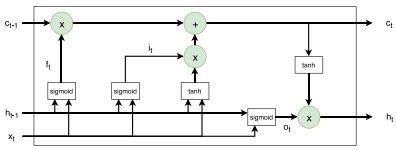


Fig. 2. LSTM cell.

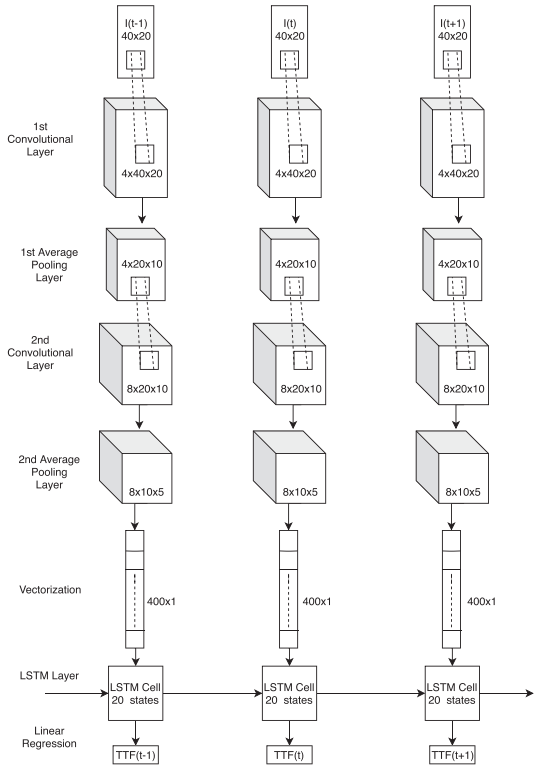


Fig. 3. Convolutional LSTM network.

C. LSTM Networks

The key component in our methodology is the LSTM network [28]. It is a subclass of recurrent neural networks (RNNs), and RNN is a special type of feed-forward neural network used for sequence processing. The RNN has an internal state parameter, namely, memory unit, to remember previous inputs, and hence, is capable of capturing temporal patterns from the sequential input.

There are various types of RNNs suggested in the literature, but the LSTM network is one of the most exploited versions because of the capability of detecting short- and long-term dependencies [24]. An LSTM cell with a forget gate is illustrated in Fig. 2. In an LSTM cell, there are two different state vectors, $c(t)$ and $h(t)$ that are known as cell state vector and output state vector, respectively. The states are updated according to following equations:

$$f(t) = \sigma_g(W_f x(t) + U_f h(t-1) + b_f)$$
$$i(t) = \sigma_g(W_i x(t) + U_i h(t-1) + b_i)$$
$$o(t) = \sigma_g(W_o x(t) + U_o h(t-1) + b_o)$$
$$c(t) = f(t) \circ c(t-1) + i_t \circ \sigma_c(W_c x(t) + U_c h(t-1) + b_c)$$
$$h(t) = o(t) \circ \sigma_h(c(t)) \quad (8)$$

where $\circ$ is the element-wise product and t is the time index. f, i, and o are known as forget gate's, input gate's, and output gate's activation, respectively. In general, the sigmoid function is used for σ_g and a hyperbolic tangent is preferred for σ_c and σ_h. Cell and output states are initialized as $c(0) = 0$ and $h(0) = 0$. If data of interest consists of sequences that have complex temporal information with short-term and long-term dependencies, it may be modeled by LSTM networks. Forget and input gates provide LSTM cell to handle both short-term and long-term relations.

Training of the LSTMs or in general RNNs, is very similar to standard feed-forward neural networks. However, because of its recursive design, the back-propagation algorithm is applied through time, and it is named as backpropagation through time (BPTT) [29]. At first, the network is unrolled through time, and then, partial derivatives of the cost with respect to trainable parameters at each time index are calculated separately using BPTT. Parameters are updated using aggregating those gradients by summation or averaging.

IV. Deep-Learning-Based Image Prognostics

The aim of our study is to estimate and update the TTF value of a system by using its degradation image stream. Two different approaches are employed to predict the TTF of industrial systems.

A. Model 1: Integrated CNN-LSTM-Based TTF Prediction

First model proposes a special type of neural network in which two time distributed convolutional layers are concatenated with an LSTM layer as shown in Fig. 1(a). The LSTM layer outputs are fed into a fully connected linearly activated layer for TTF calculation. A TTF value is generated for each time instant t after a burn-in period.

All of the convolutional, LSTM and fully connected layer parameters are trained using the BPTT algorithm. TTF estimation is a regression problem rather than a classification problem, and the mean-squared error is preferred as cost for regression problems in deep learning. However, in our TTF prediction problem, a weighted average of the squared error is adopted as the training cost function that is calculated for a sample as follows:

$$C = \sum_{k=t_b}^{t_f} \left| k(\mathrm{TTF}_i - \hat{\mathrm{TTF}}_i) \right|^2 \quad (9)$$

where t_b and t_f are burn-in and failure time, respectively. The main idea behind this cost function is that at time t, the amount of information in the sample observed at time t for the TTF estimation is more than previous samples and less than future samples. Moreover, degradation becomes more pronounced as time progresses. Therefore, TTF predictions should be more accurate for the time period close to failure, and thus, the weight of errors in the cost function increases with time.

1) Hyperparameter Tuning: The most challenging task in deep learning is that there is no systematic way to tune hyperparameters of the models. Conventionally, to determine hyperparameters, a heuristic approach is used that is followed in this research as well. First, the kernel size for the convolutional layer and pooling layer intuitively are set to 3×3 and 2×2, respectively, because the images in our dataset are 40×20. Unlike image recognition, in our application, CNN layers are expected to summarize the information all around the image, and larger kernel sizes for convolutional and pooling layers may cause loss of significant information. Then, a network is created with a single convolutional and a single LSTM layer. Number of kernels and cell state size in the LSTM layer are optimized iteratively by updating one of them and fixing the other. The initial values are selected so large that the model overfits the training data, and the optimization is stopped once the performance of the model decreases.

Once the number of kernels and cell state size are determined, an extra CNN layer is added to the model but the number of kernels is halved. It is observed that extra convolutional layer significantly improves the performance, however, adding a third convolutional layer decreases the performance even below the single convolutional layer. So, the number of layers is selected as 2 for the CNN part. Furthermore, after optimizing the CNN parameters, a second LSTM layer is added to the model. It is observed that adding an LSTM layer does not improve the result, it is decided to use a single LSTM layer in the model. With this strategy, the optimization process results in a model with two convolutional layers that has 4 and 8 3×3 convolution kernels, respectively, and 2×2 average pooling kernel. It also has a standard LSTM layer that has five neurons in its cell state. The Adam optimization technique is adopted in the training with the parameters set as learning rate $= 0.001$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$.

B. Model 2: Integrated Autoencoder-LSTM TTF Prediction

In the second model, two separate neural networks are designed for the image prognostics. First, an autoencoder is built to decrease the dimension of the images. The autoencoder is trained with the individual vectorized images in the training set, i.e., temporal information is ignored for this part. Then, the encoder part of the autoencoder is employed for transforming the images in the training and test sets to a lower dimensional space. A single layer LSTM network is designed to predict TTF values of the systems using feature streams generated by the autoencoder at each time instant after a burn in period. The TTF prediction using this method is illustrated in Fig. 4. Weighted average of squared errors as in (9) is employed as the cost for the training similar in first method.

1) Hyperparameter Tuning: The hyperparameters of this model are chosen by a heuristic approach that is similar to the one in Section IV-A1. First, a single hidden layer autoencoder is designed, and the number of neurons in the hidden layer is decreased from a large number where the model overfits the data. Then, an extra hidden layer is added to the coding part, while the number of neurons in the layers are set to be half of the value in the previous model. It is observed that the reconstruction error is decreased by almost half. However, adding more layers in both coding and decoding parts, while fixing the feed-forward computation time constant causes an increase in the reconstruction error. Therefore, a model with two encoding layers and one decoding layer is decided to be used. Moreover, the number of neurons in the hidden layers are optimized in an iterative way as discussed in Section IV-A1. The trained model has 200 and 20 neurons in its hidden layers, respectively. The LSTM cell state is set exactly to the one in the integrated CNN-LSTM model (i.e., 5) for a fair comparison. The Adam optimization technique is adopted in the training with the parameters set as learningrate $= 0.001$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$.

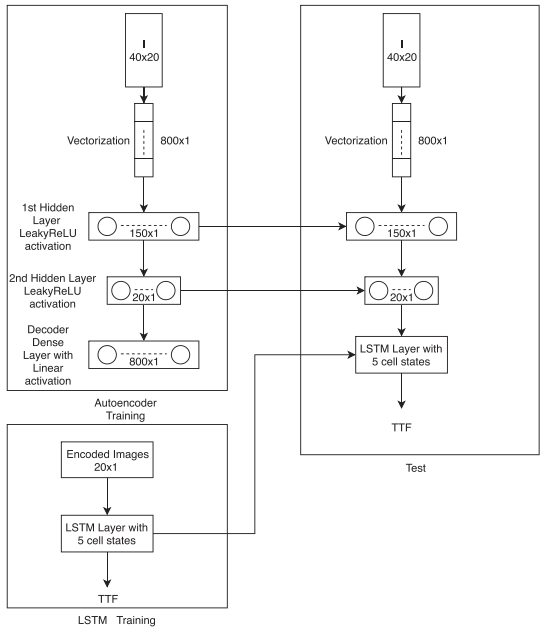


Fig. 4. Autoencoder-based feature extraction and LSTM-based TTF estimation.

V. Performance Evaluation Using Simulations

In this section, efficiency of our methods is demonstrated by using both simulated degradation image streams and a case study.

A. Simulation Study

In this part, degradation image streams are generated based on a heat transfer process as in [13]. Let $I(x, y, t)$ represent the pixel value at position (x, y) at time t. The evaluation of the pixel values is governed by the following equation:

$$\frac{\partial I}{\partial t} = \alpha\left(\frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}\right) \tag{10}$$

where α is the diffusivity coefficient. If the degradation process is modeled by this differential equation, diffusivity constant α determines the rate of degradation, and so, the TTF

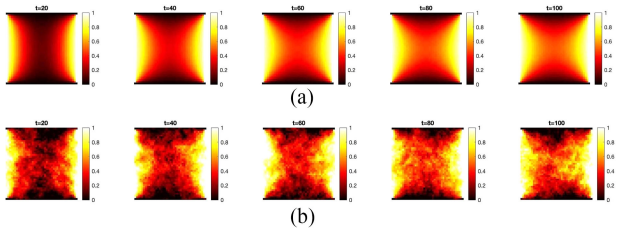


Fig. 5. Simulated image streams. (a) Without noise (b) With noise.

value. Coordinate values are set to $0 \leq x, y \leq 40$, and the images are recorded at integer values of (x, y). Initially, pixel values are set to 0, except boundary conditions that are 1 for all t. Two types of noise are added to images. First, a noise that is generated from a spatial Gaussian process with mean $\mu = 0$ and covariance function $K((x_1, y_1), (x_2, y_2)) = \sigma^2 \exp\big(-\phi\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}\big)$, where $\sigma^2 = 0.01$ and $\phi = 0.25$. The second type of noise is generated from independent and identically distributed normal distribution with variance $\sigma^2 = 0.02^2$. This leads to images of size 41×41. We have generated 500 samples of length 100 where α is randomly generated from a uniform distribution $U(0.5 \times 10^{-4}, 10^{-4})$. An example of image streams that are sampled at $t = 20, 40, 60, 80, 100$ with and without noise is illustrated in Fig. 5.

To show the generalization capability of our methods, TTF values are assigned in two different ways. In the first case, diffusivity constant α is assigned as TTF. In the second case, the TTF of the system i is set to a nonlinear function of diffusivity constant as follows:

$$\text{TTF}_i = 10 - k\alpha_i^2 \tag{11}$$

where k is a constant such that $\text{TTF}_i > 0$.

To evaluate potency of our methodologies, we compare their performance with various approaches. First approach that we use as benchmark is the log-linear scale tensor regression algorithm that have been proposed in [13]. In their study, authors used two tensor decomposition techniques, CP and Tucker decompositions with multilinear principal component analysis (MPCA) to decrease the dimensionality of the problem. For simplicity, we only replicate their algorithm with Tucker decomposition using MPCA.

Second, we employ a deep LSTM network for the same task. However, the LSTM networks may overfit the training set because of the high-dimensional input streams. To prevent this issue, the LSTM network is trained with 90% dropout between LSTM layers and between input and first LSTM layer in the training [30]. In this case, dropout simply means randomly setting some input, state, or output values of the LSTM layers to zero in the forward pass. This provides regularization for the network, and prevents overfitting.

Finally, the deep CNN is used as a benchmark. However, a standard CNN is not applicable to varying length sequences. Moreover, its computational time increases quadratically with the length of the sequence, and this causes a nonscalable network. These issues are solved by applying it with a time window as in [22]. At each time t, $\text{TTF}(t)$ is calculated by feeding the

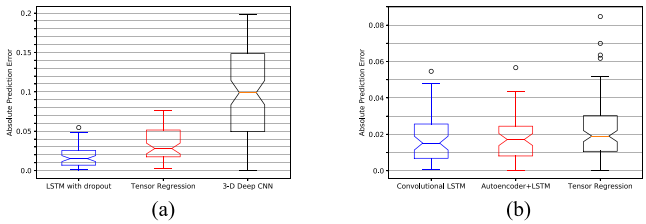


Fig. 6. Prediction errors of the proposed architectures and benchmark methods on simulated data with linear TTF. (a) Comparison of the convolutional LSTM with 3-D deep CNN and LSTM with dropout. (b) Comparison of the proposed methods with tensor regression.

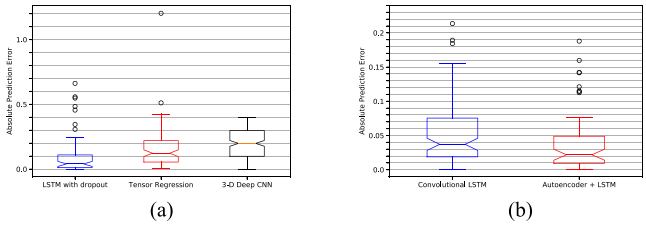


Fig. 7. Prediction errors of the proposed architectures and benchmark methods on simulated data with nonlinear TTF. (a) Tensor regression, 3-D deep CNN, and LSTM with dropout. (b) Convolutional LSTM and autoencoder-based LSTM.

network last L images. Moreover, the CNN is utilized in three-dimensional (3-D) since our image streams are 3-D tensors.

Parameters of the proposed networks are directly set to values represented in Figs. 3 and 4. Burn-in period is defined as the 20% of the stream length, i.e., 20.

Fivefold cross validation is employed for the tests, which implies we have 400 training and 100 test samples in each sample sets, and average of the experiments are presented. Absolute percentage error is used as the performance metric, which is calculated as

$$\text{Prediction Error} = \frac{|\text{Estimated TTF} - \text{Real TTF}|}{\text{Real TTF}}. \tag{12}$$

Fig. 6 illustrates the performance comparison of the LSTM network with dropout, 3-D CNN with time window, tensor regression approaches, and our methods that are convolutional LSTM network and autoencoder code-based LSTM network for the first case where $\text{TTF} = \alpha$. Median absolute prediction errors (and interquartile range) are $1.5\%(2\%)$ and $1.7\%(1.6\%)$ for the convolutional LSTM and autoencoder-based LSTM networks, and $2.2\%(2\%)$, $3\%(3.4\%)$, and $9.9\%(9.8\%)$ for tensor regression, LSTM with dropout, and 3-D deep CNN methodologies, respectively. Our methods outperform LSTM with dropout and 3-D deep CNN methods, and performs slightly better than tensor regression methods.

Fig. 7 demonstrates the performance comparison of these methods for the second case where the TTF value are determined as a nonlinear function of the diffusion parameter, α. Median absolute prediction errors (and interquartile range) are $3.7\%(5.6\%)$ and $2.2\%(4\%)$ for the convolutional LSTM and autoencoder-based LSTM networks, and $12\%(16\%)$, $4\%(9.7\%)$, and $20\%(20\%)$ for tensor regression, LSTM with

TABLE I
COMPUTATIONAL TIME

Method	Time (milisecond)
Convolutional LSTM	7.7
Autoencoer based LSTM	7
LSTM with dropout	6.2
Deep CNN	104
Tensor regression	54000

dropout, and 3-D deep CNN methodologies, respectively. Our methods outperform also tensor regression in the second simulation where the TTF value is a nonlinear function of the diffusion parameter.

It can be concluded that LSTM networks are capable of capturing temporal information without making any assumption on the distribution of the TTF value. However, if the dimension of the input is high and the number of samples is low, it overfits the data. Although the high dropout rates (90% in the simulation study) may prevent overfitting as demonstrated in Figs. 6 and 7, an underfitting is observed in this case. Dimension reduction techniques that can model the spatial correlation of the image pixels improve the results substantially. The proposed architectures model the individual images in two different ways. The other interesting result that is demonstrated is that deep CNN with a window cannot model the complex temporal relations from the image sequences. Tensor regression also failed in TTF prediction for the nonlinear case, because the model assumes a log–linear relation between image streams and TTF values.

1) Computation Time: The computational complexity of the proposed methodologies in the test part is linear with time and input dimension that makes them suitable for real life applications. Experiments are conducted using Tensorflow-GPU [33] on NVIDIA Quadro K4000 graphics board. Their total test time on simulation data for one sample $t = 1, 2, \ldots, 100$ are compared with the benchmark methods in Table I.

B. Case Study

In this section, we assess the performance of the proposed techniques in a case study. The data used in this article are collected from the rotational element thrust bearing by using the test bed presented in [31]. Because of the high cost of accelerated degradation tests, only four experiments are conducted from brand new state to failure. The degradation streams consist of 375, 611, 827, and 1478 images, respectively. The number of samples is increased by resampling the original image streams accordingly, and a total of 284 image data streams are generated as suggested in [13]. The length of the generated sequences is between 17 and 55. An example of the image streams are illustrated in Fig. 8.

The parameters of the suggested neural networks are set as in Figs. 3 and 4. The burn in time is defined as 20% of the shortest sequence length, which is 3 in this case. Absolute percentage error that is described in (12) is used as performance metric.

Fig. 9 illustrates the performance of our methods that are convolutional LSTM network and autoencoder-code-based LSTM network with 95% confidence interval for the case study with

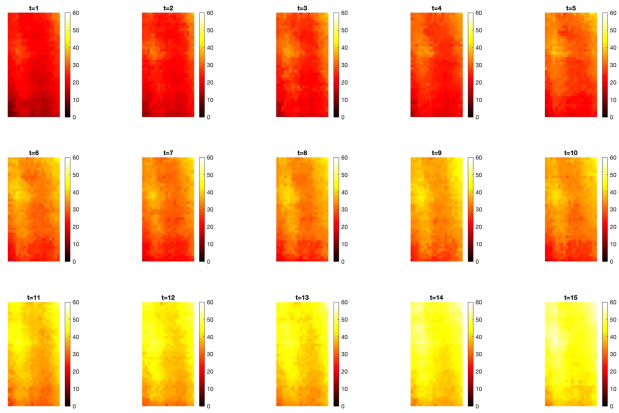
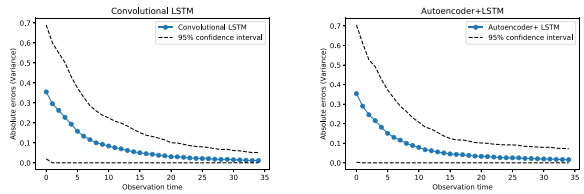
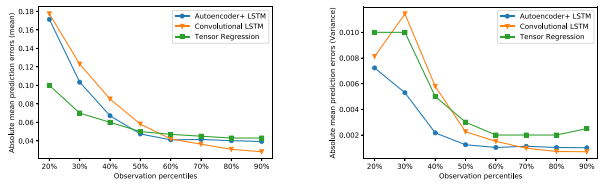


Fig. 8. Example of degradation image stream.



(a) (b)

Fig. 9. Prediction errors of with 95% confidence interval. (a) Convolutional LSTM. (b) Autoencoder+LSTM.



(a) (b)

Fig. 10. Mean and variance of the absolute prediction errors. (a) Mean. (b) Variance.

respect to the number of observations available. The results show that the accuracy of the proposed architectures increases proportionally to the number of observations available at the prediction time. Moreover, the error interval also decreases with time.

Fig. 10 compares the performance of our methods with the tensor regression with respect to observation percentiles. Only tensor regression method is used as a benchmark, because deep CNN and LSTM with dropout methodologies perform far below the proposed methods. The mean (variance) of absolute prediction errors for tensor regression, convolutional LSTM, and autoencoder-based LSTM networks are $0.05\%(0.003\%)$, $0.058\%(0.002\%)$, and $0.047\%(0.001\%)$ at 50th percentile and $0.047\%(0.003\%)$, $0.042\%(0.002\%)$, and $0.041\%(0.001\%)$ at 60th percentile. Tensor regression is slightly better at lower percentiles and worse at higher percentiles than the proposed approaches.

These results imply that the proposed architectures can be used in the real life scenarios. Both of the architectures generate reasonable TTF values after the 40th percentiles. At the same time, the results reveal the weakness of the deep learning models. When the number of samples/observations is low, they still perform worse than the classical approaches. However, on the other hand, it can be concluded that the proposed methods capture complex temporal information in the long sequences with higher dimensional inputs successfully.

Another interesting result is that the autoencoder-based LSTM is better at lower percentiles and convolutional LSTM is better at higher percentiles. It may be explained by the autoencoder's denoising effect on the original images. In the lower percentiles, LSTM with denoised lower dimensional space capture temporal structure more easily. On the other hand, the convolutional LSTM outperforms the autoencoder-based strategy in higher percentiles because of its supervised feature extraction design.

VI. Conclusion

In this article, we proposed two deep learning strategies for prognostics using image degradation streams. First, architecture consisted of convolutional layers for extracting spatial features from each image, and an LSTM layer for tracking the temporal information from this features. In the second model, the dimension of the individual degradation images was reduced by employing a deep autoencoder, and an LSTM network was trained to predict TTF values of the systems using this lower dimensional representation. Both methodologies provided a proper way of solving curse of dimensionality in image prognostics.

The effectiveness of the proposed methods were validated using a simulation and a case study data. Results implied that our architectures outperform the other types of deep learning architectures in the literature that were proposed for prognostics such as deep CNNs and raw LSTM networks with regularization and the classical frameworks such as linear/log–linear tensor regression. The tensor decomposition method was compared with classical methods such as PCA and B-spline in [13], and its superiority was shown. Since both of our approaches performed better than tensor decomposition, they would also outperform PCA and B-spline. On the other hand, the computational complexity of the proposed algorithms was linear with time and input dimension, which makes them scalable.

We believe that the proposed models encourage the practitioners to employ them in industry because of their ease of implementation and lucid math. However, as most of the deep learning architectures, the proposed methods are lack of providing a systematic way of optimizing parameters of the networks such kernel sizes, number of layers, and number of neurons. Further developments are necessary for addressing this issue.

References

[1] B. Bagheri, S. Yang, H.-A. Kao, and J. Lee, "Cyber-physical systems architecture for self-aware machines in industry 4.0 environment," *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 1622–1627, Jan. 2015.

[2] S. Ferreiro, E. Konde, S. Fernndez, and A. Prado, "INDUSTRY 4.0: Predictive intelligent maintenance for production equipment," in *Proc. 3rd Eur. Conf. Prognostics Health Manage. Soc.*, vol. 7, Jun. 2016, pp. 1–8.

[3] J. Lee, H.-A. Kao, and S. Yang, "Service innovation and smart analytics for industry 4.0 and big data environment," *Proc. CIRP*, vol. 16, pp. 3–8, Jan. 2014.

[4] S. Khan and T. Yairi, "A review on the application of deep learning in system health management," *Mech. Syst. Signal Process.*, vol. 107, pp. 241–265, Jul. 2018.

[5] N. Neogi, D. K. Mohanta, and P. K. Dutta, "Review of vision-based steel surface inspection systems," *EURASIP J. Image Video Process.*, vol. 2014, no. 1, pp. 1–19, Nov. 2014.

[6] S. A. Mcdonald, M. Preuss, E. Maire, J. Y. Buffiere, P. M. Mummery, and P. J. Withers, "X–ray tomographic imaging of Ti/SiC composites," *J. Microsc.*, vol. 209, no. 2, pp. 102–112, 2003.

[7] S. Hiasa, R. Birgul, and F. N. Catbas, "Infrared thermography for civil structural assessment: Demonstrations with laboratory and field studies," *J. Civil Struct. Health Monit.*, vol. 6, no. 3, pp. 619–636, Jul. 2016.

[8] B. Berthel, B. Wattrisse, A. Chrysochoos, and A. Galtier, "Thermographic analysis of fatigue dissipation properties of steel sheets," *Strain*, vol. 43, no. 3, pp. 273–279, 2007.

[9] S. Dutta, S. K. Pal, S. Mukhopadhyay, and R. Sen, "Application of digital image processing in tool condition monitoring: A review," *CIRP J. Manuf. Sci. Technol.*, vol. 6, no. 3, pp. 212–232, Jan. 2013.

[10] J. J. Seo, H. Yoon, H. Ha, D. P. Hong, and W. Kim, "Infrared thermographic diagnosis mechanism for fault detection of ball bearing under dynamic loading conditions," *Adv. Mater. Res.*, 2011. [Online]. Available: https://www.scientific.net/AMR.295-297.1544. Accessed: Jan. 23, 2019.

[11] H. Yan, K. Paynabar, and J. Shi, "Real-time monitoring of high-dimensional functional data streams via spatio-temporal smooth sparse decomposition," *Technometrics*, vol. 60, no. 2, pp. 181–197, Apr. 2018.

[12] X. Liu, K. Yeo, and J. Kalagnanam, "Statistical modeling for spatio-temporal degradation data," *J. Qual. Technol.*, vol. 50, no. 2, pp. 166–182, 2018.

[13] X. Fang, K. Paynabar, and N. Gebraeel, "Image-based prognostics using penalized tensor regression," *Technometrics*, vol. 61, no. 3, pp. 369–384, Nov. 2018.

[14] F. Zhou, Y. Gao, and C. Wen, "A novel multimode fault classification method based on deep learning," *J. Control Sci. Eng.*, vol. 2017, pp. 3583610-1–3583610-14, 2017.

[15] C. Lu, Z.-Y. Wang, W.-L. Qin, and J. Ma, "Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification," *Signal Process.*, vol. 130, pp. 377–388, Jan. 2017.

[16] T. Ince, S. Kiranyaz, L. Eren, M. Askar, and M. Gabbouj, "Real-time motor fault detection by 1-D convolutional neural networks," *IEEE Trans. Ind. Electron.*, vol. 63, no. 11, pp. 7067–7075, Nov. 2016.

[17] P. Wang, Ananya, R. Yan, and R. X. Gao, "Virtualization and deep recognition for system fault classification," *J. Manuf. Syst.*, vol. 44, pp. 310–316, Jul. 2017.

[18] L. Liao, W. Jin, and R. Pavel, "Enhanced restricted Boltzmann machine with prognosability regularization for prognostics and health assessment," *IEEE Trans. Ind. Electron.*, vol. 63, no. 11, pp. 7076–7083, Nov. 2016.

[19] M. Yuan, Y. Wu, and L. Lin, "Fault diagnosis and remaining useful life estimation of aero engine using LSTM neural network," in *Proc. IEEE Int. Conf. Aircraft Utility Syst.*, 2016, pp. 135–140.

[20] R. Zhao, R. Yan, J. Wang, and K. Mao, "Learning to monitor machine health with convolutional bi-directional LSTM networks," *Sensors*, vol. 17, no. 2, pp. 273–290, Feb. 2017.

[21] P. Malhotra *et al.*, "Multi-sensor prognostics using an unsupervised health index based on LSTM encoder-decoder," in *Proc. Workshop Mach. Learn. Prognostic Health Manage.*, 2016, pp. 1–10.

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

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Lake Tahoe, NV, USA: Curran Associates, Inc., 2012, pp. 1097–1105.

[24] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[25] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. Int. Conf. Mach. Learn.*, 2013, vol. 30, pp. 1–6.

[26] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 1989, pp. 593–605.
[27] Y. L. Cun *et al.*, "Handwritten digit recognition: Applications of neural network chips and automatic learning," *IEEE Commun. Mag.*, vol. 27, no. 11, pp. 41–46, Nov. 1989.
[28] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
[29] P. J. Werbos, "Backpropagation through time: What it is and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990.
[30] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Adv. Neural. Inf. Process. Syst. 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Lake Tahoe, NV, USA: Curran Associates, Inc., 2016, pp. 1019–1027.
[31] N. Gebraeel, A. Elwany, and J. Pan, "Residual life predictions in the absence of prior degradation knowledge," *IEEE Trans. Rel.*, vol. 58, no. 1, pp. 106–117, Mar. 2009.
[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–41.
[33] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. [Online]. Available: tensorflow.org

Gurkan Aydemir (S'19) received the B.S. and M.Sc. degrees in electrical and electronics engineering, in 2012 and 2014, respectively, from Bogazici University, Istanbul, Turkey, where he is currently working toward the Ph.D. degree in electrical and electronics engineering.

From 2018 to 2019, he was a Visiting Student Researcher with the Georgia Institute of Technology, Atlanta, GA, USA. His current research interests include supervised and unsupervised deep learning tools for predictive maintenance and condition monitoring.

Kamran Paynabar received the B.Sc. and M.Sc. degrees in industrial engineering from the Iran University of Science and Technology, Tehran, Iran in 2002, and Azad University, Tehran, Iran in 2004, respectively, and the M.A. degree in statistics and the Ph.D. degree in industrial and operations engineering from The University of Michigan, Ann Arbor, MI, USA, in 2010 and 2012, respectively.

He is the Fouts Family Early Career Professor and an Associate Professor with the Stewart School of Industrial and Systems Engineering, Georgia Tech, Atlanta, GA, USA. His research interests include both applied and methodological aspects of machine-learning and statistical modeling integrated with engineering principles. His current research interests include the analysis of high-dimensional complex data including multistream signals, images, point-clouds, and network data, for system modeling, monitoring, diagnosis, and prognosis.

Dr. Paynabar was the recipient of the Institute for Operations Research and the Management Sciences (INFORMS) Data Mining Best Student Paper Award, Best Application Paper Award from Institute of Industrial Engineers (IIE) Transactions, Best the Quality, Statistics, and Reliability (QSR) refereed paper from INFORMS, and Best Paper Award from Production and Operations Management Society (POMS). He has been recognized with the Georgia Tech campus level 2014 Center for Teaching and Learning/BP America (CETL)/(BP) Junior Faculty Teaching Excellence Award and the Provost Teaching and Learning Fellowship. He is serving as the Chair of QSR of INFORMS, and the President of Quality Control and Reliability Engineering (QCRE) of the Institute of Industrial and Systems Engineers (IISE).