Combining Deep Learning and Survival Analysis for Asset Health Management

Linxia Liao[1], and Hyung-il Ahn[2]

[1] *GE Digital, San Ramon, CA, 94583, USA*
linxia.liao@ge.com

[2] *Noodle Analytics, Inc., San Francisco, CA, 94105, USA*
hyungil.ahn@noodle.ai

ABSTRACT

We propose a method to integrate feature extraction and prediction as a single optimization task by stacking a three-layer model as a deep learning structure. The first layer of the deep structure is a Long Short Term Memory (LSTM) model which deals with the sequential input data from a group of assets. The output of the LSTM model is followed by mean-pooling, and the result is fed to the second layer. The second layer is a neural network layer, which further learns the feature representation. The output of the second layer is connected to a survival model as the third layer for predicting asset health condition. The parameters of the three-layer model are optimized together via stochastic gradient decent. The proposed method was tested on a small dataset collected from a fleet of mining haul trucks. The model resulted in the "individualized" failure probability representation for assessing the health condition of each individual asset, which well separates the in-service and failed trucks. The proposed method was also tested on a large open source hard drive dataset, and it showed promising result.

1. INTRODUCTION

As the Internet-of-Things technology enables us to obtain a great amount of data to monitor physical assets, there is an increasing demand for determining asset health conditions in a variety of industries. Accurate asset health assessment is one of the key elements which enable predictive maintenance strategy to increase productivity, reduce maintenance costs and mitigate safety risks.

Most analytics models for asset health assessment in the literature have relied on historical operating data, sensor data and maintenance action logs. Trappey et al. (Trappey, Trappey, Ma, & Chang, 2015) proposed an asset health prediction method for power transformers. First, principal component analysis (PCA) was used to identify key factor values such as the state of dissolved gasses. Then, a back-propagation neural network model was utilized to predict asset health condition using the identified key factor values. Hong et al. (Hong, Zhou, Zio, & Wang, 2014) presented a health trend prediction approach for rotating bearings. First, empirical mode decomposition method was used to extract features from vibration signals. Then, a self-organizing map method was used to calculate a confidence value of the bearing health state based on the extracted features. Benkedjouh et al. (Benkedjouh, Medjaher, Zerhouni, & Rechak, 2015) described a method to predict cutting tool wear. First, a nonlinear feature reduction method was used to reduce the dimension of the original features extracted from the monitoring signal. Then, a support vector regression method was used to predict the cutting tool wear based on the reduce features. Li et al. (H. Li, Pan, & Chen, 2014) proposed a method to predict battery health condition. First, a wavelet denoising approach was introduced to reduce the uncertainty and to determine trend information. Then, relevance vector machine was employed as a novel nonlinear time-series prediction model to predict the remaining life of the battery. Ahn et al. (Ahn, Leung, & Hochstein, 2014) proposed the framework of building a vital sign indicator using "individualized" cumulative failure probability, which involved two separate steps of classification and regression. The classification step was first used to calculate the classification failure probability as a way of dimensionality reduction. Then, the regression step (e.g. Cox proportional hazard regression or support vector regression), given the classification probability as an input variable, estimated the optimized hazard function and the individualized cumulative failure probability.

In general, these models tend to have two separate steps such

as feature extraction and prediction. The first step is to extract features that are indicative of failure or degradation from the data. The second step is to build a prediction model to assess or predict the health condition. This two-step approach involves two separate optimization procedures, which often requires the iteration of the two separate procedures until any acceptable result is achieved.

Based on learning multiple layers of network structures, deep learning has gained popularity in the machine learning community, especially with its success in applications such as language modeling, speech recognition, and image recognition. Deep learning has not been widely applied to asset health management or prognostics health management field. Some attempts have been made in the past. Tamilselvan et al. (Tamilselvan & Wang, 2013) applied a deep learning classification method to diagnose electric power transformer health states. Li et al. (K. Li & Wang, 2015) presented an auto-Encoder deep learning method to classify multi-class signals of spacecraft. Qiao et al. (Qiao & Xun, 2015) stacked auto-Encoder and support vector regression to estimate the state of health using a challenge competition dataset. Yan et al. (Yan & Yu, 2015) proposed an auto-Encoder method for gas turbine anomaly detection. A deep convolutional neural network based regression approach was proposed by Babu et al. (Babu, Zhao, & Li, 2016) to estimate remaining useful life of a sub-system or a system component.

Many of the proposed deep learning algorithms already consider stacking feature learning models and final models together. However, the state-of-the-art deep learning algorithms have been focused on either classification or regression problems. Stacking deep feature learning and survival analysis has not been well studied in the literature. A recent development of combining neural net and survival analysis was revealed in (Katzman et al., 2016). The proposed deep learning architecture and the survival analysis are different from our proposed method, and the application domain is different as well. In the application of asset health assessment, sequential data is a common format of the input data e.g. temperature measurements, utilization, and events over time. Long Short Term Memory (LSTM) is a good candidate to learn the past dependencies in the sequential data that may influence future events. Asset health management also often involves modeling on the data from a fleet, which survival analysis is suitable. Instead of doing feature extraction and survival analysis as two separate steps, we propose a novel 'end-to-end' deep learning structure by stacking LSTM, neural network, and survival analysis, and optimizing all the parameters together using stochastic gradient descent.

2. METHODOLOGY

We propose a model which integrates feature extraction and prediction as a single optimization task by stacking a LSTM layer, a neural network layer, and a survival model layer. The LSTM model takes the raw sequential input and extracts the features. Mean-pooling is used on the extracted features to generate input for an extra neural network layer to further learn the feature representation. The output of the neural network layer is learned by the survival model, which outputs a failure probability to indicate the health condition of an asset. The structure of the model is shown in Figure 1. This section will describe the detail of each layer, which presents how the proposed method takes the raw sequential data as input, goes through the layers, and predicts failure probability. The learning method optimizes the all the parameters using the stochastic gradient descent method.

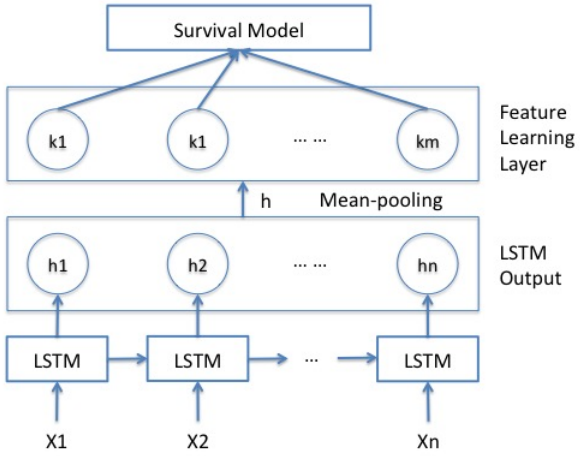

Figure 1. Overview of the proposed model. n is the length of sequence, m is the number of hidden neurons.

2.1. Long Short Term Memory (LSTM)

LSTM (Gers, Schmidhuber, & Cummins, 2000) is a type of Recurrent Neural Network (RNN), which has been successfully applied in many applications. The loop in the RNNs allows information to pass from one step of the network to the next. This information persistence enables RNNs to reason using previous information to infer later event. LSTM is a special type of RNN structure designed to learn long-term dependencies, e.g. when there are very long time lags between important events. Instead of using a single layer as in standard RNNs, LSTMs use four special and interacting layers, which are $f, i, \tilde{C}$ and o. The first layer (f) is a sigmoid layer called the "forget gate layer", which decides what information needs to be passed from the previous state (C_{t-1}). It looks at the previous output h_{t-1} and the current input x_t, and outputs a number between 0 and 1. The equation of the first layer is denoted by:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \tag{1}$$

where σ is the sigmoid function, W_f is the weight of layer

f, [] denotes the concatenate operation, and b_f is the bias of layer f. The second layer (i) of LSTM decides what information to be stored in the current state. There are two steps. Firstly, a sigmoid layer i is used to decide which value to be updated.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \tag{2}$$

where σ is the sigmoid function, W_i is the weight of layer i, and b_i is the bias of layer i.

Secondly, a tanh layer c updates the values to be stored using:

$$\tilde{C}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c), \tag{3}$$

where $tanh$ is the tangent function, W_c is the weight of layer c, and b_c is the bias of layer c.

Now we can update the previous state (C_{t-1}) to the current state (C_t) using:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t, \tag{4}$$

The last layer is a sigmoid layer (o) to determine the output of the current state. The equation of layer (o) is denoted by:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \tag{5}$$

where $sigma$ is the sigmoid function, W_o is the weight of layer o, and b_o is the bias of layer o. The final output (h_t) is determined by:

$$h_t = o_t \cdot tanh(C_t), \tag{6}$$

LSTM serves as the first layer of the proposed structure as shown in Figure 1. The purpose is to deal with the sequential data and potentially capture information in the past that may contribute to the later event. The output h_t is averaged (mean-pooling) over time as the feature representation for further steps by:

$$h = \sum_{j=1}^{n} h_j / n, \tag{7}$$

where h_j is the output of the jth sequence and n is the length of the entire sequence input.

2.2. Feature Learning Layer

The feature learning layer is a generative layer (k) which further learns the feature representation h outputted by LSTM layer. There are many possible designs for the layer. First, it can either be a single layer or multiple layers. Second, the number of neurons m can be selected differently. Last, but not the least, the activation function for each layer can be different as well. For simplicity, a single sigmoid layer (k) is used in the proposed model. The equation for layer k is denoted by:

$$P = \sigma(W_k \cdot h + b_k), \tag{8}$$

where σ is the sigmoid function, W_k is the weight of layer k, h is the output of Equation 7, and b_k is the bias of layer k.

2.3. Survival Model

Survival models have been widely used in reliability, clinic studies, and economics. This type of models analyze the expected time duration until any event happens. Sequential data contains information about events and the time when the events occurred. In asset health management applications, an event happens when an asset fails. The sequential data measures any signal that is related to the operation or condition of the asset over time. The sojourn time (time spent in a certain state) in our model is assumed to follow a Weibull distribution, which is widely accepted for product reliability analysis. The hazard rate for sojourn time t is:

$$\alpha(t) = \frac{\Lambda}{\lambda}(\frac{t}{\lambda})^{\Lambda-1} \tag{9}$$

where Λ is the shape parameter, and λ is the scale parameter. It can be adapted to model various sojourn time-dependent hazard risks.

The sojourn time is also influenced by observed covariates such as the measured signals from the asset or the extracted feature representation from the measurements. The impacts of the covariates are modeled using the Cox proportional hazard model:

$$\alpha(t|P) = \alpha(t)e^{\beta P} \tag{10}$$

where $\alpha(t)$ is the baseline hazard rate defined by the Weibull distribution, P is a vector of covariates, and β is a vector of the coefficients of the covariates. Notice that P is the output of Equation 8.

In many survival models, large portion of the observations are censored. Right censoring is the most common censoring form, when the study ends before any event happens. In our case, the asset's sequential data is used for survival analysis. Censoring is mainly caused by the incompleteness of the observation of the failed assets. The asset's health condition after the time period of the observation is unknown, hence censored. The right censoring case in our study is censored by the last time stamp of the data observed when an asset has not failed yet. In another word, how much longer this asset

can remain in service in unknown.

Censoring can be modeled by cumulative probability functions which integrates all possible outcomes. Hence, the likelihood function for the assets is defined by:

$$L = \prod_{l=1}^{N} \alpha(t_l|P_l)^{(1-\delta)} \cdot H(t_l) \tag{11}$$

where N is the total number of assets. $\alpha(t_l|P_l)$ is the probability density that the asset will fail at the time t_l given its covariates P_l. δ is the indicator for right censoring. It equals to 1 if the asset has not yet failed and otherwise 0. $H(t_l)$ is the probability that the asset stays in service for more than t_l.

$$H(t_l) = \int_{t_l}^{\infty} \alpha(t|P_l)dt \tag{12}$$

We will use the failure probability to indicate the asset's health condition. The failure probability is defined by:

$$F(t_l) = 1 - \int_{t_l}^{\infty} \alpha(t|P_l)dt \tag{13}$$

2.4. Learning Method

The objective of the learning is to minimize the negative log likelihood defined in Equation 11.

$$cost = -log(L) = -log(\prod_{l=1}^{N} \alpha(t_l|P_l)^{(1-\delta)} \cdot H(t_l)) \tag{14}$$

The covariates (P_l) for each asset is derived from the original sequential data by passing through the LSTM layer and the feature learning layer. The learning process is governed by stochastic gradient descent method. It is noted that the learning process directly minimizes the final cost function using the original data, which means the feature extraction and the asset health assessment are optimized together in the learning process.

3. CASE STUDY

Two case studies are used to validate our proposed framework. The first case study validates the method on a small dataset collected from a fleet of mining haul trucks. The model results in an "individualized" failure probability representation for assessing the health condition of each individual asset, which well separates the in-service and failed trucks. The purpose is to show the expected result from the proposed method. The second case study validates the method on an open source hard drive dataset to show the performance with a large dataset.

3.1. Case Study 1

Our proposed deep learning structure for asset health assessment was tested with one of the largest mining service companies in the world. The collected data includes the logs of daily fuel consumption, daily number of loads moved, daily meter hours, and empty drive distance on 27 mining haul trucks over the period from January 1st 2007 to November 11th 2012. Each truck is equipped with a set of sensors triggering events on a variety of vital machine conditions. All the records collected from a truck form a set of sequential data. The estimated overall cost of downtime for one of these haul trucks amounts to about 1.5 million USD per day. Hence, the financial impact of reducing the downtime is very large. The goal of our proposed method is to assess the health condition of the assets given the collected data and to estimate their future failure probability to guide the maintenance best practice.

3.1.1. Data Preparation

The service time of the trucks has been normalized to a number ranging from 0 to 1 according to the maximum length of the sequences. For shorter sequences, they are padded by zeros to ensure the same length on the input sequences. Four most important variables are selected for our study. Due to the confidentiality agreement, the actual names of the variables can not be released. The variables are also normalized to numbers ranging from 0 to 1 given their minimum and maximum values. Trucks that were not failed yet at the time stamp of the last measured log are labeled for right censoring. The data is separated into two sets by using 70% of the data for training the model, and the rest of the data for testing. Because of the very limited number of samples, we do not use a separate validation dataset.

3.1.2. Result

The model is implemented using Python Theano package. There is no well documented guidance to select the parameters of the deep learning model. We use trail-and-error to select the training parameters. The learning rate is set to 0.0001 and the model is running until the cost doesn't decrease for 5000 steps of learning. The number of neurons in the feature learning layer is set to 1 arbitrarily. The batch size for the stochastic gradient descent learning is set to 10. After the training finishes, the testing data is inputted to the trained model to calculate the failure probability for validation. Actually, the failure probability of the training data is also inputted to the trained model to validate the training result.

Ideally, the failed trucks should have higher failure probabilities, which is defined in Equation 13, than that of the non-failed trucks. The failure probabilities are shown in Figure

Table 1. Training confusion matrix.

		Prediction	
		Non-Fail	Fail
Actual	Non-Fail	14	0
	Fail	2	4

Table 2. Testing confusion matrix.

		Prediction	
		Non-Fail	Fail
Actual	Non-Fail	4	1
	Fail	1	1

2.

For the training set, the non-failed cases are shown in blue curves marked with blue up-facing triangles. All of the non-failed cases are at the bottom of Figure 2. The failed cases in the training set are shown in red curves marked with red down-facing triangles. Most of the failed cases in the training set are shown in the left upper lines. One failed case is in the middle of the figure, while the curve is still higher than the blue curves which are for the non-failed cases. Two of the failed cases in the training set are mixed with the curves of the non-failed cases, which means these two cases are not separable.

For the testing set, the non-failed cases are shown in green curves marked with green left-facing triangles. Only one of the green curves is mixed with the red curves located in the upper left part of Figure 2. It means that the result for this case is not good. All other non-failed cases in the test data are embedded in the blue curves. It means the result is good because they have similar failure probabilities as other non-failed cases. The failed cases in the test data (only two events) are shown in magenta curves marked with magenta right facing triangles. In the same way, we can see one of the result is good since the failure probability is high. The other is mixed with the blue curves, which is not good. The confusion matrices for training and testing are shown in Table 1 and Table 2, respectively. In all, the training and testing result shows that the proposed model can achieve acceptable separation between the non-failed and failed case.

Although this validation method is not well quantified, it shows good visualization. In practice, it would be more interesting to look at the future failure probabilities over time, which has been quantified, instead of comparing non-failed cases and failed cases.

3.2. Case Study 2

Our proposed framework was tested in case study 1 with a very limited dataset, while it showed the potential of using the proposed method for asset health management. In case study 2, we will use a much larger dataset to validate our proposed

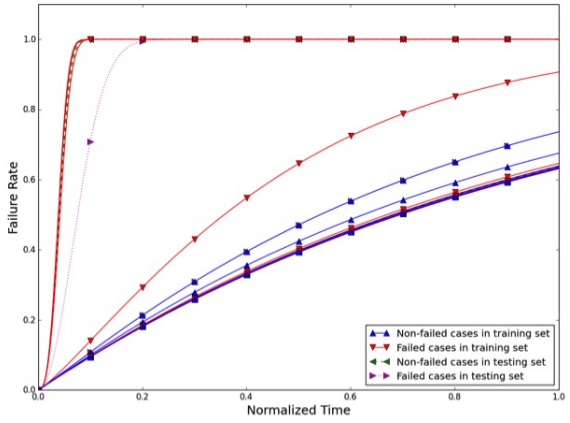

Figure 2. Training and testing result.

method.

Backblade has open sourced a reliability dataset for $41,000$ hard drives from a data center. If a hard drive fails, a new hard drive of the same model will be replaced and run until it fails. Data was recorded daily from year 2013 to year 2015. Each datatum includes date, serial number, and model, capacity, and failure, and S.M.A.R.T. (Self-Monitoring, Analysis and Reporting Technology) statistics and their normalized values which contains statistics such as reallocated sectors count, write error rate, and temperature, etc.

3.2.1. Data Preparation

In 2015, more S.M.A.R.T. columns were added to the data files. Hence, we will use the data from 2013 to 2014. During this period of time, model ST3000DM001 had the most failed hard drives comparing to other models. We focus our analysis only on data from this model.

There are $2,080,654$ rows of data. After dropping columns that have any NA value, there are 5 columns of S.M.A.R.T. raw statistics (numbered as $1, 5, 9, 194$, and 197). Each column of the S.M.A.R.T. raw statistics is normalized by subtracting the minimum value and dividing by the difference between the maximum value and minimum value of each column. Another column, which is called 'failure', indicates whether the hard drive has failed (1) or not (0). In total, there are 4703 hard drives from which 1614 hard drives failed.

3.2.2. Result

A 5-fold stratified cross validation test is performed on the dataset. The training parameter selection is still by trial-and-error. The learning rate is set to 0.001 and the model is running until the cost doesn't decrease for 500 steps of learning. The number of neurons in the feature learning layer is set to

1. The batch size for the stochastic gradient descent learning is set to 10. The failure probability at the last recorded time is calculated for each hard drive. The average Receiver Operating Characteristic (ROC) curves and area under the curve are calculated for both the training and testing dataset from all the 5 folds. The result is shown in Figure 3 and Figure 4. The area under the curve for training and testing are 0.87 and 0.72, respectively. This means we have an acceptable model which can be used for future prediction.

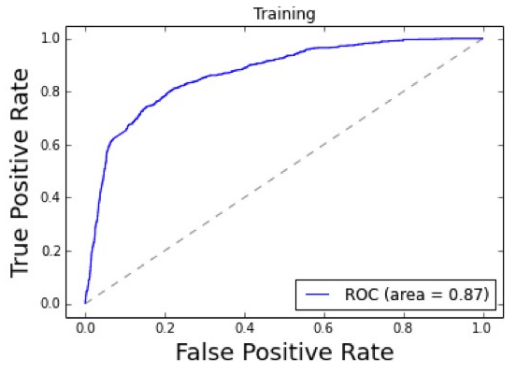

Figure 3. Training ROC of the deep model.

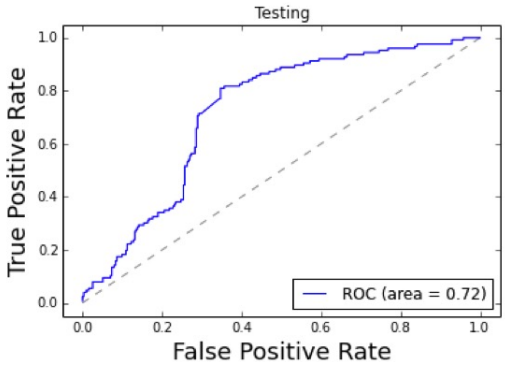

Figure 4. Testing ROC of the deep model.

To compare with the traditional cox proportional hazard (CPH) model , we prepare the features using their mean values as the covariates. The same stratified dataset was used for cross validation. The ROC curves for training and testing are shown in Figure 5 and Figure 6, respectively. The area under the curve for training and testing are 0.70 and 0.69, respectively.

The training performance of the deep model (0.87) is much better than the CPH model (0.70), which means the deep model tends to fit the data better. It is not over fitted given the performance of the testing result. The testing performance of the deep model (0.72), which is only slightly better than the CPH model (0.69) overall. If we take a closer look at the testing ROC curves, the deep model achieves over 80% true positive rate with a false positive rate around 37%. However, CPH model will have a false positive rate around 70% to be able to achieve the similar true positive rate. The deep model can achieve better accuracy by setting the optimal cut off threshold.

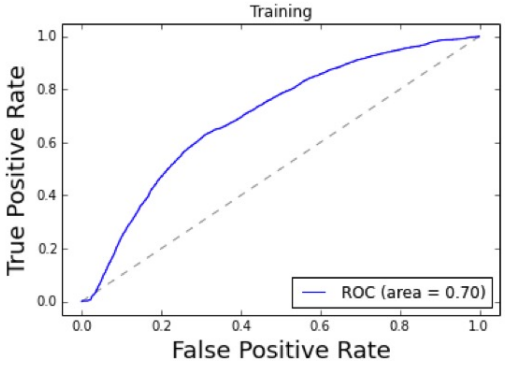

Figure 5. Training ROC of CPH.

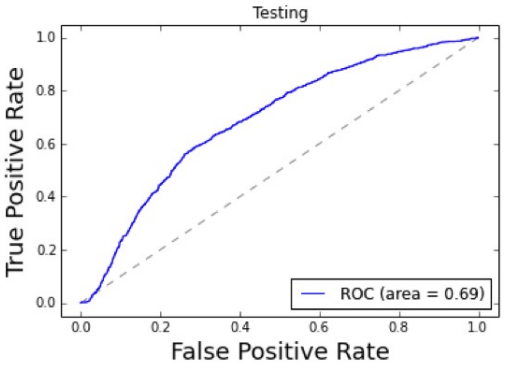

Figure 6. Testing ROC of CPH.

4. DISCUSSION

A novel deep learning structure is proposed to predict asset failure probability using sequential data by stacking deep feature learning and survival analysis. The deep learning structure consists of a Long Short Term Memory (LSTM) layer, a neural network layer, and a survival model layer. The learning process learns the feature representation and prediction task together using stochastic gradient decent. No separate feature extraction is needed. It provides an 'end-to-end' prediction model using sequential data. The proposed model is validated using data collected from a fleet of mining trucks. The result has shown that the model can predict the failure probability with acceptable result in a leave p-out test.

Given adequate sample size, the feature learning layer can be designed in a more sophisticated way to generate better feature representation. If the sequential data is not sampled in an equal time interval, zeros can be used to pad the missing sequences. A two-state model has been used in our survival analysis, which only considers failure/non-failure state. It is naturally to extend the model to deal with multi-state by modifying the likelihood function defined in Equation 11. It

should be noticed that a multi-state model will have transition probabilities among states as part of the parameters to learn. As the probabilities are bounded within 0 to 1, constraints need to be set in the learning process. If optimization with multiple non-equality constraints is not well supported in the deep learning package, alternative methods can be considered. The alternatives would be simply using a hard boundary on the parameters, or using Gibbs sampling (Carter & Kohn, 1994) which will take much longer time to train the model.

REFERENCES

Ahn, H.-i., Leung, Y. T., & Hochstein, A. (2014). Building a data-driven vital sign indicator for an economically optimized component replacement policy. In *Annual conference of the prognostics and health management (phm) society.*

Babu, G. S., Zhao, P., & Li, X.-L. (2016). Deep convolutional neural network based regression approach for estimation of remaining useful life. In *International conference on database systems for advanced applications* (pp. 214–228).

Benkedjouh, T., Medjaher, K., Zerhouni, N., & Rechak, S. (2015). Health assessment and life prediction of cutting tools based on support vector regression. *Journal of Intelligent Manufacturing*, *26*(2), 213–223.

Carter, C. K., & Kohn, R. (1994). On gibbs sampling for state space models. *Biometrika*, *81*(3), 541–553.

Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to forget: Continual prediction with lstm. *Neural computation*, *12*(10), 2451–2471.

Hong, S., Zhou, Z., Zio, E., & Wang, W. (2014). An adaptive method for health trend prediction of rotating bearings. *Digital Signal Processing*, *35*, 117–123.

Katzman, J., Shaham, U., Cloninger, A., Bates, J., Jiang, T., & Kluger, Y. (2016). Deep survival: A deep cox proportional hazards network. *arXiv preprint arXiv:1606.00931*.

Li, H., Pan, D., & Chen, C. P. (2014). Intelligent prognostics for battery health monitoring using the mean entropy and relevance vector machine. *Systems, Man, and Cybernetics: Systems, IEEE Transactions on*, *44*(7), 851–862.

Li, K., & Wang, Q. (2015). Study on signal recognition and diagnosis for spacecraft based on deep learning method. In *Prognostics and system health management conference (phm), 2015* (pp. 1–5).

Qiao, L. Q., & Xun, L. J. (2015). State of health estimation combining robust deep feature learning with support vector regression. In *Control conference (ccc), 2015 34th chinese* (pp. 6207–6212).

Tamilselvan, P., & Wang, P. (2013). Failure diagnosis using deep belief learning based health state classification. *Reliability Engineering & System Safety*, *115*, 124–135.

Trappey, A. J., Trappey, C. V., Ma, L., & Chang, J. C. (2015). Integrating real-time monitoring and asset health prediction for power transformer intelligent maintenance and decision support. In *Engineering asset management-systems, professional practices and certification* (pp. 533–543). Springer.

Yan, W., & Yu, L. (2015). On accurate and reliable anomaly detection for gas turbine combustors: A deep learning approach. In *Proceedings of the annual conference of the prognostics and health management society.*

BIOGRAPHIES

Linxia Liao is a Staff Data Scientist at GE Digital. His research interests include predictive and prescriptive analytics, prognostics, deep neural nets, and machine learning algorithms as well as their integration into cloud computing platforms. He was with the System Sciences Laboratory, Palo Alto Research Center, Palo Alto, CA, USA, developing solutions for device health prognostics and patient disease progression management. Prior to that, he was a Research Scientist at Siemens Corporate Research, building pipe-line machine learning applications to various fields in manufacturing, energy, and transportation. He received his Ph.D. degree in Industrial Engineering from the NSF I/UCR Center for Intelligent Maintenance Systems, Cincinnati, University of Cincinnati, OH, USA. He has one issued patent and over 10 pending patents, and has published one book chapter and more than 30 papers in leading journals and conference proceedings.

Hyung-il Ahn is a Principal Data Scientist at Noodle.ai. His work has focused on inventing enterprise-AI applications bringing better human experience, prediction and decision. Previously he worked at GE Digital as a senior staff data scientist, applying data-driven machine learning frameworks to industrial internet applications. He also worked at the IBM Almaden Research Center as a research staff member, developing social-media based predictive analytics and cognitive computing applications. He received his PhD degree in Media Arts and Sciences at MIT Media Lab (affective computing group). He has published broadly in the areas of affective computing, cognitive computing, predictive modeling, social media analytics, prognostics and health management, machine learning, artificial intelligence (AI) and neural computation.