Application of digital twins in the management of socio-economic systems

Sergey Barkalov[1], *Dmitry* Dorofeev[1], *Irina* Fedorova[1,*], and *Alla* Polovinkina[1]

[1]Voronezh State Technical University, Moscow Avenue, 14, Voronezh, 394026, Russia

Abstract. The article describes the use of digital twins in socio-economic processes using the example of predictive asset maintenance management. For this, the architecture of a distributed forecasting information system is proposed that collects data from digital twins and provides them with a pre-trained neural network model to obtain forecasts about the need for predictive maintenance. The article discusses two types of forecasts - about the remaining useful life and the possible failure of an asset in the considered time window. Computational experiments have been carried out, confirming that the proposed neural network model allows, due to the simultaneous training of solving two problems, to obtain more accurate forecasts than models trained to solve one problem.

1 Introduction

Industry 4.0 seeks to impact performance at three levels:

- asset level;
- the level of operational processes;
- enterprise level.

This article discusses asset digital twins. Asset management can be defined as the optimal management of the physical assets of an organization to maximize profits. Organizations can improve asset performance, reduce capital and operating costs associated with assets, extend asset life, and subsequently improve return on assets. In practice, asset management usually focuses on collecting information about assets: their location, configuration, frequency and nature of service and performance. However, the data collected alone does little to improve asset performance and decision making unless it is analyzed in depth.

1.1 Digital twins of industrial assets

The asset digital twin is its virtual image maintained throughout its entire life cycle. The asset digital twin collects detailed engineering and economic data to visualize, simulate, and analyze its functions. It is capable of collecting performance and reliability data in a variety of conditions for wind turbines, mining equipment or aircraft engines. For example,

* Corresponding author: fedorova_i@list.ru

collection and processing of reliability data aims to better understand and prevent equipment failures.

With the proliferation of Internet of Things technologies, assets have been able to exchange data, process it, and to some extent perceive the environment. These properties served as the basis for the development of such a concept as intelligent industrial assets, within which assets are given a certain degree of freedom of action.

This article presents a distributed joint forecasting system for predictive maintenance of industrial assets. In the proposed system, each asset is assigned a digital twin that processes asset data and performs predictive maintenance forecast through interaction with a social platform. The social platform collects data from digital twins and provides data exchange between them, if necessary.

1.2 Predictive maintenance

Predictive maintenance usually includes regular maintenance of industrial assets to avoid disruptions. However, in many cases this is done without real need, since the determination of the need for predictive maintenance is done based on hard-coded threshold rules. In this case, a service message is issued immediately if the measurement exceeds the threshold. This can lead, for example, to replace a piece of equipment after 8 months of operation, while it could have successfully operated for at least another 12 months. Machine learning predictive maintenance forecasting uses machine learning techniques to predict when the next failure will occur and, accordingly, carry out maintenance ahead of time.

The benefits of predictive maintenance include avoiding unplanned downtime, reducing planned downtime, increasing productivity, and optimizing the use of maintenance resources [1].

One of the key challenges of predictive maintenance is to predict impending equipment failure with a reasonable forecast horizon so that countermeasures can be taken. There are the following two forecasting problems, which are usually solved independently [2-5]:

- estimation of the remaining useful life (RUL), as a long-term forecasting problem to estimate how much time is left until the end of the useful life of equipment;
- binary forecast of asset failure in time windows (FTW), as a short-term forecasting problem to determine whether a failure will occur in the considered time window or not.

The RUL task is a regression analysis task and the FTW task is a multi-class classification task for asset failure in time windows.

It is clear that these two tasks are related and that performing them separately is not optimal and can lead to incompatible forecasts for the same equipment. To solve these problems, the authors of the article propose an algorithm based on multi-task learning (MTL) [6-9] for the regression problem and the multi-class classification problem. The RUL prediction is expected to provide useful information for predicting failures over time windows and vice versa.

2 Architecture of the system of joint distributed forecasting

The architecture used in this article is a hierarchical architecture with two layers: digital twins and social platform. The asset digital twin standardizes asset data and processes that data and requested data from other assets by its analytics engine. The asset digital twin can also communicate with a social platform that performs a variety of tasks, from collecting and presenting data at the system level to running clustering algorithms to form groups of similar assets. Let's look at the architecture levels in more detail.

2.1 Level of the digital twin

The digital twin level is represented by a software component that consists of three software modules: a data module, an analytics module, and a data exchange module.

The data module stores and manages asset data, data generated by the analytics engine, and data received by the data exchange module.

Asset data is 1) a time series of sensor values along with the time at which they were recorded; 2) a set of events associated with equipment failures or various warnings; 3) meta-information of an asset - for example, its identifier or its configuration or setting parameters.

The data generated by the analytics module are the data used as a source for building a forecast and the forecasts themselves for each time window. The forecast history can then be used to evaluate the performance of the forecasting algorithm. The analytics module can also be responsible for classifying asset events.

The forecast of the remaining useful life of the asset and the answer to the question - whether the asset will fail in the considered time window are performed in the analytics module based on a pre-trained neural network model, which is presented in section 3 of the article.

The data received by the data exchange module is information about events of interest to this digital twin, the value of the measure of similarity between the asset of the twin and other assets, and information that will be used in the forecast algorithm. The data exchange module is responsible for managing the communication between the digital twin and the social platform (higher level of the system architecture). The asset digital twin transfers data that it uses to build predictive maintenance forecasts to a social platform. As such data accumulates, the social platform can train the forecasting model of the neural network taking into account new data for each group of sufficiently homogeneous assets. Next, you can distribute the new pre-trained forecasting neural network models among the assets.

2.2 Social platform level

The social platform, as the upper level of the system architecture, is responsible for 1) ensuring and regulating communication between digital twins; 2) the formation of groups of interacting digital twins (assets); 3) training of new models of neural networks designed to perform the forecast of predictive asset maintenance; and 4) the formation of information at the enterprise level, in particular, to measure the accuracy of forecasts obtained by digital twins. Also, the social platform can store and update the technical knowledge base, providing a systematic record of failures, warnings and maintenance performed. This makes it easier to make predictive asset management decisions [10].

The social platform stores information about which assets will interact and exchange data with each other in a matrix, which also indicates scalar distances between assets.

3 Predictive maintenance model

3.1 Description and preparation of the dataset for training and evaluating

To train and evaluate the forecasting models proposed in this study, the turbofan (aircraft) engine degradation simulation dataset developed at NASA [11-12] is used. This data set was created from synthetic data collected from thermodynamic simulations. The model is called C-MAPSS (Commercial Modular Aero-Propulsion System Simulation).

The simulator consists of 14 inputs and 58 outputs, of which only 21 are specified in the dataset. The following information is provided for each turbofan engine:

- operating time (in flight cycles);
- 3 parameters of working conditions (height, engine identifier and throttle angle);
- 21 sensor signals (4 temperatures, 4 pressures, 6 speeds and 7 others).

The data schema for training and testing is shown in Table 1:

Table 1. Data schema

Index	Data field	Type	Description
1	id	integer	aircraft engine identifier
2	cycle	integer	operating time, in cycles
3	p1	real	parameter 1 working conditions
4	p2	real	parameter 2 working conditions
5	p3	real	parameter 3 working conditions
6	s1	real	sensor 1 value
7	s2	real	sensor2 value
…	…	real	…
26	s21	real	sensor 21 value

The initial dataset consists of 4 time series: FD001, FD002, FD003 and FD004 (Table 2). The time series contains sensor data from a number of aircraft engines that operate normally at the start of the recording and eventually fail. Two failure modes are possible: failure of the high pressure compressor and failure of the fan.

In the paper is used the following assumption about modeling data: the asset has a progressive degradation pattern that is reflected in the measurement values coming from the asset sensors. By studying the history of changes in sensor values of an asset over time, a machine learning algorithm can study the relationship between changes in sensor values and failures that have occurred in order to predict future failures.

Table 2. Composition of the set of initial data for modeling degradation of an aircraft engine

Dataset name	Examples in the training dataset	Examples in a test dataset	Failure modes
FD001	100	100	high pressure compressor degradation
FD002	260	259	high pressure compressor degradation
FD003	100	100	fan degradation and high pressure compressor degradation
FD004	248	249	fan degradation and high pressure compressor degradation

Each model takes three datasets as input:

- training data - data on the operation of the aircraft engine to failure;
- test data - data on aircraft engine operation without registration of failures;
- valid data - information about the ground truth remaining operation cycles for each engine from the test data.

Input data consists of "train_FD001.txt", "train_FD002.txt", "train_FD003.txt", "train_FD004.txt", "test_FD001.txt", "test_FD002.txt", "test_FD003.txt", "test_FD004.txt" and “RUL_FD001.txt"," RUL_FD002.txt"," RUL_FD003.txt ", "RUL_FD004.txt" from the original dataset.

The training data ("train_FD001.txt", "train_FD002.txt", "train_FD003.txt", "train_FD004.txt") consists of several multivariate time series with "cycle" as a unit of time along with 21 sensor values for each cycle. Each time series is generated by other equipment (aircraft engine) of the same type. Each engine is assumed to initially start and operate normally with varying degrees of initial degradation and manufacturing variance, and this information is unknown to the user. Its degradation begins at some point during a series of operating cycles. Degradation is progressing and increasing. When a predetermined threshold value of one of the sensors is reached, the engine is considered unsafe for further operation. Thus, the last cycle in each time series can be viewed as the point of failure of the corresponding engine. Taking the training data shown in Table 3 as an example, the engine with id = 1 fails in cycle 149 and the engine with id = 2 fails in cycle 269.

Table 3. Training data sample

id	cycle	p1	p2	p3	s1	s2	...	s20	s21
1	1	34.9983	0.8400	100.0	449.44	555.32	...	14.73	8.8071
1	2	41.9982	0.8408	100.0	445.00	549.90	...	10.41	6.2665
...	...	...	...	...	...	...	...	...	...
1	149	42.0017	0.8414	100.0	445.00	550.49	...	10.45	6.2285
2	1	0.0025	0.0000	100.0	518.67	642.04	...	38.88	23.2326
2	2	35.0058	0.8400	100.0	449.44	556.13	...	14.86	8.9052
..	...	...	...	...	...	...	...	...	...
2	269	42.0047	0.8411	100.0	445.00	550.11	...	10.56	6.2615

The test data ("test_FD001.txt", "test_FD002.txt", "test_FD003.txt", "test_FD004.txt") has the same data schema as the training data. The only difference is that the last time period does not represent a point of failure. Taking the data in Table 4 as an example, an engine with id = 1 runs from cycle 1 to cycle 258. It is not shown how many more cycles this engine can take before failing.

Table 4. Test data sample

id	cycle	p1	p2	p3	s1	s2	...	s20	s21
1	1	9.9987	0.2502	100.0	489.05	605.03	...	28.42	17.1551
1	2	20.0026	0.7000	100.0	491.19	607.82	...	24.29	14.8039
...	...	...	...	...	...	...	...	...	...
1	258	10.0076	0.2501	100.0	489.05	605.42	...	28.30	17.0934
2	1	42.0079	0.8408	100.0	445.00	549.22	...	10.74	6.3394
2	2	35.0021	0.8410	100.0	449.44	554.99		14.91	8.9154
...	...	...	...	...	…	...	...	...	...
2	55	0.0018	0.0000	100.0	518.67	642.67	...	38.82	23.3463

The valid data ("RUL_FD001.txt", "RUL_FD002.txt", "RUL_FD003.txt", "RUL_FD004.txt") provides information on the number of remaining engine cycles in the

test data. Taking the valid data shown in Table 5 as an example, an engine with id = 1 from the test data can complete 18 more cycles before failure occurs.

Table 5. Valid data sample

id	RUL
1	18
2	79
...	...
259	51

Data labeling

Considering the history of aircraft engine operation and the history of failure events, it is possible to predict when a running engine will fail based on two interrelated machine learning models:

the regression model (Reg model) answers the question of how many more cycles a running engine will last before it fails;

the multi-class classification model (MCC model) answers the question whether the engine will fail in the time window [1, w] of operating cycles or fail in the time window [w + 1, 2w] of operating cycles, or it will not fail in the considered time windows (w is a configurable parameter, the value of which can be selected depending on the subject area. The consumer of forecasting services must decide how far in advance the warning about the failure should be triggered before the actual event of equipment failure. In the article w = 10).

Using the engine with id = 1 as an example, Table 6 shows how the training data is labeled, where "RUL" and "Class label" are labels for regression and multi-class classification models, respectively. For the case of multi-class regression: if engine failure does not occur in the considered time window, then it is marked as class 0. Labels for test data are generated from valid data following the same procedure as when labeling training data.

Table 6. Data labeling sample

id	cycle	p1	p2	p3	s1	...	s20	s21	RUL	Class label
1	1	34.9983	0.8400	100.0	449.44	...	14.73	8.8071	148	0
1	2	41.9982	0.8408	100.0	445.00	...	10.41	6.2665	147	0
...	...	...	...	...	...	...	...	...	…	…
1	139	41.9986	0.8408	100.0	445.00	...	10.52	6.2593	10	0
1	140	35.0069	0.8406	100.0	449.44	...	14.68	8.7953	9	14
1	141	34.9991	0.8410	100.0	449.44	...	14.61	8.9616	8	14
1	142	20.0025	0.7000	100.0	491.19	...	24.13	14.6524	7	14
1	143	35.0014	0.8400	100.0	449.44	...	14.71	8.9133	6	14

1	144	0.0017	0.0018	100.0	518.67	...	38.27	23.1236	5	14
1	145	35.0007	0.8402	100.0	449.44	...	14.72	8.8240	4	14
1	146	10.0006	0.2516	100.0	489.05	...	28.16	16.9984	3	14
1	147	25.0005	0.6200	60.0	462.54	...	14.34	8.4476	2	14
1	148	0.0024	0.0013	100.0	518.67	...	38.55	23.0079	1	14
1	149	42.0017	0.8414	100.0	445.00	...	10.45	6.2285	0	14

Selection of data features

Another important task in step 1 is the selection and processing of features for training and test data.

Features that are contained or generated from training data can be grouped into two categories:

- raw features that are contained in the original dataset. In deciding which raw features should be included in the training data, both a detailed description of the data fields and knowledge of the subject area are helpful. In the set under consideration, all sensor measurements (s1-s21) are included in the training data. Other raw features are also used: operating cycle, parameter 1-3 values.
- aggregated features - the values of such features reflect the historical changes of each asset. So, for each of the 21 sensors, two types of aggregated features are created: a1-a21: a moving average of the sensor values for the last w1 cycles; sd1-sd21: standard deviation of sensor values in the most w1 recent cycles (for the parameter w1, the value w1 = 5 was used, but it can be adjusted differently depending on the subject area).

Preparing data for testing

The test dataset is formed as follows: data from "test_FD001.txt", "test_FD002.txt", "test_FD003.txt" and "test_FD004.txt" are used to generate aggregated features, and data from "RUL_FD001.txt", "RUL_FD002.txt", "RUL_FD003.txt" and "RUL_FD004.txt" are used to create RUL and Class labels for the two training models.

Similar to the training dataset, the time series data in the test dataset helps generate aggregate features.

3.2 Multi-task learning model (MTL) architecture

Data first from the training dataset and then from the test dataset X = $\{x_1, x_2, ..., x_n\}$ are fed to the input to a one-dimensional convolutional (Conv1D) layer to extract short-term regularities of the series. The Conv1D layer contains 32 one-dimensional filters of size 5. This is followed by the ReLU activation layer. After that, the Max Pooling layer is located. The Max Pooling layer allows you to reduce the dimension of the hidden representation of the data space and the amount of computations in the neural network. Then there are two layers of Bi-LSTM, 32 and 16 units each. The feature vector is then passed to the Dense layer, followed by the ReLU activation layer and the Dropout layer (a factor of 0.2 is used for the layer). All of the above layers were common to Reg and MCC models. Finally, the common feature vector is transferred to two independent Dense layers. Behind the first Dense layer is the Softmax activation layer and it gives a classification score. This is how the MTL network outputs the probability values of the classes. Behind the second Dense layer is the ReLU activation layer, which evaluates the regression and displays the time at

which the engine failed. All used names of neural network layers are standard in the field of machine learning and can be clarified in the relevant literature [13,14,15].

3.3 Training and evaluating of the MTL model

3.3.1 MTL training

Neural network training consists in simultaneous training of Reg multiple regression problems and MCC multi-class classification problems. Let us choose the loss functions for each of the problems in accordance with their type: for the MCC model, this is the cross-entropy function L_{MCC}, and for the Reg model, we will use the Huber's loss function L_{Reg} (to make training more resistant to outliers in the initial data) [16,17,18,19].

The goal of multi-task training is to minimize losses for both tasks. Therefore, the joint loss function is determined by the weighted average sum of all loss functions for the tasks:

$$L_{full} = \frac{1}{2}\left(\alpha_{MCC} L_{MCC} + \alpha_{Reg} L_{Reg}\right) \quad (1)$$

where the weighting parameters α_{MCC}, α_{Reg} are determined by the importance of the problem in total losses. For errors in the main task, additional penalties are imposed. Therefore $\alpha_{Reg} = 3\alpha_{MCC}$.

According to the architecture of the multi-task learning model, both tasks share common features and structure of the neural network during iterative learning. They are separated only on the last Dense layer.

3.3.2 Evaluating the MTL model

The following metrics were used to evaluate the quality of the classification model:

- overall accuracy and average accuracy;
- micro-averaged recall and macro-averaged recall;
- micro-averaged precision and macro-averaged precision,
- which are defined by equalities (2) - (7).

$$\text{Overall accuracy} = \frac{\sum_{s=1}^{S}(TP_s + TN_s)}{\sum_{s=1}^{S}(TP_s + TN_s + FP_s + FN_s)} \quad (2)$$

$$\text{Average accuracy} = \frac{\sum_{s=1}^{S} A_s}{S} \quad (3)$$

$$\text{Micro} - \text{average recall} = \frac{\sum_{s=1}^{S} TP_s}{\sum_{s=1}^{S}(TP_s + FN_s)} \quad (4)$$

$$\text{Macro} - \text{average recall} = \frac{\sum_{s=1}^{S} R_s}{S}, \quad (5)$$

$$\text{Micro} - \text{average precision} = \frac{\sum_{s=1}^{S} TP_s}{\sum_{s=1}^{S}(TP_s + FP_s)} \quad (6)$$

$$\text{Macro} - \text{average precision} = \frac{\sum_{s=1}^{S} P_s}{S} \quad (7)$$

where $R_s = \frac{TP_s}{TP_s + FN_s}$ is the recall for class s, is the precision for class s, $P_s = \frac{TP_s}{TP_s + FP_s}$ is the precision for class s,

and $A_S = \frac{TP_S+TN_S}{TP_S+TN_S+FP_S+FN_S}$ is the accuracy for class s.

S is the number of classes. TP_s - number of true positives for class s, FP_s- number of false positives for class s, FN_s- number of false negatives for class s, TN_s- number of true negative results for class s.

The following metrics are used to evaluate the quality of the regression model:

the mean absolute error (MAE), which is determined by equality (8) and is a measure of the proximity between the predicted output and the actual output;

root mean square error (RMSE), defined by equality (9);

R-squared is the proportion of the variance of the dependent variable explained by the model under consideration and determined by equality (10).

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{8}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{9}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i-\hat{y}_i)^2}{\sum_{i=1}^{N}(y_i-\bar{y})^2} \tag{10}$$

where $\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$ is the sample mean, $\hat{y}_i$ is the predicted value of y_i, N is the amount of data.

3.4 Experiment

To test the effectiveness of the multi-task learning model for each forecasting problem, two baseline neural network models were selected to compare the score metrics for the Reg and MCC problems. Base models are single-task models for forecasting and classification. For the problem of multi-class classification, the CNN + Bi-LSTM neural network was chosen as the basic model, and for the regression problem, the Bi-LSTM neural network was used as the basic model. All three models were implemented on the basis of tensorflow machine learning framework, trained and tested on the same datasets. The results are shown in Table 7.

Table 7. Values of evaluation measures for three models of neural networks.

Model	Overall accuracy	Average accuracy	Micro-averaged recall	Macro-averaged recall	Micro-averaged precision	Macro-averaged precision	MAE	RMSE	R^2
CNN+Bi-LSTM	-	-	-	-	-	-	15.1	18.9	0.85
Bi-LSTM	0.91	0.94	0.91	0.83	0.91	0.83	-	-	-
MTL	0.93	0.94	0.92	0.84	0.92	0.84	13	17.8	0.89

Based on the obtained results, we can conclude that the multi-task learning model in forecasting predictive maintenance is superior to single-task models on the considered dataset.

4 Conclusion

Machine Learning empowers Industry 4.0 with the ability to analyze massive amounts of data in real time and generate timely warnings about problems that may arise. The authors in the paper propose the architecture of a system for joint distributed forecasting of predictive maintenance, which allows evaluating the health and behavior of assets based on the collection and processing of information from their digital twins. Based on the proposed multi-task neural network model, the system can detect dangerous deterioration of assets before they fail. The performed computational experiment showed that the proposed neural network model allows one to simultaneously obtain a forecast of the residual useful life and a forecast of the time window in which the equipment will fail. At the same time, the metrics for evaluating the neural network model showed that while simultaneously obtaining a forecast for two predictive maintenance tasks, the accuracy of the model exceeds the models of basic neural networks trained to predict only one task.

References

1. S. Barkalov, P. Kurochka, T. Nasonova, *Optimal placement of maintenance facilities* MATEC Web of Conferences conference proceedings, 01124 (2018)
2. C. Zheng, W. Liu, B. Chen, D. Gao, Y. Cheng, Y. Yang, J. Peng, A Data-driven Approach for Remaining Useful Life Prediction of Aircraft Engines. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November, 184–189 (2018)
3. Z. Chen, S. Cao, Z. Mao, Remaining useful life estimation of aircraft engines using a modified similarity and supporting vector machine (SVM) approach. Energies, **11**, 28 (2018)
4. J. B. Ali, B. Chebel-Morello, L. Saidi, S. Malinowski, *Accurate bearing remaining useful life prediction based on Weibull distribution and artificial neural network*. Mech. Syst. Signal Process, 1–23 (2014)
5. A. Al-Dulaimi, S. Zabihi, A. Asif, A. Mohammadi, A multimodal and hybrid deep neural network model for Remaining Useful Life estimation. Comput. Ind., **108**, 186–196 (2019)
6. C. Zhao, V. Badrinarayanan, C.-Y. Lee, A. Rabinovich, *Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks*. In Proceedings of the International Conference on Machine Learning (2018)
7. A. Kendall, Y. Gal, R. Cipolla, *Multi-task learning using uncertainty to weigh losses for scene geometry and semantics*, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 7482–7491 (2018)
8. A. R Zamir, A. Sax, W. Shen, L. J. Guibas, J. Malik, S. Savarese, *Taskonomy: Disentangling task transfer learning*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3712–3722 (2018)
9. Z. Zhanpeng, L. Ping, C. L. Chen, T. Xiaoou, Facial landmark detection by deep multi-task learning. In European conference on computer vision, 94–108 Springer (2014)
10. V. E. Belousov, P. N. Kurochka, T. A. Averina, *Algorithms of a logical conclusion of knowledge in difficult technical systems on the basis of indistinct rules*. *11th IEEE International Conference on Application of Information and Communication Technologies*, AICT 2017 - Proceedings, **11**, 8687040 (2019)
11. D. K. Frederick, J. A. DeCastro, Litt, J.S. User's Guide for the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS); NASA Glenn Research Center (Cleveland, OH, USA, 2007)

12. A. Saxena, K. Goebel, Turbofan engine degradation simulation data set. NASA Ames Prognostics Data Repository (https://tiarcnasagov/tech/dash/groups/pcoe/prognostic-data-repository/), NASA Ames Research Center, Moffett Field, CA (2008)
13. L. Wen, Y. Dong, L. Gao, A new ensemble residual convolutional neural network for remaining useful life estimation. Math. Biosci. Eng., **16**, 862–880 (2019)
14. B. Hakan, V. Andrea, Integrated perception with recurrent multi-task neural networks. In Advances in neural information processing systems, 235–243 (2016)
15. X. Li, Q. Ding, J. Q. Sun, Remaining useful life estimation in prognostics using deep convolution neural networks. Reliab. Eng. Syst. Saf., **172**, 1–11 (2018)
16. S. Lambert-Lacroix, L. Zwald, "Robust regression through the Hubers criterion and adaptive lasso penalty, "Electron. J. Statist., **5**, 1015–1053 (2011)
17. A. Bhardwaj, W. Di, J. Wei, Deep Learning Essentials: Your hands-on guide to the fundamentals of deep learning and neural network modeling. Birmingham: Packt Publishing Limited (2018)
18. M. Martinez, R. Stiefelhagen, *Taming the Cross Entropy Loss*. Computer Science, Mathematics, **3** (2018)
19. J. T. Barron, *A General and Adaptive Robust Loss Function*. *InProc*. *of the IEEE Conf*. *on Computer Vision and Pattern Recognition* (CVPR) (2019)