Proceedings of The 2022 IEEE International
Conference on Real-time Computing and Robotics
July 17-22, 2022, Guiyang, China

2022 IEEE International Conference on Real-time Computing and Robotics (RCAR) | 978-1-6654-6983-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/RCAR54675.2022.9872225

Real-time Prediction Method of Remaining Useful Life Based on TinyML

Hongbo Liu, Ping Song*, Youtian Qie, Yifan Li

***Abstract*—Tiny Machine Learning (TinyML) is a new research area aimed at designing and developing machine learning (ML) techniques for embedded systems and IoT units. Due to the limited resources of embedded system, neural network pruning is widely used to reduce resource occupation. To solve the problem that the Remaining Useful Life (RUL) of the equipment is difficult to calculate accurately and in real time, a pruning method based on L1 norm weight was designed to reduce the memory footprint and computational load of the neural network, and a lightweight two-dimensional convolutional neural network was constructed. Experimental results show that compared with random pruning, this method greatly reduces the influence of neural network parameter reduction on the accuracy of inference results. Meanwhile, a retraining method based on Adam optimization was used to make the RUL curve predicted by the retrained model more close to the real RUL curve. When the weight parameters are reduced by 30%, the model still maintains good prediction accuracy, and can realize the real-time prediction of RUL in the embedded system with limited resources.**

I. INTRODUCTION

In the fields of aerospace, military equipment and industrial manufacturing, it is of great practical significance to study the RUL of key equipment for improving maintenance strategy and avoiding failure effectively. RUL is one of the important indexes in reliability analysis. It refers to the time interval from the current time when the operating conditions start to the time when the critical equipment fails [1]. Under the background of big data, the traditional life prediction technology is gradually changing to the direction of artificial intelligence prediction. In the aspect of RUL prediction using artificial intelligence technology, data-driven life prediction method has become the mainstream [2][3].

In 2010, Dutch scholar Tiedo proposed the concept of physical failure model based on using load parameters and applied it to failure analysis of aircraft blades [4]. In 2016, Cheng Yang et al. from Beijing Institute of Technology firstly studied the state identification and fault prediction of the hydro-pneumatic spring, the core component of the suspension system of special vehicles, based on the data-driven method for the demand of state-based maintenance and independent guarantee of special vehicles [5]. In 2017, Xiongjun Liu et al from Beijing Institute of Technology proposed multi-path processing, multi-time fault feature set construction and adaptive degradation degree index selection algorithm based on empirical mode decomposition based on the analysis of degradation mechanism of bearing components of integrated drive system [6]. In 2018, Juan LI et al proposed a RUL prediction method based on the consistency test of Wiener process. Based on the consistency test of degradation model, a weighted fusion RUL prediction method for aircraft fuel pump was proposed, which provided theoretical and practical guidance for engineers to predict RUL after equipment maintenance [7].

In 2020, WON et al. developed an extended prognostic model that accurately estimated the RUL of complex systems or facilities based on degradation data. They use machine-learning featuring Smoothing, logging, variable transformation and clustering to this end [8]. In 2021, CHEN et al. proposed a deep learning framework for attention-based machine RUL prediction, and adopted LSTM network to learn sequence features from raw perceptual data, thus improving the performance of RUL prediction [9].

Execution of machine learning solutions is mostly limited to high-performance computing platforms such as GPUs or FPGAs due to the high computing and memory requirements of these solutions [10]. In recent years, with the development of Internet of Things technology, people begin to combine traditional neural network with embedded system. Thus, the delay of "data production to decision" can be reduced and the edge terminal real-time computing and inference can be realized [11][12].

In this paper, a real-time prediction method of RUL based on TinyML is proposed, and a retraining compression method based on L1 norm weight pruning and Adam optimization is designed to construct a lightweight two-dimensional convolutional neural network [13]. Compared with random pruning, this method can obtain higher accuracy at the same pruning rate. The neural network can be almost restored to the inference effect before pruning by using retraining method while reducing memory usage and computational load. The trained neural network model is deployed on the related embedded platform STM32U5 to complete the inference operation. Compared with the above methods, the data transmission process is omitted and the RUL is calculated in real time by using neural network at the edge terminal.

This work was supported by The Aeronautical Science Foundation of China (Grant No. 201946072002).

HongBo Liu is studying in mechanical engineering, Beijing Institute of Technology, Beijing. (e-mail: 15522305072@163.com).

Ping Song is a Professor of mechanical engineering at Beijing Institute of Technology, Beijing (Correspondence e-mail: sping2002@bit.edu.com)

Youtian Qie is studying in mechatronic engineering, Beijing Institute of Technology, Beijing. (e-mail: qieyoutian@bit.edu.cn).

Yifan Li is studying in mechatronic engineering, Beijing Institute of Technology, Beijing. (e-mail: yifanli@bit.edu.cn).

II. METHOD

A. Model

The fault data generated by equipment in complex system is usually time series. We can usually get more information from multivariate time series data. Time series processing has greater potential to provide better predictive performance [14]. In this paper, time window is used for data preparation to deal with multi-variable time information. See Section III for details of data preparation.

A two-dimensional convolutional neural network containing five convolutional layers is used [15], and the network structure is shown in Figure 1.

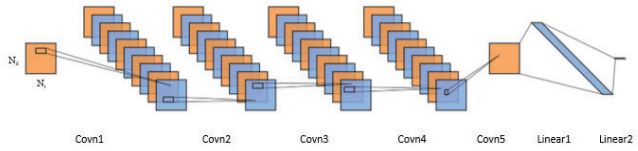


Figure 1. Neural network structure.

First, input a two-dimensional data sample with size $N_t \times N_d$, where N_t represents the time series size and N_d represents the number of selected features. Next, five convolutional layers are placed in the neural network for feature extraction, and Tanh is used as the activation function for all layers. The two-dimensional feature graph is then flattened and connected to the Linear layer to predict RUL, and the Dropout layer is used to prevent overfitting. The number of parameters of each layer in the neural network and the size of the output characteristic graph are shown in Table I.

In this paper, root mean square error (RMSE) is used to evaluate the performance of neural network, and the calculation formula is as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{N=1}^{N}\left(RUL_{TR} - RUL_{PR}\right)} \tag{1}$$

Where, RUL_{TR} is the real value of engine RUL, RUL_{PR} is the predicted value of engine RUL by the neural network.

B. Model Compression Method

This paper proposes a method based on L1 norm weight pruning and Adam optimization algorithm retraining [16][17], as shown in Figure 2. This method can realize the lightweight neural network model and ensure the computational performance of the model and the accuracy of the inference results.

Weight pruning method based on L1 norm:

- Set pruning layer and pruning ratio λ.
- The weight parameters of each layer are sorted by L1 norm.
- Assuming that the number of weight parameters in this layer is M, $M \times \lambda$ weight parameters with smaller order are pruned.

TABLE I. THE NUMBER OF PARAMETERS AT EACH LAYER AND OUTPUT SIZE

Layer(type)	Output Shape	Param
ZeroPad2d-1	[-1,1,39,14]	0
Conv2d-2	[-1,10,30,14]	110
Tanh-3	[-1,10,30,14]	0
ZeroPad2d-4	[-1,10,30,14]	0
Conv2d-5	[-1,10,30,14]	1010
Tanh-6	[-1,10,30,14]	0
ZeroPad2d-7	[-1,10,30,14]	0
Conv2d-8	[-1,10,30,14]	1010
Tanh-9	[-1,10,30,14]	0
ZeroPad2d-10	[-1,10,30,14]	0
Conv2d-11	[-1,10,30,14]	1010
Tanh-12	[-1,10,30,14]	0
Conv2d-13	[-1,1,30,14]	31
Tanh-14	[-1,1,30,14]	0
Linear-15	[-1,100]	42100
Tanh-16	[-1,100]	0
Dropout-17	[-1,100]	0
Linear-18	[-1,1]	101

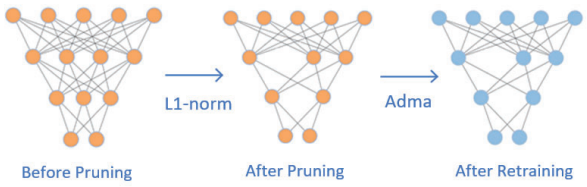


Figure 2. Model compression process.

The retraining method based on Adam optimization algorithm uses backpropagation learning to update the weights in the network. Adam optimization algorithm is used with small batch updates. The samples were randomly divided into multiple small batches, and each small batch of samples was input into the training system. Then, the network information, namely the weight of each layer, is optimized based on the average loss function of each small batch. It should be noted that the selection of batch size affects network training performance.

The network is trained by Adam optimization algorithm [18]. Adam optimization algorithm combines Momentum and RMSProp optimization algorithm advantages of fast convergence speed and strong anti-noise ability. The specific steps are as follows:

- Input learning rate LR; attenuation coefficient of moment estimation ρ_1, ρ_2; small constant term σ; initialization of neural network coefficients δ;

initialization of the first and second moment variables $s=0, r=0$; the number of iterations $t=1$

- Use the data of training set to train and output the loss value: e
- Calculate gradient and update iteration times:

$$g \leftarrow \nabla_\theta L(f_\theta(x), y) t \leftarrow t+1$$

- Update the first moment variable:

$$s \leftarrow \rho_1 s + (1-\rho_1) g$$

- Update the second moment variable:

$$r \leftarrow \rho_2 r + (1-\rho_2) g \odot$$

- Correct deviations of the first and the second moment:

$$\hat{s} \leftarrow \frac{s}{1-\rho_1^t}, \hat{r} \leftarrow \frac{r}{1-\rho_2^t}$$

- Calculate the coefficient update:

$$\Delta\theta = -LR \frac{\hat{s}}{\sqrt{\hat{r}} + \delta}$$

- Update coefficient: $\theta \leftarrow \theta + \Delta\theta$; repeat the above steps.

III. EXPERIENT

A. C-MAPSS Dataset and Preprocessing

The experimental data for this experiment were provided by NASA Center for Excellence in Fault Prediction [19], as shown in Table II. The data set consists of four sub-data sets, and each sub-data set is divided into training set and test set. And each subset includes 26 columns of data: engine number, engine operation cycle, 3 columns of operable operation environment Settings and 21 columns of time series data collected by different sensors.

TABLE II. C-MAPSS DATASET

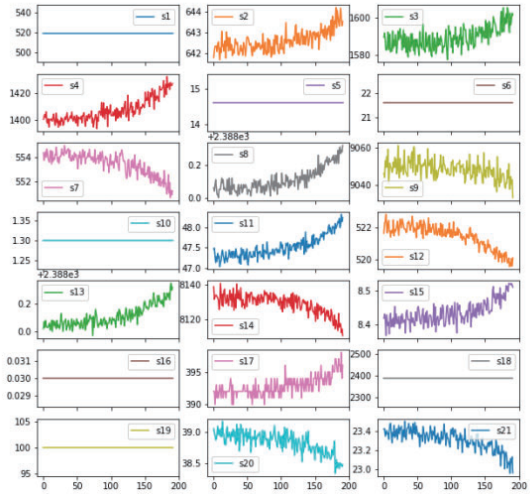
Dataset	FD001	FD002	FD003	FD004
Engine units for training	100	260	100	249
Engine units for testing	100	259	100	248
Operating conditions	1	6	1	6
Fault modes	1	1	2	2

The data collected by some sensors have a constant output over the life of the engine and cannot be used as a basis for predicting RUL. Therefore, 14 of the 21 sensors were selected to collect data as the original input features. For the four subdata sets in c-mapss, the min-max normalization method is used to normalize the measurement data collected by each sensor into the range of $[-1,1]$:

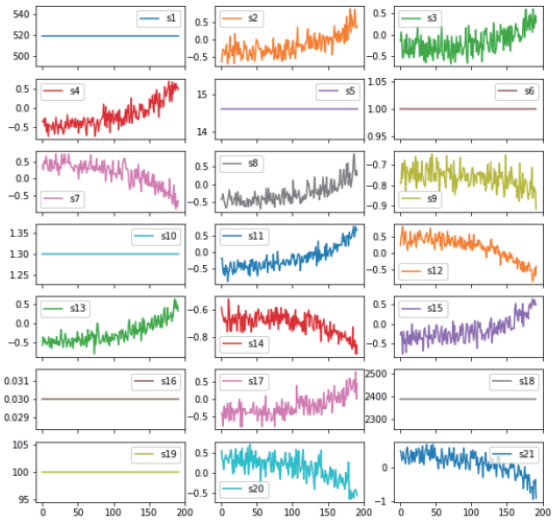
$$x_{norm}^{i,j} = \frac{2\left(x^{i,j} - x_{\min}^{i}\right)}{x_{\max}^{j} - x_{\min}^{j}} - 1 \quad (2)$$

Where, $x^{i,j}$ represents the ith original data of the jth sensor, $x_{norm}^{i,j}$ is the normalized value of $x^{i,j}$, $x_{\max}^{j}$ and $x_{\min}^{j}$ represent the maximum and minimum values of the original data of the jth sensor respectively. The original data and normalized data of 21 sensors are shown in the Figure 3.

The experiment set the time window N_t=30 and collected all sensor data within the time window. In a time window of size 30, a normalized data sample from 14 selected sensors forms a two-dimensional feature vector as the input of the network.



(a) The original data.



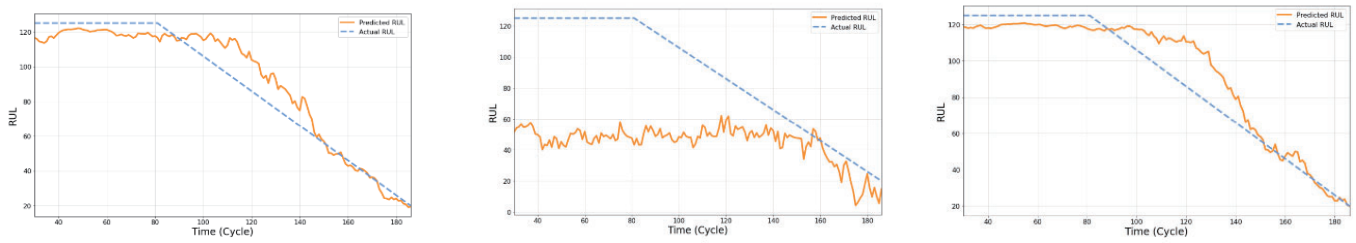
(b) The normalized data.

Figure 3. Sensors data.

B. *Weight Pruning Analysis Based on L1 Norm*

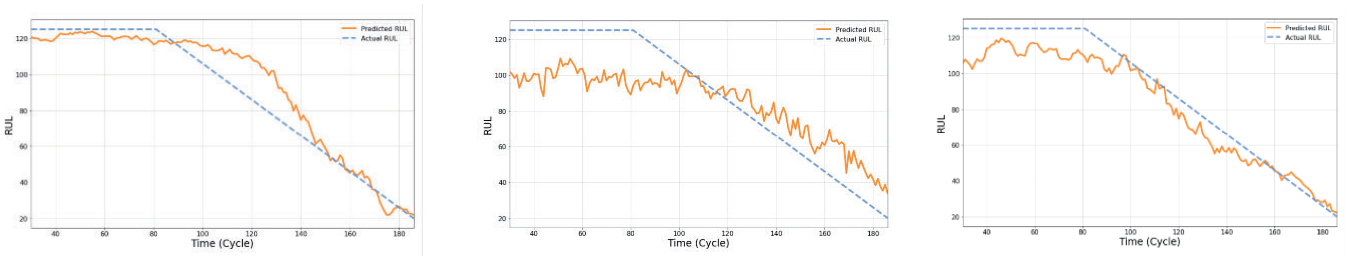
In order to verify the superiority of weight pruning based on L1 norm to network accuracy. Random pruning was designed as a comparative experiment, and the prediction results of engine RUL are shown in the Figure 4 Figure 5 and Figure 6. In the experiment, each small batch contained 512 samples, and the epoch was trained 20 times.

As can be seen from the figure above, L1 norm pruning model will lose certain accuracy with the increase of pruning proportion. This is a corollary of reducing computational load. It is very valuable to make a proper tradeoff between computational load and accuracy to achieve the feasibility of low-power execution with reduced precision loss.



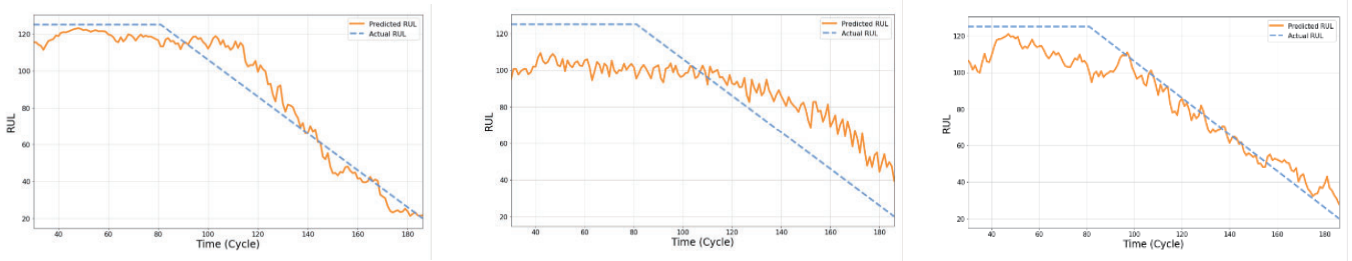
(a) Before pruning (b) Random pruning (c) L1 norm pruning

Figure 4. Convolutional layer pruning ratio of 10%.



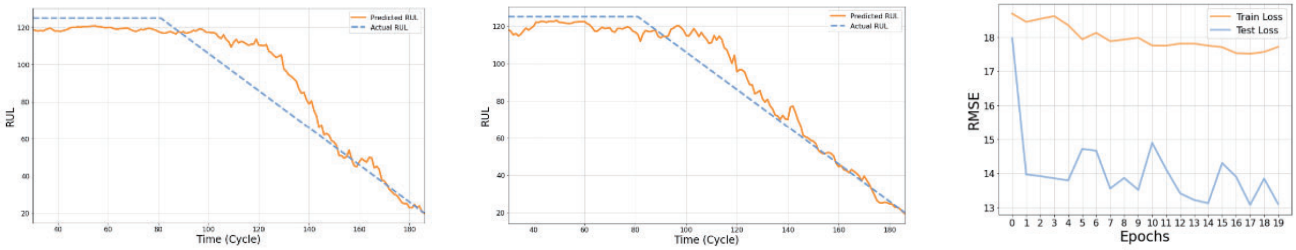
(a) Before pruning (b) Random pruning (c) L1 norm pruning

Figure 5. Convolutional layer pruning ratio of 20%.



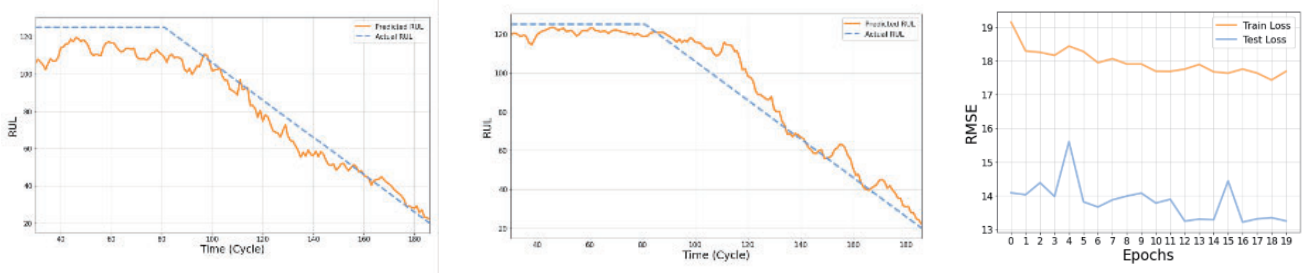
(a) Before pruning (b) Random pruning (c) L1 norm pruning

Figure 6. Convolutional layer pruning ratio of 30%.



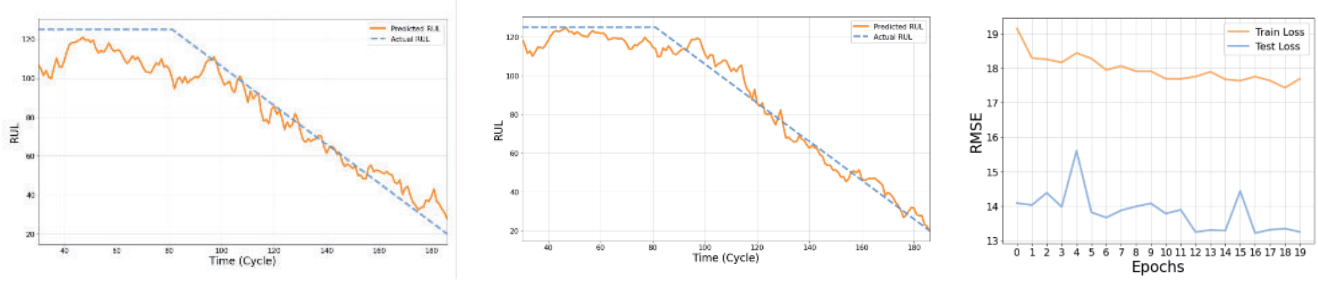
(a) Before retraining (b) After retraining (c) RMSE during retraining

Figure 7. Convolutional layer pruning ratio of 10%.



(a) Before retraining (b) After retraining (c) RMSE during retraining

Figure 8. Convolutional layer pruning ratio of 20%



(a) Before retraining (b) After retraining (c) RMSE during retraining

Figure 9. Convolutional layer pruning ratio of 30%

C. *Retraining Method Analysis Based on Adam Optimization*

After pruning based on L1 norm weight, the neural network is retrained based on Adam optimization. In the experiment, the learning rate of Adam optimization algorithm was 0.001. The prediction results of neural network before and after retraining and RMSE values during retraining are shown in Figure 7 Figure 8 and Figure 9.

According to the figures, under different pruning ratios, the predicted RUL curve after retraining is closer to the real RUL curve. The RMSE values of each epoch were observed during the retraining process, and the overall trend was downward, indicating that the retraining method based on Adam optimization could improve the accuracy of neural network. This method still applies when the pruning ratio reaches 30%.

IV. Conclusion

A retraining compression method based on L1 norm weight pruning and Adam optimization was designed for the prediction of the RUL of two-dimensional convolutional neural network. Experimental results show that this method is feasible and can effectively reduce the memory footprint of weighted parameters and the computational load of neural network. At the same time, the accuracy of RUL prediction is guaranteed. In this paper, machine learning and embedded system are combined to realize the real time inference of edge terminal neural network and the real time computation of RUL.

References

[1] Jardine, A. , Lin, D. , & Banjevic, D. . (2006). A review on machinery diagnostics and prognostics implementing condition-based maintenance - sciencedirect. Mechanical Systems and Signal Processing, 20(7), 1483-1510.

[2] Vichare, N. M. , & Pecht, M. G. . (2006). Prognostics and Health Management of Electronics. IEEE.

[3] Li, R. , Chu, Z. , Jin, W. , Wang, Y. , & Hu, X. . (2021). Temporal Convolutional Network Based Regression Approach for Estimation of Remaining Useful Life. ICPHM2021.

[4] Tinga, T. . (2010). Application of physical failure models to enable usage and load based maintenance. Reliability Engineering & System Safety, 95(10), 1061-1075.

[5] Yang, C. , Song, P. , & Liu, X. . (2017). Failure prognostics of heavy vehicle hydro-pneumatic spring based on novel degradation feature and support vector regression. Neural Computing & Applications.

[6] Liu, X. , Song, P. , Yang, C. , Hao, C. , & Peng, W. . (2017). Prognostics and health management of bearings based on logarithmic linear recursive least-squares and recursive maximum likelihood estimation. IEEE Transactions on Industrial Electronics, 1-1.

[7] (2018). Remaining useful life prediction based on variation coefficient consistency test of a wiener process. Chinese Journal of Aeronautics, 01(v.31;No.142), 112-121.

[8] Won, D. Y. , Sim, H. S. , & Kim, Y. S. . (2020). Prediction of remaining useful lifetime of membrane using machine learning. Science of Advanced Materials.

[9] Chen, Z. , Wu, M. , Zhao, R. , Guretno, F. , & Li, X. . (2020). Machine remaining useful life prediction via an attention based deep learning approach. IEEE Transactions on Industrial Electronics, PP(99), 1-1.

[10] Ren, H. , Anicic, D. , & Runkler, T. . (2021). Tinyol: tinyml with online-learning on microcontrollers.

[11] Alippi, C. , Disabato, S. , & Roveri, M. . (2018). Moving Convolutional Neural Networks to Embedded Systems: The AlexNet and VGG-16 Case. Acm/ieee International Conference on Information Processing in Sensor Networks (pp.212-223). ACM.

[12] MD Prado, Rusci, M. , Donze, R. , Capotondi, A. , Monnerat, S. , & And, L. B. , et al. (2020). Robustifying the deployment of tinyml models for autonomous mini-vehicles.

[13] Chen, Z. , Wang, D. , Fang, H. , Wang, G. , & Xie, B. . (2021). Rapid identification method of fresh tea leaves based on lightweight model. 2021 IEEE International Conference on Real-time Computing and Robotics (RCAR). IEEE.

[14] Wang, J. , Wen, G. , Yang, S. , & Liu, Y. . (2019). Remaining Useful Life Estimation in Prognostics Using Deep Bidirectional LSTM Neural Network. 2018 Prognostics and System Health Management Conference (PHM-Chongqing). IEEE.
[15] Li, X. , Ding, Q. , & Sun, J. Q. . (2018). Remaining useful life estimation in prognostics using deep convolution neural networks. Reliability Engineering & System Safety, 172(APR.), 1-11.
[16] Zhuang, L. , Li, J. , Shen, Z. , Gao, H. , & Zhang, C. . (2017). Learning efficient convolutional networks through network slimming. IEEE.
[17] Tian, G. , Chen, J. , Zeng, X. , & Y Liu. (2021). Pruning by training: a novel deep neural network compression framework for image processing. IEEE Signal Processing Letters, PP(99), 1-1.
[18] Kingma, D. , & Ba, J. . (2014). Adam: a method for stochastic optimization. Computer Science.
[19] Saxena, A. , Kai, G. , Simon, D. , & Eklund, N. . (2008). Damage propagation modeling for aircraft engine run-to-failure simulation. 2008 International Conference on Prognostics and Health Management. IEEE.